This is to certify that the

thesis entitled

An Economic Supply/Demand Balance Sheet
For Forecasting And Simulating
The United States Corn Industry

presented by

Anthony B. Washington

has been accepted towards fulfillment
of the requirements for

M.S. degree in Agricultural Economics

James H Hilker
Major professor

Date May 9, 1988

AN ECONOMIC SUPPLY/DEMAND BALANCE SHEET
FOR FORECASTING AND SIMULATING
THE UNITED STATES CORN INDUSTRY

By

Anthony B. Washington

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Agricultural Economics

1988

ABSTRACT

AN ECONOMIC SUPPLY/DEMAND BALANCE SHEET FOR FORECASTING AND SIMULATING THE UNITED STATES CORN INDUSTRY

By

Anthony B. Washington

The price and yield fluctuations faced by agricultural producers are chief sources of uncertainty. In an effort to quantify some of the sources of uncertainty, educate producers where the uncertainty exists, and aid in their production decisions, an economic supply/demand balance sheet for use on a microcomputer was designed. The balance sheet uses time series data to forecast the U.S. supply and world demand for corn in a recursive model to determine a U.S. marketing clearing farm price of corn.

The user, be it an educator or producer, can then select to do scenario analysis with the balance sheet by altering the exogenous variables or forecasted values and review the results. Further, the user can chose a particular scenario to estimate a price probability distribution. This probability distribution serves to help producers determine the probable states of nature for risk aversion/decision analysis.

ACKNOWLEDGMENTS

All praise goes to

I am indebted to the people here at Michigan State University for the opportunity to complete my Masters of Science degree. I will never forget you.

My major professor, Dr. Jim Hilker, deserves special thanks for his guidance and patience in supervising my program and reviewing manuscripts, without his help this project could never have been undertaken. I am especially thankful to my committee members, Drs. Roy Black, Gerald Schwab and Jack Meyer for their valuable comments and input.

My gratitude goes to Eric DeVuyst, Erik Zackmann and Rob Leland members of MSU's Ag Model, for the technical assistance they provided.

Finally, I must thank my family members. Each one of them are educationally oriented and have inspired me to set lofty goals and stick to them regardless of the hardships faced.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd)

Page

TABLE OF CONTENTS (cont'd)

Page

Appendices

LIST OF TABLES

LIST OF TABLES (cont'd)

LIST OF TABLES (cont'd)

LIST OF FIGURES

Figure Page

LIST OF FIGURES (cont'd)

Figure Page

CHAPTER I

THE PROBLEM AND OBJECTIVES

1.1 PROBLEM SETTING

Of the many uncertainties facing farmers, yield and price fluctuations of the products farmers sell comprise the key sources of uncertainty. The problem is that inputs must be allocated at one point in time while returns come at a later date. Because the future cannot be foretold, the inputs must be allocated without perfect assurance of a particular return. Over the last decade, dramatic price movements, both up and down, have increased interest in more and better information for both policy makers and individual producers.

When farmers face a planning period, they need to have an expectation of the price that they will receive for their efforts. A probability distribution around that price would also be helpful. This price expectation is what allows producers to choose how they will allocate their resources and the probability distribution around the price is what allows for individual risk behavior to enter the analysis. With information in this form, decision makers will be better able to comprehend the degree of uncertainty associated with future events. Furthermore, more accurate perceptions about the uncertainty associated with future events will allow decisions to be made that are more consistent with producers individual objectives and risk preferences.[1]

Recalling from classic micro economic theory, we know that there exist several market structures from which an industry may be operating. The farming industry is considered a close representation of a purely (atomistically) competitive structured market in the sense that individual farmers are price takers. The structure of this market dictates that (among other things):

1) There are a large number of firms, each producing homogeneous products,

2) New firms are free to enter and existing firms are free to exit the industry,

3) There is generally no nonprice competition, and most significantly,

4) Each firm is a price taker assuming that its actions have no effect on market price.

Being a price taker implies two major conditions occur. First, producers cannot influence the market price of their product due to the relationship between their production and the quantity of total supply. Second, in the corn subsector, producers are unable to create a unique demand

[1] A. N. Halter, and R. Mason, "Utility Measurement for Those Who Need to Know," Western Journal Of Agricultural Economics, Vol. 3, No. 2, December 1978, p. 99-109.

for their goods because of the homogeneity of their product. By understanding the structure, comparative static analysis can be conducted.

1.2 GOAL AND OBJECTIVES

The goal of this project is to develop a microcomputer economic supply/demand balance sheet for forecasting corn prices which can be used as a group-interaction teaching aide.

To reach this goal several objectives must be met. The primary objective is for the balance sheet to be simple to use and easy to update or modify yet, comprehensive enough for educational purposes. Second, nation wide corn supply and demand relationships need to be estimated from the available data. Third, these estimated equations will be combined in an algorithm for use on a microcomputer. Finally, the algorithm will be menu driven by the user to display various screens of information which present a comprehensive balance sheet as exhibited in Table 1.

1.3 PROJECT JUSTIFICATION

The art/science of forecasting corn prices is imprecise, however most successful market analysts follow an organized procedure. An economic supply/demand balance

Table 1: USDA Balance Sheet For Corn

	1981-82	1982-83	1983-84	1984-85	1985-86	PROJ. 1986-87	PROJ. 1987-88
			(MILLION ACRES)				
ACRES SET-ASIDE AND DIVERTED	--	2.1	31.7	4.2	5.9	13.6	21.5
ACRES PLANTED	84.1	81.8	60.2	80.4	83.2	76.7	66.0
ACRES HARVESTED	75.5	72.7	51.4	71.8	75.1	69.2	59.2
BU/HARVESTED ACRE	108.9	113.2	81.0	106.6	118.0	119.3	119.4
			(MILLION BUSHELS)				
BEGINNING STOCKS	1034	2174	3120	723	1648/1	4040	4882
PRODUCTION	8119	8235	4175	7674	8877	8250	7064
IMPORTS	1	1	2	3	11	2	2
TOTAL SUPPLY	9154	10410	7297	8401	10536	12291	11948
USE:							
FEED	4202	4521	3737	4117	4095	4715	4900
FOOD, SEED & IND. USES	811	898	973	1065	1160	1191	1225
TOTAL DOMESTIC	5013	5419	4710	5182	5255	5906	6125
EXPORTS	1967	1871	1864	1838	1241	1504	1700
TOTAL USE	6980	7290	6574	7020	6496	7410	7825
TOTAL ENDING STOCKS	2174	3120	723	1381/1	4040	4882	4123
TOTAL, % OF USE	31.2	42.8	11.0	19.7	62.2	65.9	52.7
STOCKS IN GOVT. PRGRAM							
CCC INVENTORY	302	1166	201	240	546	1443	
FARMER-OWNED RESERVE	1300	1550	425	443	564	1321	
UNDER GOVT. LOAN	385	109	20	567	2736	2279?	
TOTAL	1987	2825	646	1250	3846	5043	
FREE BUSHELS	187	295	77	129	192	-161?	
% OF USE	2.7	4.0	1.2	1.8	3.0	0.0	
REGULAR LOAN RATE	$2.40	$2.55	$2.65	$2.55	$2.55	$1.84	$1.82
U.S. SEASON AVERAGE FARM PRICE, $/BU	$2.50	$2.68	$3.25	$2.62	$2.35	$1.50	$1.65- $1.85

SOURCE: USDA.

1/1984-85 ending stocks do not = 1985-86 beginning stocks due to marketing year change.

sheet (henceforth called the balance sheet) such as that shown in Table 1, is the most common starting point.[2]

A balance sheet is similar to a checkbook. It records how much corn is in the bin at the start of the year, how much corn is produced during the year, how much grain disappears during the year, and how much grain is left in the bin at the end of the year. When the quantities supplied and demanded are in balance (equilibrium), a market clearing price is established.

The computerized balance sheet can be used as a practical teaching tool by University staff for educating individuals involved in cash grain enterprises. University staff can use this mobile balance sheet at extension meetings to show what factors influence the market price for corn and how changes in those factors will modify the estimated market clearing price.

The balance sheet can be used to provide information to anyone involved in the planning stage of production farming such as farmers and extension workers. The balance sheet could assist an individual identify an expected corn price and the variation around that expected price. Information about expected price and the variation associated with that price could aid producers decide upon a crop mixture based

[2] Unlike the accountant's balance sheet this does not relate to a point in time but to the balance of flows over a period of time, generally a year. The term is used in the same sense as the Food and Agriculture Organization of the United Nations uses it in their Food Balance Sheets.

on capability of diversification and relative prices, while still maintaining eligibility for government programs.

The balance sheet could provide a portion of the information needed in the marketing area by supplying both an expected price and a price probability distribution. With information in the form of expected price and probability, producers are better able to determine which marketing alternatives (i.e. hedge, forward contracts, options, storage, etc.) will be most in line with their objectives.

Each supply and demand factor of the balance sheet will be available for adjustment. This feature of the balance sheet will allow for quick testing of "what if" situations. Given an adjustment, a new solution can be found and the effect on price of that adjustment will be available for analysis.

Finally, the balance sheet could aid in management decisions. Whenever an "if" creeps into the statement of a solution, the concept of risk is implied. By providing a probability distribution associated with the price, producers could make management decisions based upon their own level of risk preference.

1.4 PROCEDURE

In Chapter 2 the theoretical and conceptual considerations for model construction are developed. Elements of supply and demand are presented to provide a basis for static analysis. A review of some of the relevant literature is also included. Chapter 3 contains the derivation of the model. The data calculations, equations used, and empirical results are presented.

The balance sheet will be designed in Chapter 4. The chapter will present case studies under different scenarios, and provide documentation on the use of the balance sheet. Finally, Chapter 5 will present a summary, conclusions and recommendations for future work on the balance sheet.

CHAPTER II

THEORETICAL AND CONCEPTUAL CONSIDERATIONS FOR MODEL CONSTRUCTION

This chapter will discuss the necessary economic theory, and present the conceptual framework needed to construct equations that will be used in the single commodity forecasting model.

2.1 ELEMENTS OF SUPPLY

The economic unit for the sale of goods and services is the firm. The firm can be a small farmer, cultivating his own land with the help of the members of his family, or it may be a nationalized organization owned by all and operated either directly or indirectly by a local or central authority. In static economic analysis, we are not directly interested in whether the firm is large or small, or if it is owned by one or many, but what happens to supply when firms revise their sales and or purchase plans.

2.1.1 Theoretical Considerations

Ryan provides the theoretical background of the firm in the short run.[1] Doll states that the short run is the time period of such length that at least one resource is varied while other resources are fixed.[2] Thus this is the most appropriate time period for the analysis in this project.

The production plan of the firm sets the quantities of each of its products that it plans to sell during any period of time, and the price per unit at which it hopes or plans to sell each of them. If 'P' is used to represent the expected selling price, 'Q' to represent the quantity that it hopes to sell, and the subscripts 1,2,3,...,n, to designate particular products, then the production plan will have the following general form:

$$P_1 * Q_1 + P_2 * Q_2 + \ldots + P_n * Q_n = R,$$

where R represents the gross revenue that the firm hopes to get from implementing the plan as a whole. This plan assumes the state of the arts in technology, a particular objective in which the firm is pursuing, and a predetermined length of time.

The revenue 'R' generated by the production plan is the value of the output produced and is determined by the expected product prices multiplied by the quantity produced.

[1] W.J.L. Ryan, Price Theory, Macmillan and Company Limited, London, 1958, Chapter 2.

[2] John P. Doll, Production Economics: Theory With Applications, John Wiley & Sons, New York, 1978, p.17.

It is assumed (in this industry structure) that the firm will be able to sell everything produced at a given market price. It is further assumed that the firm will choose that level of production which optimizes any cost-benefit analysis and generates the highest revenue. This assumption is reasonable because most farmers trade-off between maximum revenue (benefit) and the degree of risk (cost) associated with that level of revenue and it seems to approximate reality.

In order to realize the plan, factors of production are needed. The quantities of land, labor, and capital, and the price it expects to pay for them are set out by the purchase plan. If 'F' is used to represent purchase price, 'X' to represent quantity to be bought, and the subscripts 1,2,3,...,m, to designate particular inputs that are needed, then the general form of the purchase plan will be:

$$F_1 * X_1 + F_2 * X_2 + \ldots + F_m * X_m = C,$$

where C represents the total expenditure that the firm feels must be incurred to implement the production plan. This total expenditure must include all opportunity costs to land, labor, capital, and management used in producing the outputs that are listed in the production plan.

There are many alternative plans from which the firm can choose, each with a corresponding purchase plan. In the short run, alternatives are limited and a budget constraint is imposed. The budget constraint means that the capital available during the length of run is not sufficient to allow the firm to use an unlimited amount of factors on each product. One must assume that the firm will always plan to produce each product with the quantities of factors that cost least and meet the objective that the firm is pursuing.

To discover how a firm will revise its production plan given a change in information, the production possibility surface must be introduced. The production possibility surface is the combination of products which can be produced from a given amount of factors.

The concept of marginal rate of product substitution (MRPS) refers to the amount by which one product changes in quantity when another product is increased by one unit along the product possibility surface ceteris paribus.

The critical part of this analysis is the marginal criterion for resource allocation. Maximum net revenue from a limited amount of factors of production occurs when the value of the marginal product of the factor is the same for each product.

$$\frac{VMPx_1q_1}{f_1} = \ldots = \frac{VMPx_nq_1}{f_n} = \frac{VMPx_1q_2}{f_1} = \ldots = \frac{VMPx_nq_2}{f_n}$$

$$= \ldots = \frac{VMPx_1q_m}{f_1} = \ldots = \frac{VMPx_nq_m}{f_n} ,$$

where VMP is the product price multiplied by the marginal physical product for each output. This marginal resource allocation criterion specifies that when all ratios are equal to one, the optimum amount of factors are used to produce each product given the firm's objectives.

The relationship between planned sales and the expected selling price is the firm's short run supply schedule. This supply schedule shows the production plan that the firm would choose at each expected selling price, with its given production possibilities, objective and contractual obligations, and the given prices expected from the purchase plan.

The supply schedule of each existing firm can be derived in precisely the same manner. By assuming that the prices of the variable inputs are the same for all firms, the industry supply schedule may be obtained simply by horizontally summing the supply schedules for each firm at each expected selling price. While each firm operates under the assumption that the market price for its products is beyond its control, the total effect of all firms implementing their production plans is to assist in the determination of the market price for each product.

The price elasticity of supply measures the responsiveness of planned sales to changes in the expected selling price holding all other factors constant. Elasticity is valid because it causes the measurement of responsiveness to be unitless, and only records changes in terms of percentages. Given a large elasticity, firms show a great responsiveness to changes in price. With a small elasticity, firms show little responsiveness. The aggregate short run supply on farm products is extremely inelastic. Tweeten and Quance estimate that the short run price elasticity of aggregate supply of farm products in the U.S. falls somewhere between 0 and 0.2.[3] Tweeten goes on to estimate short run elasticities of supply for selected commodities and finds the elasticity for feed grains to be 0.4.[4]

In the next section the conceptual considerations needed to derive the supply equations for the model are presented.

2.1.2 Conceptual Considerations

The industry supply of corn for grain in any period is based upon the amount of corn produced in the period and the amount of carryover from the previous period plus any imports. Factors which affect the production are the price of corn, the ability to incorporate substitute crops given the relative price differences, the capital flow of the firm, the biological nature of agricultural production, and the existence or absence of government programs. The carryover can be viewed as predetermined, and the imports of corn grain into the U.S. nearly insignificant.

Soybeans and wheat are major competitors for corn land and production resources. Therefore the prices of these

[3] L.G. Tweeten, and C.L. Quance, "Positivistic Measures of Aggregate Supply Elasticities: Some New Approaches," AJAE, Vol.51, No.2, 1969, p. 351.

[4] L.G. Tweeten, Foundations of Farm Policy, University of Nebraska Press, Lincoln, 1970, p. 243.

crops are important determinants of expectations for future prices. If the expected profit of the substitutes is higher relative to the expected profit of corn the profit maximizer will substitute one or more of these alternative crops into production in order to equate the value of marginal product from an acre of the substitute crop to the value of marginal product from an acre of corn.

Nerlove examined three works based upon studies of expectations of the future value of economic variables.[5] These studies show that there is widespread underestimation of actual changes and that forecasters could generally do a better job at predicting the levels of outcomes if they used a naive model. In the study of static economic theory of supply, price changes are usually considered as permanent changes. It is argued that the profitability of responding to price changes considered only temporary is likely to be so limited that entrepreneurs forecast and react primarily to changes which they consider to be permanent, i.e., changes in the average level about which future prices are expected to fluctuate. This average is what Nerlove refers to as expected "normal" price. Nerlove raises the question as to whether entrepreneurs are really trying to forecast a particular value of an economic variable, or whether, they try to forecast the "normal" level of future values of the variable. Nerlove adds, "Any model of expectation formation should take account of the fact that these expectations

[5] Marc Nerlove, The Dynamics of Supply: Estimation of Farmers' Response To Price, The John Hopkins Press, 1958.

probably do not refer to the immediate and temporary future" (i.e. the immediate and very short run lengths of time).

Expectations of "normal" price are shaped by a multitude of influences. Expected prices cannot be observed and must be approximated by observable prices, so that a representation of expected price as a function of past prices may merely be a convenient way to summarize the effects of these many and diverse influences of price expectations. Eisner suggests that expectations may be based on a concept of the normal as a starting point in our development of a model of expectation formation.[6]

Government policy reflects an attempt to reduce surplus production, address the problem of low farm income, and the problem of price instability. Historically, plantings of corn have been substantially influenced by government policy. Mc Keon notes that increases in the level of corn price supports have increased plantings while increases in diversion payments have reduced plantings. He continues; "The significant response to diversion programs reflects the fact that producers, throughout the U.S., were willing to remove from production poorer quality crop land in return for diversion payments. As better land is bid out of production, however, the response to diversion payments

[6] Robert Eisner, Expectations, Plans, and Capital Expenditures: A Synthesis of Ex Post and Ex Ante Data presented at the Conference on Expectations, Uncertainty, and Business Behavior, October 1955.

would decline."[7] In all works reviewed, the government policy towards crop supply has received much attention.

As discussed earlier, each firms supply curve is shaped by the expected selling price. Since the farm price in recent years has been below the guaranteed support prices, producers, having the option, have found it advantageous to participate in the government programs. These producer options for reaction to price stimuli would support the testable hypothesis, as presented by Houck, Ryan, and Subotnik,[8] that the expected prices can be expressed by a linear function:

$$P^{*}_{it} = \alpha_{11} P_{it-1} + \alpha_{12} PV1_{it}$$

where P^{*}_{it} is the expected price of crop i in year t, P_{it-1} is the actual price received by farmers for crop i in year t-1, and $PV1_{it}$ is the price support policy variable for crop i in year t, the α_i's are the coefficients associated with the explanatory variables.

Price support operations have consisted of guaranteed support prices, acreage restrictions, and diversion payments for fallowing land. The guaranteed support prices include a nonrecourse loan, and direct support payments which vary according to participation level and historical production

[7] John Mc Keon, Farm Commodity Programs: Their Effect On Plantings of Feed Grains and Soybeans, doctoral dissertation, Michigan State University, 1974, Abstract.

[8] J.P. Houck, M.E. Ryan, and A. Subotnik, Soybeans and Their Products, University on Minnesota Press, Minneapolis, 1972, p.90.

levels. The diversion payments, when in operation, use the same criteria as the support payments. Because of the various elements of the price support and diversion payments, it is difficult to measure the impact of the government programs on plantings. Houck and Ryan suggest a means of weighting the announced price support by the acreage restrictions imposed, to obtain the "effective price support."[9] Similarly, they obtained an "effective diversion payment" by weighting the announced diversion rate by the eligible diversion acreage.

a) Acres Planted

A number of researchers have examined supply responses for various crops. Each of their models had feed grain submodels and some disaggregated feed grains into corn specifically. A short review of some empirical research will follow.

The Mc Keon Model

Mc Keon develops theoretical and empirical estimates of acres planted to feed grains and soybeans.[10] The Mc Keon study, cites the work of John L. Dillon to develop the necessary economic theory for calculating the input demand function. From there he modifies the function into a general statistical form which follows:

[9] J.P. Houck, and M.E. Ryan, "Supply Analysis for Corn in the United States: The Impact of Changing Government Programs," AJAE, Vol. 54, No. 2, May 1972, p. 184-91.

[10] Mc Keon, op. cit.

$$AP_{it} = \beta_o + \beta_1 P^*_{it} + \beta_2 P^*_{jt} + \beta_3 IP_{kt} + \beta_4 Z + U_t ,$$

where AP_{it} is the acreage planted to crop i in year t, P^*_{it} and P^*_{jt} are the expected prices of crop i and competing crop j, respectively, in year t, IP_{kt} is the price of input k in t, Z includes such factors as government programs, technical change and weather, the β_i's are the coefficients associated with the explanatory variables, and U_t is a random mean-zero disturbance with finite variance.

This equation offers a guide to construction estimable equations for supply response. Mc Keon notes that the form of the price expectations must be presented in observable terms as must the variables on government policy, technological change, weather and input prices.

Following a discussion of the form of the variables which are used in estimation work, Mc Keon identifies a typical equation used in his work. The form of the equation follows:

$$APC_t = \beta_o + \beta_1 PV1C_t + \beta_2 PV2C_t + \beta_3 PC_{t-1} + \beta_4 P_{kt-1} \\ + \beta_5 YC_{t-1} + \beta_6 W_t + \beta_7 IP_t + \beta_8 DV + U_t ,$$

where APC_t is the acreage planted to corn in time t, $PV1C_t$ and $PV2C_t$ are the calculated effective price support and effective diversion payment (as put forth by Houck and Ryan), respectively, for corn in time t, PC_{t-1} is the average market price of corn in time t-1, P_{kt-1} is the average market price of the competing crop k in time t-1, YC_{t-1} is the average yield of corn in time t-1, W_t is the average precipitation in April and May, IP_t reflects input prices in time t, DV represents other identifiable shift factors, and U_t is a random, mean-zero disturbance with finite variance, the β_i's are the coefficients associated with the explanatory variables.

The Wailes Model

A later study by Wailes estimates U.S. production for wheat, soybeans, and feed grains, with feed grains disaggregated into corn, sorghum, barley, and oats.[11]

The crop acreage supply relationships specified for his model were developed out of previous studies by Nerlove, Houck, and Mc Keon. These empirical estimates of the acres planted equations provide information on a set of variables which are hypothesized in explaining agricultural supply.

As noted by Wailes, the approach to analysis of crop supply functions by concentrating on the independent estimation of supply relationships for the factors of production has been developed in the seminal articles by D. G. Johnson and M. Nerlove in the 1950's. The theoretical framework is based on neoclassical economic theory which assumes that firms maximize profits and that the demand for the factors of production are determined by conditions of marginal equilibrium.

In the 1960's, Wailes notes, empirical investigations such as the Interstate Managerial Survey by Johnson

[11] E. Wailes, "A Simulation Model of United States Agriculture," doctoral dissertation, Michigan State University, 1983.

(et. al.) generally supported the expectations hypothesis but advanced that forward-looking information available to farmers, not reflected in past prices, was important in the formulation of farmer's expectations. The concept of forward prices developed by D. G. Johnson (1947) became embodied in government programs from the fifties to the present with the objective to influence crop acreage. The identification of the impact of forward price levels as reflected in loan rates and target prices is useful both in terms of how it indirectly conditions the response to past market prices and price uncertainty as well as identifies the direct impact on crop acreage (Houck, et. al.).

The general model specification for the Wailes model reflects an updating of Mc Keon's equations and the extension of the policy variable concepts to encompass the changing policy framework as legislated in the 1973 and 1977 Food and Agriculture Acts. The general acres planted (in period t) specification used in this study follows:

$$AP_t = f(P_{t-1} / CI, PV1 , PV2 , R_t , AP_{t-1}) ,$$

where AP_t = acres planted in period t

P_{t-1} = vector of lagged market prices

CI = Index of costs, CPI was used in all deflations

PV1 = vector of effective support prices, deflated

PV2 = vector of effective diversion payments, deflated

$R_t = (P_{t-1} - (\Sigma P_{t-1} / 3))^2 / (\Sigma P_{t-1} / 3)$, a vector of market price variances

AP_{t-1} = acres planted, lagged one year.

Wailes notes, "This supply framework provides a straight forward basis for simulations assuming alternative government policy strategies and objectives."

The supply sub-model which follows is directly from the Wailes discourse, and shows acres planted to corn is based upon the relationship between corn plantings and lagged corn, soybean and wheat prices and government feed grain programs

$$AP_t = \beta_o + \beta_1 CP_{t-1} + \beta_2 SP_{t-1} + \beta_3 WP_{t-1} + \beta_4 PV_1CT + \beta_5 PV_2CT + \beta_6 TL ,$$

where AP_t is acres planted to corn in thousand acres, CP_{t-1}, SP_{t-1}, and WP_{t-1} are the deflated corn price, soybean price, and wheat price received by farmers lagged one year, respectively, PV_1CT is the deflated effective support rate for corn in dollars per bushel, and PV_2CT is the deflated effective diversion payment in dollars per bushel, TL is a trend, the β_i's are the coefficients associated with the explanatory variables.

The Heady Model

The econometric simulation model by Earl O. Heady showed an alternative approach.[12] Heady designed his model with five commodity submodels at the national level. Each

[12] Earl O. Heady, An Econometric Simulation Model of the U.S. Farm Sector and its Policies and Food Exports, Vandenhoeck & Ruprecht, 1978.

commodity sub-model is divided into three categories, referred to as the pre-input, input, and output sections, corresponding to the planning, planting and harvesting decisions in the production cycle. The first category of equations is the pre-input section. The initial decisions which the producer will make, and are included in the pre-input section, include the number of acres which will be devoted to production.

Heady models the pre-input section acres planted to feed grain as:

$$FG\text{-}AC_t = \beta_o + \beta_1\ FG\text{-}PR_{t-1} + \beta_2\ S\text{-}PR_{t-1} + \beta_3\ FG\text{-}ACDIV_t + \beta_4\ FG\text{-}AC_{t-1}$$

where:

β_i = coefficients associated with the explanatory variables,

$FG\text{-}AC_t$ = feed grain acreage in million acres,

$FG\text{-}PR_{t-1}$ = feed grain average price received by farmers deflated by the implicit GNP deflator,

$S\text{-}PR_{t-1}$ = soybean average price received by farmers deflated by the implicit GNP deflator,

$FG\text{-}ACDIV_t$ = acreage diverted from production in million acres,

$FG\text{-}AC_{t-1}$ = feed grain acreage lagged one period.

Following the pre-input section, the variable resource requirements are estimated in the input section. The levels of resource demand established in the input section depend upon the levels of the fixed inputs from the pre-input section as well as the values of endogenous variables from previous years. For example, variable resource demand in the current year depend on acreage estimates from the pre-input section as well as last year's gross income, which serves as a proxy for a capital constraint. The recursive structure of the model is preserved, since the solution of the equations occurs in a sequential manner.

The FAPSIM Model

The Economic Research Service, National Economics Division developed a report which provides a detailed description of the structural equations and their statistical attributes in the current version of the Food and Agricultural Policy Simulator (FAPSIM).[13]

As noted by the authors of FAPSIM, a major shortcoming of previous research has been the failure by economists to develop acreage response equations that explicitly predict the level of farmer participation in government commodity programs. For example, Houck and Ryan (et. al.) developed response equations containing government policy variables such as the effective support price and the effective diversion payment rate. The authors continue, these equations can be used to predict total acreage response, but they cannot predict the level of participation in government programs. Central to this approach is the development of

[13] Kenneth E. Gadson, J. Michael Price, and Larry E. Salathe, U.S. Department of Agriculture, Staff Report AGES820506, May 1982.

acreage response equations with the construction of variables that reflect farmer's perceptions of expected yields and prices.

The acreage response relationships contained in FAPSIM reflect the relative profitability of either participation or not participating in a government commodity program.[14] The expected net return per acre for a program participant producing crop i is:

$$\begin{aligned} EPR_i = {} & [(EPP_i * EY_i - VC_i)\ (1.0 - (SA_i + DIV_i))] \\ & + [SR_i * PY_i\ (1.0 - (SA_i + DIV_i))]\ ALLOC_i \\ & + [DR_i * PN_i * PY_i * DIV_i] \end{aligned}$$

where:

EPR_i = expected program return per acre for crop i,

EPP_i = the maximum of the loan rate and the expected market price,

EY_i = expected yield per acre,

VC_i = variable cost per acre,

SR_i = expected deficiency payment rate (announced target price minus the maximum of the expected market price and loan rate) per acre,

PY_i = national program yield,

SA_i = proportion of each acre required to be set-aside,

DIV_i = proportion of each acre required to be diverted,

[14] The acreage response equations contained in FAPSIM follow from previous research by Robert Bancroft of the University of Vermont, while he was employed by ERS.

DR_i = diversion payment rate per bushel,

$ALLOC_i$ = minimum national program allocation factor, and

PN_i = proportion of acreage eligible for diversion payments.

The expected net return per acre for a government program nonparticipant producing crop i is given by the identity:

$$EMR_i = EMP_i * EY_i - VC_i$$

where:

EMR_i = expected market net return per acre for crop i,

EMP_i = expected market price,

VC_i = variable cost per acre, and

EY_i = expected yield per acre.

The expected net return variables are used to estimate acreage response by participants and nonparticipants. Total acreage in the program (planted plus diverted and set-aside acreage) for crop i is expressed as a behavioral relationship of the form:

$$PA_i = f\ (\ EPR_i/CPI,\ EMR_i/CPI,\ APP_i/CPI,\ NP_i(1.0 - SD_i))$$

where:

PA_i = program acreage for crop i,

APP_i = the average expected net return of competing crops,

NI_i = national program acreage for crop i,

SD_i = set-aside plus diversion rate for crop i, CPI = the all item consumer price index lagged one period;

and where EPR_i and EMR_i are defined as above.

Total acreage planted to a particular crop by program participants is a function of total program acreage multiplied by program set-aside and diversion rates. Acreage set aside and diverted is calculated endogenously as total program acreage minus acreage planted by participants. The participation rate is endogenously computed as acreage planted by participants divided by the sum of acreage planted by participants and nonparticipants.

Acreage planted to crop i by nonparticipants is a function of acreage planted to crop i by program participants, acreage set-aside and diverted, the real expected net return of competing crops, and the real expected market return for planting crop i. Acreage planted in the program, acreage set aside and diverted, and the real, expected, net return of competing crops all represent the desirability of using land for alternative purposes. Thus, acreage planted to crop i by nonparticipants is inversely related to each of these variables. Acreage planted by nonparticipants is positively related to the real, expected, market net return.

The Food Security Act of 1985 calls for a freeze of the target price at the 1985 level with subsequent reductions in the future. Along with the freeze, the Secretary of Agriculture was given authority to lower the loan, which he did. Another change was the formula for calculating base acres and base yields. According to the Act, acreage bases will be the planted and considered planted acreage in the five previous years rather than the previous two years. For the 1986 and 1987, program yields are frozen at the average of 1981-85 levels.

These efforts to lower prices and make U.S. corn more competitive in world markets will result in increased deficiency payments and encourage more farmers to comply with the program. A major uncertainty of the 1985 policy is the implementation of the Gramm-Rudman-Hollings Act to reduce the federal deficit.

The Ferris Model

Ferris uses an equation for acres planted to corn formulated in a manner similar to a partial budgeting approach familiar to many farmers.[15] The key independent variables are: (1) expected net returns from corn over variable production costs per acre outside the government program, (2) expected net returns from soybeans over variable costs per acre, (3) expected net returns on corn over variable cost per base acre for participants in the Feed Grain Program (including direct payments), and (4) percent of base acres that must be put into conserving uses in order to be in compliance.

[15] John N. Ferris, Forecasting Corn Producers' Response to the 1986 Feed Grain Program, Presented at AAEA 1986.

This review shows that there are many alternative structural approaches to estimating acres planted. In keeping with the objectives, a simple equation must be estimated. The common elements of each of the models reviewed were; a variable for expected corn price, a proxy for participation in the government program, and a proxy for revenue received from a competitive crop. This common structure presents a starting point for estimating the acres planted equation.

b) Acres Harvested

Acres harvested are estimated as a simple function of estimated planted acreage. Corn grown for silage, the ability of producers not to harvest part of the crop or any damage done to the crop results in the acres harvested being less than acres planted. The relationship between acres harvested and acres planted is expected to be positive.

The work done by previous researchers has shown that acres harvested can be a function with either an intercept and estimated acres planted, or just estimated acres planted. The function without the intercept generates a fixed percentage relationship between acres planted and acres harvested. When the intercept is included, the value will explicitly handle silage and abandoned acres in the processes of estimating acres harvested for grain. This research will examine both methods for statistical significance.

c) Yield Per Acre

Yield per acre is based upon experience of past yields. This procedure assumes a base yield on "normal" weather conditions and incorporates changes in technology. The amount of yield received from one acre can be affected by many factors. Producers can use new hybrid seed, use more or better fertilizers, and of course, the weather plays a major role on the total yield. Given the level of technology, expected yields can best be represented by regressing actual yields on time which assumes a "normal weather" production period. The technology proxy is expected to have a positive relationship with yields which reflect the increase that yields have experienced over time.

d) Production and Supply

Production and total supply are both identities. The quantity produced is generated by multiplying the number of acres harvested by the yield per acre. Total supply is exactly equal to the sum of production in the current period and the carryover from the previous period plus imports.

2.2 ELEMENTS OF DEMAND

The economic unit for the purchase of the goods and services of everyday consumption is known as the individual. It may be a sole person meeting his own individual needs or it may be a collection of individuals operating together to gather goods for the sustenance of all. Regardless of the size of the individual, the economic actions pursued are where the analysis begins.

2.2.1 Theoretical Considerations

Before beginning the analysis of economic action available to the individual, reference must be made about individual preferences and the concept of utility. This discussion provides the framework for the decision theory analysis.

Economists assume that individuals make choices that provide some form of satisfaction. It also follows that those choices are selected by the individual because it provides the maximum satisfaction given a particular situation. It is further assumed that these choices conform to some form of rational behavior, so that a unique ranking of all possible situations from most to least desirable can be made. The political theorist Bentham called this ranking of relative desirabilities utility; more desirable alternatives offer more utility than do less desirable ones.[16]

Since utility measures satisfaction, it is affected by a variety of factors such as cultural environment and psychological attitudes. Due to the complex variety of factors that determine the individual's satisfaction, there is no interpersonally valid way to measure utility. All that can be determined by utility theory is that one bundle

[16] Jeremy Bentham, Introduction to the Principles of Morals and Legislation, Hafner, London, 1948, p. 125-31.

of goods and services is preferred to another, yet, not by how much.

An individual's preferences can be represented by a utility function of the form:

$$U = f(X_1, X_2, \ldots , X_n)$$

where X_1, X_2, ... ,X_n are the quantities of each of n goods consumed.

Some classical assumptions regarding utility are: 1) More is better, 2) There exist opportunities for trade-offs, and 3) Consumers are utility maximizers subject to a budget constraint.

The assumption that more is better stands because individuals are insatiable in their wants for all goods. There is an intuitive feel that when an individual can get more of one good and not sacrifice any of another that improvement is made.

A more important aspect of individual taste that must be noted is the rate of substitution of goods, or how individuals feel about getting more of some good when they must give up an amount of some other good given their current situation. Marshall theorized that it is the marginal valuation that an individual places on a good that determines its value.[17] In light of the trade-off, the individual has an unlimited amount of choices of goods and services which yield the same utility.

[17] Alfred Marshall, Principles of Economics, Macmillan & Co., London, 1920, Book III, Chapter 6.

Typically, when an individual has a small quantity of one good relative to another, the individual is willing to give up a large amount of the excess good to get one more unit of the first good. This coincides with the notion that individuals prefer some variety in their mix of goods, i.e. some balance of consumption is desirable. Intuitively, it seems clear that an individual will tire of some good as more of it is consumed which leads to a diminishing marginal rate of substitution (MRS).

Of particular concern is how an individual allocates a fixed amount of money in purchasing goods so as to maximize the utility derived from these goods. Nicholson defines the optimizing principle for the individual as those quantities of goods which exhaust his or her total income and for which the psychic rate of trade-off between any two goods (the MRS) is equal to the rate at which the goods can be traded one for the other in the market place.[18]

Spending all of one's planned consumption expenditure is required for utility maximization, since extra goods provide extra utility, which defines the individual's budget constraint as follows:

$$P_1 * Q_1 + P_2 * Q_2 + \ldots + P_n * Q_n = I$$

where P is the market price of the good, Q is the quantity of each good purchased, and I is the level of planned

[18] Walter Nicholson, Microeconomic Theory: Basic Principles and Extensions, The Dryden Press, Chicago, 1985, p. 94.

consumption expenditure available to spend on the bundle of goods.

Since the rate at which one good can be traded for another in the market is given by the ratio of their prices, this optimizing principle can be restated to say that the individual will equate the MRS of the goods to the ratio of the prices of the goods and has the following form:

$$U = \frac{MU_{x1}}{P_1} = \frac{MU_{x2}}{P_2} = \ldots = \frac{MU_{xn}}{P_n}$$

where U is the utility derived from the bundle of goods, MU is the marginal utility and P_i is the market price of the good. This equation says that at the utility maximizing point, each good purchased should yield the same marginal utility per dollar spent on that good.

There are many factors which affect the way in which an individual will adjust the utility maximization point, among them a change in income and changes in a goods own or substitutes price are the key elements.

As total expenditures rise, it is natural to expect that the quantity of each good purchased will also increase. A normal good (as in most cases) implies as income increases, purchases increase, yet when, the quantity of a good deceases as income rises, the good is referred to as an inferior good.

The effect of price change on the quantity of a good demanded is somewhat more complex to analyze than is the effect of a change in income. This is because of two analytical affects which result from a price change; income and substitution effects

The income effect arises because a price change necessarily changes an individual's real income. The total real income will increase when there is a price decrease for any good and it will decrease when the price of any good increases. The substitution effect arises from a price change because consumption patterns are allocated so as to equate the MRS to the new price ratio.

It has been shown that an individual's demand for a good depends on the shape of the utility surface and on all prices and income. Theoretically, this can be shown as:

$$Q^*{}_1 = D_1(P_1, P_2, \ldots , P_n, I)$$

where Q^* is the quantity of the good purchased, P_i is the price of all goods available and I is the desired level of consumption expenditure, and the subscripts represent the range of possible prices on each good.

By choosing alternative prices for the good and, assuming that other determinants of demand are held constant, the quantities of the good in question can be mapped out. The relationship between the price of a good and the quantity of that good purchased is the individual's demand schedule.

The demand for each potential individual can be determined in precisely the same manner. Since the prices available to the individual are the same for everyone, and individual demand is dependent upon his or her own income, the total or market demand for the good can be obtained by horizontally summing the demand schedules for each individual at each potential price.

One of the most important applications of the elasticity concept is that of price elasticity of demand. Elasticity of demand records how quantities change (in percentage terms) in response to a percentage change in own price holding all other factors constant and is typically negative for demand functions. Given an "elastic" demand schedule, a price increase is met by more than a proportionate quantity decrease. With an inelastic demand schedule, price increases proportionally more than quantity decreases.

Another type of elasticity frequently encountered in economics is the income elasticity of demand. This concept records the relationship between income changes and quantity changes. Goods for which income elasticity is greater than 1 might be called luxury goods in the sense that purchases of these goods increase more rapidly than income. On the other hand, a good which is a necessity probably has an income elasticity of less than one.

The cross elasticity of demand is the responsiveness of the good demanded to a change in the price of some other good. The value of cross elasticity may be positive or negative, depending on the direction in which the demand schedule moves. In general, goods with a positive cross elasticity of demand are regarded as substitutes for one another, while goods with negative cross elasticities are regarded as complements for one another.

The next section presents the conceptual considerations needed to derive the demand equations used for model construction.

2.2.2 Conceptual Considerations

Conceptually, corn demand can be typed into one of two categories. First, there is the immediate demand for consumption and exports. Second, there is the demand for storage and speculation.

The immediate demand or disappearance of corn in the U.S. can be grouped into three classes. First there is the grain demanded for feed by livestock producers. Second, the food, seed, and industrial use must be accounted for. Finally, the amount of utilization in any period is dependent upon the level of exports. Total disappearance of corn (both domestic use and exports) has trended upward during the past two decades. In the past decade, corn exports grew faster than domestic use, resulting in a 15 percent increase in total disappearance.

Since corn is a seasonally produced commodity and consumed throughout the year, there is a demand for storage and speculation. Storage acts as a buffer against price fluctuations by absorbing excess in years of low prices and releasing stocks in years of high prices. Speculators hold the commodity in anticipation of a price increase that will provide them a profit.

a) Corn Consumed By Livestock

In 1982, corn consumed by livestock accounted for 84 percent of the domestic use of corn while also accounting for 82 percent of all grains fed to livestock. Competition among feed ingredients depends on relative prices and relative feed nutrition values. Producers can vary the rate in which they feed corn based upon the current price and availability of corn and substitutes (mainly soybean meal), the revenue generated by their particular enterprise, and any change in technology that structurally improves the efficiency of feed conversion.

Feed use of corn, being a derived demand, is positively related to the number of animal units on feed. By disaggregating corn consumed by livestock into separate animal species, and estimating the rate of feeding per head, an estimate of total corn consumed by livestock can be made by multiplying the rate per head by the inventory value and summing across each species. This technique should improve the overall estimate of corn consumed by livestock.

b) Food, Seed, And Industrial Use Of Corn

Food, seed, and industrial (FSI) use is expected to continue the upward trend established over the past 10 years. However, evidence is mounting that future increases will not be nearly as large as in the past. In the early 1980's, FSI use increased an average of more than 10 percent a year, but for the 1986 crop year, only a 2 percent rise is projected.

The FSI use of corn doubled in the 1970's as a result of expanding markets for high fructose corn syrup, other sweetener products, and ethyl alcohol. Each of these factors will be examined to help explain why this quantity of demand cannot easily be estimated.

i) FOOD

High fructose corn syrup (HFCS) has held the highest portion of FSI usage in this decade, and currently ranges close to 30 percent. HFCS is used in a variety of products including soft drinks, baking and cereal products, and dairy and canning products. Soft drinks can use HFCS for nearly 100 percent of their sweetener needs, however, it is not technically possible for most other products to do so. Almost all of the products containing HFCS are approaching their theoretical limits of use, therefore, continued growth can not be expected without increased future demand for end products, population growth, or development of new uses for HFCS.[19]

ii) SEED

Seed use is primarily related to the number of acres to be planted in the coming season with secondary effects from

[19] Feed Situation and Outlook Report, August 1986, p. 11-13.

trends and from crop prices. Typically, the higher the market price the higher the seeding rate. Since the seed used by producers is a special hybrid variety, the demand for hybrid seed will need to be estimated.

iii) INDUSTRIAL

Alcohol production from agricultural commodities is not new. During the Great Depression of the 1930's, there was considerable interest in producing ethanol from grains. Ethanol currently accounts for nearly 25 percent of FSI usage.

A great deal of interest in domestically produced oil substitutes was evoked by the Arab oil embargo in 1973, the Iranian crisis of the 1980's, and the decontrol of domestic crude oil prices in 1981. Each of these events caused gasoline and crude oil prices to increase rapidly. Since ethanol can be made from renewable resources such as grain and other agricultural biomass, and serve as a gasoline extender, major government efforts were undertaken to stimulate this domestic industry.

There are two types of milling processes for obtaining ethanol from corn grain. Both yield approximately the same amount of ethanol per bushel of grain.[20] The dry-milling process is the simpler process, but the wet-milling process yields more valuable by-products one of which is high fructose corn syrup.

[20] Donald R. Hertzmark, "Economic Feasibility of Agricultural Alcohol Production Within a Biomass System," AJAE, Vol. 62, No. 5, December 1980, p. 965-71.

A number of important issues are raised when considering fuel alcohol production from agricultural commodities. Whereas ethanol production is seen as a means to increase farm commodity prices and income, what impacts will ethanol production have on food prices? What are the impacts of farming land more intensively, and possibly bringing more land into production? What will be the impact on substitute commodity prices such as soybeans and sorghum?

Given these policy issues and the current worldwide glut of crude oil, enthusiasm for ethanol and other alternative fuels has diminished.

c) U.S. Corn Exports

Due to the dominant role of the U.S. corn industry in the world coarse grain market, the domestic supply, demand, and U.S. government policy changes, significantly affect world markets for coarse grain. Since 1970, the level of U.S. coarse grain exports has been closely tied to the level of the deficit outside the U.S. The implication is that, estimates of the deficit outside the U.S., provide a good handle on what U.S. exports would be.

To model the U.S. export relationship, the world coarse grain situation must be examined. The assumption made for this project is that there are three regions in the world; the U.S., the residual exporting countries, and the importing region.

Estimates will be made for the net imports on a per capita basis demanded by the importing region. It is necessary to include population in the formulations to account for any changes in demand caused by the annual trend increases in population.

By assuming that the each country in the exporting region is working toward internal price stability, exporter net exports per capita can be an identity which specifies that the region will export everything not consumed or held for stocks. This quantity in the international component should also be estimated on a per capita basis. The difference between the quantity demanded by the importing region and the quantity supplied by the exporting region will be the U.S. exports.

d) Stocks For Storage

Since the 1930's, the U.S. government has used price supports on many storable commodities and thus have held stocks to help maintain both orderly marketing, and prices above equilibrium. The current stock holding programs include grain purchased under government loan and held in the Commodity Credit Corporation (CCC) and that owned in farmer reserve (FOR). Free stocks and those held for speculative purposes also account for a portion the demand for stocks. The effect that increased stocks have on U.S. prices will depend on whether the stocks are stored under policy rules or as free stocks.

Although the government holds stocks to maintain price by soaking up excess production, the knowledge of the quantity and the release price for FOR and CCC stocks discounts its effectiveness. The quantities of grain placed in FOR and CCC are not residual ending stocks but are determined by government programs and producer expectations. Ferris notes that an adjustment must be made to account for the knowledge of the quantity stored and release price of grain in FOR and in the CCC.[21] Rather than adding all FOR and CCC grain to the ending stock equation, Ferris deletes about half of the grain held in FOR and CCC. The rational is that grain held off the market would tend to raise prices. However, the existence of grain in the government program also has a depressing effect on the market. Thus, the relevant ending stock equation for U.S. policy stocks is actually somewhere between total ending stocks and free stocks.[22]

It is necessary to develop rules for accumulation and release of FOR and CCC stocks based on producer expectations, the quantity stored and the release price. If the farm price is less than the loan rate plus interest and carrying charges, the producer has no incentive to redeem his loan and the grain held under FOR and CCC will increase. The lower the expected price relative to the loan rate, the

[21] J. Ferris, "Feed Grain, Wheat, and Oilseed Supply and Demand," MSU Staff Paper, October 1985.

[22] Total Ending Stocks = Free stocks + % CCC + % FOR.

more grain will be stored as more producers participate and a greater percentage of acres are committed to the program.

When prices are expected to be above the loan rate plus interest and storage there is little incentive to participate in the program and there is little accumulation of government stocks. If prices rise to a policy established trigger, the government would begin to release grain onto the market. The quantity of grain released would depend upon the upward pressure on price.

Shagam hypothesizes that a cubic function is the form which could provide a smooth curve with the correct shape to model the turning points in the accumulation and release of government policy stocks.[23] Shagam notes that one of the shortcomings of a cubic form for an accumulation and release of policy stocks equation is its potential to be explosive if prices are very high or very low.

Although not all inventory positions are not held for speculation, some stocks are held by individuals in the anticipation that price will rise during the marketing year. Assuming free stocks are held for speculative purposes, a relationship can be made between anticipated use, the perceived level of policy stocks and prices (relative to current prices).

When all of the demand quantities (both immediate consumption and speculative demands) are summed

[23] Shayle Shagam, "Specification and Structure of an International Agricultural Trade Model With Linkage Capabilities," master's thesis, Michigan State University, 1986, p. 51.

horizontally, the total demand for the time period can be calculated.

2.3 PRICE DETERMINATION

This section will present the theory and conceptualization needed to forecast the annual average U.S. farm price of corn.

2.3.1 Theoretical Considerations

To simplify the price determination analysis, economists use the concept of a perfectly competitive market to model agricultural product price behavior. The federal government intervenes in the pricing of grains (as well as other products) in the U.S., but models which assume competitive behavior are still useful in establishing a theoretical norm against which actual behavioral price effects can be measured. In a purely competitive market, it is assumed that every firm seeks to maximize profits by selling at as high a price as possible, and every individual seeks to maximize utility by obtaining goods at as low a price as possible. The collective actions of firms and individuals determine prices.

In the analysis of pricing it is important to decide the length of time that is to be allowed for a supply response to changing demand conditions. To further simplify the analysis, the assumption that the production period for the firm is the same as the purchase period of the individual is made. If the purchase and production plans of the firm are made at the beginning of the period, and once these plans are made, they remain unchanged until the next period, then, a theoretical static analysis can made.

When the price and quantity relationships for the firm's short run supply schedule and for the individual's current demand schedule are equated, an equilibrium price and quantity are established. An equilibrium price is one for which the quantity demanded is precisely equal to the quantity supplied and the market is cleared.

The equilibrium price serves two important functions. First, this price provides information for firms with which to decide how much should be produced. The second function of this price is to ration demand.

There are many reasons why either a supply or demand schedule might shift. Such shifts may be caused by changes in prices in other markets, in tastes and preferences, or in productive technologies. When either a supply or demand schedule does shift, equilibrium price and quantity will change. The relative magnitudes of such changes depends upon the shape of both the supply and demand schedules. The rise in price that follows any given increase in demand will be the greater the less is the price elasticity of supply, and it will be the less the greater is the price elasticity of supply. Conversely, for any given change in supply, the ensuing change in price will be the greater the less is the price elasticity of demand, and it will be the less the greater the price elasticity of demand.

In the next section the conceptual considerations needed for price determination are presented.

2.3.2 Conceptual Considerations

A common approach to estimating the current period price is through the ratio of ending stocks to total demand. Where current supply is a function of lagged price, current demand is a function of current price, current ending stocks are the identity current supply less current demand and lagged ending stocks. This procedure is commonly referred to as a "disequilibrium" model stemming from the failure of the price equation to necessarily equate supply and demand.

The FAPSIM model notes that this failure results from the estimating of the above price equation as a structural, rather than a reduced form, equation.[24] The FAPSIM model uses an alternative framework where the market price equation is determined by substituting the supply, demand, and ending stock functions into the supply/demand identity and solving for price. Comparison of the two price equations indicates that the original formulation will not necessarily provide a market clearing price consistent with specified demand and supply schedules.

Ferris hypothesizes that in any given year, world prices will affect U.S. crop exports, seeding rates, and

[24] FAPSIM, op. cit., p. 14.

feeding rates.[25] These changes in world price in turn modify total domestic utilization and ending stocks. In essence, the world price is established by determining a margin over the U.S. loan rate. If the farm price of corn drops below the loan rate in this computation, the farm price is set at a level reflecting the influence of the loan, having the effect of putting a floor under the market. The loan rate is set exogenously in response to domestic pressure, and because the U.S. is the dominant world exporter, this same support price will also act as a floor on the world price by soaking up and storing excess supply before it goes into export channels.

These price determination schemes can be used as a starting point for market clearing (equilibrium) within the balance sheet.

[25] J. Ferris, "Feed Grain, Wheat, and Oilseed Supply and Demand," MSU Staff Paper, October 1985.

CHAPTER III

SIMULATION MODEL CONSTRUCTION

Chapter three examines how economic and statistical theory can be used to guide in the construction of estimable equations capable of forecasting the farm price of corn in the U.S. The data calculations, equations used, and empirical results are presented.

3.1 THE SIMULATION MODEL

Construction of a simulation model begins with the specification of a set of individual supply and demand relationships, which explain the behavior of agricultural product prices. Applied econometric techniques are used to estimate specific economic coefficients (parameters) and provide forecasts of prices or the variables affecting prices. These quantitative techniques serve the purpose of making the supply and demand relationships among variables explicit.

The simulation model should be consistent with the logic and theory underlying the commodity being analyzed. According to Tomek and Robinson, model building may be viewed as having two parts.[1] One involves the specification of the economic model, that is, the general economic relationships. Economic theory suggests the general types of functions that may be appropriate for a particular research problem. The second part of model building involves the explicit definition of structural equations which are to be estimated. These structural issues are presented with each equation.

Following the development of the conceptual framework, observations which reflect the hypothesized variables in the model must be collected. In practice, the specification of the equations to be estimated is likely to be influenced by the data available. Finally, coefficients which relate the variables to one another are estimated, and evaluated according to statistical procedures.

3.2 GENERAL ECONOMETRIC PROCEDURES

Econometrics is the science of abstracting or modeling the real world through the use of (typically) quantitative methods.[2] By concentrating on the types of models that can be expressed in functional form, data can be used to estimate the parameters of the equations and theoretical relationships can be tested statistically.

[1] W.G. Tomek, and K.L. Robinson, Agricultural Product Prices, Cornell University Press, Ithaca, NY, 1972, p. 303.

[2] Robert S. Pindyck, and Daniel L. Rubinfeld, Econometric Models and Economic Forecasts, McGraw-Hill, New York, 1981.

Conventional economic theory, regardless of form, postulates exact functional relationships between variables. Any introductory examination of economic data, however, indicates that points do not lie exactly on straight lines or other smooth functions. Thus a stochastic term must be introduced into economic relationships.

The stochastic term may take on positive or negative values and helps explain reality by accounting for excluded variables in the explicit function, unpredictable randomness in human responses, and errors in measurement of relevant variables. The important assumptions about the stochastic term concern its probability distribution. The simplest assumption is to assume that the term has an expected value of zero, zero correlation among different observations, and a finite variance.

The objective is a hypothesis test that the relationship among certain variables can be predicted or explained to a large extent by other variables. Many rules can be created to define the relationships among variables, however, the least squares criterion is the procedure said to give the line of best fit. The line of best fit is defined by those parameters which minimize the sum of the squared deviations of the observations and the estimated relationship. Least squares estimated parameters have the properties of being linear functions of the actual observations, and being unbiased linear estimates of the actual parameter.

The procedure of choosing the line of best fit is often referred to as ordinary least squares (OLS). OLS estimates of parameters have the minimum variance and conform to the Gauss-Markov theorem, which states that given the assumptions about the stochastic term, the parameter estimates are the best (most efficient) linear unbiased estimators of the actual parameters. OLS estimates of the parameters are appropriate due to the recursive nature of the model and the iterative process used to solve the price equation.

Using OLS estimating procedures provides information about the distribution of the parameters, and makes possible the construction of confidence intervals and testing of hypotheses concerning those regression parameters.

3.3 STATISTICAL CONCEPTS AND RANDOM VARIABLES

The most important objective of a statistical analysis is to draw inferences or generalities about a population from the partial information embodied within sample data.[3] The statistical concepts needed to do analysis and make inferences about a particular population are the sample mean and sample standard deviation.

The sample mean is a measure of central tendency and represents the center of a data set. The mean is generated as:

[3] Gouri K. Bhattacharyya, and Richard A. Johnson, Statistical Concepts and Methods, Whiley & Sons, New York, 1977.

MEAN = (sum of all observations / number of observations).

The sample standard deviation measures the extent of variation around the center and serves as a basic measure of variability. The sample standard deviation gives a measure of variability in the same unit as the data and is calculated as the square root of the sample variance. The sample variance is generated as:

VAR = (sum of (each observation less the MEAN)^2) / (number of observations less one).

Using these calculations, some inference can be made about the population of yields and prices from the sample of yields and prices generated within the model.

The central limit theorem states that when a sample of observations is large, the distribution of those observations is approximately normal, regardless of the shape of the population distribution. The yield of each of the world regional areas is based on (among other factors) annual weather patterns. If a single annual weather pattern constitutes an observation from the population of all annual weather patterns, it can be assumed that the distribution of annual weather patterns is approximately normal and each of the world's regional yields is approximately normal.

Because weather plays such a central role, the actual yield does not always take on the point forecast estimate, and is actually a continuous random variable. Continuous random variables have the characteristics of being able to assume all values in an interval, have a probability distribution, a mean, a variance and a standard deviation.

The assumption of a normal distribution for yields allows the use of a standardized random variable for generating random yields. The standardized random variable has a mean of zero and a standard deviation of one. It is customary to denote the standardized random variable by Z.

Knowing the point estimate and the standard deviation for the yields will allow the world supply to be randomly shocked. To calculate the random shocks, a computer generated pseudorandom number can be used. The pseudorandom number has the property of being a number between zero and one, inclusive. The pseudorandom number with the prescribed property will have a population mean of 0.5 and a standard deviation of the square root of 1/12. By sampling the pseudorandom number generator a distribution of values (between 0 and 1, inclusive) is obtained, and a sample mean can be calculated from this distribution.

To obtain the standard random variable Z, the following equation may be applied:

Z = (sample mean - population mean) / (standard error / square root of the number of observations).

Once the Z value is known, the point estimate and the standard deviation of the estimated equation are used to determine the random shock. The equation follows:

Shock = standard deviation * Z + point estimate.

By assuming yield is a random variable, it holds that supply will also be a random variable and thus, price will be a random variable. Using the random shocks the model will determine a price based on those random supplies and provide a distribution for price. Once the price distribution has been generated, statistical confidence intervals can be placed on the range of prices. These confidence intervals on the price distribution will be the basis of the risk aversion/decision analysis.

3.4 DATA SOURCE CONSIDERATIONS

The type of data used in this research is time series. The sources of each data series are listed in Appendix 1. As noted in the general econometric procedures, the consistency of the data is important in the functional form of the equation and in the level of bias of the estimation parameters. An attempt was made to obtain observations for each series from a common and original source.

Another aspect of data sources is the particular time series selected for estimation purposes. Parameter estimates generated from different sample periods may be substantially different because the structural relationship between the dependent and explanatory variables is subject to change over time. Given this structure, the sample period for each equation will be chosen independently.

The passage of The Food Security Act of 1985 mandates a change in the marketing year for corn from October 1 through September 30 to September 1 through August 31. This change, as mandated, is due to producers harvesting earlier in the year and new crop corn being used before October 1. The change in the feed grain data base has led to changes in the U.S. supply, disappearance and stocks estimates. The challenge involved in adjusting to the new data surveys lay in estimating past years' supply and disappearance balance for the newly defined periods so that a consistent historical data base could be developed. Because the marketing year in 1985 began on October 1, this research will not use data under the new market year surveys and most of the equations will be estimated through 1985.

3.5 APPRAISING THE RESULTS

The interpretation of the estimated parameters will determine the usefulness of a particular result within the theoretical and conceptual framework developed.

The coefficients of each explanatory variable are estimates of the net relationship between the respective variable and the dependent variable. These estimates are interpreted as the change in the dependent variable given a one unit change in the respective explanatory variable, ceteris paribus. For an estimated relationship to have a high degree of explanatory power, the variance of the regression and the variance of the coefficients should be as small as possible. Small standard errors gives more confidence that the true, but unknown, parameter is within a relatively small range of the estimate.

The corrected coefficient of multiple determination is a measure of the degree of the variation in the dependent variable associated with the variation in the explanatory variables. This coefficient, also known as adjusted R-squared can be used to select among alternative model specifications.

The estimated parameters must stand the test of logic according to economic theory. If the sign of an estimated coefficient is not consistent with the logic of the economic theory, then the estimated equation is suspected of being misspecified. Further, the estimated parameters must stand the statistical evaluation on the Students t test (henceforth called the t statistic). The t statistic provides measures of the probability of obtaining an parameter coefficient when the true parameter is indeed zero. The t statistic of the estimated parameter is presented in the tables parenthetically below the respective estimate.

Given the necessary assumptions about the stochastic term for OLS, we know that the term is to exhibit a constant variance over the range of observations. Serial correlation is the lagged correlation of a series with itself. Such behavior is usually a sign of specification error in the estimated relationship which can be caused by a variety of factors. The Durbin Watson test statistic (DW) is a common test for the existence of first order serial correlation. Any nonrandom behavior of the error term is a sign of misspecification and should be investigated.

When one or more lagged endogenous variables are present in an estimated equation, the DW statistic is no longer useful. The test statistic to be used in this case is the Durbin h statistic. The Durbin h statistic is approximately normally distributed with unit variance and the test can be done directly by using the normal distribution table.

3.6 U.S. CORN SUPPLY MODEL

Each structural equation used in the model represents an attempt to explain how the endogenous variables change from year to year. The estimation of acres planted contains the behavioral relationships for corn supply and therefore, must contain the relevant economic variables to explain this equation. To facilitate ease of updating and modification, the equation used to forecast this response should be relatively simple.

3.6.1 Results For Acres Planted

The structure used to estimate the acres planted in this project used a variation of the Ferris model discussed in Chapter 2. The form of the equation is:

APCT = f (REVCOR(-1), REVWHT(-1), PARTREV(-1), DVCRN)

where: APCT = Acres planted to corn in million acres in the current year,

REVCOR(-1) = Real net corn revenue per acre, lagged one year,

REVWHT(-1) = Real net wheat revenue per acre, lagged one year,

PARTREV(-1) = Real net corn revenue for participating in the government program, lagged one year,

DVCRN = Percent of acreage devoted to conserving uses.

Revenue received from corn and competing crops provide expectations for producers to base decisions upon. Since policy and expected revenue affect the decision of acreage committed to corn, there must be some variable or proxy of these included in the equation. The signs on the variables for the acres planted equation are expected to be positive on the corn revenue and negative on the substitute's revenue, the participant's revenue and the conserving acres.

The revenue per base acre for a participant in the government program is calculated from the gross from the market given the acreage limitations imposed, revenue from diversion payments and revenue from any deficiency payment. With this summation the variable costs are subtracted from the gross revenue to yield net revenue.

The non-government program participant revenue and competitor crop revenue received in this estimation is the real farm price multiplied by the yield per acre less the variable costs. These revenues do not include any income from government programs nor is it restricted by acreage limitations and therefore, is strictly a net return per acre. Each revenue is deflated by the consumer price index.

Equation 1 of Table 2 shows the estimation of acres planted using net corn revenue for non participants, net wheat revenue, the participant's net revenue and the percentage of conservation acres. Dummy variables were used on two of the observations. The Food and Agriculture Act of 1973 caused an anomaly and the Russian grain embargo of 1979 caused the other. This estimate shows the net revenue for non participants to be negative which immediately dismissed the equation from consideration.

The second attempt (equation 2 of Table 2) used net soybean revenue as the competing crop. Again the negative sign on the net corn revenue for non participants dismissed the equation from consideration.

Several attempts (not shown in Table 2) were made at estimating acres planted using various combinations of revenues and dummy variables with unsatisfactory results.

By assuming that the revenue for the participants is the most important variable and that most producers will be participants in the government program the net corn revenue for non participants was eliminated from the equation. Equation 3 of Table 2 shows this formulation. In this estimation, both dummy variables and net wheat revenue are

TABLE 2: Ordinary Least Squares Estimates for U.S. ACRES PLANTED TO CORN in million acres

Equation	Time Period	Independent Variables								$\bar{R}^2$	S. E.	DW
		Constant	DV 73	DV 79	RCNREV(-1)	RWHREV(-1)	RSBREV(-1)	RTOREV(-1)	DVCRN			
1	1961-85	91.349 (54.91)	-5.522 (-2.44)	4.331 (2.21)	-0.048 (0.86)	-0.103 (-1.72)	N/A	0.004 (0.08)	-60.432 (-19.55)	.948	1.865	2.44
2	1961-85	92.496 (36.74)	-4.219 (-1.83)	5.338 (2.54)	-0.0698 (-1.16)	N/A	-0.0317 (-0.71)	-0.0211 (-0.38)	-61.696 (-16.30)	.941	1.985	2.06
3	1961-85	90.993 (56.87)	-6.539 (-3.41)	4.044 (2.11)	N/A	-0.126 (-2.39)	N/A	-0.028 (-0.79)	-60.333 (-19.67)	.948	1.852	2.47

where: DV 73 = A dummy variable to account for the 1973 Food and Agriculture Act
1 for 1973; otherwise 0

DV 79 = A dummy variable to account for the Russian grain embargo
1 for 1979; otherwise 0

RCNREV(-1) = Real net corn revenue for non participants, lagged one year

RWHREV(-1) = Real net wheat revenue, lagged one year

RSBREV(-1) = Real net soybean revenue, lagged one year

RTOREV(-1) = Real net revenue from participation in the government program
Deficiency payments + Diversion payments + Return from market

DVRCN = Acreage devoted to conserving uses, in percentage terms used. The t statistic on wheat price tested significant at the 5% level. The revenue generated when participating in the government program tested low at the 5% level but due to its importance, was left in the equation. The explanatory power of equation 3 was high with adjusted R-squared's of 0.95 and DW statistic's of 2.47, which show inconclusive results to signs of serial correlation.

3.6.2 Results For Acres Harvested

The Agricultural Act of 1961 which called for specific acreage diversion programs for corn growers appears in Figure 1. This legislation can be viewed as a structural change and makes looking at two estimation periods, 1951 to 1986 and 1961 to 1986, interesting.

The structure used to estimate acres harvested is:

$$AHCT = f\ (\ APCT,\ TIME\ )$$

where: AHCT = Acres harvested to corn in million acres in the current year,

APCT = Acres planted to corn in million acres in the current year,

TIME = Time trend.

Each estimate of acres harvested using the longer time period was lacking in explanatory power because of the structural change. This made the 1961 to 1986 time period more desirable.

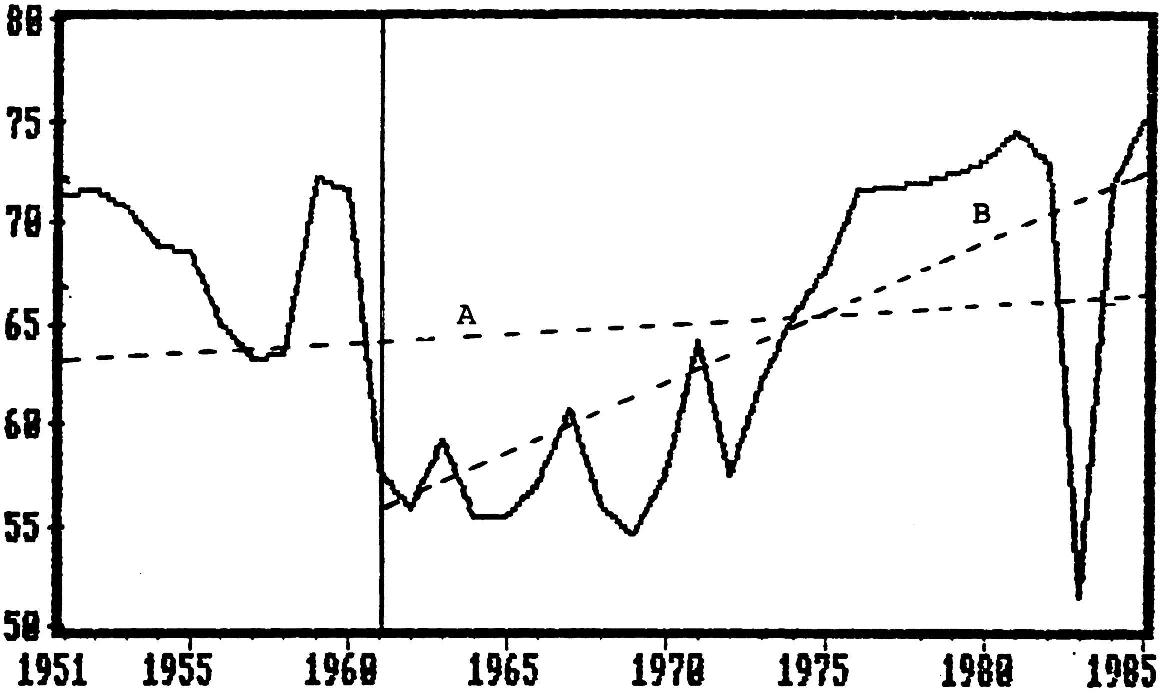


Figure 1: Acres Harvested To Corn In Million Acres

The vertical line indicates the year (1961) which called for specific acreage diversion programs. The dashed lines represent the trend forecasts from (A) 1951 to 1985 and (B) 1961 to 1985. These dashed lines show a change in slope and thus a structural change in acres harvested.

By using the assumption that only the acres planted is related to the acres that can be harvested, equation 1 of Table 3 shows positive serial correlation. The positive serial correlation indicates a misspecified equation.

If the assumption is that technology also contributes to the amount of harvested acres, then time can serve as a proxy. It is assumed that time would have a positive relationship in this equation to represent increases in producers ability to incorporate increased cultural practices.

Equation 2 of the same table shows this relationship. An adjusted R-squared of 0.98 and a DW of 1.22 show a greater explanatory power and decreased evidence of positive serial correlation and make this the equation of choice.

3.6.3 Results For Yield Per Acre

An estimate was made of the actual yield per acre using one of the simplest approaches in forecasting. By assuming that events from the past will continue in the future, a trend estimate may be used. Unfortunately, forecasts based on trends miss all turning points for the variable, however since yield estimates are based on random patterns, this need not be a factor. The functional form follows:

$$\text{CORNYT} = f\ (\ \text{TIME}\ )$$

where : CORNYT = Corn yield in bushels per acre in the current year

TIME = Time trend.

TABLE 3: Ordinary Least Squares Estimates for U.S. Acres Harvested in million acres

Equation	Time Period	Independent Variables			$\bar{R}^2$	S. E.	DW
		Constant	APCT	TIME			
1	1961-86	-5.645 (-2.36)	0.9429 (29.22)	N/A	.971	1.298	0.788
2	1961-86	-8.456 (-3.52)	0.8722 (22.08)	0.1092 (2.63)	.977	1.163	1.22

where: APCT = Acres planted to corn in the current year

TIME = Time trend, 1961 = 61

Here time serves as a proxy to productivity for the period of 1950 to 1985 (See Table 4). A long time period in a trend estimate helps to reduce the standard error and increase degrees of freedom. The estimate shows that an increase in yield of 2.15 bu/acre per year can be expected.

3.6.4 Production and Supply

Production and supply are both identities. Production estimates are generated by using the estimate of acres harvested multiplied by the estimate of the yield per acre. It takes the following form:

$$PRODT = AHCT * CORNYT$$

where: PRODT = Corn production of the current year,

AHCT = Acres harvested in the current year,

CORNYT = Corn yield in the current year.

The supply is the beginning stocks (ending stocks from the previous year) plus the current production plus any imports. For this model, the U.S. imports will be considered predetermined. The supply identity has the following form:

$$TSPLYT = PRODT + EST(-1) + IPORTT$$

where: TSPLYT = Total supply in the current year,

PRODT = Production in the current year,

EST(-1) = Ending stocks lagged one year.

IPORTT = Domestic imports in the current year.

TABLE 4: Ordinary Least Squares Estimate for U.S. Yield in bushesls/acre

Equation	Time Period	Independent Variable		$\bar{R}^2$	S. E.	DW
		Constant	TIME			
1	1950-85	-71.376 (-8.50)	2.15 (17.51)	0.897	7.661	2.03

where: TIME = Time trend, 1950 = 50

3.7 U.S. CORN DEMAND MODEL

The equation used for the demand component is the identity:

$$TDMDT = FEEDT + FSIT + XPORTT + STOCKST$$

Total use (TDMDT) is the sum of corn used for livestock consumption (FEEDT), food, seed, and industrial use (FSIT), exports (XPORTT) and ending stocks (STOCKST).

In turn each of these demand components are estimated functions or identities composed of other variables.

3.7.1 Results For Corn Consumed By Livestock

The livestock feeding rate per head was conceptually developed to be; that amount of corn which a producer will feed in any time period based upon the value of output from feeding, the cost of the feed, the cost of a substitute feed, and any information regarding technological changes. The functional form of the feed use equation is as follows:

CONSUMPTION = FEEDING RATE/HEAD * ANIMAL
NUMBERS
where: FEEDING RATE = f (OUTPUT VALUE, OWN PRICE,
SUBSTITUTE PRICE, TIME)
ANIMAL NUMBERS = Inventory values of the animal
species.

By separating each animal species and estimating individual rates of feeding and multiplying the feeding rate per head by animal inventory numbers, each species can be summed to give the total combined corn use.

The corn price is expected to be inversely related to the rate of feeding while the substitute price is expected to be positively related, which follows from the marginal rate of substitution discussed earlier. The sign of the output price is expected to be positive since the short run response of producers to increases in product price is to feed animals out more heavily. The technology variable is expected to have a positive sign, reflecting the increase over time in the ability of an individual animal to utilize more feed.

Price series data for this component of demand are collected with different months (other than the corn marketing year) establishing the market year. In order to make meaningful aggregations of prices and quantities, all time t periods cannot be aggregated together. Because of this, it is necessary to "overlap" price and inventory data series where possible. This overlapping means to include the time period of price and inventory which falls within the consumption year.

The time map in Table 5 shows the relationships between the marketing year and the various output price and inventories. By referring to the time map, the association between the corn marketing year and relevant price and inventory observations can be made.

TABLE 5: TIME RELATIONSHIP MAP

```
        Calendar year t                    t+1
 [------+------+------+------][------+------+------+------]
JAN    APR    JUL    OCT     JAN    APR    JUL    OCT

          t-1             corn marketing year t
          t-1                 consumption t
 -------+------+-----][-------+------+------+-----][------
JAN    APR    JUL    OCT     JAN    APR    JUL    OCT

                      dairy animal number
                        beef cow number
                   cattle and calves on feed
                              *
                            JAN 1

        hen, pullet, and other chicken number
                 hog and pig number
                          *
                        DEC 1


                                   broiler production
                                   turkey production
                              [------+------+------+------]
                             JAN    APR    JUL    OCT


        milk price t                       t+1
      milk cow price t                     t+1
   steer & heifer price t                  t+1
      beef cow price t                     t+1
 [------+------+------+------][------+------+------+------]
JAN    APR    JUL    OCT     JAN    APR    JUL    OCT


      egg price t                      t+1
    broiler price t                    t+1
    turkey price t                     t+1
      hog price t                      t+1
-+------+------+------+--][---+------+------+------+--][--
JAN    APR    JUL    OCT     JAN    APR    JUL    OCT
```

For example, in aggregating the hog consumption per head, the corn consumed by hogs in corn marketing year t is divided by the hog and pig number in calendar year t. Dairy animal consumption per head used corn consumed by dairy animals in corn marketing year t divided by the dairy animal number in calendar year t+1.

Likewise, the corn to output price ratios used the output price in calendar year t+1 divided by the corn price in marketing year t. This shows an overlapping because the corn marketing year contained between 9 and 11 months of output price year t+1. Without using the values that fall in the corn price marketing year and simply using all time t periods, the aggregated variable would be biased due to insufficient or misinformation.

a) Dairy animal consumption per head

The dependent variable dairy animal consumption per head used dairy animal numbers for milk cows and heifers that have calved as of January 1.

The estimates of dairy animal consumption used time to proxy technological changes, real farm price, the real milk price, and the real wheat price. Several attempts were tried using various combinations of variables and time periods. The most favorable equation used the longer time period which was 1970-84 and had signs that were all correct (see equation 2 in Table 6). The adjusted R-squared is high and shows a good fit for the estimate, the standard error is

TABLE 6: Ordinary Least Squares Estimates for DAIRY ANIMAL CORN CONSUMPTION PER HEAD

		Independent Variables							
Equation	Time Period	Constant	TIME	OUTPUT	RFARMP	RWHTPT	$\bar{R}^2$	S. E.	DW
1	1971-84	-87.328 (-2.79)	1.840 (6.30)	3.444 (1.05)	-25.795 (-2.47)	10.890 (1.73)	.863	3.67	1.86
2	1970-84	-86.515 (-3.19)	1.832 (7.37)	3.415 (1.11)	-25.609 (-2.69)	10.746 (1.93)	.890	3.49	1.96

where: TIME = Time trend, 1970 = 70

OUTPUT = Real annual average all milk price, $/CWT

RFARMP = Real farm price of corn $/BU

RWHTPT = Real farm price of wheat $/BU

the lowest among the attempts and the DW statistic shows no sign of serial correlation.

b) Cattle on feed consumption per head

The dependent variable cattle on feed consumption per head used cattle and calves on feed as of January 1.

The estimates of cattle on feed used time, the real farm price, the real soybean meal price, the real steer and heifer price, and a lagged dependent variable. Again several attempts were tried using these variables. The estimate which showed the most favorable results used the time period from 1971-84 and stood the test of logic (equation 2, Table 7). The sign on the time variable is negative which could indicate that a substitute for corn as a feed has been incorporated into the cattle on feed ration over time. The signs of the other variables are as expected.

In equation 2, the standard error was the smallest among all attempts and the adjusted R-squared was the highest at 0.61. The equation used a lagged dependent variable so the DW statistic was not applicable. The Durbin h statistic of 1.31 shows that at the 5 percent level (the critical value of the normal distribution being 1.645), the null hypothesis of no serial correlation cannot be rejected (i.e. not significantly different from zero).

TABLE 7: Ordinary Least Squares Estimates for CATTLE ON FEED CORN CONSUMPTION PER HEAD

	Time	Independent Variables								
Equation	Period	Constant	TIME	RATIO	RFARMP	RSMP	LAGGED	$\bar{R}^2$	S. E.	DW
1	1971-84	63.431 (4.98)	N/A	N/A	-13.590 (-1.41)	0.199 (2.06)	N/A	.175	11.86	2.60
2	1971-84	240.119 (4.47)	-2.126 (-3.49)	1.665 (3.82)	N/A	N/A	-0.6594 (-2.95)	.607	8.18	N/A

where: TIME = Time trend, 1971 = 71

RATIO = Real steer and heifer price ($/CWT) :
real farm price of corn ($/BU) ratio

RFARMP = Real farm price of corn, $/BU

RSMP = Real soybean meal price, $/TON

LAGGED = Cattle on feed corn consumption per head, lagged one year

c) Other beef cattle consumption per head

The dependent variable other beef cattle consumption per head used beef cows and beef cow replacements on January first.

The estimates of other beef cattle used the beef cow price, the real farm price, real soybean and wheat prices as substitutes. The attempt which showed the best results used the 1971-84 time period, a lagged substitute price and also stood the test of logic (equation 2, Table 8).

In this estimation, the adjusted R-squared was highest (at 0.83) among all attempts while the standard error was approximately the same for all estimates. The estimate did not have enough observations to show a critical value on the DW statistic table.

d) Hen, pullet, and other chicken consumption per head

The dependent variable HPC consumption per head used inventories on December 1.

The estimates for hen, pullet and other chicken consumption used the real egg price, the real farm price, the real soybean meal price, real wheat price and a lagged dependent variable. The time period of 1971-84 showed the best results with logical signs on all variables (equation 2, Table 9). This equation had a smaller standard error and a higher adjusted R-squared which all indicate a better fit. Again because of the lagged dependent variable, the Durbin h

TABLE 8: Ordinary Least Squares Estimates for
OTHER BEEF CATTLE CORN CONSUMPTION PER HEAD

Equation	Time Period	Independent Variables			$\bar{R}^2$	S. E.	DW
		Constant	RATIO	RSMP			
1	1971-84	3.225 (10.60)	0.114 (5.55)	0.0014 (0.57)	.752	0.310	2.41
2	1971-84	2.875 (9.73)	0.115 (7.27)	0.0049 (2.27) *	.826	0.260	2.30

where: RATIO = Beef cow for slaughter & herd replacement, and cull cow sold for slaughter ($/CWT) : real farm price of corn ($/BU) ratio

RSMP = Real soybean meal price, $/TON

* Real soybean meal price, $/Ton, lagged one year

TABLE 9: Ordinary Least Squares Estimates for HEN, PULLET AND OTHER CHICKEN CORN CONSUMPTION PER HEAD

Equation	Time Period	Independent Variables						$\bar{R}^2$	S. E.	DW
		Constant	OUTPUT	RFARMP	RSMP	RWHTPT	LAGGED			
1	1971-84	-0.8065 (-3.24)	0.0042 (0.37)	-0.2248 (-1.24)	0.0006 (0.56)	N/A	0.2398 (1.68)	.095	0.125	N/A
2	1971-84	0.9349 (5.97)	N/A	-0.7245 (-2.16)	0.0012 (1.25)	0.3409 (1.72)	0.2219 (1.97)	.308	0.109	N/A

where: OUTPUT = Real egg price (includes hatching eggs), CENTS/DOZ

RFARMP = Real farm price of corn, $/BU

RSMP = Real soybean meal price, $/TON

RWHTPT = Real farm price of wheat, $/BU

LAGGED = Hen, pullet and >ther chicken consumption per head, lagged one year test statistic must be used. The Durbin h showed that the null hypothesis of no serial correlation cannot be rejected.

e) Broiler consumption per head

The dependent variable broiler consumption per head used the number of birds produced from December 1 to November 30.

The variables used in the broiler estimation were time, the real farm price of corn and the real wheat price. The time period chosen was 1971-84 and it showed fair results with a small number of observations. The signs of all independent variables were correct for the test of logic (equation 1, Table 10). The adjusted R-squared showed a good fit of the data to the dependent variable while the Durbin Watson indicated no signs of serial correlation.

f) Turkey consumption per head

The dependent variable turkey consumption per head used annual production counts from December 1 to November 30.

The estimate uses both the real farm price and a lagged dependent variable, and the real wheat price. With an estimation period of 1971 to 1984, each variable showed significant t statistics (equation 1, Table 11). The explanatory power of the equation is only fair in being able to attribute less than 64% of the variability of the turkey consumption to the variability of the independent variables. The DW is not applicable in this case due to the lagged dependent variable and the Durbin h is inclusive.

TABLE 10: Ordinary Least Squares Estimate for BROILER CORN CONSUMPTION PER HEAD

Equation	Time Period	Independent Variables				$\bar{R}^2$	S. E.	DW
		Constant	TIME	RFARMP	RWHTPT			
1	1971-84	0.0025 (0.06)	0.0014 (3.16)	-0.0431 (-2.63)	0.0162 (1.65)	.785	0.006	2.01

where: TIME = Time trend, 1971 = 71

RFARMP = Real farm price of corn, $/BU

RWHTPT = Real farm price of wheat, $/BU

TABLE 11: Ordinary Least Squares Estimate for TURKEY CORN CONSUMPTION PER HEAD

Equation	Time Period	Independent Variables				$\bar{R}^2$	S. E.	DW
		Constant	RFARMP	RWHTPT	LAGGED			
1	1971-84	0.7086 (12.82)	-0.3428 (-3.07)	0.1260 (1.83)	0.1039 (1.42)	.639	0.041	N/A

where: RFARMP = Real farm price of corn, $/BU

RWHTPT = Real farm price of wheat, $/BU

LAGGED = Turkey corn consumption per head, lagged one year

g) Hog consumption per head

The dependent variable hog consumption per head used the hog and pig inventory on December 1.

Hog consumption was estimated using time, real hog price, and real farm price from 1970-84. The estimate with the best statistics among those attempted used the 1971-84 time period and had the highest adjusted R-squared and the lowest standard error (equation 3, Table 12) of all estimates. The signs were all logical but the DW statistic showed inclusive results while a pattern existed in the residual plot.

h) Other and unallocated consumption

To estimate the consumption of each species on a per head basis, the animal inventories must be known. The only data obtainable for other and unallocated was sheep and lambs on feed as of January 1. Since other species of animals consume corn during the year (i.e. horses, mules, etc.), the other and unallocated consumption cannot be estimated on a feeding rate per head basis.

Equation 1 in Table 13 shows the estimate of other and unallocated consumption. The independent variables of real corn farm price, real wheat price, and time show good statistics when dummy variables are used for the years 1982 and 1984. The dummy variables were necessary because an unusually large amount of corn was placed in this category during those years. The real farm price was negative while all other variables were positive which is expected from the

TABLE 12: Ordinary Least Squares Estimates for HOG CORN CONSUMPTION PER HEAD

Equation	Time Period	Independent Variables						$\bar{R}^2$	S. E.	DW
		Constant	TIME	OUTPUT	RFARMP	RSMP	RWHTPT			
1	1970-84	-52.755 (-1.69)	0.970 (2.82)	0.365 (0.90)	-20.511 (-1.59)	N/A	10.650 (1.39)	.503	4.71	1.67
2	1970-84	-95.776 (-3.34)	1.395 (4.30)	0.361 (0.96)	-4.157 (-0.83)	0.075 (1.87)	N/A	.559	4.43	2.44
3	1971-84	-14.624 (-1.81)	0.499 (5.43)	0.394 (4.05)	-7.612 (-5.70)	N/A	N/A	.879	1.16	1.44

where: TIME = Time trend, 1970 = 70

OUTPUT = Real average price of hogs received by farmers, $/CWT

RFARMP = Real farm price of corn, $/BU

RSMP = Real soybean meal price, $/TON

RWHTPT = Real farm price of wheat, $/BU

TABLE 13: Ordinary Least Squares Estimate for OTHER AND UNALLOCATED CORN CONSUMPTION

Equation	Time Period	Independent Variables						$\bar{R}^2$	S. E.	DW
		Constant	TIME	RFARMP	RWHTPT	DV82	DV84			
1	1971-84	-412.497 (-4.01)	6.839 (5.54)	-63.265 (-1.53)	26.481 (1.08)	326.601 (19.91)	189.265 (10.89)	.986	14.23	2.51

where: TIME = Time trend, 1971 = 71

RFARMP = Real farm price of corn, $/BU

RWHTPT = Real farm price of wheat, $/BU

DV82 and DV84 = Dummy variables to account for anomalies in the USDA distribution of corn disappearance
1 for 1982 and 1984; otherwise 0

theoretical framework. The t statistic on the price coefficients are suspect of insignificance yet not extremely low. The adjusted R-squared shows good explanatory power. The Durbin Watson showed indeterminate results and was slightly suspect of showing negative serial correlation among the error terms however no pattern was evident in the residual plot of the equation.

3.7.2 Food, Seed, And Industrial Use Of Corn

Given the evidence about FSI in Chapter 2, it would be difficult to forecast its future demand. It is known that the quantity consumed by FSI will level off, yet it is not known when or by how much. Therefore, this model will not attempt an estimate of FSI and will provide a predetermined quantity for purposes of total use.

3.7.3 Results For U.S. Corn Exports

a) Assumptions

Recalling the export formulation from Chapter 2, the assumption is that there are 3 regions. 1) The United States, 2) Residual Exporters, and 3) Importers (see Appendix 2 for a listing of the international regional groupings). Using the three regions, we assume that the U.S. will provide the balance of the world needs after the "dumpers" have exported their product.

An equation can be formulated which shows that the U.S. exports (XPORTT) are equal to importer net imports (INIPC) less exporter net exports (ENEPC). The form of this demand component follows:

$$XPORTT = INIPC - ENEPC.$$

b) Aggregations and Limitations

There is a trade-off involved when aggregating the consumption and production of the entire world into three regions. The trade-off involved is; the simplicity of the model for the purposes of an easy to use educational tool, versus the precision of a complex system of equations and linkages. A simple model looses much of; the robustness of the more complex model, explanatory power, and estimates are likely to be less reliable, however, the complex system would be more difficult to maintain and use.

To maintain some consistency in the numerous international data series, the data for gross domestic product and consumer price index for each region was indexed with 1980 equal to 100.

It was necessary to weigh the regional gross domestic product data by feed grain usage and aggregate across regional areas. This procedure is essential to represent the degree of importance feed grain has in each regional area of the region in relation to their level of wealth. An example of the weighting procedure follows:

Importer Gross Domestic Product (IGDP)=

SUM ((feed grain net imports of the past 5 years / world net imports of the past 5 years) * real gross domestic product).

The weight is an average of the past five years net imports into the regional area divided by the total world net imports. This product is multiplied by the real gross domestic product of the regional area. Once the weighted real gross domestic product has been calculated for each regional area, the values are summed across each regional area to obtain the real gross domestic product for the importing region.

The same type of aggregation was used for the residual exporter's gross domestic product. Again, a five year average of the net exports from the dumpers is divided by the total net exports and multiplied by the residual exporter's real gross domestic product. Finally, the value for gross domestic product is summed across the three countries to get the real gross domestic product for the exporting region.

Importer and Exporter population was aggregated as the sum of the population of each region to obtain an importer and exporter regional population.

SUM (pop i) and SUM (pop j)

Where: i is the importing regions and j is the exporting regions.

The prices used in each U.S. export equation reflect the f.o.b. U.S. Gulf Ports feed grain price perceived by the respective importing and exporting region. To maintain consistency with the prices of the other components of U.S. demand a linkage was needed to relate the U.S farm price of corn to the export price of feed grain. The linkage is provided by calculating a simple average of the difference between the Gulf price and the farm price.

The importer and exporter perceived price variable called for multiplying the U.S. real feed grain price by a regional normalized exchange rate. The aggregation is necessary because the varying in the exchange rate of the U.S. dollar to foreign currencies could cause the price that the particular region pays on the world market to vary even if the Gulf Port price stays constant. The price aggregation has the following form:

Feed grain and wheat price = the real f.o.b. U.S. Gulf price * normalized exchange rate for feed grain and wheat respectively.

The lagged revenue generated by the crop and alternative crops affect the amount of acres planted (and consequently supply) in the current period. This is due to producers having expectations of the profitability of the corp and being able to adjust production accordingly. The revenue variable generated summed the revenue of the respective crop for each country in the region and has the following form:

Revenue per acre = The lagged real crop price * the
lagged yield per acre for the crop
* the lagged normalized exchange
rate for the crop all divided by the
lagged local consumer price index.

c) Importers net imports per capita

Theoretical components for a demand function state that demand is a function of own price, substitute price and income. The importers net imports per capita equation takes the form:

$$INIPC = f\ (\ IFP,\ IWP,\ IGDPPC\ )$$

where: INIPC = Importers net imports per capita in thousand metric tons,

IFP = Importers feed grain price,

IWP = Importers wheat price,

IGDPPC = Importers gross domestic product per capita.

The theoretical framework for demand developed in Chapter 2 showed good results. The dependent variable importer net imports per capita used prices perceived by the importing region for corn and wheat, and per capita income. In equation 1 of Table 14, the signs were correct and all the coefficients were significant. The adjusted R-squared shows good explanatory power. The Durbin Watson statistic

TABLE 14: Ordinary Least Squares Estimates for IMPORTER NET IMPORTS PER CAPITA in thousand metric tons

Equation	Time Period	Independent Variables						$\bar{R}^2$	S. E.	DW
		Constant	IFP	IWP	IGDPPC	IPROPC	Rho			
1	1961-84	0.0028 (0.68)	-0.00013 (-3.24)	0.00009 (3.17)	6917650 (5.12)	N/A	N/A	.920	0.0015	1.40
2	1961-84	0.0072 (1.29)	-0.00011 (-2.46)	0.00008 (2.90)	8656992 (4.29)	-0.0847 (-1.15)	N/A	.921	0.0014	1.12
3	1962-84	0.0101 (1.31)	-0.00004 (-1.19)	0.00005 (1.92)	9209967 (3.05)	-0.1431 (-2.46)	0.7277 (3.46)	.937	0.0013	1.80

where: IFP = Importer feed grain price

IWP = Importer wheat price

IGDPPC = Importer gross domestic product per capita

IPROPC = Importer production per capita

Rho = The first-order serial correlation coefficient of 1.4 showed indeterminate results. There was however, a pattern in the residual plot and the most recent observation having one of largest residuals which is undesirable.

It seems rational to assume that the importers own supply would have an impact on the amount that they import in order to fulfill their demand, therefore an indication of their production can serve as a proxy. The sign of the production variable is expected to be negative because the more corn produced in the region, the less imports would be needed. When importer production is added to the conceptual formulation (equation 2 of Table 14), the standard error is large and the coefficient was suspected of insignificance. The adjusted R-squared of 0.92 showed a slight increase in explanatory power and the Durbin Watson statistic which, still tested indeterminate, was lower. Again there was a pattern to the residual plot and the last observation had the largest residual.

The final estimate tried (equation 3 of Table 14) used a correction for first-order serial correlation. This estimate took 7 iterations for rho to converge. the standard error was smaller and the adjusted R-squared was larger than the other estimates which make this the equation of choice.

d) Importers yield

The importers yield was estimated using a time trend. The estimate took the following form:

IYIELD = f (TIME)

where: IYIELD = Importers yield in metric tons per hectare,

TIME = Time trend.

This estimate is necessary in order to calculate the importing region's production and to determine the standard deviation around the yield for purposes of randomly shocking the world yield. The estimate (see equation 1 of Table 15) had great explanatory power at 0.97 but the DW statistic showed negative serial correlation. Equation 2 of Table 15 shows the correction for first-order serial correlation and was the equation of choice.

e) Exporters net exports per capita

Since the exporters will export everything not consumed or held for stocks, the exporters net exports per capita is an identity which takes the following form:

ENEPC = ESPLYPC - ECONPC - EESPC

where: ENEPC = Exporters net exports per capita in thousand metric tons,

ESPLYPC = Exporters supply per capita in thousand metric tons,

ECONPC = Exporters consumption per capita in thousand metric tons,

EESPC = Exporters ending stocks per capita in thousand metric tons.

TABLE 15: Ordinary Least Squares Estimates for
IMPORTER YIELD in metric tons/hectare

Equation	Time Period	Independent Variables			$\bar{R}^2$	S. E.	DW
		Constant	TIME	Rho			
1	1960-84	-0.8987 (-9.90)	0.0329 (26.21)	N/A	0.966	0.0452	2.73
2	1961-84	-0.8944 (-13.17)	0.0328 (35.06)	-0.3820 (-1.87)	0.966	0.0438	1.82

where: TIME = Time trend, 1960 = 60

Rho = The first-order serial correlation coefficient

i) Exporter Supply Per Capita

The Exporter supply aggregation was made by summing the beginning stocks (lagged ending stocks) and the production of the current period for each country in the region. The identity follows:

$$ESPLYPC = EPROPC + EESPC(-1)$$

where: ESPLYPC = Exporter supply per capita in thousand metric tons,

EPROPC = Exporter production per capita in thousand metric tons,

EESPC(-1) = Exporter ending stocks per capita in thousand metric tons, lagged one year.

ii) Exporters production per capita

By multiplying the estimated yield and acres harvested together then dividing by the population, the estimation of the exporter's production per capita can be obtained. The exporters production per capita has the form:

$$EPROPC = (EYIELD * EHA) / EPOP$$

where: EPROPC = Exporter production per head in thousand metric tons,

EYIELD = Exporter yield in metric tons per hectare,

EHA = Exporter harvested area in thousand hectares,

EPOP = Exporter population.

The exporters yield was a function of time to represent the technology changes made in the production practices. That equation takes the following form:

EYIELD = f (TIME)

where: EYIELD = Exporters yield in metric tons per hectare,

TIME = Time trend.

As expected the sign on the technology proxy was positive indicating increased cultural practices (as shown in equation 1 of Table 16). The adjusted R-squared and the Durbin Watson statistic were 0.89 and 2.03 respectively. This indicates a good fitting equation with no presence of serial correlation among the residuals.

To complete the production equation the exporters harvested area was estimated with the following form:

EHA = f (EFREV(-1), EWREV(-1), EHA(-1), EES(-1))

where: EHA = Exporters harvested area in thousand hectares,

EFREV(-1) = Exporters feed grain revenue lagged one year,

EWREV(-1) = Exporters wheat revenue lagged one year,

EHA(-1) = Exporters harvested area lagged one year,

EES(-1) = Exporters ending stocks lagged one year.

Equation 1 of Table 17 show the estimate of exporter area harvested. This equation used lagged feed grain revenue, lagged wheat revenue, lagged harvested acres, and lagged ending stocks. The signs of all variables were correct while the standard errors on the feed revenue and

TABLE 16: Ordinary Least Squares Estimate for
EXPORTER YIELD in metric tons/hectare

Equation	Time Period	Independent Variables		$\bar{R}^2$	S. E.	DW
		Constant	TIME			
1	1961-84	-1.260 (-5.18)	0.0460 (13.78)	.891	0.1132	2.03

where: TIME = Time trend, 1961 = 61

TABLE 17: Ordinary Least Squares Estimate for
EXPORTER HARVESTED AREA in thousand hectares

Equation	Time Period	Independent Variables					$\bar{R}^2$	S. E.	DW
		Constant	EFREV(-1)	EWREV(-1)	EHA(-1)	EES(-1)			
1	1965-84	8508.920 (3.53)	204.54 (1.28)	-343.17 (-2.15)	0.6984 (4.64)	-0.2998 (-1.72)	.680	901.216	N/A

where: EFREV(-1) = Exporter feed grain revenue, lagged one year

EWREV(-1) = Exporter wheat revenue, lagged one year

EHA(-1) = Exporter Harvested area, lagged one year

EES(-1) = Exporter ending stocks, lagged one year ending stock variables were higher than desired, however within a tolerable range. The Durbin Watson statistic cannot be used because of a lagged dependent variable in the equation. The Durbin h test statistic shows that the null hypothesis of no serial correlation cannot be rejected. The adjusted R-squared of 0.68 indicates fair explanatory power.

iii) Exporters consumption per capita

As indicated in the theoretical considerations, the consumption function is dependent on income and some proxy for prices. The exporters consumption per capita takes the form:

$$ECONPC = f\ (\ EGDPPC,\ EFP,\ EWP,\ ESPLYPC\ )$$

where: ECONPC = Exporters consumption per capita in thousand metric tons,
EGDPPC = Exporters gross domestic product per capita,
EFP = Exporters feed grain price,
EWP = Exporters wheat price,
ESPLYPC = Exporters supply per capita in thousand metric tons.

The estimates of exporter consumption per capita are shown in Table 18. Equation 1 has problems which is evidenced by the signs of the own and substitute prices. Theory says that the own price should have a negative sign while substitute price should be positive. This equation

TABLE 18: Ordinary Least Squares Estimates for EXPORTER CONSUMPTION PER CAPITA in thousand metric tons

		Independent Variables								
Equation	Time Period	Constant	EGDPPC	EFP	EWP	ESPYPC	DV740N	$\bar{R}^2$	S. E.	DW
1	1961-84	0.1436 (2.28)	1848507 (5.60)	0.0015 (1.62)	-0.0013 (-1.65)	N/A	N/A	.557	0.040	1.41
2	1961-84	0.1057 (2.07)	147468.8 (0.27)	0.0008 (1.10)	-0.0003 (-0.41)	0.3416 (3.57)	N/A	.721	0.0318	1.50
3	1961-84	0.0578 (2.13)	1222063 (3.87)	-0.0010 (-2.14)	0.00083 (2.17)	0.3140 (6.37)	-0.0793 (-7.34)	.926	0.0164	2.45

where: EGDPPC = Exporter gross domestic product per capita

EFP = Exporter feed grain price

EWP = Exporter wheat price

ESPYPC = Exporter supply per capita

DV740N = Spline variable to account for the European wheat subsidy
0 for 1961-73; 1 for 1974 on

has the opposite of what theory would predict. When an equation does not stand the test of logic, it must be dismissed from consideration.

It seems rational that the consumption of the exporting region is also based upon the amount of corn which is available for use. This consideration is shown in equation 2 and again the signs of the own and substitute prices are switched. The formulation must be eliminated but important information is gained. This equation shows the exporter supply is significant and helps to reduce the serial correlation among the residuals, the standard error is smaller and the adjusted R-squared is larger. This indicates that the supply variable helps explain the exporters consumption.

In 1974 the European sector started to subsidize wheat for feed grain. This policy change marked a structural change in the formulation of exporters consumption. By using a spline variable on the years 1974 on, the conceptual model worked better. The spline variable shows a negative sign because that policy caused less corn to be consumed at the effectively higher price. Equation 3 of Table 18 shows this formulation and becomes the equation of choice. It had an adjusted R-squared of 0.93 and a Durbin Watson statistic of 2.45 indicating inclusive evidence of serial correlation among the residuals.

iv) Exporters ending stocks per captia

The exporters ending stocks was estimated using the following form:

EESPC = f (EPROPC, ESUDR(-1))

where: EESPC = Exporter ending stocks per capita in thousand metric tons,

EPROPC = Exporter production per capita in thousand metric tons,

ESUDR(-1) = Exporter ending stocks per capita to exporter utilization per capita ratio, lagged one year.

By using the lagged ending stocks to utilization ratio and the current production to estimate the ending stocks, all signs were correct (see equation 1 Table 19). The adjusted R-squared was relatively low but the t statistics were significant. The DW statistic shows no signs of serial correlation.

In equation 1, the 1983 observation is a severe negative outlier. This is because of the U.S. 1983 PIK program which limited acreage planted and consequently supply. Due to the reduced U.S. supply, the exporters "dumped" more grain on the market and hence depleted their stock holdings. When the 1983 observation is dummied out (in equation 2 of Table 19), the explanatory power and the t statistics both improve. Again the DW statistic shows no serial correlation.

TABLE 19: Ordinary Least Squares Estimates for EXPORTER ENDING STOCKS PER CAPITA in thousand metric tons

Equation	Time Period	Independent Variables						$\bar{R}^2$	S. E.	DW
		Constant	EPROPC	ESUDR(-1)	DV83	EESPC(-1)	EUDPC(-1)			
1	1961-84	0.0201 (0.61)	0.0268 (2.23)	0.2280 (2.04)	N/A	N/A	N/A	.189	0.0218	1.58
2	1961-84	-0.0344 (-0.97)	0.0527 (3.69)	0.2963 (2.92)	-0.0712 (-2.68)	N/A	N/A	.374	0.0191	1.78
3	1961-84	0.0293 (1.23)	0.0728 (4.31)	N/A	-0.0932 (-3.54)	0.3516 (2.07)	-0.1338 (-3.66)	.485	0.0173	N/A

where: EPROPC = Exporter production per capita

ESUDR(-1) = Exporter ending stocks per capita:
exporter utilization per capita ratio, lagged one year

DV83 = Dummy variable accounting for the reduced U.S. supply due to the 1983 PIC program
1 for 1983; otherwise 0

EESPC(-1) = Exporter ending stocks per capita, lagged one year

EUDPC(-1) = Exporter utilization per capita, lagged one year

In equation 3 the ending stock to utilization ratio is disaggregated. By separating the variable, the equation becomes richer in the information provided yet it looses one degree of freedom, which can be sacrificed in this case. The signs of all the variables were correct and each variable was significant. The adjusted R-squared was 0.52 and the Durbin Watson statistic indicated a rejection of the hypothesis of positive serial correlation among the error terms.

An examination of the U.S. feed grain exports shows that approximately 86% of the feed grain exported is corn (see Table 20). Since the international component estimates feed grain demand, this conversion factor must be provided to show the actual amount of corn that the U.S. will export.

3.7.4 Results For Policy And Free Ending Stocks

To complete the demand identity, ending stock quantities must be estimated. The government will hold stocks in both the CCC and FOR while speculators will hold free stocks. Since policy stocks are not residual and are established by policy, some rule must be established for the accumulation and release of these stocks. Speculative stock demand is related to prices and use so an estimate of this is also needed.

a) Policy Stocks

By combining the concepts of both Shagam and Ferris (as developed in Chapter 2), a function with three major turning

TABLE 20: U.S. FEED GRAIN EXPORTS in Thousand Bushels

YEAR	CORN	SORGHUM	BARLEY	OATS	FEED GRAIN EXPORTS	CORN AS A PERCENT OF EXPORTS
1975	1,699,187.24	229,042.96	22,800.92	12,276.72	1,963,307.84	0.8655
1976	1,672,545.41	246,121.27	64,787.74	8,294.65	1,991,749.07	0.8397
1977	1,935,283.61	213,500.45	55,497.49	10,002.97	2,214,284.53	0.8740
1978	2,122,154.40	206,579.96	24,649.21	10,399.51	2,363,783.08	0.8978
1979	2,418,907.98	324,981.03	52,820.27	2,757.65	2,799,466.92	0.8641
1980	2,338,298.70	304,588.80	75,660.39	13,268.16	2,731,816.05	0.8560
1981	1,954,322.40	249,078.21	98,382.31	2,686.87	2,304,469.79	0.8481
1982	1,856,734.78	214,468.87	44,169.94	750.42	2,116,124.01	0.8774
1983	1,850,856.65	246,414.03	88,817.74	941.57	2,187,029.99	0.8463
1984	1,823,083.04	299,299.84	71,654.91	494.91	2,194,532.69	0.8307
1985	N/A	N/A	N/A	N/A	N/A	

AVERAGE PERCENT = 0.8600

points can be developed. The functional form used in estimating policy stocks follows:

$$\text{Stocks}_t = f\left(\frac{P_t}{CLR_t} , \frac{P_t}{CLR_t}^2 , \frac{P_t}{CLR_t}^3 \right)$$

where: Stocks_t = change in policy ending stocks in time t,

P_t = world price of corn in time t,

CLR_t = corn loan rate in time t.

Since the loan rate establishes a floor under price, combining the price and loan rate, creates the ratio needed to do analysis. The rules used to adjust the cubic function (as discussed in Chapter 2) were as follows:

A) CCC stocks in the years before 1976 were viewed as being 10% available. Stocks after 1976 were viewed as being 80% available unless; a) the current stocks accumulated are less than the previous year and/or b) the price over loan rate ratio is greater than 1.55 in which case the stocks were seen as being 10% available and

B) FOR stocks before 1976 were viewed and being non existent. Stocks after 1976 were viewed as being 75% available unless; a) the current stocks accumulated are less than the previous year and/or b) the price over loan rate ratio is greater than 1.55 in which case the stocks were seen as being non-existent.

Shagam cites a problem with the policy stock equation in that the formula is not a smooth functional rule but rather the choice of one of two percentages. He continues that several attempts were made to derive a smooth function but all attempts failed.

Both of the policy stock equations used the price loan rate ratio (linearly, squared, and cubed) as the independent variables. The time period estimated was 1977 to 1997. Each equation estimated the change in stocks from accumulation or release as the dependent variable.

The equation for the change in FOR stocks had good results (equation 1 of Table 21). Again each variable was significant and the R-squared was 0.96, the estimate was inconclusive in determining the correlation between residual error terms.

The equation for the change in CCC stocks also showed good results (see equation 2 Table 21). Each variable had significant t statistics, the adjusted R-squared was high at 0.91 and the standard error small. The DW statistic showed some signs of negative serial correlation.

b) Free Stocks

Once the rules had been established on the release and accumulation of policy stocks, the equation used to estimate free stocks could be developed. The functional form needed for this relationship follows:

$$FS_t = f(P_t, CLR_t, CCC_t, FOR_t, Util_t)$$

where: FS_t = Free Stocks in time t,

P_t = world price of corn in time t,

CLR_t = corn load rate in time t,

TABLE 21: Ordinary Least Squares Estimates for A CHANGE IN POLICY STOCKS in Thousand Bushels

Equation	Time Period	Independent Variables				$\bar{R}^2$	S. E.	DW
		Constant	FPLR	FPLR2	FPLR3			
1	1977-97	28637.804 (2.72)	-53497.37 (-2.21)	35532.478 (1.95)	-8431.004 (-1.89)	0.962	281.254	1.52
2	1977-97	34330.23 (3.05)	-68153.12 (-2.63)	45967.279 (2.37)	-10569.07 (-2.22)	0.913	300.393	1.09

where: FPLR = Feed grain price: corn loan rate ratio

FPLR2 = Feed grain price: corn loan rate ratio squared

FPLR3 = Feed grain price: corn loan rate ratio cubed

CCC_t = estimated CCC stocks in time t,

FOR_t = estimated FOR stocks in time t,

$Util_t$ = utilization in time t.

The independent variables in this equation were a summation of the ruled policy stocks to utilization for one variable and the price over the loan rate to utilization as the second (see Table 22).

The results show very significant t statistics and the DW statistic indicates that the null hypothesis of zero serial correlation can be accepted. The adjusted R-squared was 0.73 which shows good explanatory power of the equation.

3.8 U.S. CORN PRICE DETERMINATION

The basic model framework used in this approach is:

$$S_t = f\ (\ P_{t-1}\ )$$

$$D_t = f'\ (\ P_t\ )$$

$$ST_t = f''\ (\ P_t\ )$$

$$P_t = f\ (P_{t-1}\ ) + f''\ (\ P_{t-1}\ ) - f'\ (\ P_t\ ) - f''\ (\ P_t\ )$$

where: S_t = Total supply in the current year,

D_t = Total demand in the current year,

P_t = The average farm price of corn in the current year,

ST_t = Total ending stocks in the current year.

By using this system of equations as a recursive model, and the supply being essentially predetermined by the price in the previous period, the price can be determined by the

TABLE 22: Ordinary Least Squares Estimate for FREE STOCKS in Thousand Bushels

Equation	Time Period	Independent Variables				$\bar{R}^2$	S. E.	DW
		Constant	ALPHA	DELTA	DV83			
1	1964-84	1127.423 (12.80)	-1873.69 (-4.57)	-1301390 (-5.17)	-619.161 (-5.48)	0.727	111.965	2.16

where: ALPHA = Weighted policy stocks: domestic utilization ratio

DELTA = Feed grain price over corn loan rate:
domestic utilization ratio

DV83 = Dummy variable to account for the 1983 PIC program procedure outlined in the FAPSIM model. This price equation structure allows the model to be solved by starting price at an very high level and successively reducing price until the quantity demanded equals the amount available.

CHAPTER IV

THE ECONOMIC SUPPLY/DEMAND BALANCE SHEET WITH APPLICATIONS

Chapter four discusses the development of the actual microcomputer balance sheet. Ex post analysis is performed on the forecasts and sample runs with analyses under differing scenarios is provided.

4.1 THE ECONOMIC SUPPLY/DEMAND BALANCE SHEET

The first step in designing an economic supply and demand balance sheet was to plan a way to input the relevant variables and to display the relevant information. A commercial screen generator, SCREEN SCULPTURE, was used in screen development.[1] This screen generator was used to define the screens, including; the variable names, the variable types (e.g. real, integer, yes vs. no, etc.) the default values, the allowable range for input values, and screen layout.

The screen generator then automatically generated the input/output computer code associated with the screens in the Pascal language.

[1] Screen Sculpture, Copyright 1985, The Software Bottling, Company of New York.

The second step was to write an algorithm which would equate the supply and the demand, determine the equilibrium corn price, and tied the screens together.[2] The algorithm was written with TURBO PASCAL and utilized the equations discussed in Chapter 3.[3] In the U.S. corn industry, the annual supply is set for the year and is known with some certainty which essentially presents an inelastic supply curve. The level of disappearance is an aggregate of the domestic consumption, and the exports. The ending stocks are added to the disappearance to support equilibrium.

Through a series of menu options, the user will input all of the relevant predetermined and exogenous variables for the forecast year. The user can then choose to solve the model with these values and view each of the supply and demand factors estimated (as discussed in Chapter 3) and the market clearing farm price. A supply/demand balance sheet in this form allows the user to perform case studies under different scenarios by changing the predetermined and exogenous variables.

A review of the literature on decision-making theory and agricultural economics research contrasted with a survey of farm management instructors and Extension specialists indicates the existence of a large "gap" between theory and

[2] See Appendix 3 for documentation to the Supply/Demand Balance Sheet and the input/output code of the Pascal program.

[3] Turbo Pascal System, Copyright 1985, Borland Inc.

practice in risky farm decision-making.[4] A major step in closing this gap is to help quantify the probabilities for these farm decision makers. To this end, one objective of this project was to develop probabilities for and a confidence interval of the expected price generated by the model.

The probability (i.e. the numerical value) assigned to a particular event reflect the degree of belief that that event will occur. For example, the model forecasts an expected farm price, the probability would be an interval of prices for which there is a 99, 95, 90, 80, 70, ... percent chance of occurring. The decision maker is then allowed to use his level of risk preference to make planning decisions.

The accuracy of the probability forecasts could be improved if the approach provides an opportunity to revise initial probability estimates when new information becomes available.

Weather is a predominant cause of yield variations from year to year. Since long-range weather forecasting is not yet very dependable we must resort to probability statements about the weather (and hence price forecasts) by reporting

[4] Walker, O.L. and A.G. Nelson, Agricultural Research and Education Related to Decision-Making Under Uncertainty: An Interpretive Review of Literature, Oklahoma State University Agricultural Experiment Station Research Report P-747, March 1977; Walker, O.L. "Teaching Decision Making Under Uncertainty in the Farm Management Curriculum." Proceedings of Farm Management Teaching Workshop, Michigan State University, East Lansing, Michigan, April 14-15 1977, New York: Agricultural Development Council, Inc., p. 124-130.

yields in the form of probability distributions.[5] When the relevant variables have been inputed, the model will randomly shock the importer, the exporter and the U.S. yields. This procedure assumes a normal distribution for the weather (as it affects yields) since it can be said to conform to the central limit theorem. The process results in prices being displayed as probabilities.

A reasonable question at this time would be what does the user do with this information? The use of any marketing alternative essentially reshapes the price probability distribution (by cutting off the tails) that one faces. Although moving into that area is beyond the scope of this project, the subject will be addressed in the recommendations of the next chapter.

4.2 EX POST ANALYSIS

The ex post forecast covers the period between the estimation period and the present. A distinction can be made between conditional and unconditional forecasts. In an unconditional forecast, values for all the explanatory variables in the forecasting equation are known with certainty. In a conditional forecast, values for one or more explanatory variables are not known with certainty, so that guesses or forecasts of them must be used to produce the forecast of the dependent variable.

[5] John Ferris, "Probability Forecasts on U.S. Corn Prices," Ag Econ Staff Paper No. 82-4, Michigan State University, December 1983.

Using explanatory variables which are known with certainty provides a measure of the balance sheet's ability to forecast accurately.

4.2.1 The 1985-86 Marketing Year

Since most of the equations in the balance sheet are estimated through 1984, the 1985-86 marketing year would be the first ex post forecast.

For the 1985-86 marketing year, the simple acres planted equation did a reasonable job of forecasting considering the amount of time researchers have committed to an efficient acres planted equation (See run 1 of Table 23). The balance sheet forecasted the acres planted as 80.7 million acres which is 3% less than the actual amount of 83.2 million.

Consequently, the acres harvested forecast (which is essentially a percentage of acres planted) was 5.2% less than the actual acres harvested. The yield is based on a time trend and was 5% less than the actual record yield per acre of 118.0 bushels. These errors in domestic production resulted in the U.S. production forecast being 10% less than the actual production and the U.S. supply being 927 million bushels short of the actual supply.

Given the short supply, the balance sheet estimated a farm price of $2.65. Based on this price, the feed consumption was forecasted as 4046 million bushels and the exports were forecasted as 1797 million bushels. The CCC,

TABLE 23: BALANCE SHEET FOR CORN: THE 1985-86 MARKETING YEAR

RUN NUMBER	1	2	3	4	5	6	7
			(MILLION ACRES)				
ACRES SET-ASIDE	5.9	5.9	5.9	5.9	5.9	5.9	5.9
ACRES PLANTED	80.7	83.2	83.2	83.2	83.2	83.2	83.2
ACRES HARVESTED	71.2	75.1	75.1	75.1	75.1	75.1	75.1
YIELD PER ACRE	111.6	118.0	118.0	118.0	118.0	118.0	118.0
			(MILLION BUSHELS)				
BEGINNING STOCKS	1648	1648	1648	1648	1648	1648	1648
PRODUCTION	7950	8877	8877	8877	8877	8877	8877
IMPORTS	11	11	11	11	11	11	11
TOTAL SUPPLY	9609	10536	10536	10536	10536	10536	10536
USE							
FEED	4046	4187	4095	4115	4095	4205	4095
FOOD, SEED & IND. USES	1160	1160	1160	1160	1160	1160	1160
TOTAL DOMESTIC	5206	5347	5255	5275	5255	5365	5255
EXPORTS	1797	1825	1241	1811	1241	1241	1241
TOTAL USE	7004	7173	6496	7087	6496	6606	6496
TOTAL ENDING STOCKS	2604	3362	4040	3448	4040	3929	4040
TOTAL, % OF USE	37.2	46.9	62.2	48.6	62.2	59.5	62.2
U.S. FARM PRICE	$2.65	$2.41	$2.21	$2.53	$2.35	$2.38	$2.35
STOCKS IN GOVT. PROG.							
CCC INVENTORY	702	1102	1580	1172	1591	1514	1591
FARMER-OWNED RES.	1442	1945	2434	1999	2423	2343	2423
FREE STOCKS	459	313	25	276	24	71	24
EFFECTIVE LOAN RATE	$2.55	$2.55	$2.55	$2.69	$2.69	$2.69	$2.69

FOR and free stocks were forecasted as 702, 1442 and 459 million bushels respectively.

To correct for the discrepancies in the ex post forecasts, add factors can be used. The domestic production is corrected for by adding 2.5 million acres to the acres planted. Acres harvested needed an add factor of 3.9 million acres and yield needed 6.4 bushels per acre to force the time trend forecast up to the record yield.

With the domestic supply corrected to equal the actual supply, the price and disappearance can be analyzed. Given the corrected supply the balance sheet forecasted a price of $2.41 (Run 2 of Table 23). As expected, with a lower price, the disappearance forecasts increased. In this case, the feed consumption was 2.2% greater than the actual amount while exports were 45% greater.

By using the override feature and forcing the feed consumption and exports equal to the actual amounts, the balance sheet gives an interesting policy result (Run 3 of Table 23). Using all the correct supply and demand amounts, the farm price generated was $2.21. This indicates that the corn loan rate of $2.55 is not the effective loan rate. Producers know that there is an amount of corn held off the market by the government "PIK and ROLL" program. The PIK certificate has a face value in cash and the number of bushels a dollar amount will redeem varies with prices. Thus the producer is indifferent as to the actual farm price of corn if he is covered under the "PIK and ROLL" program and the price can fall below the loan rate.[6]

By increasing the corn loan rate to the effective amount of $2.69 and allowing the disappearance to be determined by market forces a new analysis can be done (Run 4 of Table 23). A farm price of $2.53 was generated under this analysis. The feed consumption was less than 1% greater than the actual amount and exports just over 45% greater than the actual.

When the feed consumption and export forecasts are overridden to equal the actual amounts, a farm price of $2.35 was generated (Run 5 of Table 23). This shows that the policy of the government program has an affect on the producer expectation of the effective corn loan rate.

The interesting ex post analysis was the way the international component forecasted U.S. exports. This component is a conditional forecast because of the lag in the publication of international data. By using the corrected U.S. supply and the effective corn loan rate of $2.69, the forecasted U.S. exports exceeded the actual amount by 45 percent (See run 5 of Table 23). Examining the components of U.S. exports, the balance sheet forecasted U.S. exports as being exactly equal to the importing

[6] For discussion on the "PIK and ROLL" program, see the Agricultural Economics Staff Paper No. 86-90, Michigan State University, by Jim Hilker and Mary Schultz, November 1986.

region's net imports (Run 5 of Table 24). Further analysis showed the exporting region had zero exports.

The structure of the exporting region's net exports is that all grain not consumed or held in stocks will be exported. The equations for consumption and ending stocks were reasonably consistent with historical data however, the supply forecast was significantly lower (60% lower) than historical exporter supply (Run 5 of Table 25).

The exporting region's supply is the sum of the current production and the lagged ending stocks. The equation used to forecast the exporter production is a function of lagged revenues for feed grain and wheat and a lagged price proxy. With the hyper-inflation that Argentina is experiencing, the lagged revenues generated were essentially nullified by the Argentine consumer price index. And thus the overall production was reduced. To correct for this effect, the exporter harvested area add factor needed to be used.

Adding 12.8 million hectares to the production of the exporting region moved the exporter supply in line with the historical record. Since the exporting region's consumption is a function of (among others) the supply, it was necessary to decrease the exporters consumption through the use of an add factor. By subtracting 0.109 thousand metric tons per capita from the exporting region's consumption, the quantity consumed was more historically accurate. Run 6 of Table 25 shows the corrected exporting region's net exports.

TABLE 24: U.S. EXPORT DEMAND: THE 1985-86 MARKETING YEAR

RUN NUMBER	5	6
(MILLION BUSHELS)		
IMPORTERS NET IMPORTS	1811	2017
EXPORTERS NET EXPORTS	0	776
U.S. EXPORTS	1811	1241

TABLE 25: EXPORTING REGION'S NET EXPORTS:
THE 1985-86 MARKETING YEAR

RUN NUMBER	5	6
(MILLION BUSHELS)		
EXPORTERS SUPPLY	752	1900
EXPORTERS CONS.	820	911
EXPORTERS STOCKS	129	213
EXPORTERS EXPORTS	0	776

With the unconditional knowledge that the U.S. exports were roughly 33% less than the historical average and assuming that the exporting region is continuing the same policy of "dumping" their residual grain, the implication is that the importing region imported much less in the 1985-86 marketing year. The balance sheet forecasted a much lower net import by the importing region and the use of an add factor of 0.0012 thousand metric tons per capita was needed to increase the net imports (Run 6 of Table 24).

The final ex post analysis for the 1985-86 marketing year examined the forecast of corn consumed by livestock. By correcting the domestic supply, using the effective corn loan rate, and adjusting the international component as discussed, the balance sheet forecast the price as $2.38 (Run 6 of Table 23). With the price higher actual, theory says that the quantity demanded would be lower, however, the balance sheet over forecast the quantity of corn consumed by livestock by 2% (See run 6 of Table 26). By using a negative add factor of 129.37 million bushels on the other and unallocated (i.e. residual) consumption the forecast for corn consumed by livestock reached the actual level of 4095 million bushels (Run 7 of Table 26). Run 7 of Table 23 shows the actual supply and demand factors for the 1985-86 marketing year using all add factors discussed and a market clearing farm price of $2.35.

TABLE 26: LIVESTOCK CONSUMPTION OF CORN IN THE U.S.: THE 1985-86 MARKETING YEAR

RUN NUMBER	6	7
(MILLION BUSHELS)		
DAIRY ANIMALS	830	833
CATTLE ON FEED	658	664
OTHER BEEF CATTLE	187	188
HENS, PULLETS, & OTHER CHICKENS	334	337
BROILERS	481	483
TURKEYS	119	119
HOGS	1445	1449
OTHER AND UNALLOCATED	148	19
TOTAL CONSUMPTION	4205	4095

4.2.2 The 1986-87 Marketing Year

The 1986-87 marketing year is another unconditional forecast for the balance sheet however, due to the lag in the reporting of international component statistics, some of the disaggregated factors of the balance sheet are conditional upon educated guesses of the exogenous variables.

In 1986-87 the acres set-aside for participation in the government program were 13.6 million acres. The acres planted equation forecasted that 75.0 million acres would be planted to corn, which again under estimates the amount of acreage actually committed to corn by 2.2% percent or 1.7 million acres (See run 1 of Table 27).

The acres harvested equation also under estimated the actual quantity by 4.2% due to the low estimate of acres planted. The trend used to forecast yield was 113.8 bushels per acre which is 5.5 bushels per acre less than the actual record yield of 119.3 bushels. With the reduced acres planted forecast, the reduced acres harvested forecast and the low yield forecast, the domestic production was forecasted as 8.5% lower than the actual amount. This forecast caused the U.S. supply to be under estimated by 707 million bushels.

Using these figures for domestic supply, the U.S. farm price was forecasted at $1.99. Since this price was higher than the actual, both feed consumption and U.S. exports were under forecast as 4404 million and 1225 million bushels respectively. The estimates of CCC and FOR stocks were 1765 million and 3001 million bushels respectively while, at this price, the free stocks were forecasted at a zero level.

When the add factors are imposed on the domestic production forecasts, the actual production amounts can be obtained. An add factor of 1.7 million acres must be added to the acres planted forecast. The acres harvested needed an add factor of 2.9 million acres and the yield trend needed 5.5 bushels per acre added to the equation to equal the record yield.

TABLE 27: BALANCE SHEET FOR CORN: THE 1986-87 MARKETING YEAR

RUN NUMBER	1	2	3	4	5	6	7
			(MILLION ACRES)				
ACRES SET-ASIDE	13.6	13.6	13.6	13.6	13.6	13.6	13.6
ACRES PLANTED	75.0	76.7	76.7	76.7	76.7	76.7	76.7
ACRES HARVESTED	66.3	69.2	69.2	69.2	69.2	69.2	69.2
YIELD PER ACRE	113.8	119.3	119.3	119.3	119.3	119.3	119.3
			(MILLION BUSHELS)				
BEGINNING STOCKS	4040	4040	4040	4040	4040	4040	4040
PRODUCTION	7547	8253	8253	8253	8253	8253	8253
IMPORTS	2	2	2	2	2	2	2
TOTAL SUPPLY	11588	12295	12295	12295	12295	12295	12295
USE:							
FEED	4404	4517	4650	4650	4785	4749	4650
FOOD, SEED & IND. USES	1191	1191	1191	1191	1191	1191	1191
TOTAL DOMESTIC	5595	5708	5841	5841	5976	5940	5840
EXPORTS	1225	1255	1525	1525	1315	1525	1525
TOTAL USE	6821	6963	7366	7366	7292	7465	7366
TOTAL ENDING STOCKS	4766	5331	4929	4929	5002	4829	4928
TOTAL, % OF USE	69.9	76.6	66.9	66.9	68.4	64.7	66.9
U.S. FARM PRICE	$1.99	$1.83	$1.94	$1.51	$1.50	$1.54	$1.51
STOCKS IN GOVT. PROG.							
CCC INVENTORY	1765	1991	1826	1837	1854	1791	1837
FARMER-OWNED RES.	3001	3339	3102	3091	3132	3038	3090
FREE STOCKS	0	0	0	0	0	0	0
EFFECTIVE LOAN RATE	$1.84	$1.84	$1.84	$1.51	$1.51	$1.51	$1.51

Once the domestic production is corrected, a new analysis can be conducted (Run 2 of Table 27). Given the corrected production, the supply is now increased to the actual and the balance sheet forecasts a farm price of $1.83. With a farm price higher than actual, the quantities of feed consumption and exports were reduced. In this analysis the feed consumption was 2.9% lower than the actual amount and the exports were 17.7% less. This farm price increased the amount of policy stocks from the original forecasted amount to 1991 million bushels for CCC holdings and 3339 million bushels for FOR stocks. Again the free stocks were forecasted at a zero level.

By overridding all supply and demand factors equal to the known amounts, the forecasted farm price should equal the actual farm price of $1.51. When this scenario is run the farm price is forecasted as $1.94 which again indicates that corn loan rate of $1.84 is not the effective corn loan rate (Run 3 of Table 27). After several runs of trial and error the effective corn loan rate was determined to be $1.51 which is essentially a floor under the amount that the farm price will drop (Run 4 of Table 27).

A new analysis can be conducted using the corrected supply quantities, the effective corn loan rate and allowing the demand factors to be determined by market forces (Run 5 of Table 27). In this analysis, the farm price dropped to $1.50.

Since the international component is still conditional upon the estimates of the exogenous variables some educated guesses were required for these factors. Examination of the importing region's net imports shows that it is low by historical comparisons (Run 5 of Table 28). An add factor must be used to increase the importing region's net imports. Adding 0.00273 thousand metric tons per capita to the importing region's net imports brings it into line with historical values (Run 6 of Table 28).

Run 5 of Table 29 shows that the exporting region's net exports is too low given the add factor used to increase the importing region's net imports. Since the exporting region's consumption is a function of their supply, an add factor was needed to reduce the regional consumption. Subtracting 0.129 thousand metric tons per capita from the exporting regions consumption creates historical comparisons. Finally, the exporting region's ending stocks needed an add factor of 0.05 thousand metric tons per capita (Run 6 of Table 29) to complete the exporter net exports.

Once all the corrections have been made on the domestic supply, the corn loan rate and the international component, the analysis of corn consumed by livestock can be made.

The farm price for this scenario is $1.54 and the forecast for feed consumption is 4749 million bushels which is 2.1% greater than the actual amount (Run 6 of Table 27 and Table 30). A negative add factor of 126.42 applied to the other and unallocated consumption would reduce the

TABLE 28: U.S. EXPORT DEMAND: THE 1986-87 MARKETING YEAR

RUN NUMBER	5	6
(MILLION BUSHELS)		
IMPORTERS NET IMPORTS	2102	2522
EXPORTERS NET EXPORTS	788	996
U.S. EXPORTS	1314	1525

TABLE 29: EXPORTING REGION'S NET EXPORTS:
THE 1986-87 MARKETING YEAR

RUN NUMBER	5	6
(MILLION BUSHELS)		
EXPORTERS SUPPLY	2163	2163
EXPORTERS CONS.	1314	973
EXPORTERS STOCKS	62	194
EXPORTERS EXPORTS	788	996

TABLE 30: LIVESTOCK CONSUMPTION OF CORN IN THE U.S.: THE 1986-87 MARKETING YEAR

RUN NUMBER	6	7
(MILLION BUSHELS)		
DAIRY ANIMALS	843	845
CATTLE ON FEED	879	892
OTHER BEEF CATTLE	225	227
HENS, PULLETS, & OTHER CHICKENS	390	392
BROILERS	527	529
TURKEYS	144	145
HOGS	1576	1580
OTHER AND UNALLOCATED	165	40
TOTAL CONSUMPTION	4750	4650

forecast to 4650 million bushels and a market clearing price of $1.51 (Run 7 of Table 27 and Table 30).

4.2.3 The 1987-88 Marketing Year

At the time of this publication, the 1987-88 marketing year supply factors are known with certainty, however the final demand factors are unknown. This makes the supply factors an unconditional forecast while the domestic and international components are conditional upon the estimates of the user. This creates a situation where the farm price is also not known with certainty and the USDA provides a range of prices for the 1987-88 marketing year.

In this ex post forecasts the acres planted equation over estimated the quantity of acres committed to corn (See run 1 of Table 31). The 1987-88 forecast of acres planted was 2.6% greater than the actual amount. Due to the quantity forecasted to be planted, the acres harvested forecast was also over estimated. This forecast was less than 1 percent greater than reality. The yield per acre was a new record of 119.9 bushels and thus the trend yield forecast was low by 4 bushels per acre.

The 1987-88 forecast for domestic production was under estimated and is 2.5% less than the actual amount. This caused the U.S. supply to be forecast as 175 million bushels less than the actual. Given the short U.S. supply forecast the balance sheet forecast a farm price of $2.18, 36 cents over the policy established corn loan rate of $1.82. The corn consumed by livestock and U.S. exports were forecast as 4252 million and 1414 million bushels respectively. The total ending stocks as a percent of use were 72.9 percent and the government held all the stocks (1802 for CCC and 3212 for FOR).

Add factors can be used on the domestic supply with certainty to determine how well the balance sheet forecasts given an accurate supply. By subtracting 1.7 million acres from the acres planted, subtracting 0.5 million acres from the acres harvested and adding 4.0 bushels per acre to the yield gives the correct domestic supply (Run 2 of Table 31).

TABLE 31: BALANCE SHEET FOR CORN: THE 1987-88 MARKETING YEAR

RUN NUMBER	1	2	3	4	5	6	7	8	9
				(MILLION ACRES)					
ACRES SET-ASIDE	21.5	21.5	21.5	21.5	21.5	21.5	21.5	21.5	21.5
ACRES PLANTED	67.7	66.0	66.0	66.0	66.0	66.0	66.0	66.0	66.0
ACRES HARVESTED	60.1	59.6	59.6	59.6	59.6	59.6	59.6	59.6	59.6
YIELD PER ACRE	115.9	119.9	119.9	119.9	119.9	119.9	119.9	119.9	119.9
				(MILLION BUSHELS)					
BEGINNING STOCKS	4929	4929	4929	4929	4929	4929	4929	4929	4929
PRODUCTION	6966	7141	7141	7141	7141	7141	7141	7141	7141
IMPORTS	2	2	2	2	2	2	2	2	2
TOTAL SUPPLY	11897	12072	12072	12072	12072	12072	12072	12072	12072
USE:									
FEED	4252	4285	4700	4700	4485	4700	4523	4508	4700
FOOD, SEED & IND. USES	1215	1215	1215	1215	1215	1215	1215	1215	1215
TOTAL DOMESTIC	5467	5500	5915	5915	5640	5915	5738	5723	5915
EXPORTS	1414	1423	1600	1600	1459	1600	1482	1607	1600
TOTAL USE	6882	6924	7515	7515	7099	7515	7221	7330	7515
TOTAL ENDING STOCKS	5015	5147	4557	4557	4972	4557	4850	4741	4556
TOTAL, % OF USE	72.9	74.3	60.6	60.6	70.0	60.9	67.2	64.7	60.6
U.S. FARM PRICE	$2.18	$2.13	$2.57	$2.02	$1.93	$1.86	$1.80	$1.82	$1.86
STOCKS IN GOVT. PROG.									
CCC INVENTORY	1802	1845	1402	1416	1556	1409	1513	1481	1409
FARMER-OWNED RES.	3212	3302	2346	2331	2619	2339	2536	2456	2339
FREE STOCKS	0	0	807	808	797	808	800	803	808
EFFECTIVE LOAN RATE	$1.82	$1.82	$1.82	$1.49	$1.49	$1.39	$1.39	$1.39	$1.39

With the new results the analysis shows that the farm price falls to $2.13 and the livestock consumption and exports fall. Given the a price of $2.13, the livestock consumption was 8.9% lower and the exports were 11% lower than the USDA forecast of 4700 million and 1600 million bushels respectively. This lower price also increased policy stock holdings by 1.4 percent.

When the demand factors are forced to equal the USDA forecast, the farm price of the balance sheet shows that $2.57 would be needed to clear the market (Run 3 of Table 31). Under this scenario, the CCC, FOR and free stocks were 1402, 2346 and 807 million bushels respectively.

The analysis could stop here (due to the conditional nature of the demand factors) except the balance sheet forecast price does not fall within the range of the USDA forecast price. Assuming the effective corn loan rate carries over from the 1986-87 ex post forecast and drops by two cents as directed by U.S. policy, the 1987-88 effective corn loan rate of $1.49 would be used (Run 4 of Table 31). In this scenario, the domestic supply is corrected with add factors and the demand factors are overridden to equal the USDA forecasts. The balance sheet farm price is forecast as $2.02 and the policy stocks are forecast as 1416 million for CCC and 2331 million bushels for FOR while the free stocks are forecast as 808 million bushels. Again the farm price forecast does not fall within the USDA forecast range. Using the same corn loan rate assumption and allowing the market to determine the demand factors the farm price clears the market at $1.93 while both the corn consumed by livestock and exports are under the USDA forecast by 4.6% and 8.8% respectively (Run 5 of Table 31). The total ending stocks were 70.0% of use with the free stocks forecast as 797 million bushels.

To establish a farm price within the USDA forecast range, the effective corn loan rate was reduced ten cents more (Run 6 of Table 31). With the effective corn loan rate set at $1.39, using the corrected domestic supply and overridding the demand factors, the balance sheet forecast a market clearing farm price of $1.86 which is within the USDA forecasted range.

When the market determines the demand factors, the balance sheet forecast a farm price of $1.80 (Run 7 of Table 31). At this price the corn consumed by livestock and exports were 4523 and 1482 million bushels respectively, still lower than the USDA forecasts. Ending stocks were 67.2% of total use and free stocks were forecast as 800 million bushels.

The add factors for the international component are conditional upon the knowledge of the user however, following historical trends for values leads to reasonable results. The initial forecast of the importing region's net imports shows that it is low compared to historical records (Run 7 of Table 32). The importing region needed an add

TABLE 32: U.S. EXPORT DEMAND: THE 1987-88 MARKETING YEAR

RUN NUMBER	7	8
(MILLION BUSHELS)		
IMPORTERS NET IMPORTS	2324	2587
EXPORTERS NET EXPORTS	841	987
U.S. EXPORTS	1482	1600

TABLE 33: EXPORTING REGION'S NET EXPORTS:
THE 1986-87 MARKETING YEAR

RUN NUMBER	7	8
(MILLION BUSHELS)		
EXPORTERS SUPPLY	2195	2195
EXPORTERS CONS.	1330	1025
EXPORTERS STOCKS	24	183
EXPORTERS EXPORTS	841	987

factor of 0.00174 thousand metric tons per capita to show growth (Run 8 of Table 32).

Given the add factor increase to the importing region's net imports, the exporting region's net export forecast was too low to equal the actual USDA forecast. The exporting region needed 0.1114 thousand metric tons subtracted from their regional consumption and 0.059 added to their regional ending stocks (Run 8 of Table 33) using these corrections to the international component brought the balance sheet U.S. export forecast in line with the USDA forecast.

With the described corrections made to the balance sheet forecast, the corn consumed by livestock could be analyzed. Given these corrections the balance sheet forecast a price of $1.82 and a livestock consumption quantity of 4508 million bushels (Run 8 of Table 34). Each of the feeding rates per head followed historical trends except the cattle on feed and the other beef cattle figures (Run 8 of Table 35 and Table 36). These feeding rate per head forecasts were lower than what would be expected. By using add factors of 10.0 bushels per head for cattle on feed and 0.3 for other beef cattle, the forecast went from 52.9 to 61.8 bushels per head in run 9 of Table 35 and 5.3 to 5.6 bushels per head in run 9 of Table 36. An add factor of 84.2 was needed on the other and unallocated forecast to account for the residual quantity of corn unaccounted for by the other equations (Run 9 of Table 34).

TABLE 34: LIVESTOCK CONSUMPTION OF CORN IN THE U.S.:
THE 1987-88 MARKETING YEAR

RUN NUMBER	8	9
(MILLION BUSHELS)		
DAIRY ANIMALS	813	801
CATTLE ON FEED	666	778
OTHER BEEF CATTLE	224	234
HENS, PULLETS, & OTHER CHICKENS	390	387
BROILERS	525	523
TURKEYS	156	154
HOGS	1566	1562
OTHER AND UNALLOCATED	168	251
TOTAL CONSUMPTION	4508	4650

TABLE 35: CATTLE ON FEED CONSUMPTION PER HEAD:
THE 1987-88 MARKETING YEAR

RUN NUMBER	8	9
FEEDING RATE PER HEAD	52.9	61.8
CATTLE & CALVES ON FEED (MIL. HEAD)	12.6	12.6
CATTLE ON FEED CONSUMPTION (MIL. BU.)	666.2	778.1

TABLE 36: OTHER BEEF CATTLE CONSUMPTION PER HEAD:
THE 1987-88 MARKETING YEAR

RUN NUMBER	8	9
FEEDING RATE PER HEAD	5.3	5.6
BEEF COWS AND REPLACEMENTS (MIL. HEAD)	42.0	42.0
OTHER BEEF CATTLE CONS. (MIL. BU.)	223.6	234.2

After making the corrections to the domestic supply, the international component and the corn consumed by livestock, the balance sheet forecasted a clearing farm price of $1.86 which is within the USDA forecast range.

4.3 SCENARIO ANALYSES AND SAMPLE RUNS

The 1988-89 balance sheet forecast will be a completely conditional forecast, and it will use the partially conditional 1987-88 forecast as the lagged exogenous variable vector. The first step in making the forecast was to estimate the exogenous variables for the 1988-89 marketing year. These exogenous variables are based on historical trends and economic knowledge of the user. Once the values have been entered into the exogenous variable section, the model solves for the supply and demand factors and presents the information in the balance sheet.

The initial forecast for the 1988-89 marketing year (See run 1 of Table 37) showed an acres planted forecast of 70.5 million acres. The acres harvested was 62.6 million acres and the yield was forecasted as 118.1 bushels per acre. In order to bring the acres planted forecast plus the acres set-aside equal to a 7 year average of total corn acreage (as estimated by the USDA) an add factor of 4.2 million acres was subtracted form the acres planted.

This add factor adjusted forecast will be known as the control (See 'Control' run of Table 37). In the control, the balance sheet forecast a market clearing price of $2.05.

TABLE 37: BALANCE SHEET FOR CORN: THE 1988-89 MARKETING YEAR

RUN NUMBER	1	CONTROL	2	3	4	5	6	7	8
				(MILLION ACRES)					
ACRES SET-ASIDE	21.0	21.0	21.0	21.0	21.0	21.0	21.0	21.0	21.0
ACRES PLANTED	70.5	66.3	66.3	66.3	66.3	66.3	66.3	66.3	66.3
ACRES HARVESTED	62.6	58.9	58.9	58.9	58.9	58.9	58.9	58.9	58.9
YIELD PER ACRE	118.1	118.1	118.1	118.1	105.0	124.0	118.1	118.1	118.1
				(MILLION BUSHELS)					
BEGINNING STOCKS	4557	4557	4557	4557	4557	4557	4557	4557	4557
PRODUCTION	7389	6956	6965	6965	6185	7304	6956	6956	6956
IMPORTS	2	2	2	2	2	2	2	2	2
TOTAL SUPPLY	11948	11515	11515	11515	10744	11863	11515	11515	11515
USE									
FEED	4602	4518	4699	4244	4273	4588	4562	4516	4490
FOOD, SEED & IND. USES	1235	1235	1235	1235	1235	1235	1235	1235	1363
TOTAL DOMESTIC	5837	5753	5934	5479	5508	5823	5797	5751	5853
EXPORTS	1576	1557	619	2465	1495	1573	1632	1557	1551
TOTAL USE	7414	7311	6554	7945	7004	7396	7429	7309	7404
TOTAL ENDING STOCKS	4534	4204	4961	3570	3739	4467	4086	4205	4114
TOTAL, % OF USE	61.2	57.5	75.7	44.9	53.4	60.4	55.0	57.5	55.5
U.S. FARM PRICE	$1.93	$2.05	$1.80	$2.49	$2.44	$1.95	$2.10	$2.05	$2.09
STOCKS IN GOVT. PROG									
CCC INVENTORY	1593	1464	1777	1041	1103	1568	1419	1464	1428
FARMER-OWNED RES.	2921	2673	3184	1691	1833	2869	2560	2674	2587
FREE STOCKS	20	66	0	838	802	28	106	66	98
EFFECTIVE LOAN RATE	$1.80	$1.80	$1.80	$1.80	$1.80	$1.80	$1.80	$1.80	$1.80

The corn consumed by livestock was forecast at 4518 million, exports were forecast at 1557 million, CCC stocks were forecast at 1464 million, FOR stocks were forecast at 2673 and free stocks were forecast at 66 million bushels. The probability distribution for this forecast used 100 observations. The Control run of Table 38 shows a price range of $1.24 to $2.52 with a mean of $2.03 and standard deviation of 0.178.

Because the 1988-89 market year information is unknown, some scenario's were chosen to determine their effect on farm price. The scenarios to be examined include changes in world income, U.S. yield, competitor crop prices and industrial alcohol use. These scenarios will be discussed and compared to the control forecast which used the add factor on the acres planted equation.

4.3.1 Change In World Income

The first scenario is a change in the rest-of-the-world income. This action could come about from a change in exchange rates, a change in the gross domestic product of a region or a change in the consumer price index of a region. For purposes of analysis, the exogenous variable importer and exporter gross domestic product is adjusted by the desired amounts.

If the rest-of-the-world income were to decrease by 20% the analysis shows that the farm price would decrease by 12% to $1.80 (Run 2 of Table 37). Corn consumed by livestock is

TABLE 38: PROBABILITY DISTRIBUTIONS FOR VARYING SCENARIOS

RUN NUMBER	1	CONTROL	2	3	4	5	6	7	8
PROBABILITY THAT THE PRICE WILL BE THAT LISTED OR BELOW	$/BU.	$/BU.	$/BU.	$/BU.	$/BU.	$/BU.	$/BU.	$/BU.	$/BU.
1		1.24	1.56	1.91			1.74	1.78	1.76
5		1.61	1.61	1.98			1.82	1.80	1.81
10		1.79	1.66	2.05			1.87	1.83	1.88
20		1.85	1.71	2.13			1.92	1.88	1.95
30		1.90	1.74	2.17			1.96	1.92	2.00
40		1.95	1.77	2.22			2.01	1.95	2.04
50		2.00	1.79	2.27			2.06	1.99	2.09
60		2.06	1.82	2.33			2.11	2.02	2.13
70		2.10	1.86	2.51			2.19	2.07	2.18
80		2.14	1.89	2.55			2.25	2.11	2.24
90		2.20	1.95	2.59			2.48	2.15	2.30
95		2.24	1.99	2.62			2.51	2.23	2.50
99		2.52	2.14	2.72			2.63	2.50	2.62
MEAN		2.03	1.81	2.34			2.11	2.03	2.15
STANDARD DEVIATION		0.178	0.114	0.208			0.213	0.158	0.200

forecast to increase by 4% and ending stocks are forecast to increase by 18.2 percent. If the importing region had 20% less income, one would assume that U.S. exports world decrease. The analysis supports this assumption where U.S. exports drops by over 60 percent. The probability distribution had a range from $1.56 to $2.14 with a mean price of $1.81 and standard deviation of 0.114 (Run 2 of Table 38).

If the rest-of-the-world income were to increase by 20% the analysis shows an increase of 21% in farm price to $2.49 (Run 3 of Table 37). Corn consumed by livestock in this case will decrease by 6% and ending stocks will decrease by 15 percent. The assumption that U.S. exports will increase is supported by the forecast which shows a jump of 58 percent. In this scenario the probability distribution has a mean of $2.34 and a standard deviation of 0.208 (Run 3 of Table 38).

4.3.2 Change In U.S. Yield

The second scenario will examine changes in supply and demand if the U.S. does and does not continue the record setting yields of the recent past. To perform the analysis the override feature of the balance sheet yield per acre is used.

If the U.S. experiences a yield of 105 bushels per acre, the analysis shows that the domestic supply would drop by 6% and farm price would climb 19% to $2.44 (Run 4 of Table 37). Corn consumed by livestock would drop 5.4% and U.S. exports would suffer from the inflated price by falling 4 percent. The U.S. government would benefit from this action in that CCC stocks would fall 25% and FOR stocks would fall 31 percent. With the inflated price free stocks climb by 736 million bushels.

If the U.S. were to continue the record making yield of the recent past, and yield were to reach 124.0 bushels per acre the analysis shows an increase in domestic supply of 3 percent (Run 5 of Table 37). In this scenario the price drops 10 cents due to the increased supply and the demand factors increase due to the lower price. The analysis shows that corn consumed by livestock will increase 1.5 percent, exports increase 1 percent and ending stocks increase 6.3 percent.

4.3.3 Change In Competitor Crop Prices

In the third scenario a change of competitor crop prices is examined. In the first case the wheat gulf and farm price are adjusted in the exogenous variable section.

If the wheat gulf price and farm price increased 16% and 31% respectively, until they were somewhat comparable to the 1985-86 marketing year, the analysis shows that corn farm price rises 2.4 percent (Run 6 of Table 37). If one were to just examine the corn farm price change the assumption would be for a decrease in the demand factors. Yet due to the increase in wheat prices,this scenario shows the corn consumed by livestock increased over the control by about 1 percent, the exports increased by almost 5% and ending stocks dropped by 2.8 percent. The probability distribution showed a mean farm price of $2.11 and standard deviation of 0.213 in 100 observations (Run 6 of Table 38).

In the second case the soybean meal price (a proxy for soybeans) falls by 8 percent and is adjusted in the exogenous variable section. In this analysis, the corn farm price is $2.05, the same as the control run (Run 7 of Table 37).

In this scenario the corn consumed by livestock decreases by 2 million bushels, while exports and ending stocks remain the same. This is due to certain livestock corn consumption equations using soybean meal price as an exogenous variable. The probability distribution showed a mean farm price of $2.03 and standard deviation of 0.158 in 100 observations (Run 7 of Table 38).

4.3.4 Change In Alcohol Use

The final scenario examines an increase in industrial alcohol use. This action comes about from the increased interest in using ethyl alcohol as a gasoline extender. The alcohol use is increased in the exogenous variable section.

By assuming a 40% increase in alcohol production and all other food, seed, and industrial uses remain the same, the total FSI use will increase 10 percent. If that scenario occurs, the farm price is forecast to increase 2 percent (Run 8 of Table 37). The demand factors drop slightly with corn consumed by livestock declining 0.6 percent, exports decreasing 0.4% and ending stocks decreasing 2.2 percent. This makes an interesting policy analysis because the farm price increases which will make the producers happy while the government benefits by holding less policy stocks. The probability distribution shows a mean price of $2.15 and a standard deviation of 0.200 which is 2.2 cents above the control standard deviation (Run 8 of Table 38).

CHAPTER V

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

This chapter will summarize the Economic Supply/Demand Balance Sheet project and includes recommendations for future work. Suggestions include increasing the accuracy of the balance sheet and discussion on decision theory for handling risk.

5.1 SUMMARY AND CONCLUSIONS

This project accomplished the goal of developing a micro-computer economic supply/demand balance sheet for forecasting corn prices and use as a group-interaction teaching aide.

In so doing, supply and demand factors for corn were estimated. The U.S. supply consisted mainly of estimating the acres that would be planted to corn. Numerous hypotheses were examined to estimate this equation, yet due to the dynamic nature of government policy and farmer expectations an equation to estimate acres planted has never been accurately developed. The equation provided in this project cannot necessarily be considered the best, but it provides ease of use and is good for educational purposes.

Corn disappearance consists of the four components discussed below.

The corn consumed by livestock is disaggregated into seven animal species and an unallocated (i.e. residual) amount. Each animal species consumption is estimated on a per head basis with the unallocated consumption being estimated based on relative prices and a trend. Estimating on a per head basis and multiplying by animal inventories allows a more accurate explanation of corn as a feed grain disappearance.

Food, seed, and industrial use is not econometrically estimated but is based upon knowledge provided by the user as to changing economic conditions for industrial and seed use and theoretical limits and constraints on food use.

The international component is included by linking a simple rest-of-the-world supply and demand model with the U.S. and equating U.S. exports to balance world needs.

Stock holdings were estimated as what the government holds due to policy and the demand for free stocks held for speculative purposes. The policy stock equations are based upon price over the corn loan rate ratios while the free stock equation is based upon the price over the loan rate ratio and the policy stocks over utilization ratio.

Attempts were made to overcome the limitations of the system by allowing the user to adjust the intercept of each equation with an add factor. If the estimate is known to be inaccurate of if more current information becomes available, a positive or negative add factor could be included in each equation to increase the accuracy of the forecast.

An override feature is provided which allows the user to input a value for any of the supply and demand estimates using a posteriori information. That override value can then be used when the model solves which allows the user to do "what if" scenario analysis by calculating supply and demand factors using the new value.

The original results of each ex post analysis were suspect of being inaccurate, yet when add factors are used, the results became more tolerable. Several scenarios and analyses were performed on the 1988-89 marketing year. The add factor adjusted results of that marketing year became the control from which to compare the rest of the scenarios.

5.2 RECOMMENDATIONS FOR INCREASING BALANCE SHEET PERFORMANCE

The estimated equations should be updated on a regular basis using the most current data available. This allows the annual forecasts to be as accurate as possible and will account for any structural changes that may have occurred since the last estimate. The food, seed, and industrial use could be estimated in order to prevent the misinformation of the user affecting the accuracy of the forecasts.

The farm price forecast could be more accurate if all feed grains and soybeans were included in the model and the system of equations were solved simultaneously.

Sensitivity analysis could be done on the add factors to determine the amount of change from one year to another given an add factor adjustment and if the adjustments used aid in giving information from year to year.

The computer algorithm calculates all values as whole numbers whereas the balance sheet presents some information as integer values. This condition creates rounding errors to appear in the balance sheet. If the values which are presented as integers were calculated as such the rounding errors could be eliminated.

End user input should be incorporated to the design of the computer balance sheet as it is used by extension workers to make sure that the information provided is helpful as well as desired. On line help windows would be useful to guide the user through the program. Once the program is mastered, the use of the menus to recalculate a market clearing price tends to slow the user down. If a recalculate feature could be included which allows the user to make changes in the balance sheet, recalculate the supply and demand factors and resume in the same spot, the analysis process could be made more user friendly.

5.3 RECOMMENDATIONS FOR HANDLING RISK

The literature shows that it is quite difficult to develop a universally agreed-upon set of principles of individual behavior under conditions of uncertainty. However decision theory provides some principles for making decisions under risk and uncertainty. Decision theory can be used to determine optimal strategies when a decision maker is faced with several decision alternatives and an uncertain or risk-filled pattern of future events.

Using this information, approaches to incorporate the data in the balance sheet with decision theory are presented with the intent that at some future time someone can expand upon this rudimentary analysis.

5.3.1 The Traditional Analysis

Much of the pioneering works of Frank H. Knight define the traditional approach to decision theory. Knight divided uncertain situations into two subclasses: risk and uncertainty. Knight defines that an event is risky if both, the possible future outcomes that could result and the probability associated with each outcome are known.[1] Thus, risk exists when the outcome of the process, say yield per acre, is a random variable with a known probability distribution. Risk requires that outcomes can be identified and a probability can be associated with each outcome. The probabilities needed to define the risk situation are not known with certainty but are subject to all the usual vagaries of empirical estimation. Knight defines uncertainty as that event in which no probabilities can be assigned to the outcomes. In this case the possible outcomes may also be unknown.

[1] Frank H. Knight, Risk, Uncertainty, and Profit, Houghton-Mifflin, 1921, quoted by Doll, op. cit., p. 238-39.

With the theoretical acceptance of subjective knowledge (Anderson et. al.[2]), Knight's distinction between risk and uncertainty is not relevant, and the terms are used interchangeably. The argument for subjective knowledge assumes that it is impossible for the manager to know the alternative actions, the possible outcomes, and their consequences and yet be entirely ignorant of the probabilities associated with these outcomes. Since typical agricultural decision processes focus on the most probable outcome a more modern analysis is needed.

5.3.2 The Modern Analysis

Modern theorists argue that risky decisions must be made in uncertain situations even when the decision maker lacks objective empirical knowledge about outcomes and probability distributions. In these cases the decision maker will formulate subjective estimates of possible outcomes and their probability of occurrence thus making all risky events subsets of uncertain events.

These probabilities reflect the decision maker's degree of belief that a particular event will occur. The quality of information is a consideration when assigning probabilities. To the extent that they are an expression of personal judgment, it is possible, if one decision maker improperly assesses the quality of information, that two

[2] J.R. Anderson, J.L. Dillon, and J.B. Hardaker, Agricultural Decision Analysis, Iowa State University Press, 1977.

decision makers may assign different probabilities to the same event. However, "When two reasonable men have had roughly the same experience with a certain kind of event, they assign roughly the same probability."[3]

5.3.3 Decision Problems

Modern decision theory has three components applicable to risky decision making. First the decision maker must make a choice, or sequence of choices, among various possible courses of action each with an individual payoff. Second, the consequence of any course of action depends on an unpredictable event or "state of nature". The unpredictableness of the state of nature is what generates probabilities that are estimated from the given information. Finally, the decision maker will choose a strategy for experimentation and action that is consistent with his level of risk aversion.[4] To illustrate a decision problem, let $s_1,\ldots,s_m$ represent the m possible states of nature with probabilities $p(s_1),\ldots,p(s_m)$ of occurring. The decision maker's choices are identified as $A_1,\ldots,A_n$ and each choice yields a unique outcome O_{ij} ($i=1,\ldots,m$; $j=1,\ldots,n$) in the respective state of nature. These components are shown in Table 39.

[3] Schlaifer, Robert, Introduction to Statistics for Business Decisions, Mc Graw-Hill Book Co., 1961, p. 17.

[4] Raiffa, Howard, Decision Analysis, Addison-Wesley Publishing Co., Reading, Mass., 1968, Chapter 1.

Table 39: The Decision Table

	Probability of State And Nature State		
	$p(s_1)$	. . . . .	$p(s_m)$
DECISION ALTS.	s_1	. . . . .	s_m
A_1	O_{11}	. . . . .	O_{m1}
.	.		.
.	.		.
.	.		.
A_n	O_{1n}	. . . . .	O_{mn}

Having identified the possible choices, the outcomes, and their likelihoods, the decision maker must compare outcomes. A common denominator must be established in order to make comparisons.

One approach is to maximize expected returns. This approach requires the outcomes to be expressed in dollar equivalents and again the problem of decision maker subjectiveness arises. The ordering of choices for the maximum of expected returns is based on the index expressed as follows:

$$E(A_j) = \sum_{i=1}^{m} p(s_i)y_{ij}$$

where: A_j is the jth choice,

$p(s_i)$ is the probability for the state of nature,

y_{ij} is the dollar equivalent for the outcome of the ith state of nature and the jth choice in Table 39.

5.3.4 Expected Utility Model

Daniel Bernoulli argues that individuals do not care directly about the dollar payoff of a particular decision but that they respond to the utility these dollars provide.[5] An individual's utility is ranked by their cultural environment and psychological attitudes. Due to these factors, the terms risk averse, risk neutral, and risk preferring do not necessarily describe concave, linear and convex utility functions. An individual may have diminishing marginal utility and a preference for risk taking which, when combined, may result in either a concave or a convex utility function.

The von Neumann-Morgenstern Theorem developed an expected utility model in an axiomatic system. The basic axioms assume "rational" behavior among decision makers and allow an ordinal ranking of utility functions. At minimum the conditions include:[6]

Ordering of choices: For any two choice A_1 and A_2 the decision maker either prefers A_1 to A_2, prefers A_2 to A_1, or

[5] Daniel Bernoulli, "Exposition of a New Theory on the Measurement of Risk," Econometrica 22, January 1954, p. 23-36.

[6] Robison, L.J., and P.J. Barry, The Competitive Firm's Response to Risk, Macmillian, New York, 1987, Chapter 2.

is indifferent.

Transitivity of choices: If A_1 is preferred to A_2, and A_2 is preferred to A_3, then A_1 must be preferred to A_3.

Substitution of choices: If A_1 is preferred to A_2, and A_3 is some other choice, then a risky choice $pA_1 + (1-p)A_3$ is preferred to another risky choice $pA_2 + (1-p)A_3$, where p is the probability of occurrence of A_1 or A_2.

Certainty equivalent of choices: If A_1 is preferred to A_2, and A_2 is preferred to A_3, then some probability p exists that the decision maker is indifferent to having A_2 for certain or receiving A_1 with probability p and A_3 with probability 1-p. Thus A_2 is the certainty equivalent of $pA_1 + (1-p)A_3$.

Suppose a decision maker faces a risky event and the possible outcomes of the event are ranked from least desirable to most desirable. By arbitrarily assigning a utility value to the two extremes (for convenience assign 0 the the least desirable and 1 to the most desirable) and ranking the the values between the extremes based on the individual's certainty equivalents for differing likelihoods of gains and losses allows the utility to be ordinally ranked. This procedure assumes that the utility of the certainty equivalent (y_{CE}) equals the expected utility of the risky alternative:

$$U(y_{CE}) = p(1.0) + (1-p)(0) = p.$$

The index of preference of choices can be constructed as:

$$EU(A_1) = p(s_1)U(y_{11}) + \cdot \cdot \cdot + p(s_m)U(y_{m1})$$

$$\vdots$$

$$EU(A_n) = p(s_1)U(y_{1n}) + \cdot \cdot \cdot + p(s_m)U(y_{mn}).$$

Using this analysis, the optimizing principle is for the individual to choose that option which maximizes the expected value of their utility.

5.3.5 Expected Value-Variance Analysis

Expected value-variance (E-V) analysis starts with the assumption that the decision maker seeks to maximize utility where utility if a function of expected income and the variance of expected income. The individual's utility will increase as expected income increases and decreases as the variance of expected income increases. Therefore it can be assumed that a larger expected return is preferred to a smaller return and a smaller variance is preferred to a larger variance. Using this information in graphical form, a boundary which locates the minimum possible variances associated with each possible level of expected income, can be created (see Figure 2).

The E-V boundary can be determined conceptually by computing the expected income and the variance of that income for all alternative choices and connecting those points that represent the minimum possible variance for each expected income level. Points above the E-V boundary may be

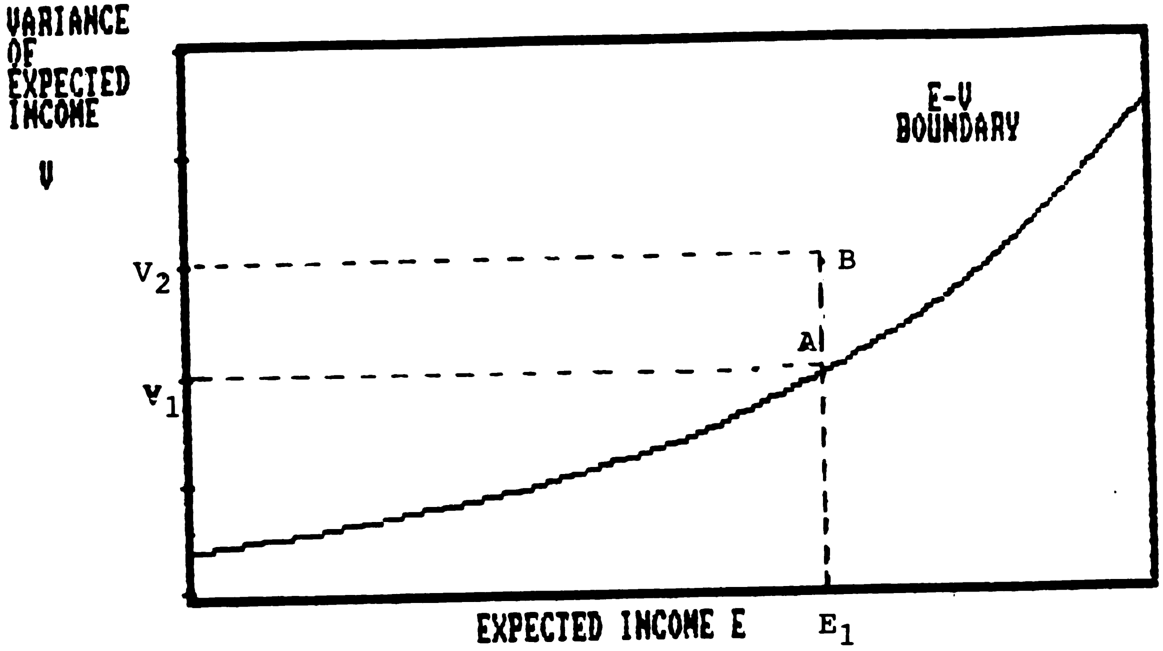


Figure 2: The E-V Boundary

If expected income E_1 is desired, the least variance that must be tolerated is V_1. Point A represents the expected income and variance resulting from a particular farm plan, given the available resoureces. Other points above the E-V boundary may be attainable, in the sense that the resources may be available to undertake farm plans with those particular income and variance values, but because less variance is preferred to more, a farm plan above the boundary will never be undertaken. Point B may be attainable but is irrational because given expected income E_1, V_1 is preferred to V_2.

attainable given the resources to undertake the production plan for a particular income and variance value, however since it is assumed that a producer prefers less variance to more variance, said producer will never use a production plan above the boundary.

5.4 CONCEPTUAL ASSISTANCE TO DECISION MAKING

With price information in the form of probabilities, decision makers will be better able to comprehend the degree of uncertainty associated with the states of nature (i.e. future prices). It is these perceptions about future prices that determine what and how much farmers produce and when they sell their production. With more accurate perceptions about the uncertainty associated with future events, decisions can be made more consistent with farmers' individual objectives and risk preferences.[7]

In the sense that this is an aggregated data model that forecasts the U.S. farm price, the burden of decision alternatives is placed on the user. However, the balance sheet was designed to help in choosing actions or decisions to reduce risk in three areas. The categories are the government program participation decisions, crop mixture decisions and marketing decisions.

The payoff matrix in Table 40 combines all of the components of modern decision analysis.

[7] Halter, A.N., and R. Mason, loc. cit.

Table 40: The Payoff Matrix

	STATES OF NATURE				
DECISION ALTS.	LOW PRICE S_1	S_2	MEAN PRICE S_3	S_4	HIGH PRICE S_5
d_1:Participate in the Government program	+	+?	?	-?	-
d_2:Increase the acres committed to corn vs. an alternative	-	-?	?	+?	+
d_3:Increase the acres committed to an alternative vs. corn	+	+?	?	-?	-
d_4:Repay loan & sell	-	-?	?	+?	+
d_5:Default loan	+	+?	?	-?	-
d_6:Forward contract the mean price S_3	+	+?	?	-?	-
d_7:Hedge (basis narrows)	+	+?	?	+?	+
d_8:Spot market	-	-?	?	+?	+

The decision alternatives accompany the categories discussed above. The states of nature (i.e. the price range and associated probabilities) are provided by the balance sheet. The outcomes must be assigned by the individual user but for conceptual assistance, the outcomes should range from favorable (+) to unfavorable (-) levels of utility.

APPENDICES

Appendix 1

Data Source Considerations

The data for corn consumption of each animal species was obtained from the USDA FEED OUTLOOK AND SITUATION YEARBOOK. The data provided here is in million metric tons and, to be consistent with other data, was converted to bushel equivalents.

The animal inventory data was obtained from AGRICULTURAL STATISTICS. For most of the animal species classes, the inventory values are stock variables which assumes that each animal on inventory day will be carried throughout the year. However, the broiler and turkey values are production flow variables and they measure consumption of all birds produced in the 12 month period.

The price data was obtained from AGRICULTURAL PRICES and the USDA OUTLOOK AND SITUATION REPORT. The milk, milk cow, steer & heifer, and beef cow prices are all on a January to December market year using average of the year in standard units for each species. Egg, broiler, turkey, and hog prices are on a December to November market year.

The data used for the export section was collected from three sources. All of the commodity data (i.e. regional consumption, production, ending stocks, net imports and exports) came from the FOREIGN AGRICULTURE SERVICE and specifically, the FOREIGN PRODUCTION, SUPPLY, AND DISTRIBUTION OF AGRICULTURAL COMMODITIES. All price data comes from the FOREIGN AGRICULTURAL TRADE OF THE UNITED STATES. All macro economic data (i.e. consumer price index, population, and exchange rates) comes from the INTERNATIONAL FINANCIAL STATISTICS.

The marketing year of feed grain in this model is October 1 to September 30. It is important to note that future updates of the model will need to reflect the change in the marketing year from September 1 to August 31, (starting in calendar year 1986) however, the change in data collection should not significantly affect this models' forecasting reliability.

Appendix 2

International Regional Groupings

Region	Countries
United States	United States
Residual Exporters	
Canada	Canada
Australia	Australia
Argentina	Argentina
Importers	
Brazil	Brazil
Developed Markets	United Kingdom, Belgium, Denmark, Netherlands, Finland, Luxembourg, Portugal, Ireland, Greece, Iceland, Austria, France, West Germany, Italy, Switzerland, Sweden, Norway, Malta, Spain, Japan, South Africa
Soviet Bloc	Albania, Bulgaria, East Germany, Hungary, Poland, Romania, Yugoslavia, Czechoslovakia, USSR
China	China
Oil-Exporting LDC's	Algeria, Ecuador, Indonesia, Iran, Iraq, Libya, Oman, Saudia Arabia, Venezuela, Nigeria, United Arab Emirates, Kuwait

International Regional Groupings (cont'd)

Region	Countries
Importers	
Newly Industrialized	Hong Kong, Singapore, Malaysia, Taiwan, South Korea
LDC's	all others

Appendix 3

Supply/Demand Balance Sheet Documentation

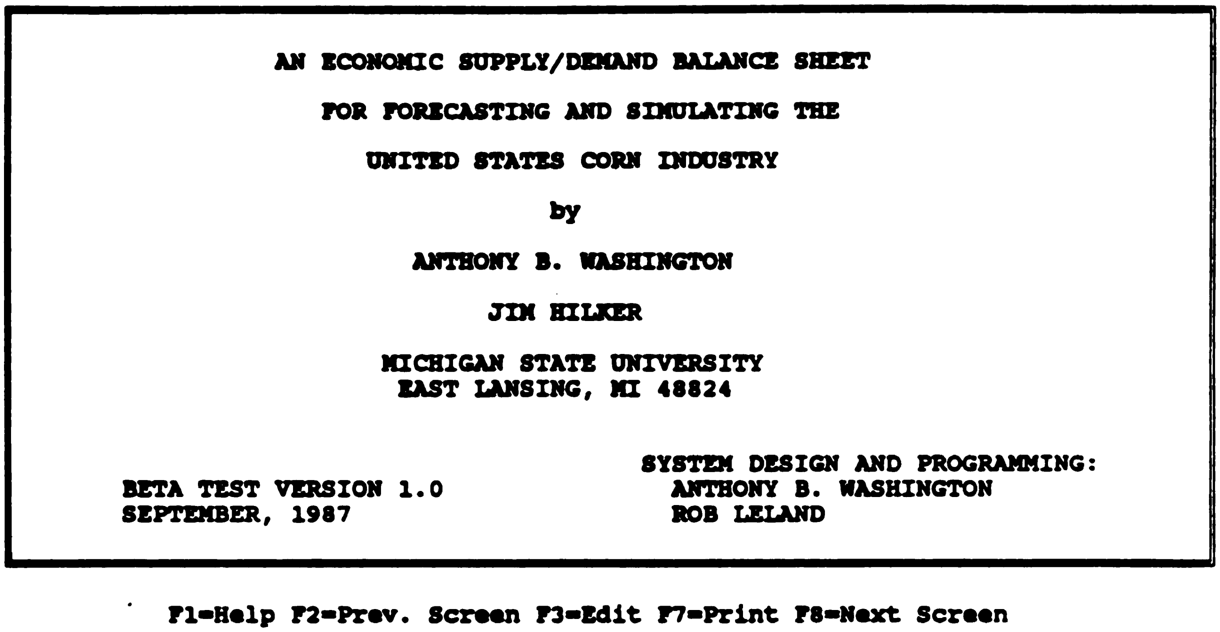
A.1 Getting Started

To use the balance sheet, you will need an IBM or compatible microcomputer with at least 256K of RAM memory. At least one disk drive will be needed and an 80-column monitor.

The balance sheet is started by placing the floppy disk into the boot up drive and powering on your computer. When you see the DOS prompt, type "**corn**" and press return. You will see some resident application files loaded into memory and then the title screen. On the bottom of the title screen are key codes for performing various operations. To move forward, press F8.

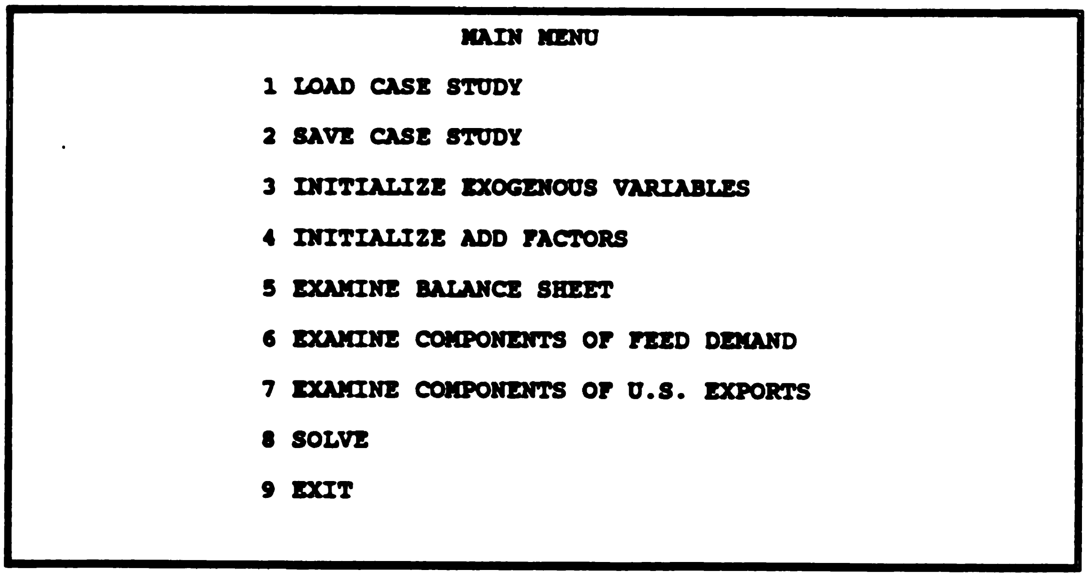
After pressing F8 on the title screen, the main menu appears. To move through the menu, either select the number in front of the option or use the direction keys to highlight an option and press return. Depending on the main menu selection, you will see another menu or the screen selected.

The display of the balance sheet is arranged into four columns of data. The fourth column is always the forecasted



```
                AN ECONOMIC SUPPLY/DEMAND BALANCE SHEET
                  FOR FORECASTING AND SIMULATING THE
                     UNITED STATES CORN INDUSTRY
                                 by
                        ANTHONY B. WASHINGTON
                             JIM HILKER
                      MICHIGAN STATE UNIVERSITY
                       EAST LANSING, MI 48824

                                       SYSTEM DESIGN AND PROGRAMMING:
    BETA TEST VERSION 1.0                ANTHONY B. WASHINGTON
    SEPTEMBER, 1987                      ROB LELAND

    F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 3: Title Screen



```
                     MAIN MENU
          1 LOAD CASE STUDY
          2 SAVE CASE STUDY
          3 INITIALIZE EXOGENOUS VARIABLES
          4 INITIALIZE ADD FACTORS
          5 EXAMINE BALANCE SHEET
          6 EXAMINE COMPONENTS OF FEED DEMAND
          7 EXAMINE COMPONENTS OF U.S. EXPORTS
          8 SOLVE
          9 EXIT
```

Figure 4: Main Menu

vector while column three holds the lagged vector which the forecast is based. On some screens there are fields for yes/no responses to the override feature. If you choose to override an estimated value, the override value column is highlighted and the new value can be inputted.

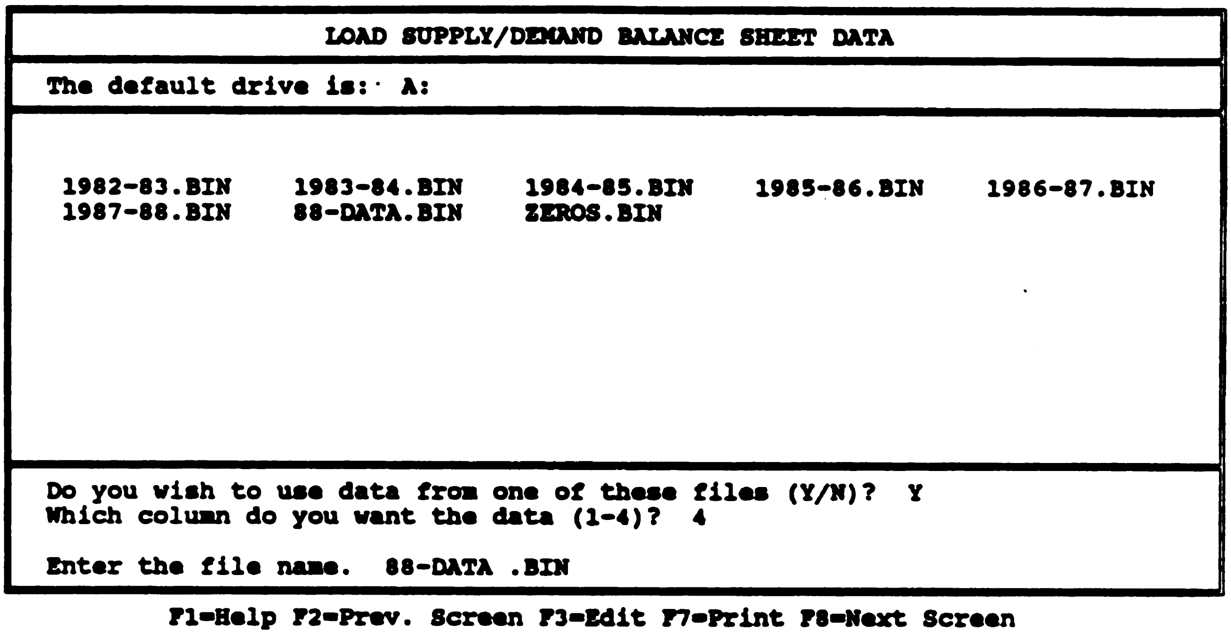
Each main menu option will be discussed below.

A.2 Loading and Saving Data

A.2.1 Load Case Study

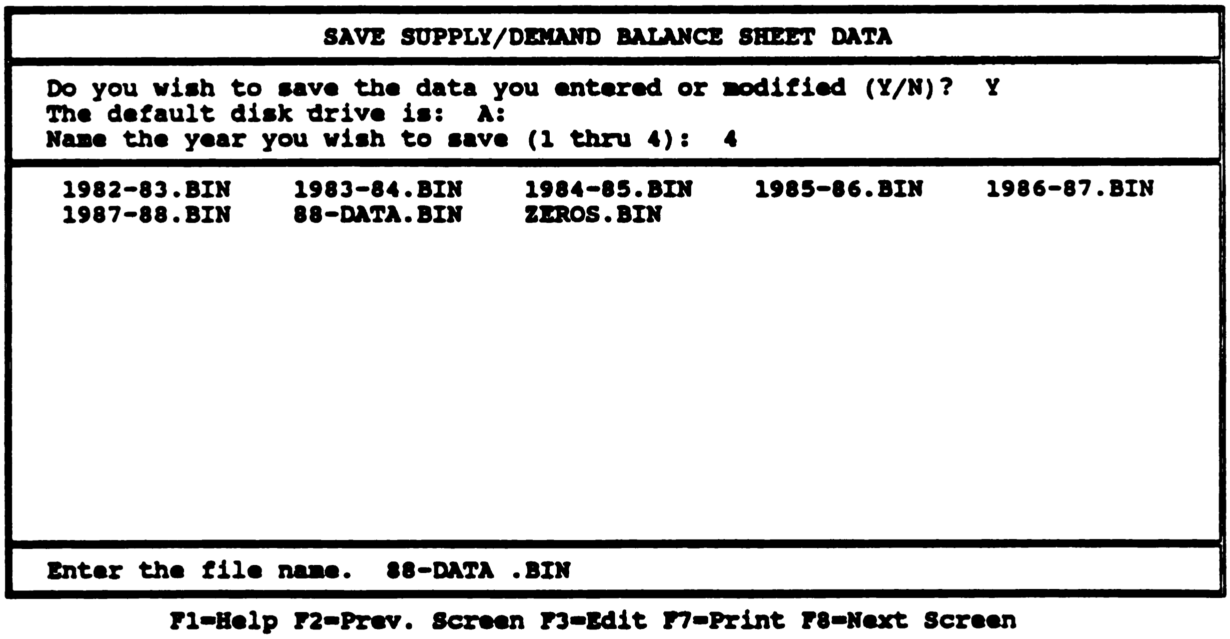
When the Load screen appears, you are asked to input the default drive. The historical data is saved on the balance sheet disk and therefore the return key is entered. The data files appear on the screen and you are asked if any of these files will be used. You are then prompted to enter which column the data should go. You may enter a file name and by striking return, the vector is loaded. The file extension is not needed. A message appears at the bottom of the screen to confirm the transfer of the file.

You are required to load a lagged vector of historical data to prevent any errors while running the calculations. Columns one and two can be used to load historical data or can be used to hold a particular scenario for on screen comparisons. In order to update or edit a vector with new information, it must be loaded into the fourth column.



```
LOAD SUPPLY/DEMAND BALANCE SHEET DATA

The default drive is:· A:

1982-83.BIN    1983-84.BIN    1984-85.BIN    1985-86.BIN    1986-87.BIN
1987-88.BIN    88-DATA.BIN    ZEROS.BIN

Do you wish to use data from one of these files (Y/N)?  Y
Which column do you want the data (1-4)?  4

Enter the file name.  88-DATA .BIN
       F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 5: Load Screen



```
SAVE SUPPLY/DEMAND BALANCE SHEET DATA

Do you wish to save the data you entered or modified (Y/N)?  Y
The default disk drive is:  A:
Name the year you wish to save (1 thru 4):  4

1982-83.BIN    1983-84.BIN    1984-85.BIN    1985-86.BIN    1986-87.BIN
1987-88.BIN    88-DATA.BIN    ZEROS.BIN

Enter the file name.  88-DATA .BIN
       F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 6: Save Screen

A.2.2 Save Case Study

The Save screen is much like the Load screen. You are asked if any of the data vectors should be saved. You are then prompted to input which column should be saved and then a file name. Again, the file extension is not needed. Since editions are made to the fourth column, it is generally the one to save. A message at the bottom of the screen confirms the transfer of the file.

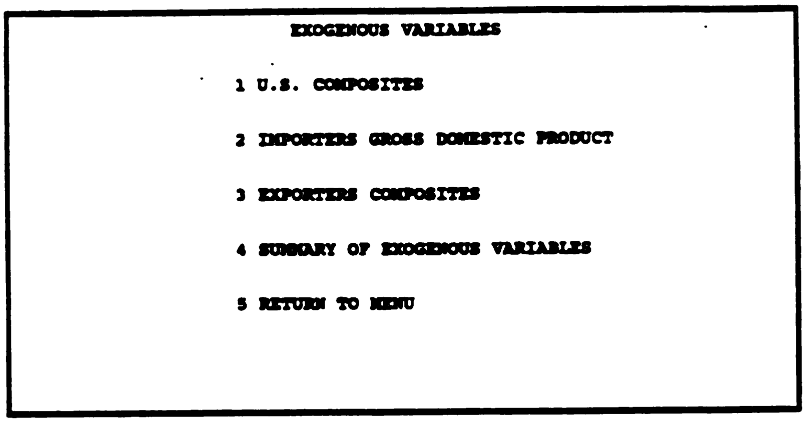
A.3 Initialize Exogenous Variables

In order to forecast the next time period, you must input the relevant exogenous and predetermined variables.

The Summary Of Exogenous Variables displays all the variables used and sets the year of the forecast vector. You begin by entering the marketing year you wish to forecast and then one-by-one entering the exogenous variables. When you get to the bottom of the screen, press F8 to continue.

The other menu options available for initializing the exogenous variables are essentially work-sheets which carry the generated value to the exogenous summary. If you wish, the work-sheets can be bypassed and all relevant inputs done in the exogenous variable summary.

The U.S. Composites menu option allows you to determine the gross revenue from participating in the government program and the food, seed, and industrial use. The U.S. acres planted to corn is a function of (among others) net



EXOGENOUS VARIABLES

1 U.S. COMPOSITES

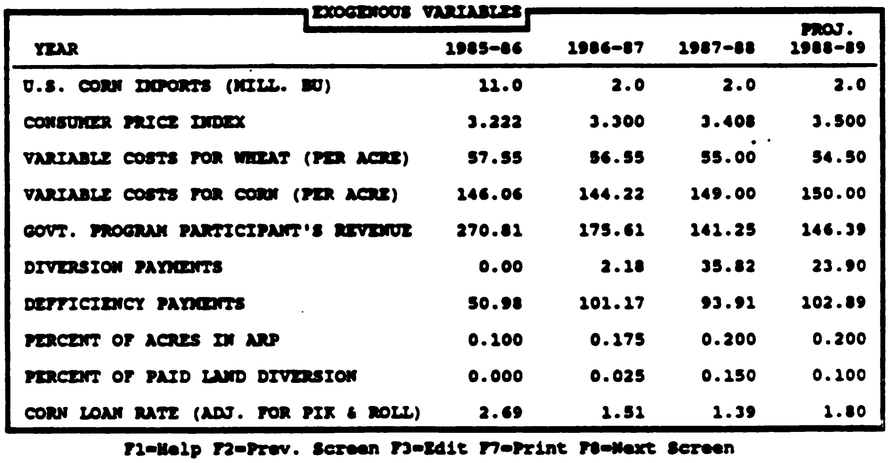
2 IMPORTERS GROSS DOMESTIC PRODUCT

3 EXPORTERS COMPOSITES

4 SUMMARY OF EXOGENOUS VARIABLES

5 RETURN TO MENU

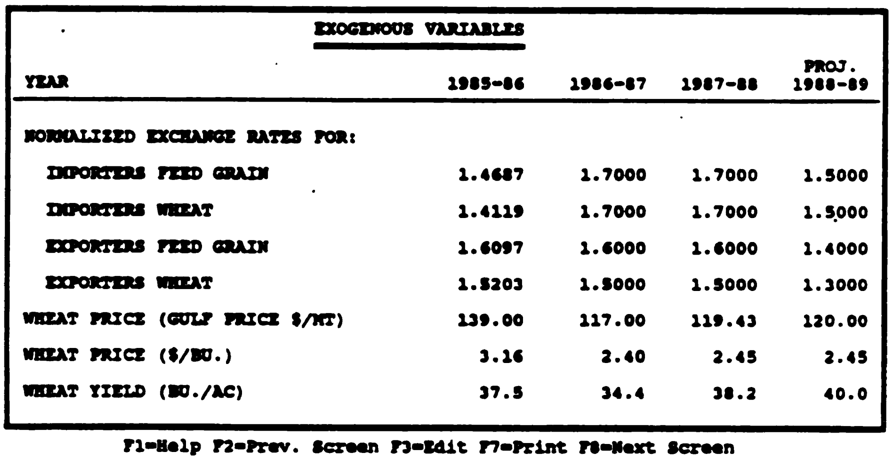
Figure 7: Exogenous Variable Menu



EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
U.S. CORN IMPORTS (MILL. BU)	11.0	2.0	2.0	2.0
CONSUMER PRICE INDEX	3.222	3.300	3.408	3.500
VARIABLE COSTS FOR WHEAT (PER ACRE)	57.55	56.55	55.00	54.50
VARIABLE COSTS FOR CORN (PER ACRE)	146.06	144.22	149.00	150.00
GOVT. PROGRAM PARTICIPANT'S REVENUE	270.81	175.61	141.25	146.39
DIVERSION PAYMENTS	0.00	2.18	35.82	23.90
DEFFICIENCY PAYMENTS	50.98	101.17	93.91	102.89
PERCENT OF ACRES IN ARP	0.100	0.175	0.200	0.200
PERCENT OF PAID LAND DIVERSION	0.000	0.025	0.150	0.100
CORN LOAN RATE (ADJ. FOR PIK & ROLL)	2.69	1.51	1.39	1.80

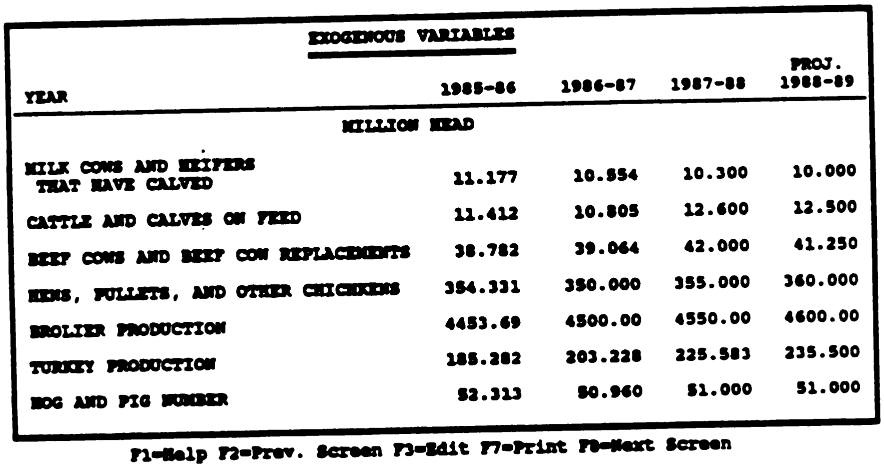
Figure 8: Exogenous Variable Screen 1



EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
NORMALIZED EXCHANGE RATES FOR:				
IMPORTERS FEED GRAIN	1.4687	1.7000	1.7000	1.5000
IMPORTERS WHEAT	1.4119	1.7000	1.7000	1.5000
EXPORTERS FEED GRAIN	1.6097	1.6000	1.6000	1.4000
EXPORTERS WHEAT	1.5203	1.5000	1.5000	1.3000
WHEAT PRICE (GULF PRICE $/MT)	139.00	117.00	119.43	120.00
WHEAT PRICE ($/BU.)	3.16	2.40	2.45	2.45
WHEAT YIELD (BU./AC)	37.5	34.4	38.2	40.0

Figure 9: Exogenous Variable Screen 2

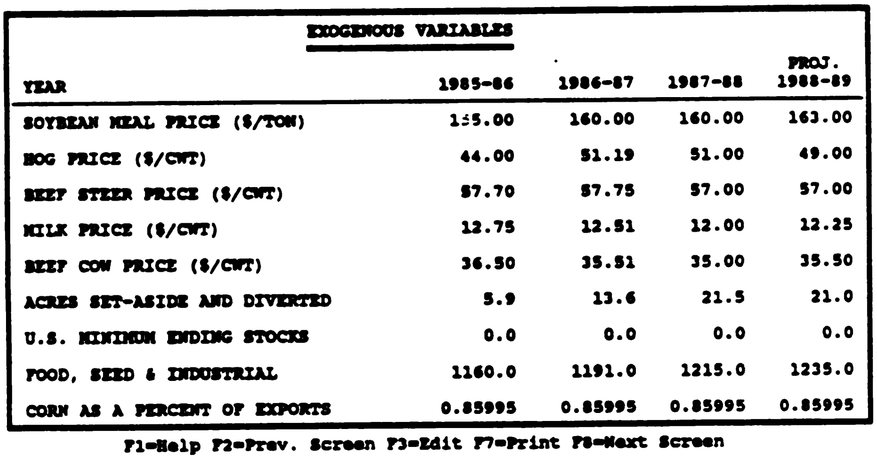


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
MILLION HEAD				
MILK COWS AND HEIFERS THAT HAVE CALVED	11.177	10.554	10.300	10.000
CATTLE AND CALVES ON FEED	11.412	10.805	12.600	12.500
BEEF COWS AND BEEF COW REPLACEMENTS	38.782	39.064	42.000	41.250
HENS, PULLETS, AND OTHER CHICKENS	354.331	350.000	355.000	360.000
BROLIER PRODUCTION	4453.69	4500.00	4550.00	4600.00
TURKEY PRODUCTION	185.282	203.228	225.583	235.500
HOG AND PIG NUMBER	52.313	50.960	51.000	51.000

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 10: Exogenous Variable Screen 3

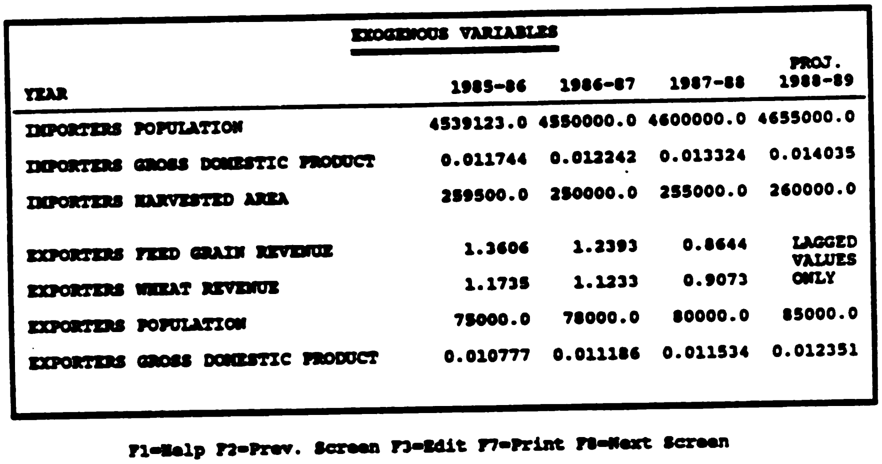


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
SOYBEAN MEAL PRICE ($/TON)	155.00	160.00	160.00	163.00
HOG PRICE ($/CWT)	44.00	51.19	51.00	49.00
BEEF STEER PRICE ($/CWT)	57.70	57.75	57.00	57.00
MILK PRICE ($/CWT)	12.75	12.51	12.00	12.25
BEEF COW PRICE ($/CWT)	36.50	35.51	35.00	35.50
ACRES SET-ASIDE AND DIVERTED	5.9	13.6	21.5	21.0
U.S. MINIMUM ENDING STOCKS	0.0	0.0	0.0	0.0
FOOD, SEED & INDUSTRIAL	1160.0	1191.0	1215.0	1235.0
CORN AS A PERCENT OF EXPORTS	0.85995	0.85995	0.85995	0.85995

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 11: Exogenous Variable Screen 4

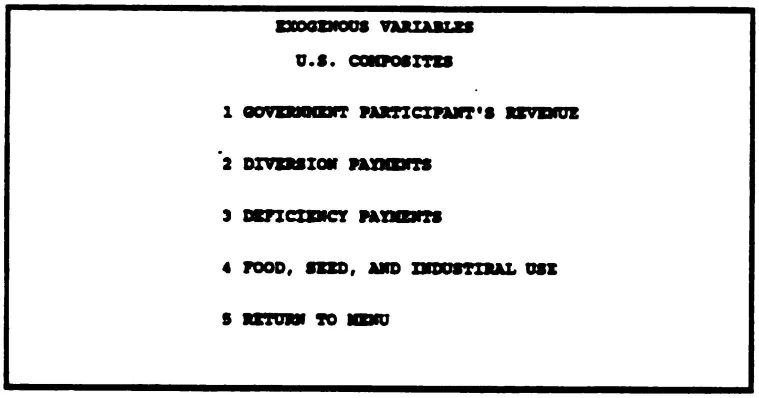


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
IMPORTERS POPULATION	4539123.0	4550000.0	4600000.0	4655000.0
IMPORTERS GROSS DOMESTIC PRODUCT	0.011744	0.012242	0.013324	0.014035
IMPORTERS HARVESTED AREA	259500.0	250000.0	255000.0	260000.0
EXPORTERS FEED GRAIN REVENUE	1.3606	1.2393	0.8644	LAGGED VALUES ONLY
EXPORTERS WHEAT REVENUE	1.1735	1.1233	0.9073	
EXPORTERS POPULATION	75000.0	78000.0	80000.0	85000.0
EXPORTERS GROSS DOMESTIC PRODUCT	0.010777	0.011186	0.011534	0.012351

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 12: Exogenous Variable Screen 5



EXOGENOUS VARIABLES

U.S. COMPOSITES

1 GOVERNMENT PARTICIPANT'S REVENUE

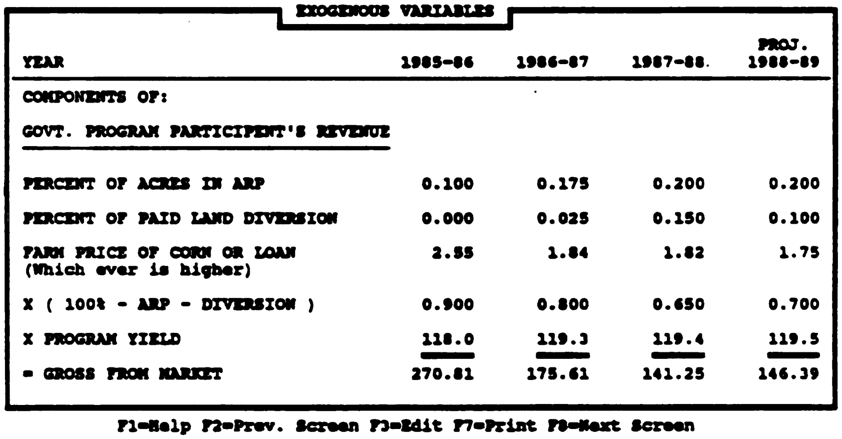
2 DIVERSION PAYMENTS

3 DEFICIENCY PAYMENTS

4 FOOD, SEED, AND INDUSTIRAL USE

5 RETURN TO MENU

Figure 13: U.S. Composites Sub Menu

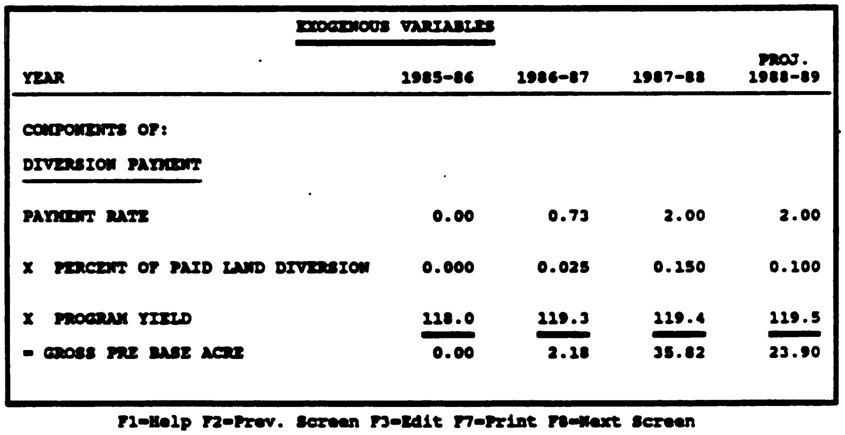


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
COMPONENTS OF:				
GOVT. PROGRAM PARTICIPENT'S REVENUE				
PERCENT OF ACRES IN ARP	0.100	0.175	0.200	0.200
PERCENT OF PAID LAND DIVERSION	0.000	0.025	0.150	0.100
FARM PRICE OF CORN OR LOAN (Which ever is higher)	2.55	1.84	1.82	1.75
X (100% - ARP - DIVERSION)	0.900	0.800	0.650	0.700
X PROGRAM YIELD	118.0	119.3	119.4	119.5
= GROSS FROM MARKET	270.81	175.61	141.25	146.39

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 14: Govt. Program Participant's Revenue Screen

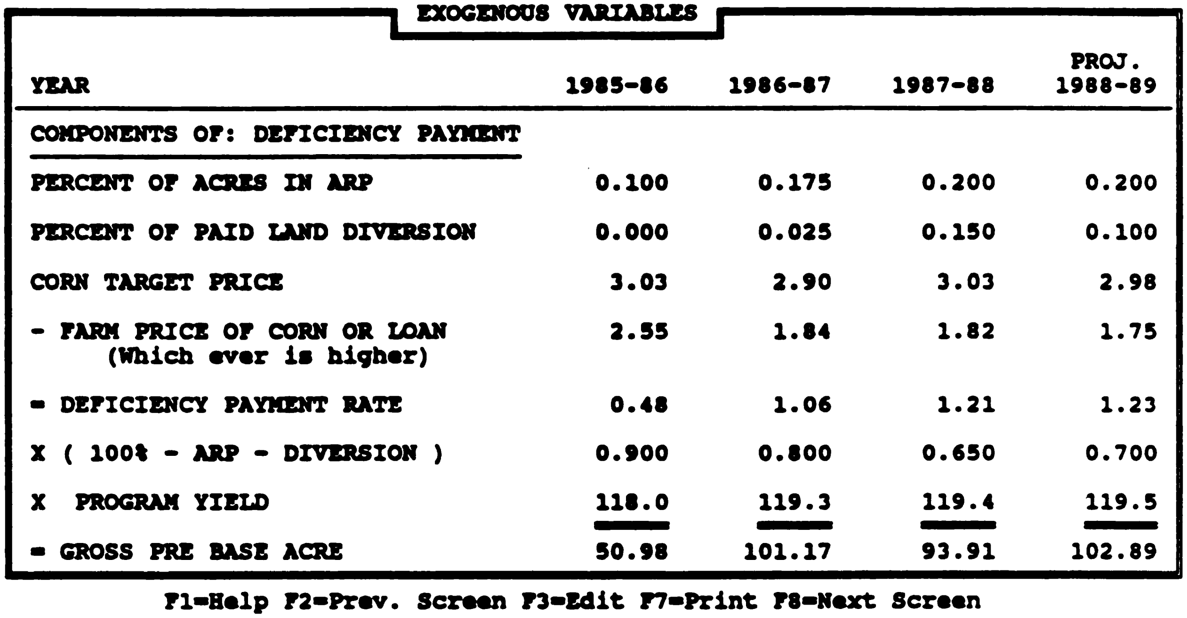


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
COMPONENTS OF:				
DIVERSION PAYMENT				
PAYMENT RATE	0.00	0.73	2.00	2.00
X PERCENT OF PAID LAND DIVERSION	0.000	0.025	0.150	0.100
X PROGRAM YIELD	118.0	119.3	119.4	119.5
= GROSS PRE BASE ACRE	0.00	2.18	35.82	23.90

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 15: Diversion Payments Screen

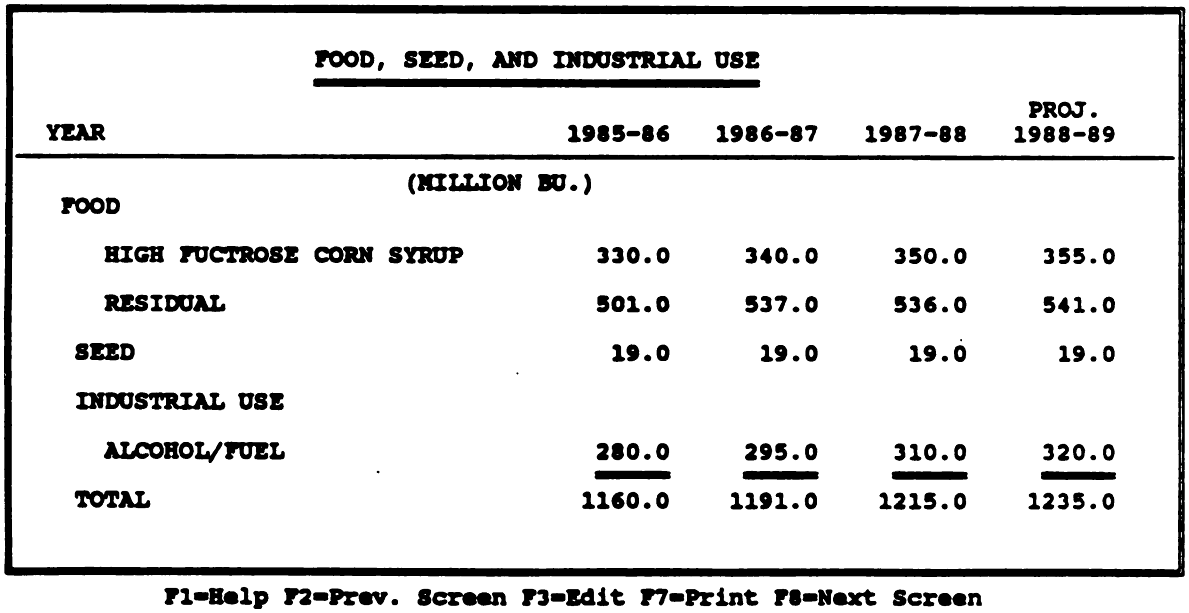


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
COMPONENTS OF: DEFICIENCY PAYMENT				
PERCENT OF ACRES IN ARP	0.100	0.175	0.200	0.200
PERCENT OF PAID LAND DIVERSION	0.000	0.025	0.150	0.100
CORN TARGET PRICE	3.03	2.90	3.03	2.98
- FARM PRICE OF CORN OR LOAN (Which ever is higher)	2.55	1.84	1.82	1.75
= DEFICIENCY PAYMENT RATE	0.48	1.06	1.21	1.23
X (100% - ARP - DIVERSION)	0.900	0.800	0.650	0.700
X PROGRAM YIELD	118.0	119.3	119.4	119.5
= GROSS PRE BASE ACRE	50.98	101.17	93.91	102.89

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 16: Deficiency Payments Screen



FOOD, SEED, AND INDUSTRIAL USE

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
(MILLION BU.)				
FOOD				
HIGH FUCTROSE CORN SYRUP	330.0	340.0	350.0	355.0
RESIDUAL	501.0	537.0	536.0	541.0
SEED	19.0	19.0	19.0	19.0
INDUSTRIAL USE				
ALCOHOL/FUEL	280.0	295.0	310.0	320.0
TOTAL	1160.0	1191.0	1215.0	1235.0

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 17: Food, Seed, And Industrial Use Screen

corn revenue form government payments. The work-sheets help you calculate revenues from the market, diversion payments and any deficiency payment.

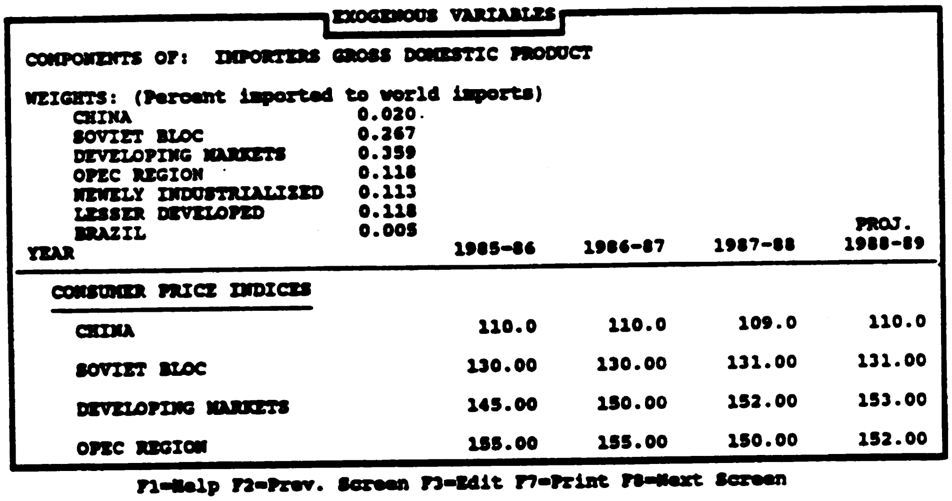
The food, seed, and industrial work-sheet disaggregates FSI use into high fructose corn syrup, seed, industrial use and a residual. This disaggregation will allow you to more accurately determine the total FSI use given the conceptual constraints.

The Importers Gross Domestic Product menu option is a work-sheet that requires the percentage (or weights) that each area in the region imports. These weights must be changed to represent a change in an area's importance to the total regional imports. Each area CPI and gross domestic product is needed to calculate the regional gross domestic product.

The Exporters Composites menu option shows work-sheets which help you to calculate the exporters crop revenue and exporters gross domestic product. The balance sheet uses lagged values for exporters crop revenue, however this menu option requires user input in order to have values when the vector being edited is used in the lagged vector column.

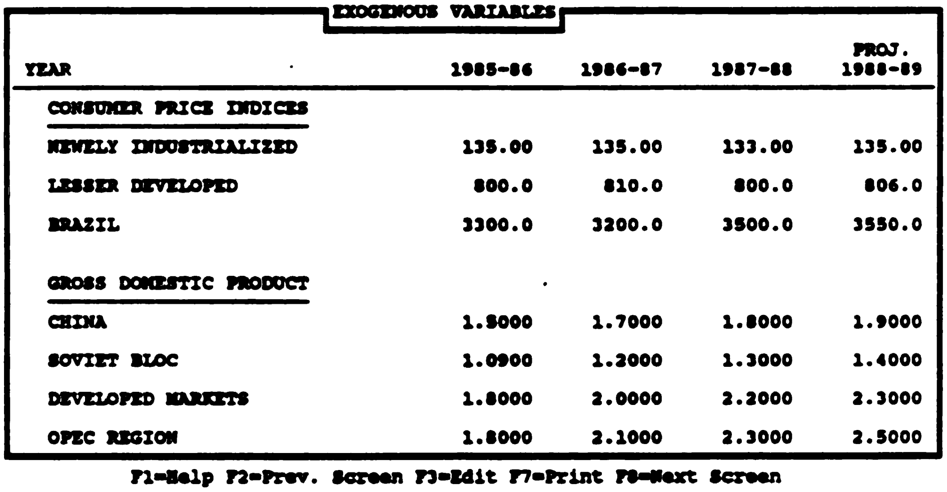
The Exporter Gross Domestic Product menu option is a work-sheet much like the Importers Gross Domestic Product. It requires the use of weights for area importance as well as CPI's and gross domestic product.

Once these variables are entered, they can be saved in a vector for future use or scenario analysis.



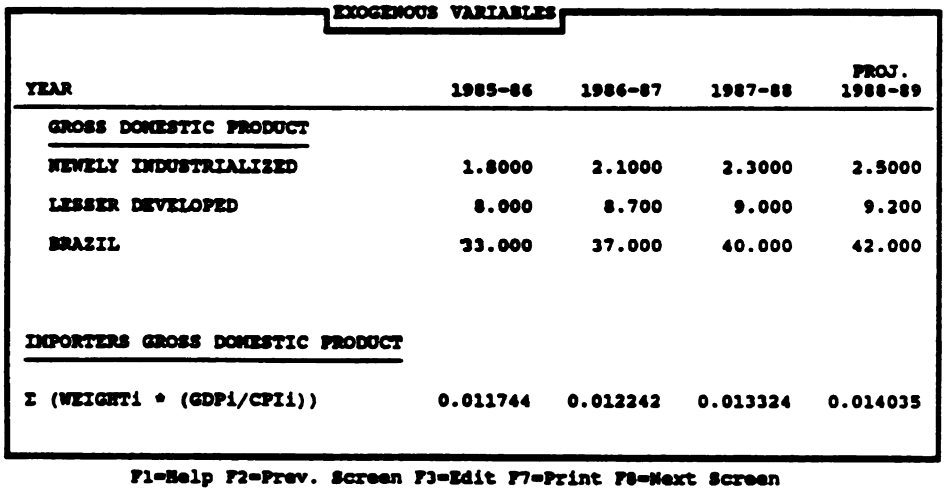
```
                              EXOGENOUS VARIABLES

COMPONENTS OF:  IMPORTERS GROSS DOMESTIC PRODUCT

WEIGHTS: (Percent imported to world imports)
    CHINA                       0.020
    SOVIET BLOC                 0.267
    DEVELOPING MARKETS          0.359
    OPEC REGION                 0.118
    NEWELY INDUSTRIALIZED       0.113
    LESSER DEVELOPED            0.118
    BRAZIL                      0.005                                        PROJ.
YEAR                                     1985-86    1986-87    1987-88    1988-89
----------------------------------------------------------------------------------
  CONSUMER PRICE INDICES
  ______________________

    CHINA                                 110.0      110.0      109.0      110.0

    SOVIET BLOC                           130.00     130.00     131.00     131.00

    DEVELOPING MARKETS                    145.00     150.00     152.00     153.00

    OPEC REGION                           155.00     155.00     150.00     152.00

          F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 18: Importers Gross Domestic Product Screen 1



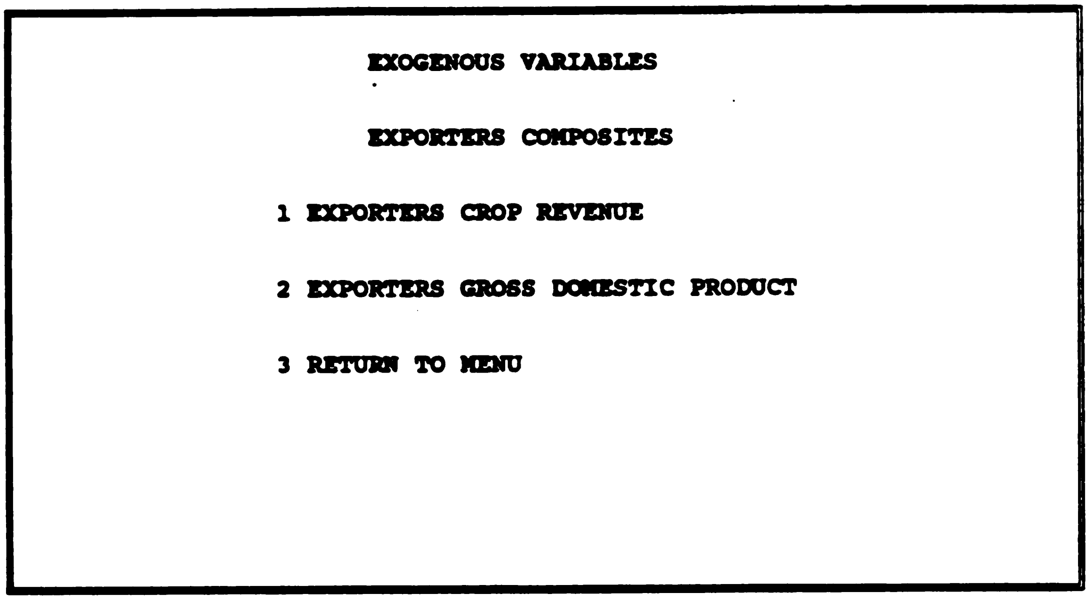
```
                              EXOGENOUS VARIABLES
                                                                             PROJ.
YEAR                                     1985-86    1986-87    1987-88    1988-89
----------------------------------------------------------------------------------
  CONSUMER PRICE INDICES
  ______________________
  NEWELY INDUSTRIALIZED                   135.00     135.00     133.00     135.00

  LESSER DEVELOPED                        800.0      810.0      800.0      806.0

  BRAZIL                                 3300.0     3200.0     3500.0     3550.0

  GROSS DOMESTIC PRODUCT
  ______________________
  CHINA                                   1.5000     1.7000     1.8000     1.9000

  SOVIET BLOC                             1.0900     1.2000     1.3000     1.4000

  DEVELOPED MARKETS                       1.8000     2.0000     2.2000     2.3000

  OPEC REGION                             1.8000     2.1000     2.3000     2.5000

          F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 19: Importers Gross Domestic Product Screen 2



```
                              EXOGENOUS VARIABLES
                                                                             PROJ.
YEAR                                     1985-86    1986-87    1987-88    1988-89
----------------------------------------------------------------------------------
  GROSS DOMESTIC PRODUCT
  ______________________
  NEWELY INDUSTRIALIZED                   1.8000     2.1000     2.3000     2.5000

  LESSER DEVELOPED                        8.000      8.700      9.000      9.200

  BRAZIL                                 33.000     37.000     40.000     42.000



IMPORTERS GROSS DOMESTIC PRODUCT
________________________________

Σ (WEIGHTi * (GDPi/CPIi))               0.011744   0.012242   0.013324   0.014035


          F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 20: Importers Gross Domestic Product Screen 3



EXOGENOUS VARIABLES

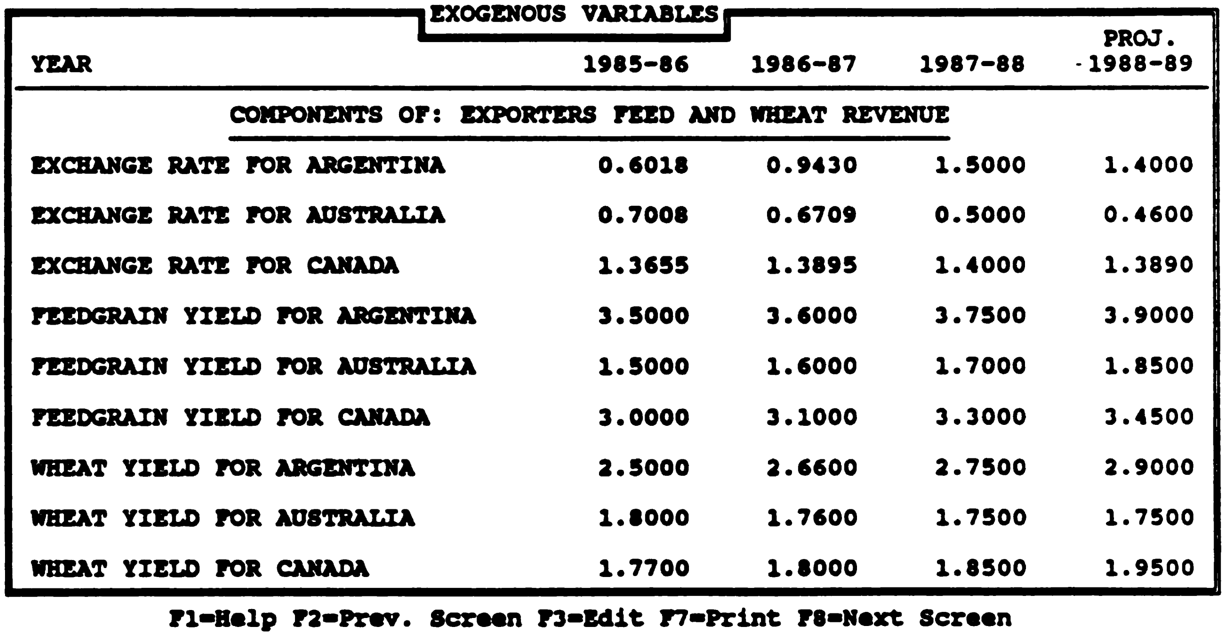
EXPORTERS COMPOSITES

1 EXPORTERS CROP REVENUE

2 EXPORTERS GROSS DOMESTIC PRODUCT

3 RETURN TO MENU

Figure 21: Exporters Composites Sub Menu

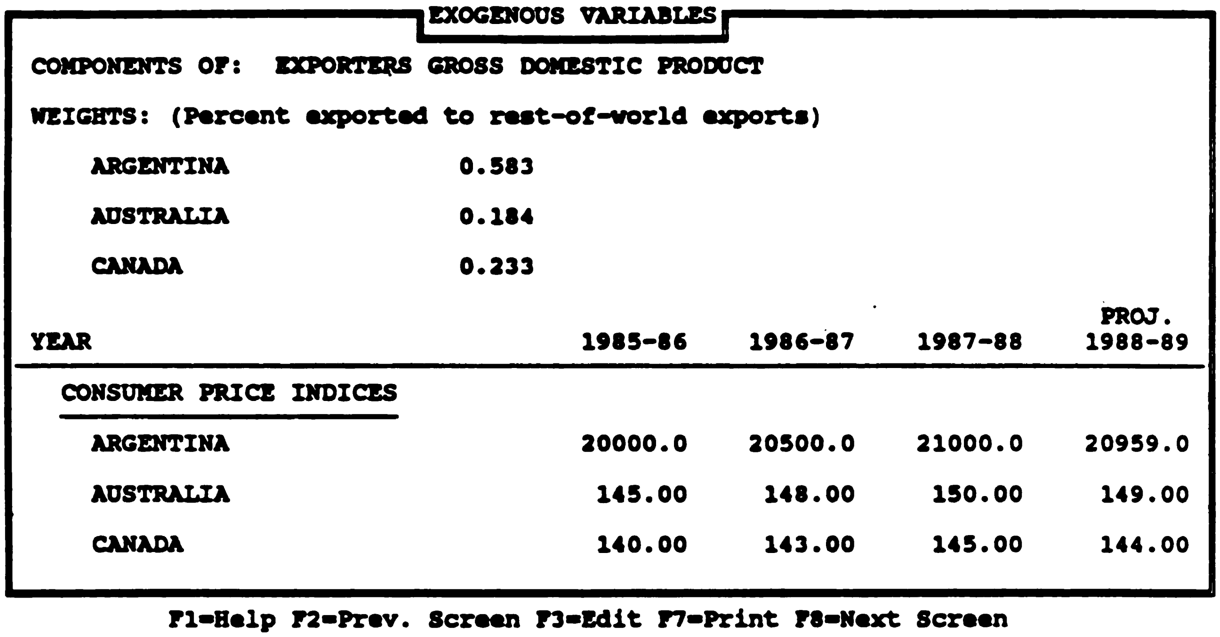


EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
COMPONENTS OF: EXPORTERS FEED AND WHEAT REVENUE				
EXCHANGE RATE FOR ARGENTINA	0.6018	0.9430	1.5000	1.4000
EXCHANGE RATE FOR AUSTRALIA	0.7008	0.6709	0.5000	0.4600
EXCHANGE RATE FOR CANADA	1.3655	1.3895	1.4000	1.3890
FEEDGRAIN YIELD FOR ARGENTINA	3.5000	3.6000	3.7500	3.9000
FEEDGRAIN YIELD FOR AUSTRALIA	1.5000	1.6000	1.7000	1.8500
FEEDGRAIN YIELD FOR CANADA	3.0000	3.1000	3.3000	3.4500
WHEAT YIELD FOR ARGENTINA	2.5000	2.6600	2.7500	2.9000
WHEAT YIELD FOR AUSTRALIA	1.8000	1.7600	1.7500	1.7500
WHEAT YIELD FOR CANADA	1.7700	1.8000	1.8500	1.9500

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 22: Exporter Crop Revenue Screen



EXOGENOUS VARIABLES

COMPONENTS OF: EXPORTERS GROSS DOMESTIC PRODUCT

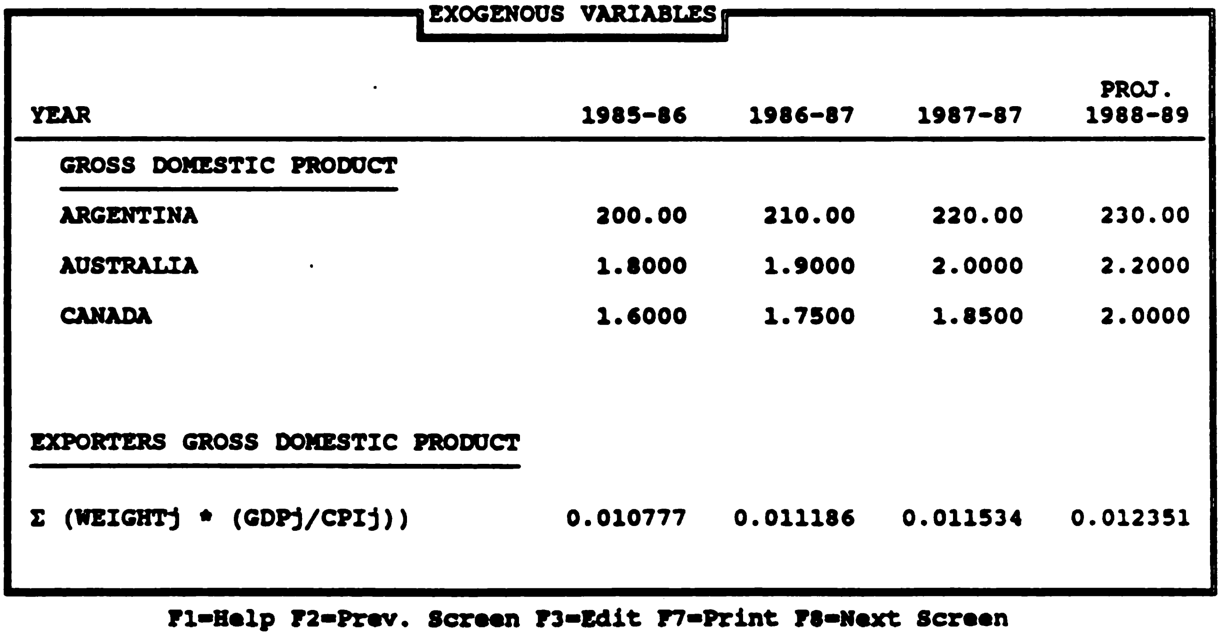
WEIGHTS: (Percent exported to rest-of-world exports)

ARGENTINA	0.583
AUSTRALIA	0.184
CANADA	0.233

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
CONSUMER PRICE INDICES				
ARGENTINA	20000.0	20500.0	21000.0	20959.0
AUSTRALIA	145.00	148.00	150.00	149.00
CANADA	140.00	143.00	145.00	144.00

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 23: Exporter Gross Domestic Product Screen 1



EXOGENOUS VARIABLES

YEAR	1985-86	1986-87	1987-87	PROJ. 1988-89
GROSS DOMESTIC PRODUCT				
ARGENTINA	200.00	210.00	220.00	230.00
AUSTRALIA	1.8000	1.9000	2.0000	2.2000
CANADA	1.6000	1.7500	1.8500	2.0000
EXPORTERS GROSS DOMESTIC PRODUCT				
Σ (WEIGHTj * (GDPj/CPIj))	0.010777	0.011186	0.011534	0.012351

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 24: Exporter Gross Domestic Product Screen 2

A.4 Initialize Add Factors

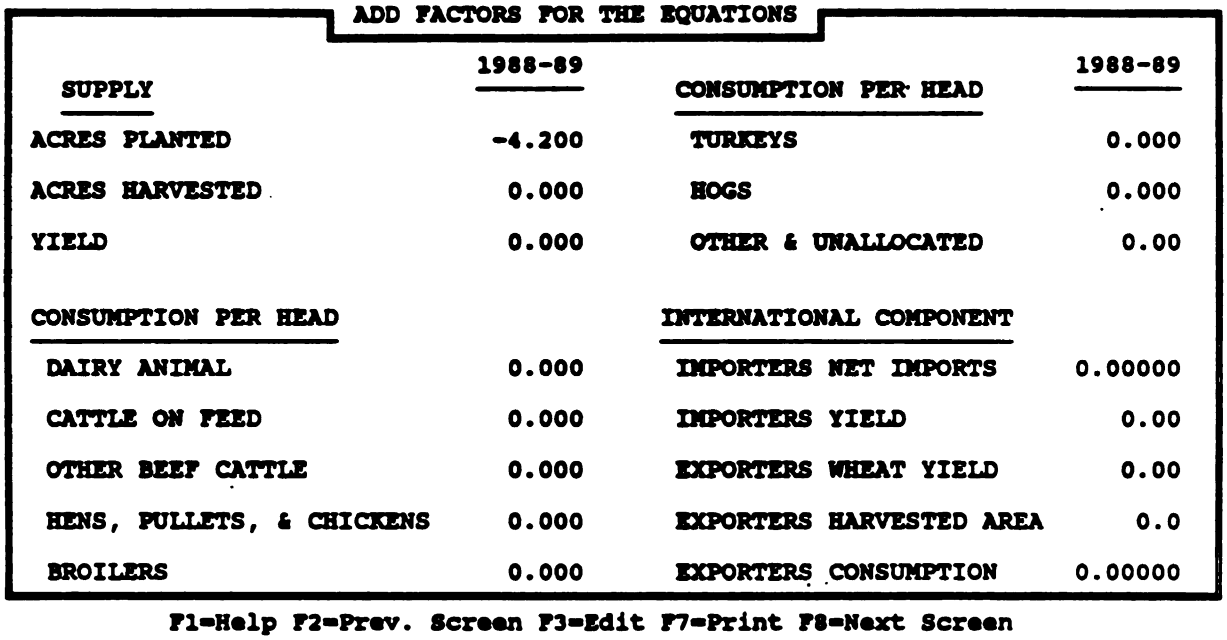
Each of the estimated equations has a mechanism which can be used the adjust the intercept of the equation. By using a positive add factor the function will return a higher value and vice versa for a negative add factor. This feature is added so you can adjust each equation with a posteriori information.

A.5 Examine Balance Sheet

You can choose to examine screens which show a balance sheet similar to that shown in Table 1 (page 4). This menu option displays three screens which show U.S. supply and utilization factors. The first screen shows strictly the U.S. production situation. The second screen shows the U.S. supply as well as the estimated and predetermined uses of corn. This screen also shows the estimated price which is calculated when supply and usage are equated. The final screen shows information that relates to the level of ending stocks. On each of these screens there are places to override the estimated values presented. If the override is used, the model will solve using these new values. This feature allows you to do "what if" scenarios.

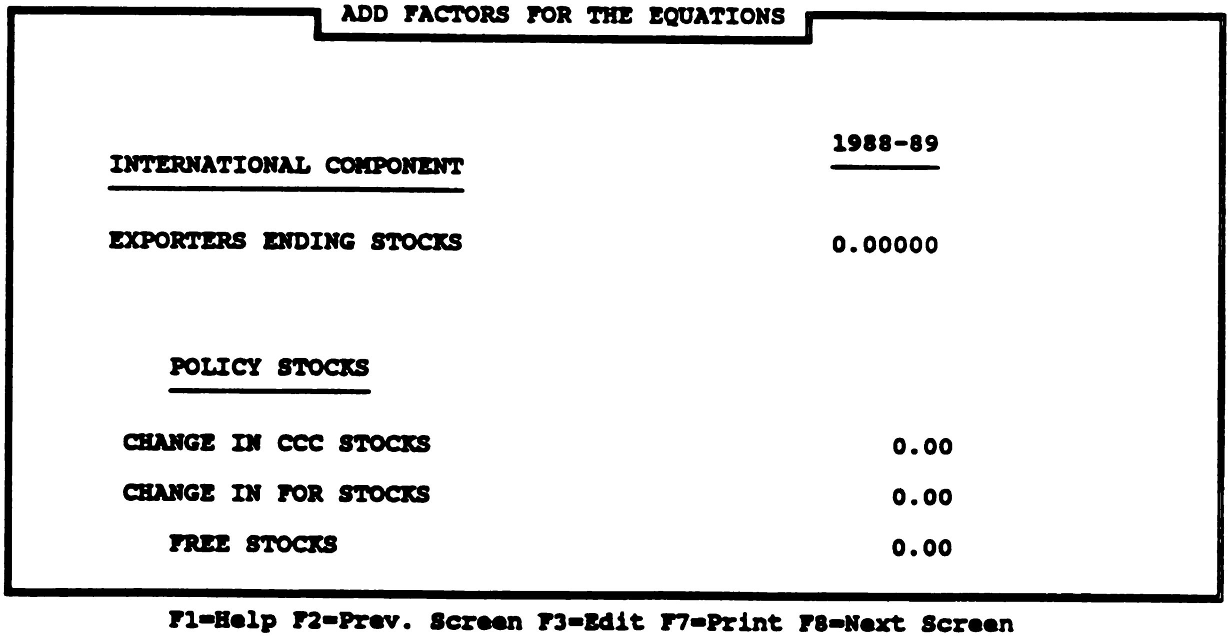
A.6 Examine Components of Feed Demand

This menu option allows you to view screens that show the livestock corn consumption both in aggregate form and on a per head basis for each livestock species. If you view



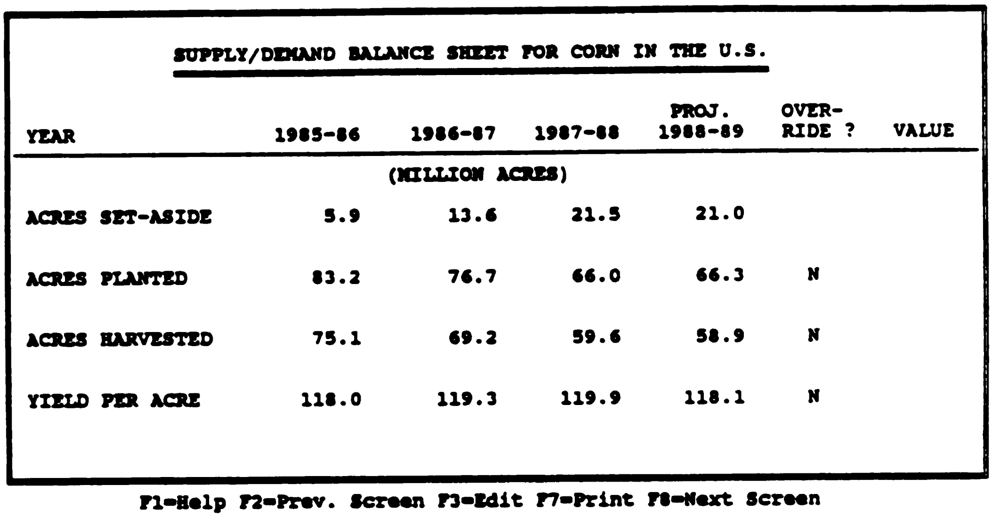
```
                         ADD FACTORS FOR THE EQUATIONS

                                  1988-89                                  1988-89
 SUPPLY                                       CONSUMPTION PER HEAD

ACRES PLANTED                      -4.200      TURKEYS                       0.000

ACRES HARVESTED                     0.000      HOGS                          0.000

YIELD                               0.000      OTHER & UNALLOCATED           0.00

CONSUMPTION PER HEAD                          INTERNATIONAL COMPONENT

 DAIRY ANIMAL                       0.000     IMPORTERS NET IMPORTS       0.00000

 CATTLE ON FEED                     0.000     IMPORTERS YIELD                0.00

 OTHER BEEF CATTLE                  0.000     EXPORTERS WHEAT YIELD          0.00

 HENS, PULLETS, & CHICKENS          0.000     EXPORTERS HARVESTED AREA        0.0

 BROILERS                           0.000     EXPORTERS CONSUMPTION       0.00000

            F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 25: Add Factor Screen 1



```
                         ADD FACTORS FOR THE EQUATIONS


                                                    1988-89
     INTERNATIONAL COMPONENT

     EXPORTERS ENDING STOCKS                        0.00000


        POLICY STOCKS

      CHANGE IN CCC STOCKS                             0.00

      CHANGE IN FOR STOCKS                             0.00

         FREE STOCKS                                   0.00

            F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 26: Add Factor Screen 2

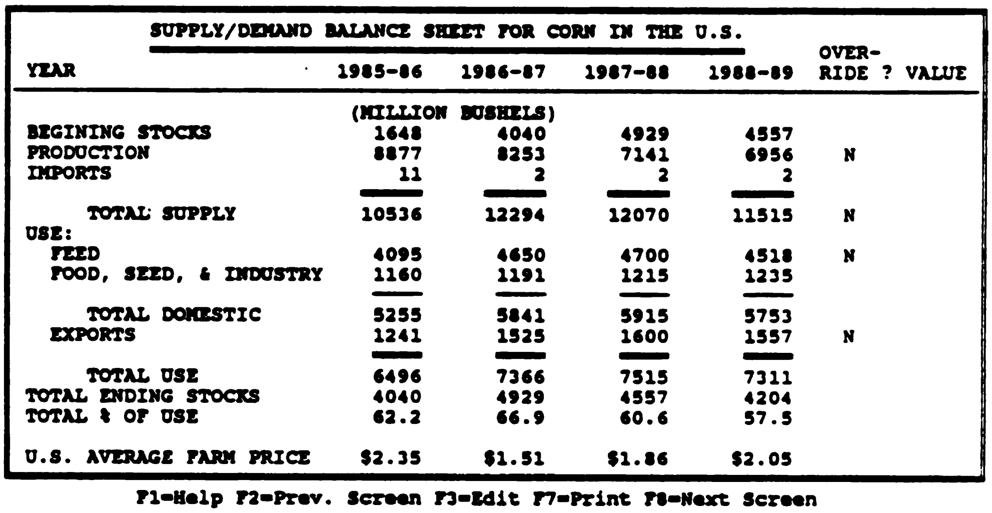


SUPPLY/DEMAND BALANCE SHEET FOR CORN IN THE U.S.

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89	OVER- RIDE ?	VALUE
		(MILLION ACRES)				
ACRES SET-ASIDE	5.9	13.6	21.5	21.0		
ACRES PLANTED	83.2	76.7	66.0	66.3	N	
ACRES HARVESTED	75.1	69.2	59.6	58.9	N	
YIELD PER ACRE	118.0	119.3	119.9	118.1	N	

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 27: Balance Sheet Screen 1

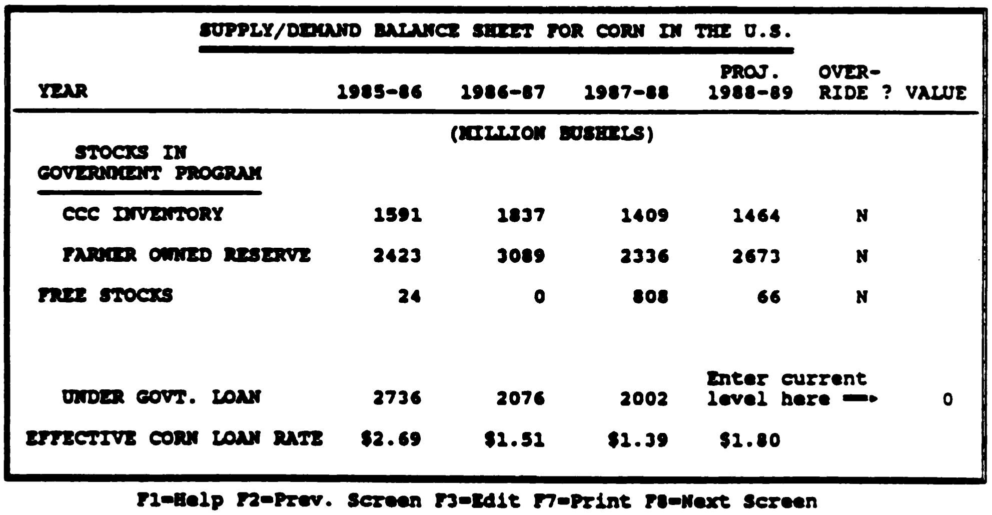


SUPPLY/DEMAND BALANCE SHEET FOR CORN IN THE U.S.

YEAR	1985-86	1986-87	1987-88	1988-89	OVER- RIDE ?	VALUE
	(MILLION BUSHELS)					
BEGINING STOCKS	1648	4040	4929	4557		
PRODUCTION	8877	8253	7141	6956	N	
IMPORTS	11	2	2	2		
TOTAL SUPPLY	10536	12294	12070	11515	N	
USE:						
FEED	4095	4650	4700	4518	N	
FOOD, SEED, & INDUSTRY	1160	1191	1215	1235		
TOTAL DOMESTIC	5255	5841	5915	5753		
EXPORTS	1241	1525	1600	1557	N	
TOTAL USE	6496	7366	7515	7311		
TOTAL ENDING STOCKS	4040	4929	4557	4204		
TOTAL % OF USE	62.2	66.9	60.6	57.5		
U.S. AVERAGE FARM PRICE	$2.35	$1.51	$1.86	$2.05		

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 28: Balance Sheet Screen 2

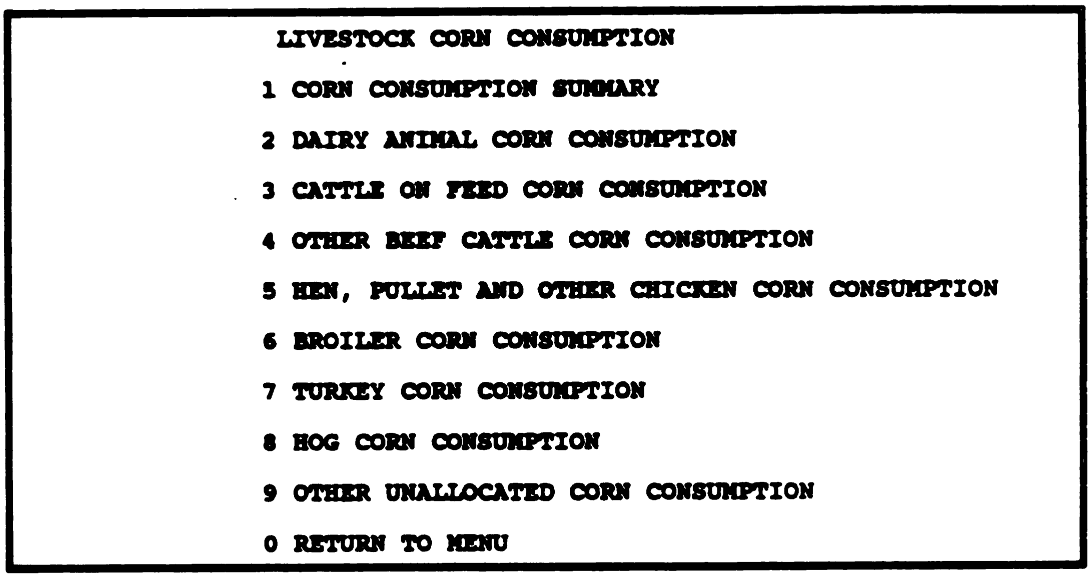


SUPPLY/DEMAND BALANCE SHEET FOR CORN IN THE U.S.

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89	OVER- RIDE ?	VALUE
		(MILLION BUSHELS)				
STOCKS IN GOVERNMENT PROGRAM						
CCC INVENTORY	1591	1837	1409	1464	N	
FARMER OWNED RESERVE	2423	3089	2336	2673	N	
FREE STOCKS	24	0	808	66	N	
UNDER GOVT. LOAN	2736	2076	2002	Enter current level here ──►		0
EFFECTIVE CORN LOAN RATE	$2.69	$1.51	$1.39	$1.80		

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 29: Balance Sheet Screen 3



LIVESTOCK CORN CONSUMPTION

1 CORN CONSUMPTION SUMMARY

2 DAIRY ANIMAL CORN CONSUMPTION

3 CATTLE ON FEED CORN CONSUMPTION

4 OTHER BEEF CATTLE CORN CONSUMPTION

5 HEN, PULLET AND OTHER CHICKEN CORN CONSUMPTION

6 BROILER CORN CONSUMPTION

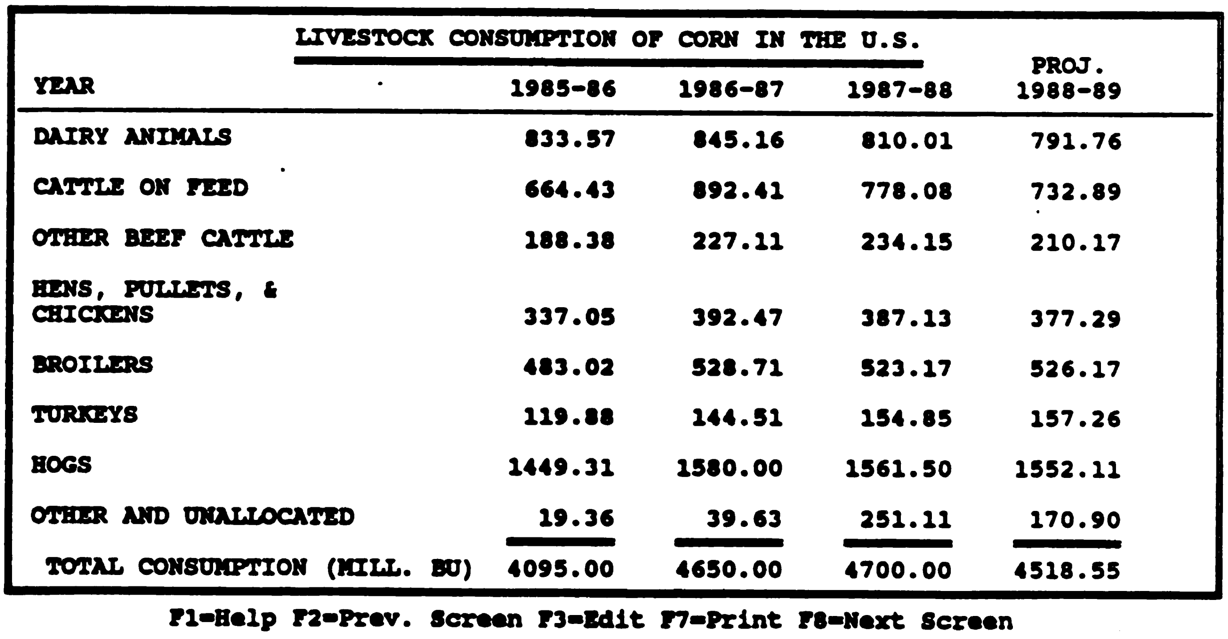
7 TURKEY CORN CONSUMPTION

8 HOG CORN CONSUMPTION

9 OTHER UNALLOCATED CORN CONSUMPTION

0 RETURN TO MENU

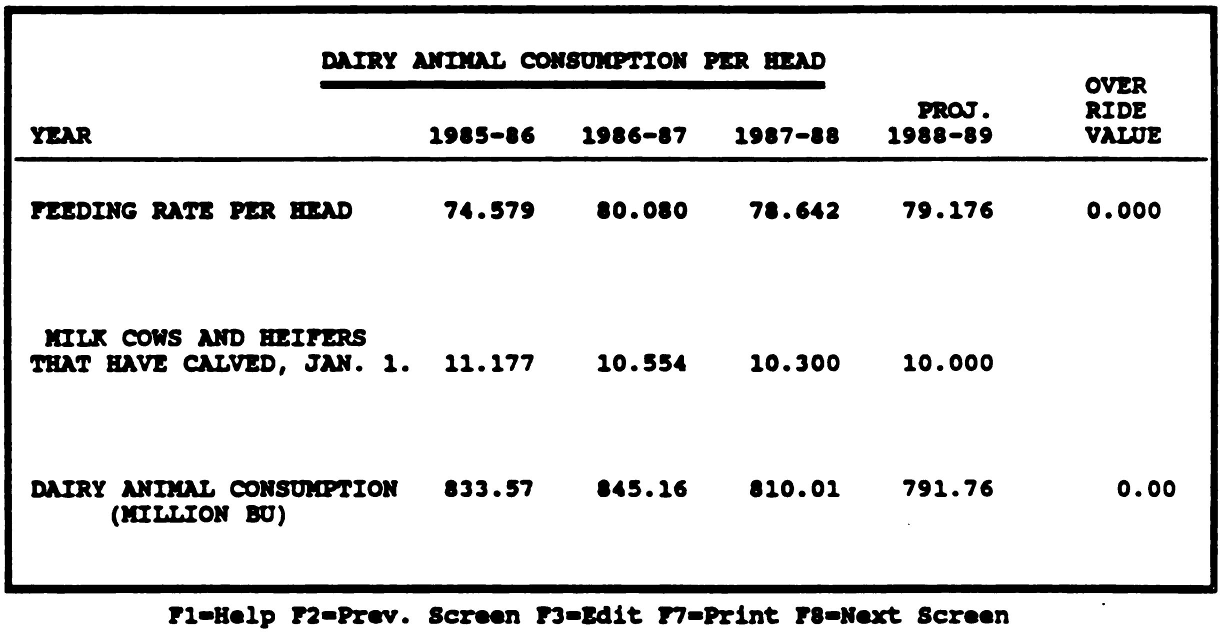
Figure 30: Livestock Corn Consumption Menu



LIVESTOCK CONSUMPTION OF CORN IN THE U.S.

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89
DAIRY ANIMALS	833.57	845.16	810.01	791.76
CATTLE ON FEED	664.43	892.41	778.08	732.89
OTHER BEEF CATTLE	188.38	227.11	234.15	210.17
HENS, PULLETS, & CHICKENS	337.05	392.47	387.13	377.29
BROILERS	483.02	528.71	523.17	526.17
TURKEYS	119.88	144.51	154.85	157.26
HOGS	1449.31	1580.00	1561.50	1552.11
OTHER AND UNALLOCATED	19.36	39.63	251.11	170.90
TOTAL CONSUMPTION (MILL. BU)	4095.00	4650.00	4700.00	4518.55

Figure 31: Corn Consumption Summary Screen

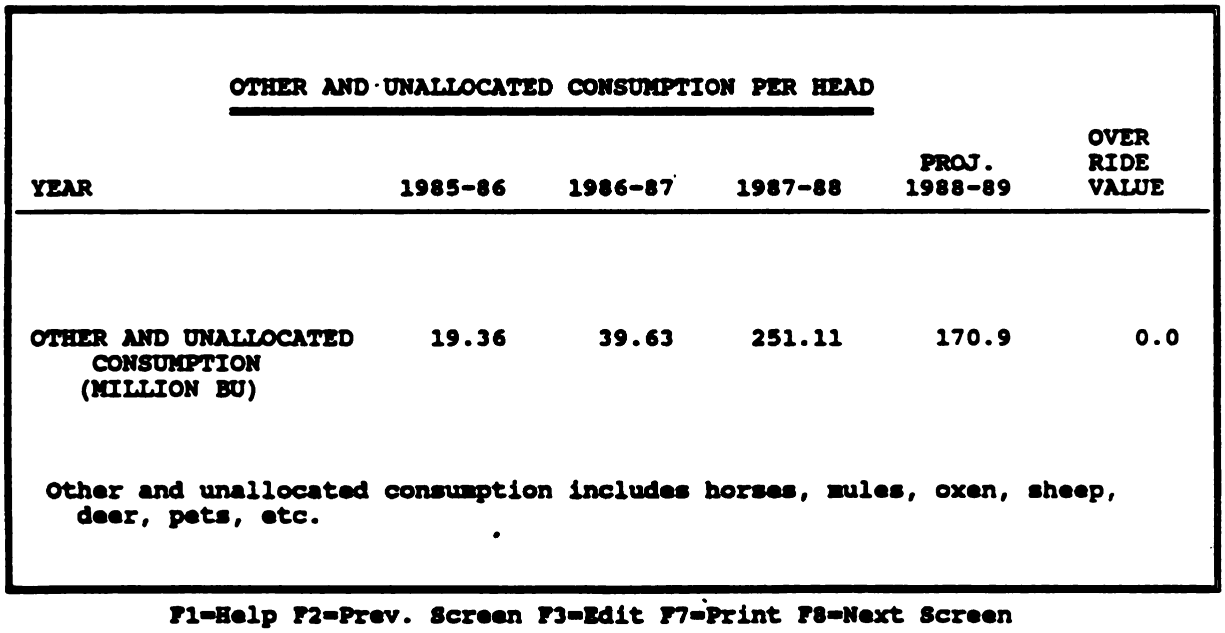


DAIRY ANIMAL CONSUMPTION PER HEAD

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89	OVER RIDE VALUE
FEEDING RATE PER HEAD	74.579	80.080	78.642	79.176	0.000
MILK COWS AND HEIFERS THAT HAVE CALVED, JAN. 1.	11.177	10.554	10.300	10.000	
DAIRY ANIMAL CONSUMPTION (MILLION BU)	833.57	845.16	810.01	791.76	0.00

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 32: Dairy Animal Consumption Per Head Screen



OTHER AND UNALLOCATED CONSUMPTION PER HEAD

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89	OVER RIDE VALUE
OTHER AND UNALLOCATED CONSUMPTION (MILLION BU)	19.36	39.63	251.11	170.9	0.0

Other and unallocated consumption includes horses, mules, oxen, sheep, deer, pets, etc.

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 33: Other and Unallocated Consumption Screen

the per head information, there are places to override the per head estimates for "what if" scenarios.

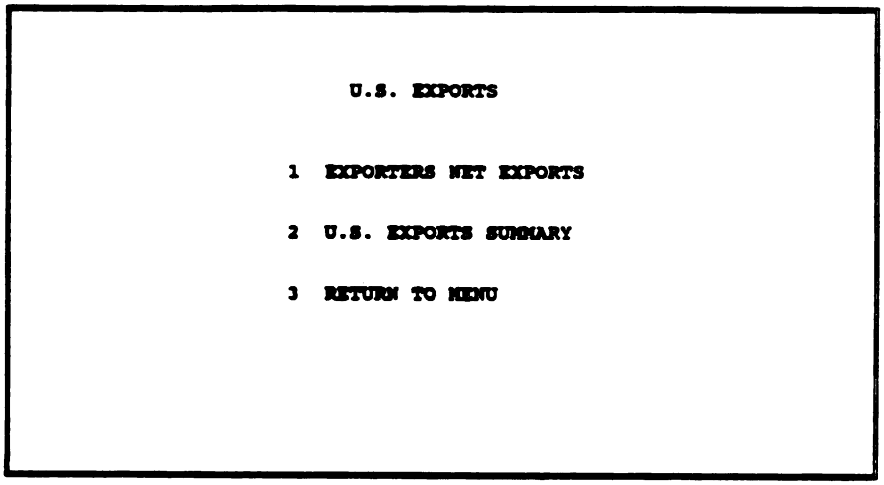
A.7 Examine Components of U.S. Exports

This menu option allows you to examine the international import/export component. One sub menu option shows a screen which disaggregates the exporting regions supply, consumption, ending stocks and exports. The other sub menu option shows a screen with the aggregate net world imports, exports and the U.S. exports.

A.8 Solve

There are two choices for solving the model depending upon your goals. One solve menu option allows you to solve for a single price. The second menu option determines a price probability given a chosen number of iterations.

In the single price option, you have either input the relevant exogenous and predetermined variables or overridden an estimated value. You input the initial "high" price and the price by which to decline on each iteration. When these inputs have been made, the algorithm will start with the "high" price and compare the estimated supply to the estimated demand. If the demand is less than the supply, the price will drop by the reduction price and recheck the supply and demand. When the demand is equal to the supply the algorithm will stop and you can return to the display portion of the program.



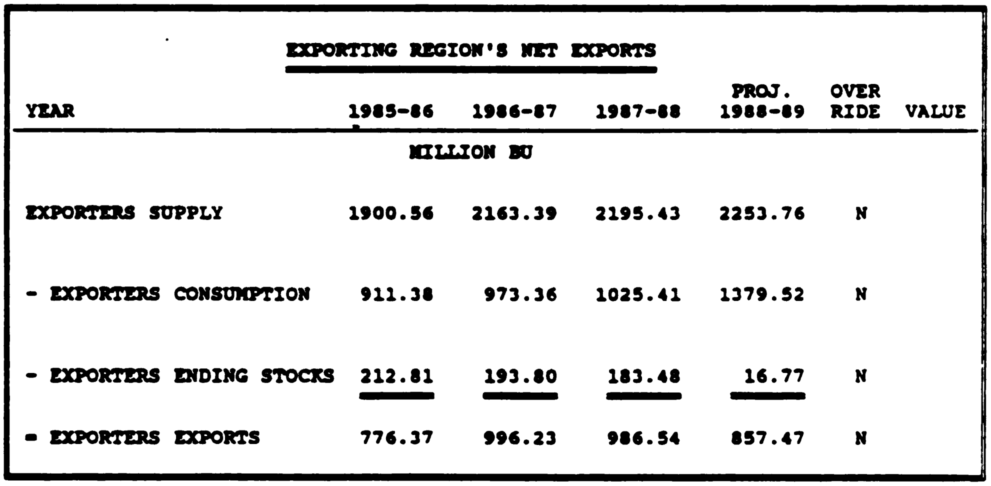
U.S. EXPORTS

1 EXPORTERS NET EXPORTS

2 U.S. EXPORTS SUMMARY

3 RETURN TO MENU

Figure 34: U.S. Export Sub Menu

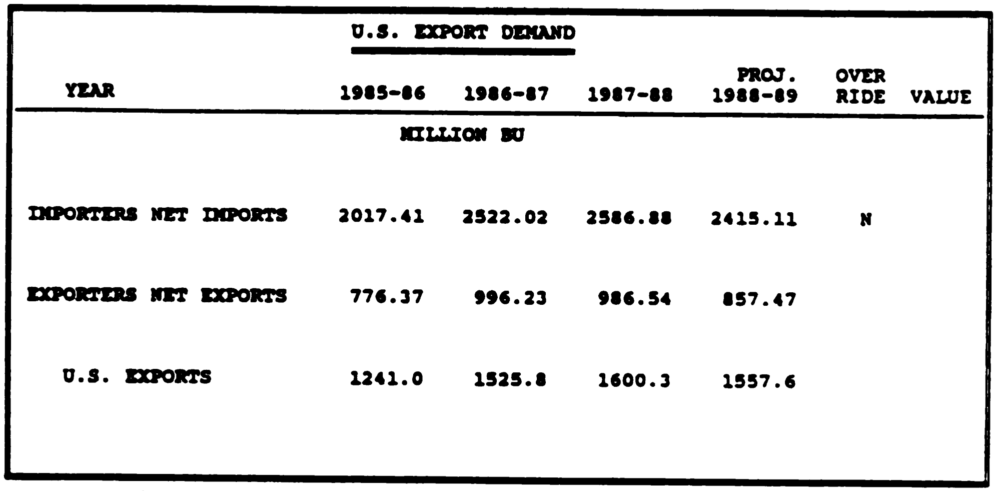


EXPORTING REGION'S NET EXPORTS

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89	OVER RIDE	VALUE
	MILLION BU					
EXPORTERS SUPPLY	1900.56	2163.39	2195.43	2253.76	N	
- EXPORTERS CONSUMPTION	911.38	973.36	1025.41	1379.52	N	
- EXPORTERS ENDING STOCKS	212.81	193.80	183.48	16.77	N	
= EXPORTERS EXPORTS	776.37	996.23	986.54	857.47	N	

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 35: Exporters Net Exports Screen



U.S. EXPORT DEMAND

YEAR	1985-86	1986-87	1987-88	PROJ. 1988-89	OVER RIDE	VALUE
	MILLION BU					
IMPORTERS NET IMPORTS	2017.41	2522.02	2586.88	2415.11	N	
EXPORTERS NET EXPORTS	776.37	996.23	986.54	857.47		
U.S. EXPORTS	1241.0	1525.8	1600.3	1557.6		

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 36: U.S. Exports Summary Screen

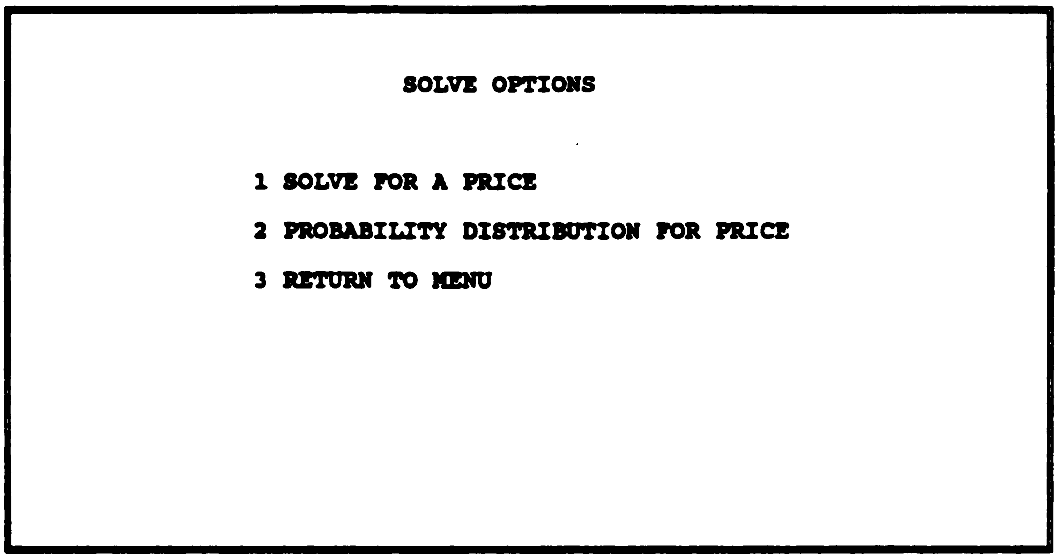
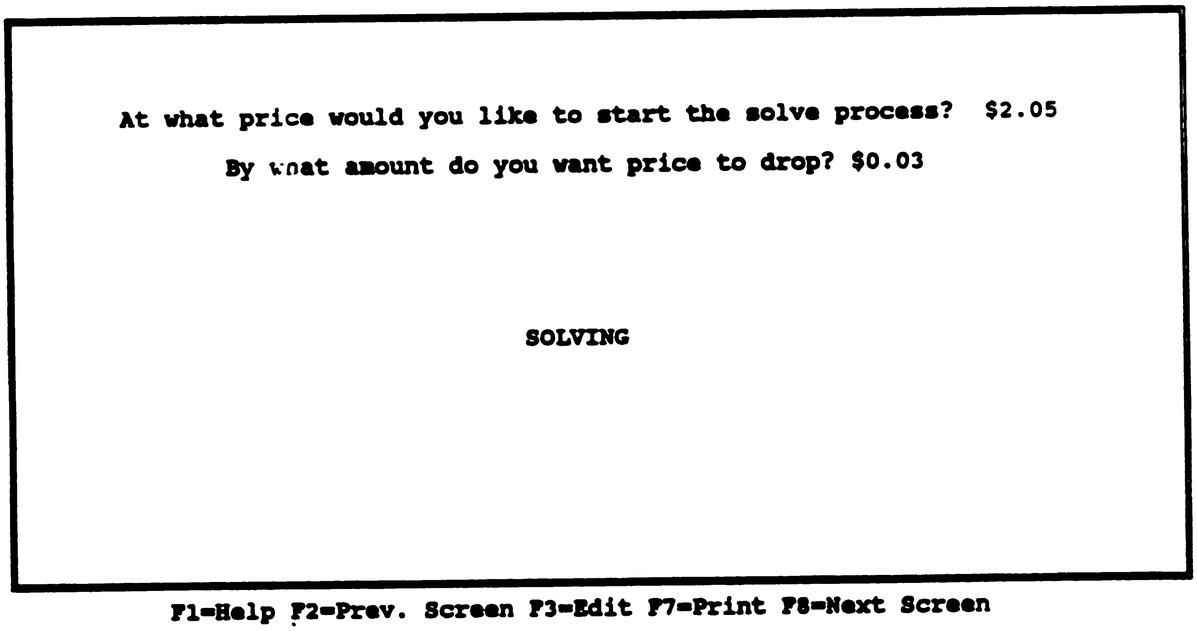


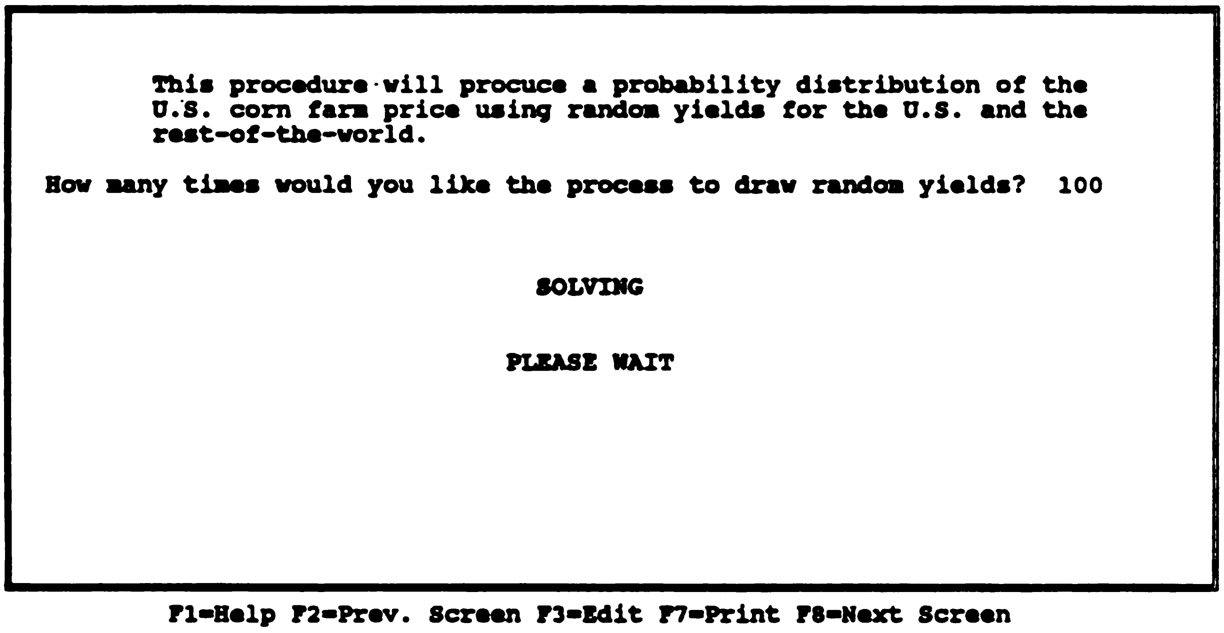
Figure 37: Solve Options Sub Menu


At what price would you like to start the solve process? $2.05

By wnat amount do you want price to drop? $0.03

SOLVING

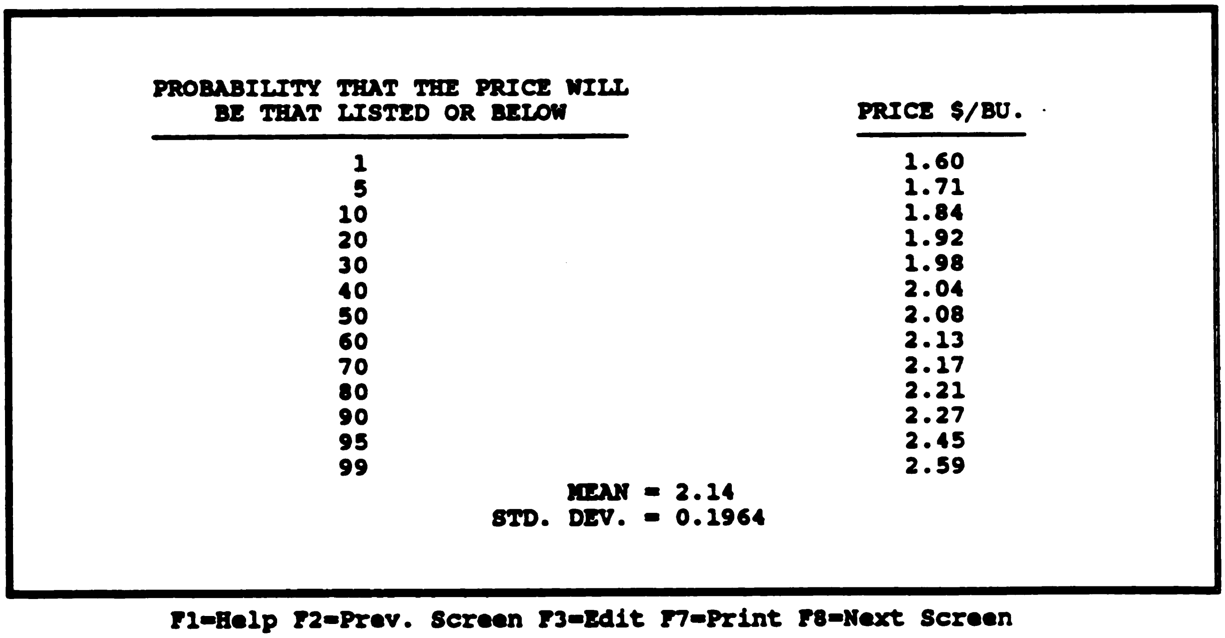
Figure 38: Single Price Solve Screen



```
          This procedure will procuce a probability distribution of the
          U.S. corn farm price using random yields for the U.S. and the
          rest-of-the-world.

How many times would you like the process to draw random yields?  100


                                   SOLVING


                                 PLEASE WAIT
```

```
F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 39: Probability Iteration Screen



```
PROBABILITY THAT THE PRICE WILL
    BE THAT LISTED OR BELOW                 PRICE $/BU.
_______________________________             ___________

              1                                1.60
              5                                1.71
             10                                1.84
             20                                1.92
             30                                1.98
             40                                2.04
             50                                2.08
             60                                2.13
             70                                2.17
             80                                2.21
             90                                2.27
             95                                2.45
             99                                2.59
                         MEAN = 2.14
                    STD. DEV. = 0.1964
```

```
F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen
```

Figure 40: Price Probability Distribution Display Screen

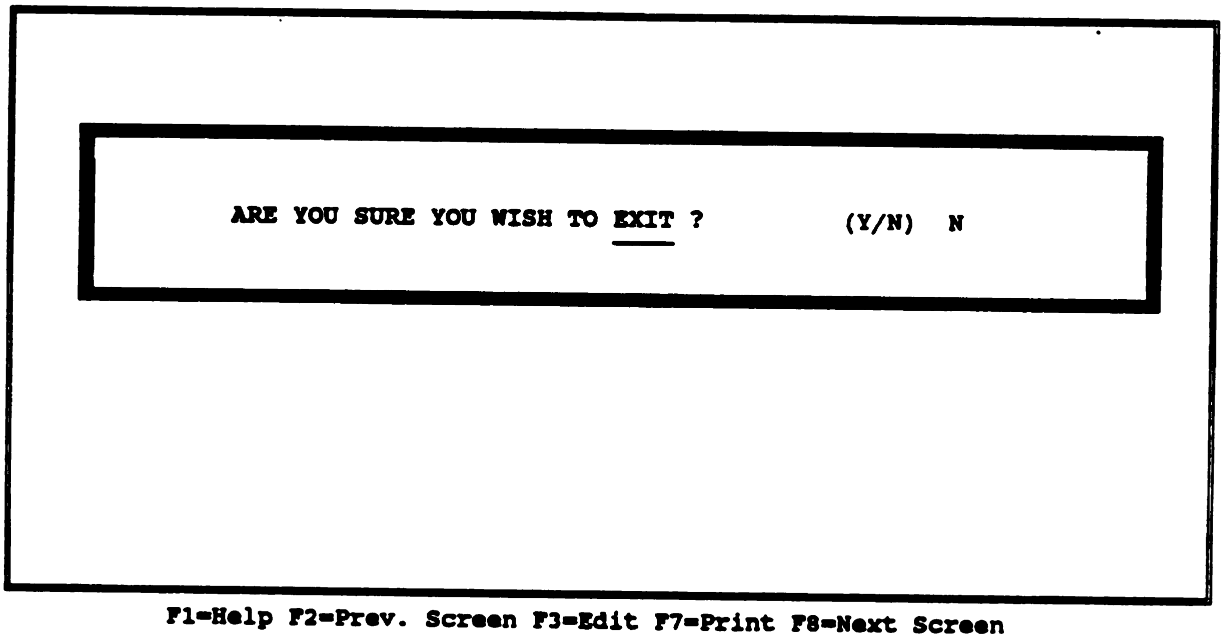
A trade-off is involved when choosing the amount by which to reduce on each iteration. If you choose a low reduction price, the model will solve slowly. If you are unsure how your overrides will affect the model, you can set a high price and a large reduction price. This will allow the model to solve quickly but the accuracy of the forecast suffers. By choosing the later alternative, you can wait until the balance sheet has reached an equilibrium price and the key codes appear at the bottom of the screen. Choosing the F3 key code, you will be returned to this solve screen and the current equilibrium price will be displayed. This time you can choose a price higher than the previous reduction price and a new reduction price.

The second solve option allows you to select a number of iterations for the purpose of calculating and displaying a range of prices and probabilities based upon random production shocks. The model will accept 300 or less iterations. By increasing the number of iterations, the standard error of the price distribution decreases and the confidence interval becomes smaller. However depending upon the number of iterations chosen, the model will take some time to solve.

When the model solves using this menu option the random shocks, affect the world production. Each iteration will have a different equilibrium price a price range is created. You can move to the price probability display by pressing F8 when the key codes appear at the bottom of the screen. The range of prices have a related probability distribution which indicates the probability of the price shown being at or below that percentage. This information provides subjective probabilities which can be used in the users decision analysis.

A.9 Exit

This option displays a screen that makes sure you are ready to exit the program. If not, you inputs the response for no and the main menu will return.



ARE YOU SURE YOU WISH TO EXIT ?   (Y/N) N

F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen

Figure 41: Exit Screen

Appendix 4

Computer Input/Output Code

```
PROGRAM FIRST(INPUT,OUTPUT);

(**************************************************************************
*                                                                         *
*                AN ECONOMIC SUPPLY/DEMAND BALANCE SHEET                  *
*                                                                         *
*                   FOR FORECASTING AND SIMULATING THE                    *
*                                                                         *
*                      UNITED STATES CORN INDUSTRY                        *
*                                                                         *
*                                  by                                     *
*                                                                         *
*                         ANTHONY B. WASHINGTON                           *
*                                                                         *
*                 Submitted to Michigan State University                  *
*               in partial fulfillment of the requirements                *
*                           for the degree of                             *
*                                                                         *
*                           MASTER OF SCIENCE                             *
*                                                                         *
*                                                                         *
*                   Department of Agricultural Economics                  *
*                                                                         *
*                                  1988                                   *
*                                                                         *
**************************************************************************)
{Procedures Generated By Screen Sculptor,Version 1.2 4/2/1987 - 23:33:4 }
CONST CopyrightSS='(C)Copyright 84,85 The Software Bottling Company Of New York';
{    DO NOT REMOVE The Above Copyright Notice
     This Program may not be used without the above Copyright Notice
}
{  SCREEN SCULPTOR(C) Version 1.2
   (C) COPYRIGHT, THE SOFTWARE BOTTLING COMPANY OF NEW YORK, 1984, 1985
   WARNING: Do not attempt to make any changes to this procedure unless you
   have made a backup. The Software Bottling Company Of New York can not
   and will not support such changes made to this program.
   Turbo(tm) Pascal Version, Trade Mark Of Borland International}

{ Global Required Definition of variables and constants }

TYPE  STR2 = STRING[2]; STR80 = STRING[80]; STR79 = STRING[79];
      resSS = (staySS, prevSS, exitSS, nextSS);
      AR4 = ARRAY[1..4] OF REAL;

CONST { Esc, Up Arrow Key, Left Arrow Key , Page Up Key       }
      escSS=#27;   uSS='H'; lSS='K'; puSS='I';
      { Blank, Down Arrow Key, Right Arrow Key, Page Down Key }
      blankSS=' '; dSS='P'; rSS='M'; pdSS='Q';
      { Function keys  F1-F10 }
      f1SS=';'; f2SS='<'; f3SS='='; f4SS='>'; f5SS='?';
      f6SS='@'; f7SS='A'; f8SS='B'; f9SS='C'; f10SS='D';
      retSS : STR2='';
      ES = 1;

        ***** Beginning of Include file LISTING.INC *****
{$I LISTING.INC}
VAR   actionSS, last_field_actionSS : resSS;
      hiSS, loSS : REAL;
      vtypeSS, screenSS, use_screenSS, screen_fieldSS, varSS : INTEGER;
      file_existSS, last_fieldSS, retrieveSS : BOOLEAN;
      rangeSS : STR80;
      BeepOnSS: BOOLEAN;

{ ENDOGENOUS AND EXOGENOUS VARIABLES }
PCNT_OF_XPORTS,
APCT:AR4;
AHCT:AR4;
PYIELD,
CORNYT:AR4;
WHTYT,
GROSS,
ARP,
DIVERSION,
BAL,
DVPCN,
DVRCN,
DIVPAY,
DPAY,
DEFPAY,
BEGINSTOCKS,
MIN_FREE_STOCKS,
END_STOCKS,
PRODUCT:AR4;
IMPORTS,
TSUPPLY,
ONE:AR4;
ONEPH,
TWO:AR4;
TWOPH,
THREE:AR4;
THREEPH,
FOUR:AR4;
FOURPH,
FIVE:AR4;
FIVEPH,
SIX:AR4;
```

```
SIXPH,
SEVEN:AR4;
SEVENPH,
EIGHT:AR4;
DANUM:AR4;
COFNUM:AR4;
OBCNUM:AR4;
HPCNUM:AR4;
BRONUM:AR4;
TURNUM:AR4;
HOGNUM:AR4;
DAIRYCONS,
CATONFED,
OTHERBEEFCAT,
CHICKENCONS,
BROILERCONS,
TURKEYCONS,
HOGCONS,
OTHERCONS,
FEEDT:AR4;
HFCS,
SEED_USE,
RESIDUAL,
FUEL_USE,
FSI_USE,
DOMESTIC_USE,
SETASIDE_ACRES:AR4;
DV73:REAL;
DV82:REAL;
DV83:REAL;
DV84:REAL;
DV74ON:REAL;
VCCN:AR4;
VCHW:AR4;
C_TARGETP:AR4;
CORN_LOAN_RATE:AR4;
PARREV:AR4;
REVCOR:AR4;
REVWHT:AR4;
YEAR:AR4;
YEAR1B,YEAR2B,YEAR3B,YEAR4B:REAL;
YEARB,
CORNPT:AR4;
RFARMP,
MILKP:AR4;
WHTPT:AR4;
BEFSTRP:AR4;
BCOWP:AR4;
SMPT:AR4;
HOGP:AR4;
IPOP:AR4;
IGDP:AR4;
IHA,
IYIELD,
IPRO:AR4;
IFP,
IWP,
IGDPPC,
IPROPC,
INIPC:AR4;
INI_IN_BU,
NXXRF:AR4;
NXXRW:AR4;
NMXRF:AR4;
NMXRW:AR4;
CHWT,SBWT,DMWT,LOWT,NIWT,LDWT,BRWT,ARWT,AUWT,CAWT:REAL;
CPISB:AR4;
CPIDM:AR4;
CPILO:AR4;
CPINI:AR4;
CPILD:AR4;
CPIBR:AR4;
CPIAR:AR4;
CPIAU:AR4;
CPICA:AR4;
CPICH:AR4;
CPIT:AR4;
GDPCH,
GDPSB,
GDPDM,
GDPLO,
GDPNI,
GDPLD,
GDPBR,
GDPAR,
GDPAU,
GDPCA,
FP,
WP,
EYIELD:AR4;
EFREV,
EWREV:AR4;
WYAR,
WYAU,
WYCA,
FYAR,
FYAU,
FYCA,
XRAR,
XRAU,
XRCA,
E_END_STK,
E_NET_EXPORT,
EHA:AR4;
ECON,
ECON_IN_BU,
ECONPC:AR4;
EES,
EES_IN_BU,
EESPC:AR4;
ESPLY_IN_BU,
ESPLYPC,
EPRO,
EPROPC,
EUDPC,
ENE:AR4;
ENE_IN_BU,
EXPORTS:AR4;
EGDP:AR4;
EPOP:AR4;
EWYIELD:AR4;
TOTAL_USE:AR4;
PERCENT_USE:AR4;
FPLR,
FPLR2,
FPLR3:REAL;
CCCSTOCKS,
DCCCSTOCKS,
FORSTOCKS,
DFORSTOCKS:AR4;
FREESTOCKS:AR4;
GOVTSTOCKS:AR4;
POLICY_STOCKS:AR4;
TRIGGERP:REAL;
           { ADD FACTOR VARIABLES }
APCT_AF,
AHCT_AF,
CORNYT_AF,
ONEPH_AF,
TWOPH_AF,
THREEPH_AF,
FOURPH_AF,
FIVEPH_AF,
SIXPH_AF,
SEVENPH_AF,
EIGHT_AF,
IYIELD_AF,
INIPC_AF,
EYIELD_AF,
EWYIELD_AF,
EHA_AF,
ECONPC_AF,
EESPC_AF,
DCCCSTOCKS_AF,
DFORSTOCKS_AF:AR4;
FREESTOCKS_AF:AR4;
           { OVERRIDE VARIABLES }
CORNYT_OR,
APCT_OR,
AHCT_OR,
BEGINSTOCKS_OR,
PRODUCT_OR,
TSUPPLY_OR,
IMPORTS_OR,
ONEPH_OR,
DAIRYCONS_OR,
TWOPH_OR,
CATONFED_OR,
THREEPH_OR,
OTHERBEEFCAT_OR,
FOURPH_OR,
CHICKENCONS_OR,
FIVEPH_OR,
BROILERCONS_OR,
SIXPH_OR,
TURKEYCONS_OR,
SEVENPH_OR,
HOGCONS_OR,
OTHER_UNALLOC_OR,
FEEDT_OR,
DOMESTIC_USE_OR,
INI_IN_BU_OR,
ESPLY_IN_BU_OR,
ECON_IN_BU_OR,
EES_IN_BU_OR,
ENE_IN_BU_OR,
EXPORTS_OR,
TOTAL_USE_OR,
CCCSTOCKS_OR,
FORSTOCKS_OR,
FREESTOCKS_OR,
ENDSTOCKS_OR:AR4;

          { VARIABLES FOR THE RANDOM SECTION }
ALPHA,ALPHA1,ALPHA2:ARRAY[1..30] OF REAL;
BETA,BETA1,BETA2,
GAMMA,GAMMA1,GAMMA2,
YIELD,YIELD1,YIELD2,
Z,Z1,Z2:REAL;
REDUCE_PRICE:REAL;
INFO:STRING[7];
PAUSE:STRING[11];
RANDOM_PULLS:REAL;

          { FLAG VARIABLES FOR OVERRIDE VALUES }
CORNYT_FLAG,
APCT_FLAG:STRING[1];
AHCT_FLAG,
BEGINSTOCKS_FLAG,
PRODUCT_FLAG,
TSUPPLY_FLAG,
IMPORTS_FLAG,
FEEDT_FLAG,
DOMESTIC_USE_FLAG:STRING[1];
INI_IN_BU_FLAG,
ESPLY_IN_BU_FLAG,
ECON_IN_BU_FLAG,
EES_IN_BU_FLAG,
ENE_IN_BU_FLAG,
EXPORTS_FLAG,
TOTAL_USE_FLAG,
CCCSTOCKS_FLAG,
FORSTOCKS_FLAG,
FREESTOCKS_FLAG:STRING[1];
ENDSTOCKS_FLAG:STRING[1];

      { MISCELLANEOUS VARIABLES }
FG,
BG,
FLAG,
WA,I,S,F:INTEGER;
P,J:REAL;
NY:INTEGER;
USCORNPT:ARRAY[1..300] OF REAL;
EQUILIBRIUM:REAL;
PERCENTILE_ANSWER:REAL;
        ***** End of Include file LISTING.INC *****

{$V-,C-,R-} { Pascal Directives. See Compiler Manual }
{$I TURPROCS.TUR  Include Procedures In This File. See Manual }

(*  This is a summary of the procedures in TURPROCS.TUR

PROCEDURE BEEP(BeepOn: BOOLEAN);            { Sound Beep if BeepOn=TRUE }
PROCEDURE CLEAR_KBD;                        { Clear Keyboard Buffer }
PROCEDURE COLOR(foregr,backgr:BYTE);        { Set Color }
PROCEDURE WRITEC(vtext: STR80);             { Write Chars Using Color }
FUNCTION  SET_MONITOR_TYPE: INTEGER;        { Determine Monitor Type }
{ Display A Screen Sculptor Screen }        { 2=Color, 3=Mono }
PROCEDURE DISPLAY_SCREEN(screen_name: STR80; VAR file_existSS: BOOLEAN);
{ Display And Get An Item From Screen. See Detailed Desription In Manual }
PROCEDURE GETITEM(
                COL,LIN,LEN :           BYTE;     { Column, Line, Length }
                ITYPE :                 CHAR;     { Type= C, N, D, Y, M  }
            VAR WITEM :                 STR80;    { Variable Name        }
                PICT :                  STR80;    { Picture X, U, L, 9, 8 # }
                ITEM_LOW,ITEM_HIGH :    STR80;    { Range - Numerics/Date Only}
            VAR RET :                   STR2;     { Returned Code        }
                RETRIEVE :              BOOLEAN;  { False=Disp Only, True=Get }
                FGR_COLOR,BGR_COLOR :   BYTE      { Colors Foregr, Backgr }
                ); EXTERN;

*)

PROCEDURE RET_STATUS;
{ Check Status Of Variable retSS and return a code in 'actionSS' & set 'varSS'
  This procedure is called immediately following GETITEM }

{ Input to this procedure:
  when retSS is length 1 the values are any of the ASCII chars
```

```
 when retSS is length 2 the values are uSS, lSS, puSS, pdSS, function keys
                                     dSS, rSS
                                     ( See CONST Section For Meanings ) }
{ Output:
 The following codes are returned in actionSS : nextSS, prevSS,
                                                exitSS, staySS }
{ Based upon 'actionSS' this procedure will then set 'varSS' to an integer,
 which represents the next item (variable ) to get.  }
BEGIN
 last_field_actionSS:=exitSS;
 actionSS:=nextSS; { Initialize Action Code }
 if retrieveSS then { Is retrieveSS TRUE? }
 begin
  if ord(retSS[0])=2 then { Is retSS length 2 ? }
  begin
   CASE retSS[2] of
   { Action to be taken depending on the last key pressed }
    uSS, lSS: actionSS:=prevSS;  { Up Key, Left Key }
    dSS, rSS: actionSS:=nextSS;  {Down Key, Right Key}
    puSS: actionSS:= { Page Up   }exitSS;
    pdSS: actionSS:= { Page Down }exitSS;
    f1SS,f2SS,f3SS,f4SS,f5SS,
    f6SS,f7SS,f8SS,f9SS,f10SS: actionSS:=staySS; { Function Key }
   END { Case ret };
  end else { retSS is length 1 }
  begin
   if retSS=escSS { Escape Key } then actionSS:=exitSS
  end;
 { Any other key not in the above list will keep actionSS=nextSS }
 end; {retrieveSS}
 CASE actionSS of
  staySS: ;
  nextSS: begin
       varSS:=varSS+1; if varSS>screen_fieldSS then varSS:=1;
       if last_fieldSS and retrieveSS then actionSS:=last_field_actionSS
      end;
  prevSS: begin varSS:=varSS-1; if varSS<1 then varSS:=screen_fieldSS end;
  exitSS: ;
 END; { CASE }
END; {PROCEDURE RET_STATUS}

PROCEDURE GETNUM(
        COL,LIN,LEN :         BYTE;     { Column, Line, Length }
        ITYPE :               CHAR;     { Type= C, N, D, Y, M  }
      VAR WITEM :             REAL;     { Numerci Variable Name }
        PICT :                STR80;    { Picture X, U, L, 9, 8 # }
        ITEM_LOW,ITEM_HIGH :  REAL;     { Range - Numerics/Date Only}
      VAR RET :               STR2;     { Returned Code        }
        RETRIEVE :            BOOLEAN;  { False=Disp Only, True=Get }
        FGR_COLOR,BGR_COLOR : BYTE      { Colors Foregr, Backgr }
        );
{ This Procedure converts numeric to string before calling GETITEM }
{ It then converts the result back to numeric }
VAR numSS,numloSS,numhiSS: STR80; errorcodeSS,dec_posSS: INTEGER;
```

```
BEGIN
  { Get # of Decimal Positions }
  dec_posSS:=ord(pict[0])-pos('.',pict);
  { Convert item, low and high range to string }
  STR(witem:0:dec_posSS,numSS);
  STR(item_low:0:dec_posSS,numloSS);
  STR(item_high:0:dec_posSS,numhiSS);
  GETITEM(col,lin,len,itype,numSS,pict,numloSS,numhiSS,
          ret,retrieve,fgr_color,bgr_color);
  { Convert string to numeric item }
  VAL(numSS, witem, errorcodeSS);
END; { GETNUM }

        ***** Beginning of Include file MENU.UTL *****
{$I MENU.UTL}
const
      IOerr:  boolean = false;

type
 str255 = string[255];
 str3 = string[3];
 str8 = string[8];
 str1 =string[1];
 char3 = array [1..3] of char;

var
    {** variables used for whatnext **}
  Bscreen: integer;
  response: char;
    {** variables used for datasp.scr and the Exit screen **}
  Option,Exit_Response: STR1;

    {** variables used in LOAD **}
  DefaultDrive,
  LoadYn,
  SaveYn: STR1;
  FileName: STRING[80];
  FileExt: STR3;
  bactive,
  edit,
  exit: boolean;
  ub: integer;


Procedure INIT_VAR_UTIL;
    {** Procedure to initialize the above variables **}
   Begin
     FileExt:='BIN';
     Option:='N';
     Exit_Response:='N';
     getdir(0,DefaultDrive);
     LoadYn:='N';
     Filename:='';
     SaveYn:='Y';
    bscreen:=1;
    bactive:=false;
    exit:=false;
    edit:=false;
  End;

 Procedure Flashup(flash_command:Str255);
 {This procedure receives a string and performs a FLASHUP WINDOW function
  to it. The write stmt. is the procedure which activates the flash
  command.
  See a FLASUP WINDOW manual (Software Bottling company)}

  var tilde: char;
  begin
   gotoxy(1,1);
   tilde:=chr(126);
   write(tilde,flash_command,'/');
  end;

{*******************************************************************************
*  FUNCTION FreeDiskSpace uses MsDos and functions of MsDos to get the am-    *
*  ount of diskspace available on a disk. This function returns a negative    *
*  number if no diskspace is available. It also returns a negative number     *
*  if the drive bieng accessed is not available on the computer.              *
*******************************************************************************}

FUNCTION FreeDiskSpace( filespec : str2 ) : real;

   type
      regpack = record
                  ax, bx, cx, dx, bp, si, di, ds, es, flags : integer;
                end;
   var
      regs           :  regpack;
      drivenum,
      colon          :  integer;
      availclusters,
      totalclusters,
      bytespersect,
      sectpercluster  :  real;
      ch             :  char;
   begin
      drivenum := 0;                  {start with default drive}
      colon := pos( ':', filespec );
      if colon <> 0 then begin        {if drive in filespec then...}
         ch := copy( filespec, colon-1, 1 );
         drivenum := ord( ch ) - 64;
      end { if };
      with regs do begin
         ax := $3600;
         dx := drivenum;
         msdos( regs );
         if ax = $FFFF then begin      {got an invalid drive}
            freediskspace := -99999.0;
            end { then }
         else begin
            availclusters := bx;
            totalclusters := dx;
            bytespersect := cx;
            sectpercluster := ax;
            freediskspace := bytespersect * sectpercluster * availclusters;
         end; { if };
      end { with };
   end; { FreeDiskSpace }


{*******************************************************************
*  PROCEDURE Line25 writes the string-parameter passed to it on the *
*  25th line of the screen.                                        *
*******************************************************************}
procedure Line25(bottom : str80);
BEGIN
  gotoxy(1,25);
  clreol;
  gotoxy(((80-length(bottom)) div 2),25);
  write(bottom);
END;

{*******************************************************************
* PROCUDURE Presskey loops until the user presses any key.         *
*                                                                  *
*******************************************************************}
procedure Presskey;
VAR
  cmd : char;
  test : boolean;

BEGIN
  repeat
    if keypressed then test:=true;
  until test;
  read(kbd,cmd);
END;

{*******************************************************************
* PROCEDURE Continue writes a message on line 25 by calling the    *
* procedure Line25. It then waits for a response by calling the    *
* procedure Presskey.                                              *
*******************************************************************}
procedure Continue;
BEGIN
  Line25('Press any key to continue...');
  Presskey;
  gotoxy(1,25);
  clreol;
END;

{*******************************************************************
*  PROCEDURE Function_Key_Pressed awaits the user input and then    *
*  emulates the pressing of <RETURN>.                               *
*********************************************************************}
procedure Function_Key_Pressed(VAR retSS:Str2);

VAR
  answerSS:string[1];

BEGIN
  answerSS:=blankSS;
  GetItem(78,25,1,'C',answerSS,'U','','',retSS,TRUE,7,0);
  if ord(retSS[0])=2
    then
      retSS:=retSS[2]
    else
      retSS:=blankSS;
END;

{*********************************************************************
*  PROCEDURE IOcheck checks for a number of input/output errors as  *
*  listed below. In order for this procedure to be valuable the     *
*  operating systems input/output error checking system must be     *
*  temporarily turned off. This is done by placing $I- in parenth   *
*  before IOcheck is called. To turn the system back on use $I+.    *
*  The procedure writes an error message on line 25 of the screen   *
*   and awaits user input to continue.                              *
*********************************************************************}
procedure IOcheck(pass: integer);
 VAR
   IOcode:Integer;

 BEGIN
   IOcode:=IOresult;
   IOerr:=(IOcode<>0);
   If IOerr
    then begin
     Case IOcode of
       $01 :Line25('File does not exist-Press any key to continue');
       $02 :Line25('File not open for input-Press any key to continue');
       $03 :Line25('File not open for output-Press any key to continue');
       $04 :Line25('File not open-Press any key to continue');
       $10 :Line25('Error in numeric format-Press any key to continue');
       $20 :line25('Operation not allowed on logical device-Press any key to continue');
       $21 :Line25('Not allowed in direct mode-Press any key to continue');
       $22 :Line25('Assign to standard files not allowed-Press any key to continue');
       $90 :Line25('Record length mismatch-Press any key to continue');
       $91 :Line25('Seek beyond end-of-file -Press any key to continue');
       $99 :Line25('Unexpected end-of-file -Press any key to continue');
       $F0 :Line25('Disk write error-Press any key to continue');
       $F1 :Line25('Directory is full-Press any key to continue');
       $F2 :Line25('File size overflow-Press any key to continue');
       $FF :Line25('File disapeared-Press any key to continue')
     else
        Line25('Unknown I/O error-Press any key to continue');
     end; {case}
      Presskey;{pressing of any key to continue}
      end {then begin}
  else begin
       if (pass = 2)
        then begin
             Line25('File transfer successful-Press any key to continue');
             Presskey;{pressing of any key to continue}
           end;
       end; {else if}
  Line25('                                                                        ');
  end;{of proc IOcheck}

{*********************************************************************
* Procedure Helpscreen  prints the the screen file HELP1.SCR        *
* and HELP2.SCR to the monitor, if they exist.                      *
*********************************************************************}
 PROCEDURE Helpscreen;

BEGIN
  DISPLAY_SCREEN('HELP1.SCR',file_existSS); {display screen}
   if not file_existSS then
    BEGIN
      gotoxy(1,1);
      write('Missing screen Helpscreen');
    end;
 Continue;
 DISPLAY_SCREEN('HELP2.SCR',file_existSS); {display screen}
  if not file_existSS then
    BEGIN
      gotoxy(1,1);
      write('Missing screen Helpscreen');
    end;
 Continue;
 screenSS:=100;
end; {Helpscreen}

{ ********************************************************************
*   Print: This routine is to be used to emulate the pressing of    *
*   the <SHIFT> and <PrtSc> keys.  It works by using the system     *
*   interupt number five (5) in memory location hex CD05.           *
*********************************************************************}

overlay procedure Print;
TYPE
  regpack = record
              ax,bx,cx,dx,bp,si,di,ds,es,flags: integer;
            END;
VAR
  recpack : regpack;             {assign record}

BEGIN
  intr($cd05,recpack);           {call interrupt}
END;
```

```
{****************************************************************************
* PROCEDURE EXITSCREEN DETERMINES IF USER ACTUALLY INTENDED ON EXITING THE *
* PROGRAM-IF NOT ALLOWS USER TO RETURN TO PREVIOUS SCREEN.                 *
****************************************************************************}
overlay procedure ExitScreen;

 Begin
   exit_response:='N';
   flashup('S=exit');
   gotoxy(1,25);
   clreol;
   GETITEM(63,9,1,'Y',EXIT_RESPONSE,
   'U','','',retSS,TRUE,0,7);
   if  (Exit_Response = 'Y') then
     begin
       exit:= true;        { Exits the program }
       textcolor(15);
       textbackground(0);
       clrscr;
     end;
END; { PROCEDURE ExitScreen }


{*********************************************************************
*  PROCEDURE WhatMenu functions to allow smooth passage from one     *
*  screen to another. It also allows editing of input values, calls *
*  the procedure Print to print the screen bieng viewed to a print-  *
*  er, allows exiting from the program and calls the Help Screens.   *
*********************************************************************}
PROCEDURE WhatMenu(VAR screen:integer; var bactive: boolean; var ub: integer);

VAR
  retSS: Str2;

BEGIN
 Line25(' F1=Help F2=Prev. Screen F3=Edit F7=Print F8=Next Screen ');
 if (not bactive)
   then ub:=1;
  repeat
    Function_Key_Pressed(retSS);
    CASE retSS[1] of
      f1ss:  Helpscreen;
      f2ss:  if (screen > 1)
               then screen:=screen-1 { previous screen }
               else begin
                    bactive:= false;
                    screen:=1;
                    end;
      f3ss:  edit:= true;
      f7ss: Print;
      f8ss:  if (screen < Ub)
               then screen:=screen+1  {next screen}
               else begin
```

```
          bactive:=false;
          screen:=1;
          end;
  else  Beep(true);
  END;
 until retSS[1] in [f1ss,f2ss,f3ss,f8ss];
 if retSS[1]<>f3ss
  then edit:=false;

 gotoxy(1,25);
 clreol;
END;

{*********************************************************************
* Procedure Directory checks the specified defaultdrive for the     *
* files with the given file extension. The procedure uses MsDos to  *
* search and find the specified files. The procedure then prints the*
* file names on the screen. This procedure was taken from the Turbo *
* Tutor manual section 20-4                                          *
*********************************************************************}


 procedure Directory(DefaultDrive:Str2;
          fileext:str3;
          ymin,ymax:integer);
TYPE
 Char12arr          = array [ 1..12 ] of Char;
 String20           = string[ 20 ];
 RegRec =
  record
   AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : Integer;
  END;

VAR
 Regs               : RegRec;
 DTA                : array [ 1..43 ] of Byte;
 Mask               : Char12arr;
 NamR               : String20;
 Error, I, X, Y     : Integer;
 originaldirectory  : Str80;
 ext                : char3;

 BEGIN { main body of PROCEDURE Directory }
for i:= 1 to 3 do
 ext[i]:= fileext[i];



  FillChar(DTA,SizeOf(DTA),0);        { Initialize the DTA buffer }
  FillChar(Mask,SizeOf(Mask),0);      { Initialize the mask }
  FillChar(NamR,SizeOf(NamR),0);      { Initialize the file name }
  getdir(0,originaldirectory);        { Get the original directory}
  Chdir(DefaultDrive);
  Regs.AX := $1A00;         { Function used to set the DTA }
```

```
   Regs.DS := Seg(DTA);      { store the parameter segment in DS }
   Regs.DX := Ofs(DTA);      {   "    "       "    offset in DX }
   MSDos(Regs);              { Set DTA location }
Error := 0;
   Mask := '????????.???';    { Use global search }
   mask[10] := ext[1];
   mask[11] := ext[2];
   mask[12] := ext[3];
   Regs.AX := $4E00;          { Get first directory entry }
   Regs.DS := Seg(Mask);      { Point to the file Mask }
   Regs.DX := Ofs(Mask);
   Regs.CX := 22;             { Store the Option }
   MSDos(Regs);               { Execute MSDos call }
{IOcheck;}
   Error := Regs.AX and $FF;  { Get Error return }


  I := 1;                    { initialize 'I' to the first element }
   x:=5;
   y:=ymin;
   if (Error = 0) then
     repeat
       NamR[I] := Chr(Mem[Seg(DTA):Ofs(DTA)+29+I]);
       I := I + 1;
     until not (NamR[I-1] in [' '..'~']) or (I>20);

   NamR[0] := Chr(I-1);          { set string length because assigning }
                             { by element does not set length }

    gotoxy(x,y);
    write(namr);
    x:=x+15;

   while (Error = 0) do BEGIN
     Error := 0;
     Regs.AX := $4F00;           { Function used to get the next }
                                 { directory entry }
     Regs.CX := 22;              { Set the file Option }
     MSDos( Regs );              { Call MSDos }
     Error := Regs.AX and $FF;   { get the Error return }
     I := 1;
     repeat
       NamR[I] := Chr(Mem[Seg(DTA):Ofs(DTA)+29+I]);
       I := I + 1;
     until not (NamR[I-1] in [' '..'~'] ) or (I > 20);
     NamR[0] := Chr(I-1);
     if (Error = 0)
       then
         BEGIN
           if x>65
             then
               BEGIN
                 x:=5;
                 y:=y+1;
```

```
              END;
            if y>ymax
              then
                BEGIN
                  Line25('Press any key to continue...');
                  presskey;
                  window(2,ymin,79,ymax);
                  clrscr;
                  window(1,1,80,25);
                  x:=5;
                  y:=ymin;
              END;
            gotoxy(x,y);
            write(namr);
            x:=x+15;
        END;
    END;
    chdir(originaldirectory);
  END; { of PROCEDURE Directory }

  {*********************************************************************
  * Procedure LoadBEV gets the specified file on the specified de-     *
  * fault drive and puts it into a record or variables. Commands to    *
  * turn off the systems input/output error checking routine are found *
  * in this procedure in order to keep fatal program terminating errors*
  * from occuring. Procedure IOcheck replaces the systems I/O check.   *
  *********************************************************************}
 overlay procedure LoadBEV(FileName:str80;DefaultDrive:Str1);
  {$I-} {turns off system I/O error check procedure}
  VAR
    BevFile:  text;
    Drive: Str2;
    OriginalDirectory: Str80;
    i:integer;

  BEGIN
    GetDir(0,originaldirectory);
    Drive:=DefaultDrive+':';
    ChDir(Drive);
    FileName:=FileName+'.'+fileext;
    assign(BevFile,FileName);
    reset(BevFile);
    IOcheck(1);
    If not IOerr then
      begin
      { REQUIRES READLN STATEMENTS OF WHAT I WILL LOAD  }
      { ENDOGENOUS AND EXOGENOUS VARIABLES }

readln(BevFile,PCNT_OF_XPORTS[S]);
readln(BevFile,APCT[S],AHCT[S],PYIELD[S],CORNYT[S],WHTYT[S],GROSS[S]);
readln(BevFile,BAL[S],ARP[S],DIVERSION[S]);
readln(BevFile,DVPCN[S],DVRCN[S],DIVPAY[S],DPAY[S],DEFPAY[S],BEGINSTOCKS[S]);
readln(BevFile,MIN_FREE_STOCKS[S],END_STOCKS[S],PRODUCT[S],IMPORTS[S]);
readln(BevFile,TSUPPLY[S],ONEPH[S],TWOPH[S],THREEPH[S]);
readln(BevFile,FOURPH[S],FIVEPH[S],SIXPH[S]);
readln(BevFile,SEVENPH[S],DANUM[S],COFNUM[S],OBCNUM[S],HPCNUM[S]);
readln(BevFile,BRONUM[S],TURNUM[S],HOGNUM[S],DAIRYCONS[S],CATONFED[S]);
readln(BevFile,OTHERBEEFCAT[S],CHICKENCONS[S],BROILERCONS[S],TURKEYCONS[S]);
readln(BevFile,HOGCONS[S],OTHERCONS[S],FEEDT[S],HFCS[S],SEED_USE[S],FSI_USE[S]);
readln(BevFile,DOMESTIC_USE[S],SETASIDE_ACRES[S],VCCN[S],VCHW[S],C_TARGETP[S]);
readln(BevFile,CORN_LOAN_RATE[S],PARREV[S],REVCOR[S],REVWHT[S],YEAR[S]);
readln(BevFile,YEARB[S]);
readln(BevFile,CORNPT[S],RFARMP[S],MILKP[S],WHTPT[S],BEFSTRP[S],BCOWP[S]);
readln(BevFile,SMPT[S],HOGP[S],IPOP[S],IGDP[S],IPRO[S],IFP[S]);
readln(BevFile,IHA[S]);
readln(BevFile,IWP[S],IGDPPC[S],IPROPC[S],INIPC[S],NXXRF[S]);
readln(BevFile,NXXRW[S]);
readln(BevFile,NMXRF[S],NMXRW[S],CPISB[S],CPIDM[S],CPILO[S],CPINI[S]);
readln(BevFile,CPILD[S]);
readln(BevFile,CPIBR[S],CPIAR[S],CPIAU[S],CPICA[S]);
readln(BevFile,CPICH[S],CPIT[S],GDPCH[S]);
readln(BevFile,GDPSB[S],GDPDM[S],GDPLO[S],GDPNI[S],GDPLD[S],GDPBR[S]);
readln(BevFile,GDPAR[S]);
readln(BevFile,GDPAU[S],GDPCA[S],FP[S],WP[S],EYIELD[S],EFREV[S]);
readln(BevFile,EWREV[S]);
readln(BevFile,WYAR[S],WYAU[S],WYCA[S],FYAR[S],FYAU[S],FYCA[S]);
readln(BevFile,XRAR[S],XRAU[S]);
readln(BevFile,XRCA[S],E_END_STK[S],E_NET_EXPORT[S],EHA[S],ECON[S],ECONPC[S]);
readln(BevFile,EES[S],EESPC[S],ESPLYPC[S],EPRO[S],EPROPC[S]);
readln(BevFile,EUDPC[S]);
readln(BevFile,ENE[S],EXPORTS[S],EGDP[S],EPOP[S]);
readln(BevFile,EWYIELD[S]);
readln(BevFile,TOTAL_USE[S],PERCENT_USE[S],CCCSTOCKS[S],DCCCSTOCKS[S]);
readln(BevFile,FORSTOCKS[S],DFORSTOCKS[S],FREESTOCKS[S],GOVTSTOCKS[S]);
readln(BevFile,POLICY_STOCKS[S]);
readln(BevFile,ONE[S],TWO[S],THREE[S]);
readln(BevFile,FOUR[S],FIVE[S],SIX[S]);
readln(BevFile,SEVEN[S],EIGHT[S],RESIDUAL[S],FUEL_USE[S]);
readln(BevFile,INI_IN_BU[S],ECON_IN_BU[S],EES_IN_BU[S]);
readln(BevFile,ESPLY_IN_BU[S],ENE_IN_BU[S]);

readln(BevFile,DV73,DV82,DV83,DV84,DV74ON);
readln(BevFile,YEAR1B,YEAR2B,YEAR3B,YEAR4B);
readln(BevFile,CHWT,SBWT,DMWT,LOWT,NIWT);
readln(BevFile,LDWT,BRWT,ARWT,AUWT,CAWT);
readln(BevFile,FPLR,FPLR2,FPLR3);
readln(BevFile,TRIGGERP);

          { ADD FACTOR VARIABLES }
readln(BevFile,APCT_AF[S],AHCT_AF[S],CORNYT_AF[S],ONEPH_AF[S],TWOPH_AF[S]);
readln(BevFile,THREEPH_AF[S]);
readln(BevFile,FOURPH_AF[S]);
readln(BevFile,FIVEPH_AF[S],SIXPH_AF[S],SEVENPH_AF[S],EIGHT_AF[S],INIPC_AF[S]);
readln(BevFile,EYIELD_AF[S]);
readln(BevFile,EWYIELD_AF[S],EHA_AF[S],ECONPC_AF[S],EESPC_AF[S],DCCCSTOCKS_AF[S]);
readln(BevFile,DFORSTOCKS_AF[S],FREESTOCKS_AF[S]);

          { OVERRIDE VARIABLES }
```

```
readln(BevFile,CORNYT_OR[S],APCT_OR[S],AHCT_OR[S],BEGINSTOCKS_OR[S],PRODUCT_OR[S]);
readln(BevFile,TSUPPLY_OR[S]);
readln(BevFile,IMPORTS_OR[S],ONEPH_OR[S],DAIRYCONS_OR[S],TWOPH_OR[S],CATONFED_OR[S]);
readln(BevFile,THREEPH_OR[S]);
readln(BevFile,OTHERBEEFCAT_OR[S],FOURPH_OR[S],CHICKENCONS_OR[S],FIVEPH_OR[S]);
readln(BevFile,BROILERCONS_OR[S]);
readln(BevFile,SIXPH_OR[S],TURKEYCONS_OR[S],SEVENPH_OR[S],HOGCONS_OR[S]);
readln(BevFile,OTHER_UNALLOC_OR[S]);
readln(BevFile,FEEDT_OR[S],DOMESTIC_USE_OR[S]);
readln(BevFile,INI_IN_BU_OR[S],ESPLY_IN_BU_OR[S],ECON_IN_BU_OR[S]);
readln(BevFile,EES_IN_BU_OR[S],ENE_IN_BU_OR[S]);
readln(BevFile,EXPORTS_OR[S],TOTAL_USE_OR[S],CCCSTOCKS_OR[S],FORSTOCKS_OR[S]);
readln(BevFile,FREESTOCKS_OR[S]);
readln(BevFile,ENDSTOCKS_OR[S]);

          { FLAG VARIABLES FOR OVERRIDE VALUES }
readln(BevFile,CORNYT_FLAG);
readln(BevFile,APCT_FLAG);
readln(BevFile,AHCT_FLAG);
readln(BevFile,BEGINSTOCKS_FLAG);
readln(BevFile,PRODUCT_FLAG);
readln(BevFile,TSUPPLY_FLAG);
readln(BevFile,IMPORTS_FLAG);
readln(BevFile,FEEDT_FLAG);
readln(BevFile,DOMESTIC_USE_FLAG);
readln(BevFile,INI_IN_BU_FLAG);
readln(BevFile,ESPLY_IN_BU_FLAG);
readln(BevFile,ECON_IN_BU_FLAG);
readln(BevFile,EES_IN_BU_FLAG);
readln(BevFile,ENE_IN_BU_FLAG);
readln(BevFile,EXPORTS_FLAG);
readln(BevFile,TOTAL_USE_FLAG);
readln(BevFile,CCCSTOCKS_FLAG);
readln(BevFile,FORSTOCKS_FLAG);
readln(BevFile,FREESTOCKS_FLAG);
readln(BevFile,ENDSTOCKS_FLAG);
      IOcheck(2);
      end;
    close(BevFile);

    {$I+} {turns system I/O error check back on}
    ChDir(originaldirectory);
  END;


  {*********************************************************************
  * Procedure SaveBEV functions the same as Load exept that SaveBEV   *
  * creates and writes to a specified file.                           *
  *********************************************************************}
  overlay procedure SaveBEV(FileName:str80;DefaultDrive:Str1);

    VAR
        BevFile:  text;
        Drive: Str2;
        originaldirectory: Str80;
        fileexists :boolean;
        overwriteyn :str1;
        i:integer;

  BEGIN
    fileexists:=false;
    GetDir(0,originaldirectory);
    Drive:=DefaultDrive+':';
    ChDir(drive);
    FileName:=FileName+'.'+fileext;
   {$I-} {turns system I/O error trapping procedure off}
    assign(BevFile,FileName);
    reset(bevfile);
    if IOresult=0
        then begin {file exists}
              Line25('The file exists. Do you wish to OVERWRITE it?  (Y/N)');
              GETITEM (75,25,1,'Y',overwriteyn,
               'U','','',retSS,true,0,7);
                if overwriteyn='N'
                   then begin
                         fileexists:=true;
                         close (BEVFILE);
                        end;
            end;
   if (not fileexists)
       then begin
    rewrite(BevFile);
    IOcheck(1);
   if not IOerr then
     begin
      { REQUIRES WRITELN STATEMENTS FOR WHAT WILL BE SAVED  }
      { ENDOGENOUS AND EXOGENOUS VARIABLES }

writeln(BevFile,PCNT_OF_XPORTS[S]:9:5);
writeln(BevFile,APCT[S]:9:2,AHCT[S]:9:2,PYIELD[S]:9:2,CORNYT[S]:9:2,WHTYT[S]:9:2,GROSS[S]:9:2);
writeln(BevFile,BAL[S]:9:3,ARP[S]:9:3,DIVERSION[S]:9:3);
writeln(BevFile,DVPCN[S]:9:2,DVRCN[S]:9:3,DIVPAY[S]:9:3,DPAY[S]:9:2,DEFPAY[S]:9:3,BEGINSTOCKS[S]:9:2
);
writeln(BevFile,MIN_FREE_STOCKS[S]:9:2,END_STOCKS[S]:9:2,PRODUCT[S]:9:2,IMPORTS[S]:9:2);
writeln(BevFile,TSUPPLY[S]:9:2,ONEPH[S]:9:3,TWOPH[S]:9:3,THREEPH[S]:9:3);
writeln(BevFile,FOURPH[S]:9:3,FIVEPH[S]:9:3,SIXPH[S]:9:3);
writeln(BevFile,SEVENPH[S]:9:3,DANUM[S]:9:3,COFNUM[S]:9:3,OBCNUM[S]:9:3,HPCNUM[S]:9:3);
writeln(BevFile,BRONUM[S]:9:3,TURNUM[S]:9:3,HOGNUM[S]:9:3,DAIRYCONS[S]:9:2,CATONFED[S]:9:2);
writeln(BevFile,OTHERBEEFCAT[S]:9:2,CHICKENCONS[S]:9:2,BROILERCONS[S]:9:2,TURKEYCONS[S]:9:2);
writeln(BevFile,HOGCONS[S]:9:2,OTHERCONS[S]:9:2,FEEDT[S]:9:2,HFCS[S]:9:2,SEED_USE[S]:9:2,FSI_USE[S]:
9:2);
writeln(BevFile,DOMESTIC_USE[S]:9:2,SETASIDE_ACRES[S]:9:2,VCCN[S]:9:2,VCHW[S]:9:2,C_TARGETP[S]:9:2);
writeln(BevFile,CORN_LOAN_RATE[S]:9:2,PARREV[S]:9:2,REVCOR[S]:9:2,REVWHT[S]:9:2,YEAR[S]:3:0);
writeln(BevFile,YEARB[S]:3:0);
writeln(BevFile,CORNPT[S]:9:2,RFARMP[S]:9:2,MILKP[S]:9:2,WHTPT[S]:9:2,BEFSTRP[S]:9:2,BCOWP[S]:9:2);
writeln(BevFile,SMPT[S]:9:2,HOGP[S]:9:2,IPOP[S]:11:2,IGDP[S]:11:6,IPRO[S]:11:2,IFP[S]:9:3);
writeln(BevFile,IHA[S]:11:2);
writeln(BevFile,IWP[S]:9:3,IGDPPC[S]:11:6,IPROPC[S]:11:6,INIPC[S]:11:3,NXXRF[S]:9:6);
writeln(BevFile,NXXRW[S]:9:6);
writeln(BevFile,NMXRF[S]:9:6,NMXRW[S]:9:6,CPISB[S]:9:3,CPIDM[S]:9:3,CPILO[S]:9:3,CPINI[S]:9:3);
writeln(BevFile,CPILD[S]:9:3);
writeln(BevFile,CPIBR[S]:9:3,CPIAR[S]:11:2,CPIAU[S]:11:2,CPICA[S]:11:2);
writeln(BevFile,CPICH[S]:9:3,CPIT[S]:9:3,GDPCH[S]:9:6);
writeln(BevFile,GDPSB[S]:9:6,GDPDM[S]:9:6,GDPLO[S]:9:6,GDPNI[S]:9:6,GDPLD[S]:9:6,GDPBR[S]:9:3);
writeln(BevFile,GDPAR[S]:9:3);
writeln(BevFile,GDPAU[S]:9:6,GDPCA[S]:9:6,FP[S]:9:3,WP[S]:9:3,EYIELD[S]:9:6,EFREV[S-1]:9:6);
writeln(BevFile,EWREV[S-1]:9:6);
writeln(BevFile,WYAR[S]:11:5,WYAU[S]:11:5,WYCA[S]:11:5,FYAR[S]:11:5,FYAU[S]:11:5,FYCA[S]:11:5);
writeln(BevFile,XRAR[S]:11:5,XRAU[S]:11:5);
writeln(BevFile,XRCA[S]:7:5,E_END_STK[S]:9:2,E_NET_EXPORT[S]:9:2,EHA[S]:11:2,ECON[S]:11:2,ECONPC[S]:
11:3);
writeln(BevFile,EES[S]:11:2,EESPC[S]:11:6,ESPLYPC[S]:11:6,EPRO[S]:11:2,EPROPC[S]:9:6);
writeln(BevFile,EUDPC[S]:11:6);
writeln(BevFile,ENE[S]:11:2,EXPORTS[S]:9:2,EGDP[S]:11:6,EPOP[S]:11:2);
writeln(BevFile,EWYIELD[S]:9:6);
writeln(BevFile,TOTAL_USE[S]:9:2,PERCENT_USE[S]:9:2,CCCSTOCKS[S]:9:2,DCCCSTOCKS[S]:9:2);
writeln(BevFile,FORSTOCKS[S]:9:2,DFORSTOCKS[S]:9:2,FREESTOCKS[S]:9:2,GOVTSTOCKS[S]:9:2);
writeln(BevFile,POLICY_STOCKS[S]:9:2);
writeln(BevFile,ONE[S]:9:3,TWO[S]:9:3,THREE[S]:9:3);
writeln(BevFile,FOUR[S]:9:3,FIVE[S]:9:3,SIX[S]:9:3);
writeln(BevFile,SEVEN[S]:9:3,EIGHT[S]:9:3,RESIDUAL[S]:9:2,FUEL_USE[S]:9:2);
writeln(BevFile,INI_IN_BU[S]:9:2,ECON_IN_BU[S]:9:2,EES_IN_BU[S]:9:2);
writeln(BevFile,ESPLY_IN_BU[S]:9:2,ENE_IN_BU[S]:9:2);

writeln(BevFile,DV73:9:2,DV82:9:2,DV83:9:2,DV84:9:2,DV74ON:9:2);
writeln(BevFile,YEAR1B:9:2,YEAR2B:9:2,YEAR3B:9:2,YEAR4B:9:2);
writeln(BevFile,CHWT:9:4,SBWT:9:4,DMWT:9:4,LOWT:9:4,NIWT:9:4);
writeln(BevFile,LDWT:9:4,BRWT:9:4,ARWT:9:4,AUWT:9:4,CAWT:9:4);
writeln(BevFile,FPLR:9:2,FPLR2:9:2,FPLR3:9:2);
writeln(BevFile,TRIGGERP:9:2);

           { ADD FACTOR VARIABLES }
writeln(BevFile,APCT_AF[S]:9:3,AHCT_AF[S]:9:3,CORNYT_AF[S]:9:3,ONEPH_AF[S]:9:3,TWOPH_AF[S]:9:3);
writeln(BevFile,THREEPH_AF[S]:9:3);
writeln(BevFile,FOURPH_AF[S]:9:3);
writeln(BevFile,FIVEPH_AF[S]:9:3,SIXPH_AF[S]:9:3,SEVENPH_AF[S]:9:3,EIGHT_AF[S]:9:3,INIPC_AF[S]:9:5);
writeln(BevFile,EYIELD_AF[S]:9:3);
writeln(BevFile,EWYIELD_AF[S]:9:3,EHA_AF[S]:11:3,ECONPC_AF[S]:9:5,EESPC_AF[S]:9:5,DCCCSTOCKS_AF[S]:9
:3);
writeln(BevFile,DFORSTOCKS_AF[S]:9:3,FREESTOCKS_AF[S]:9:3);

         { OVERRIDE VARIABLES }
writeln(BevFile,CORNYT_OR[S]:9:2,APCT_OR[S]:9:2,AHCT_OR[S]:9:2,BEGINSTOCKS_OR[S]:9:2,PRODUCT_OR[S]:9
:2);
writeln(BevFile,TSUPPLY_OR[S]:9:2);
writeln(BevFile,IMPORTS_OR[S]:9:2,ONEPH_OR[S]:9:3,DAIRYCONS_OR[S]:9:2,TWOPH_OR[S]:9:3,CATONFED_OR[S]
:9:2);
writeln(BevFile,THREEPH_OR[S]:9:3);
writeln(BevFile,OTHERBEEFCAT_OR[S]:9:2,FOURPH_OR[S]:9:3,CHICKENCONS_OR[S]:9:2,FIVEPH_OR[S]:9:3);
writeln(BevFile,BROILERCONS_OR[S]:9:2);
writeln(BevFile,SIXPH_OR[S]:9:3,TURKEYCONS_OR[S]:9:2,SEVENPH_OR[S]:9:3,HOGCONS_OR[S]:9:2);
writeln(BevFile,OTHER_UNALLOC_OR[S]:9:2);
writeln(BevFile,FEEDT_OR[S]:9:2,DOMESTIC_USE_OR[S]:9:2);
writeln(BevFile,INI_IN_BU_OR[S]:9:2,ESPLY_IN_BU_OR[S]:9:2,ECON_IN_BU_OR[S]:9:2);
writeln(BevFile,EES_IN_BU_OR[S]:9:2,ENE_IN_BU_OR[S]:9:2);
writeln(BevFile,EXPORTS_OR[S]:9:2,TOTAL_USE_OR[S]:9:2,CCCSTOCKS_OR[S]:9:2,FORSTOCKS_OR[S]:9:2);
writeln(BevFile,FREESTOCKS_OR[S]:9:2);
writeln(BevFile,ENDSTOCKS_OR[S]:9:2);

           { FLAG VARIABLES FOR OVERRIDE VALUES }
writeln(BevFile,CORNYT_FLAG);
writeln(BevFile,APCT_FLAG);
writeln(BevFile,AHCT_FLAG);
writeln(BevFile,BEGINSTOCKS_FLAG);
writeln(BevFile,PRODUCT_FLAG);
writeln(BevFile,TSUPPLY_FLAG);
writeln(BevFile,IMPORTS_FLAG);
writeln(BevFile,FEEDT_FLAG);
writeln(BevFile,DOMESTIC_USE_FLAG);
writeln(BevFile,INI_IN_BU_FLAG);
writeln(BevFile,ESPLY_IN_BU_FLAG);
writeln(BevFile,ECON_IN_BU_FLAG);
writeln(BevFile,EES_IN_BU_FLAG);
writeln(BevFile,ENE_IN_BU_FLAG);
writeln(BevFile,EXPORTS_FLAG);
writeln(BevFile,TOTAL_USE_FLAG);
writeln(BevFile,CCCSTOCKS_FLAG);
writeln(BevFile,FORSTOCKS_FLAG);
writeln(BevFile,FREESTOCKS_FLAG);
writeln(BevFile,ENDSTOCKS_FLAG);
     IOcheck(2);
     end;
   close(BevFile);

  end;
   {$I+} {turns system error trapping back on}
   ChDir(originaldirectory);
   Line25('                                                                      ');
  END;

  {**********************************************************************
  * PROCEDURE CheckDisk checks if the specified defaultdrive is a       *
  * legal drive. If it is it will call the Procedure Directory and      *
  * continue normally. Else the procedure will write an error mes-      *
  * age.                                                                *
  *********************************************************************}

 overlay procedure CheckDisk(defaultdrive:strl);
 Var drive: Str2;
     result:real;
     temp: Strl;

Begin {checkdisk}
  drive:=defaultdrive+':';
  result:=freediskspace(drive);
  GetDir(0,temp);
```

```
if((result<=0)or(defaultdrive>'E'))and((result<=0)or(defaultdrive<>temp))
       then begin
          Line25('Disk Full/Invalid Drive-press any key to contiue');
          Presskey;
          gotoxy(1,25);
          clreol;
          if (retrieveSS) then varSS:=screen_fieldSS;
        end
    else
        if (retrieveSS) then Directory (drive,fileext,8,16);
 end;{checkdisk}

{overlay Procedure end_on_line_help;}
 { begin}
 { flashup('C=ALL');}
 { flashup('K=CLEAR');}
 { if retrieveSS then Line25('Press <ESC> for Command Line');}
 { end;}


{overlay Procedure on_line_help(window_name: str8);}
(*{   begin}
{   if retrieveSS}
{     then begin}
{          flashup('K={F10},'+window_name);}
{          Line25('On line help available. Press <F10>');}
{          end;}
{   end;}*)
        ***** End of Include file MENU.UTL *****
        ***** Beginning of Include file INITIAL.INC *****
{$I initial.inc}
overlay procedure initial;  { A PROCEDURE TO INITIALIZE ALL VARIABLES TO ZERO }
begin
for i:=1 to 4 do
   begin
     PCNT_OF_XPORTS[i]:=0.0;
     APCT[i]:=0.0;
     AHCT[i]:=0.0;
     PYIELD[i]:=0.0;
     CORNYT[i]:=0.0;
     WHTYT[i]:=0.0;
     GROSS[i]:=0.0;
     ARP[i]:=0.0;
     DIVERSION[i]:=0.0;
     BAL[i]:=0.0;
     DVPCN[i]:=0.0;
     DVRCN[i]:=0.0;
     DIVPAY[i]:=0.0;
     DPAY[i]:=0.0;
     DEFPAY[i]:=0.0;
     BEGINSTOCKS[i]:=0.0;
     MIN_FREE_STOCKS[i]:=0.0;
     END_STOCKS[i]:=0.0;
     PRODUCT[i]:=0.0;
```

```
IMPORTS[i]:=0.0;
TSUPPLY[i]:=0.0;
ONE[i]:=0.0;
ONEPH[i]:=0.0;
TWO[i]:=0.0;
TWOPH[i]:=0.0;
THREE[i]:=0.0;
THREEPH[i]:=0.0;
FOUR[i]:=0.0;
FOURPH[i]:=0.0;
FIVE[i]:=0.0;
FIVEPH[i]:=0.0;
SIX[i]:=0.0;
SIXPH[i]:=0.0;
SEVEN[i]:=0.0;
SEVENPH[i]:=0.0;
EIGHT[i]:=0.0;
DANUM[i]:=0.0;
COFNUM[i]:=0.0;
OBCNUM[i]:=0.0;
HPCNUM[i]:=0.0;
BRONUM[i]:=0.0;
TURNUM[i]:=0.0;
HOGNUM[i]:=0.0;
DAIRYCONS[i]:=0.0;
CATONFED[i]:=0.0;
OTHERBEEFCAT[i]:=0.0;
CHICKENCONS[i]:=0.0;
BROILERCONS[i]:=0.0;
TURKEYCONS[i]:=0.0;
HOGCONS[i]:=0.0;
OTHERCONS[i]:=0.0;
FEEDT[i]:=0.0;
HFCS[i]:=0.0;
SEED_USE[i]:=0.0;
RESIDUAL[i]:=0.0;
FUEL_USE[i]:=0.0;
FSI_USE[i]:=0.0;
DOMESTIC_USE[i]:=0.0;
SETASIDE_ACRES[i]:=0.0;
VCCN[i]:=0.0;
VCHW[i]:=0.0;
C_TARGETP[i]:=0.0;
CORN_LOAN_RATE[i]:=0.0;
PARREV[i]:=0.0;
REVCOR[i]:=0.0;
REVWHT[i]:=0.0;
YEAR[i]:=0.0;
YEARB[i]:=0.0;
CORNPT[i]:=0.0;
RFARMP[i]:=0.0;
MILKP[i]:=0.0;
WHTPT[i]:=0.0;
BEFSTRP[i]:=0.0;
```

```
BCOWP[i]:=0.0;
SMPT[i]:=0.0;
HOGP[i]:=0.0;
IPOP[i]:=0.0;
IGDP[i]:=0.0;
IHA[i]:=0.0;
IYIELD[i]:=0.0;
IPRO[i]:=0.0;
IFP[i]:=0.0;
IWP[i]:=0.0;
IGDPPC[i]:=0.0;
IPROPC[i]:=0.0;
INIPC[i]:=0.0;
INI_IN_BU[i]:=0.0;
NXXRF[i]:=0.0;
NXXRW[i]:=0.0;
NMXRF[i]:=0.0;
NMXRW[i]:=0.0;
CPISB[i]:=0.0;
CPIDM[i]:=0.0;
CPILO[i]:=0.0;
CPINI[i]:=0.0;
CPILD[i]:=0.0;
CPIBR[i]:=0.0;
CPIAR[i]:=0.0;
CPIAU[i]:=0.0;
CPICA[i]:=0.0;
CPICH[i]:=0.0;
CPIT[i]:=0.0;
GDPCH[i]:=0.0;
GDPSB[i]:=0.0;
GDPDM[i]:=0.0;
GDPLO[i]:=0.0;
GDPNI[i]:=0.0;
GDPLD[i]:=0.0;
GDPBR[i]:=0.0;
GDPAR[i]:=0.0;
GDPAU[i]:=0.0;
GDPCA[i]:=0.0;
FP[i]:=0.0;
WP[i]:=0.0;
EYIELD[i]:=0.0;
EFREV[i]:=0.0;
EWREV[i]:=0.0;
WYAR[i]:=0.0;
WYAU[i]:=0.0;
WYCA[i]:=0.0;
FYAR[i]:=0.0;
FYAU[i]:=0.0;
FYCA[i]:=0.0;
XRAR[i]:=0.0;
XRAU[i]:=0.0;
XRCA[i]:=0.0;
E_END_STK[i]:=0.0;
```

```
E_NET_EXPORT[i]:=0.0;
EHA[i]:=0.0;
ECON[i]:=0.0;
ECON_IN_BU[i]:=0.0;
ECONPC[i]:=0.0;
EES[i]:=0.0;
EES_IN_BU[i]:=0.0;
EESPC[i]:=0.0;
ESPLY_IN_BU[i]:=0.0;
ESPLYPC[i]:=0.0;
EPRO[i]:=0.0;
EPROPC[i]:=0.0;
EUDPC[i]:=0.0;
ENE[i]:=0.0;
ENE_IN_BU[i]:=0.0;
EXPORTS[i]:=0.0;
EGDP[i]:=0.0;
EPOP[i]:=0.0;
EWYIELD[i]:=0.0;
TOTAL_USE[i]:=0.0;
PERCENT_USE[i]:=0.0;
CCCSTOCKS[i]:=0.0;
DCCCSTOCKS[i]:=0.0;
FORSTOCKS[i]:=0.0;
DFORSTOCKS[i]:=0.0;
FREESTOCKS[i]:=0.0;
GOVTSTOCKS[i]:=0.0;
POLICY_STOCKS[i]:=0.0;
       { ADD FACTORS }
APCT_AF[i]:=0.0;
AHCT_AF[i]:=0.0;
CORNYT_AF[i]:=0.0;
ONEPH_AF[i]:=0.0;
TWOPH_AF[i]:=0.0;
THREEPH_AF[i]:=0.0;
FOURPH_AF[i]:=0.0;
FIVEPH_AF[i]:=0.0;
SIXPH_AF[i]:=0.0;
SEVENPH_AF[i]:=0.0;
EIGHT_AF[i]:=0.0;
INIPC_AF[i]:=0.0;
IYIELD_AF[i]:=0.0;
EYIELD_AF[i]:=0.0;
EWYIELD_AF[i]:=0.0;
EHA_AF[i]:=0.0;
ECONPC_AF[i]:=0.0;
EESPC_AF[i]:=0.0;
DCCCSTOCKS_AF[i]:=0.0;
DFORSTOCKS_AF[i]:=0.0;
FREESTOCKS_AF[i]:=0.0;
      { OVERRIDE VARIABLES }
CORNYT_OR[i]:=0.0;
APCT_OR[i]:=0.0;
AHCT_OR[i]:=0.0;
```

```
      BEGINSTOCKS_OR[i]:=0.0;
      PRODUCT_OR[i]:=0.0;
      TSUPPLY_OR[i]:=0.0;
      IMPORTS_OR[i]:=0.0;
      ONEPH_OR[i]:=0.0;
      DAIRYCONS_OR[i]:=0.0;
      TWOPH_OR[i]:=0.0;
      CATONFED_OR[i]:=0.0;
      THREEPH_OR[i]:=0.0;
      OTHERBEEFCAT_OR[i]:=0.0;
      FOURPH_OR[i]:=0.0;
      CHICKENCONS_OR[i]:=0.0;
      FIVEPH_OR[i]:=0.0;
      BROILERCONS_OR[i]:=0.0;
      SIXPH_OR[i]:=0.0;
      TURKEYCONS_OR[i]:=0.0;
      SEVENPH_OR[i]:=0.0;
      HOGCONS_OR[i]:=0.0;
      OTHER_UNALLOC_OR[i]:=0.0;
      FEEDT_OR[i]:=0.0;
      DOMESTIC_USE_OR[i]:=0.0;
      INI_IN_BU_OR[i]:=0.0;
      ESPLY_IN_BU_OR[i]:=0.0;
      ECON_IN_BU_OR[i]:=0.0;
      EES_IN_BU_OR[i]:=0.0;
      ENE_IN_BU_OR[i]:=0.0;
      EXPORTS_OR[i]:=0.0;
      TOTAL_USE_OR[i]:=0.0;
      CCCSTOCKS_OR[i]:=0.0;
      FORSTOCKS_OR[i]:=0.0;
      FREESTOCKS_OR[i]:=0.0;
      ENDSTOCKS_OR[i]:=0.0;
   end;
          { VARIABLES FOR THE RANDOM SECTION }
FOR I:=1 TO 30 DO
   BEGIN
      ALPHA[I]:=0.0;
      ALPHA1[I]:=0.0;
      ALPHA2[I]:=0.0;
   END;
BETA:=0.0;
BETA1:=0.0;
BETA2:=0.0;
GAMMA:=0.0;
GAMMA1:=0.0;
GAMMA2:=0.0;
YIELD:=0.0;
YIELD1:=0.0;
YIELD2:=0.0;
Z:=0.0;
Z1:=0.0;
Z2:=0.0;
DV73:=0.0;
DV82:=0.0;
```

```
DV83:=0.0;
DV84:=0.0;
DV74ON:=0.0;
YEAR1B:=0.0;
YEAR2B:=0.0;
YEAR3B:=0.0;
YEAR4B:=0.0;
CHWT:=0.0;
SBWT:=0.0;
DMWT:=0.0;
LOWT:=0.0;
NIWT:=0.0;
LDWT:=0.0;
BRWT:=0.0;
ARWT:=0.0;
AUWT:=0.0;
CAWT:=0.0;
FPLR:=0.0;
FPLR2:=0.0;
FPLR3:=0.0;
TRIGGERP:=0.0;
     { MISCELLANEOUS VARIABLES }
FG:=15;
BG:=0;
flag:=0;
NY:=4;
CORNPT[4]:=3.00;
REDUCE_PRICE:=0.03;
INFO:='SOLVING';
end;
        ***** End of Include file INITIAL.INC *****
        ***** Beginning of Include file RECALC.INC *****
{$I RECALC.INC}
overlay procedure RE_CALCULATE; { THE FORECAST CALCULATIONS }

var conserved_acres: real;   { A LOCAL VARIABLE FOR CONSERVING ACRES
                                (ARP + DIVERSION) }

procedure FREESTOCK_PROC;
        { A PROCEDURE TO CALCULATE THE FREE STOCKS BASED ON
        WEIGHTS USED FOR POLICY }

VAR
CCCWT, FORWT: REAL;

   BEGIN

{        RULE                              CCC WEIGHT       FOR WEIGHT }
{        ----                              ----------       ---------- }
{     1) YEAR <= 76                          0.10              0.0     }
{     2) YEAR >= 77                          0.80              0.75    }
{     3) YEAR >= 77, CCCt <= CCCt-1,                                    }
{        TRIGGER PRICE > $1.55                0.10              0.75    }
{     4) YEAR >= 77, FORt <= FORt-1,                                    }
{        TRIGGER PRICE > $1.55               0.80               0.0     }

      IF ( year[NY] <= 76 ) THEN CCCWT:= 0.1 ELSE CCCWT:= 0.8;

      IF ( year[NY] >= 77 ) AND ( CCCstocks[NY] <= CCCstocks[NY-1] )
          AND ( fplr > 1.55 ) THEN CCCWT:= 0.1;

      IF ( year[NY] <= 76 ) THEN FORWT:= 0.0 ELSE FORWT:= 0.75;

      IF ( FORstocks[NY] <= FORstocks[NY-1] ) AND ( fplr > 1.55 ) THEN
      FORWT:= 0.0;

     FREESTOCKS[NY]:=1127.4231 - 1873.6898 * (( CCCWT * CCCSTOCKS[NY]
                     + FORWT * FORSTOCKS[NY] ) / TOTAL_USE[NY] )
                     - 1301390.2 * ( FPLR / TOTAL_USE[NY] ) + FREESTOCKS_AF[NY];

      IF FREESTOCKS[NY] < MIN_FREE_STOCKS[NY] THEN FREESTOCKS[NY]:=MIN_FREE_STOCKS[NY];

   END;  { AREA WHICH CALCULATES FUNCTION Fx FOR GIVEN WEIGHTS }

BEGIN
           {  DOMESTIC  SUPPLY  }

   { APCT = f ( DV83, LAGGED CORN REVENUE(NON-PARTICIPANTS), WHEAT REVENUE,
            LAGGED CORN REVENUE(PARTICIPANTS) }

   CONSERVED_ACRES:=ARP[NY]+DIVERSION[NY];

   IF APCT_FLAG='N' THEN

  APCT[NY]:= 90.993360 {-6.539 * dv73 + 4.04 * dv79} - 0.1262719 *
  (((WHTPT[NY-1] * WHTYT[NY-1]) - VCHW[NY-1]) / CPIT[NY-1]) - 0.0281170 *
  (((PARREV[NY-1] + DIVPAY[NY-1] + DEFPAY[NY-1]) - VCCN[NY-1]) / CPIT[NY-1])
  - 60.333270 * CONSERVED_ACRES + APCT_AF[NY]

    ELSE APCT[NY]:= APCT_OR[NY];

   IF AHCT_FLAG='N' THEN
    AHCT[NY]:= -8.412272 + 0.8729 * APCT[NY] + 0.1078 * YEAR[NY] + AHCT_AF[NY]
    ELSE AHCT[NY]:= AHCT_OR[NY];

   IF CORNYT_FLAG='N' THEN
    CORNYT[NY]:= -71.3758 + 2.153 * YEAR[NY] + CORNYT_AF[NY]
    ELSE CORNYT[NY]:= CORNYT_OR[NY];

   if year[ny]=85 then beginstocks[NY]:=1648.0
     else BEGINSTOCKS[NY]:=END_STOCKS[NY-1];

   IF PRODUCT_FLAG='N' THEN
   PRODUCT[NY]:= AHCT[NY] * CORNYT[NY]
   ELSE PRODUCT[NY]:=PRODUCT_OR[NY];

   IF IMPORTS_FLAG='Y' THEN IMPORTS[NY]:= IMPORTS_OR[NY];
IF TSUPPLY_FLAG='N' THEN
 TSUPPLY[NY]:= BEGINSTOCKS[NY] + PRODUCT[NY] + IMPORTS[NY]
 ELSE TSUPPLY[NY]:=TSUPPLY_OR[NY];

        {  FEED USE  }

{ DACON/HEAD = f ( TIME, REAL MILK PRICE, REAL CORN FARM PRICE, REAL WHEAT PRICE ) }

 if oneph_or[ny]=0.0 then
 begin
 ONEPH[NY]:= -86.515 + 1.832 * YEAR[NY] + 3.415 * (MILKP[NY] / CPIT[NY])
 - 25.61 * (CORNPT[NY] / CPIT[NY]) + 10.746 * (WHTPT[NY] / CPIT[NY]) + ONEPH_AF[NY];
 dairycons[NY]:=oneph[NY]*danum[NY];
 end
 else dairycons[ny]:=oneph_or[ny]*danum[ny];

{ COFCON/HEAD = f ( TIME, BEEF-STEER/CORN RATIO, LAGGED DEPENDENT ) }

 if twoph_or[ny]=0.0 then
 begin
 TWOPH[NY]:= 240.119 - 2.126 * YEAR[NY] + 1.665 * (BEFSTRP[NY] / CORNPT[NY])
 - 0.659 * TWOPH[NY-1] + TWOPH_AF[NY];
 catonfed[NY]:=twoph[NY]*cofnum[NY];
 end
 else catonfed[ny]:=twoph_or[ny]*cofnum[ny];

{ OBCCON/HEAD = f ( BEEF COW/CORN RATIO, LAGGED REAL SOYBEAN MEAL PRICE ) }

 if threeph_or[ny]=0.0 then
 begin
 THREEPH[NY]:= 2.875 + 0.115 * (BCOWP[NY] / CORNPT[NY])
 + 0.00487 * (SMPT[NY-1] / CPIT[NY-1]) + THREEPH_AF[NY];
 otherbeefcat[NY]:=threeph[NY]*obcnum[NY];
 end
 else otherbeefcat[ny]:=threeph_or[ny]*obcnum[ny];

{ HPCCON/HEAD = f ( REAL CORN FARM PRICE, REAL SOYBEAN MEAL PRICE, REAL WHEAT PRICE,
                    LAGGED DEPENDENT }

 if fourph_or[ny]=0.0 then
 begin
 FOURPH[NY]:= 0.9349 - 0.724 * (CORNPT[NY] / CPIT[NY]) + 0.00121
 * (SMPT[NY] / CPIT[NY]) + 0.3409 * (WHTPT[NY] / CPIT[NY]) + 0.222 * FOURPH[NY-1]
 + FOURPH_AF[NY];
 chickencons[NY]:=fourph[NY]*hpcnum[NY];
 end
 else chickencons[ny]:=fourph_or[ny]*hpcnum[ny];

{BROCON/HEAD = f ( TIME, REAL CORN FARM PRICE, REAL WHEAT PRICE ) }

 if fiveph_or[ny]=0.0 then
 begin
 FIVEPH[NY]:= 0.00245 + 0.00143 * YEAR[NY] - 0.0431 * (CORNPT[NY] / CPIT[NY])
 + 0.0162 * (WHTPT[NY] / CPIT[NY]) + FIVEPH_AF[NY];
 broilercons[NY]:=fiveph[NY]*bronum[NY];
 end
 else broilercons[ny]:=fiveph_or[ny]*bronum[ny];

{ TURCON/HEAD = f ( REAL CORN FARM PRICE, REAL WHEAT PRICE, LAGGED DEPENDENT ) }

 if sixph_or[ny]=0.0 then
 begin
 SIXPH[NY]:= 0.709 - 0.3428 *  (CORNPT[NY] / CPIT[NY]) + 0.126 * (WHTPT[NY] / CPIT[NY])
 + 0.104 * SIXPH[NY-1] + SIXPH_AF[NY];
 turkeycons[NY]:=sixph[NY]*turnum[NY];
 end
 else turkeycons[ny]:=sixph_or[ny]*turnum[ny];

{ HOGCON/HEAD = f ( TIME, REAL HOG PRICE, REAL CORN FARM PRICE, ) }

 if sevenph_or[ny]=0.0 then
 begin
 SEVENPH[NY]:= -14.624 + 0.5 * YEAR[NY] + 0.394 * (HOGP[NY] / CPIT[NY])
 - 7.612 * (CORNPT[NY] / CPIT[NY]) + SEVENPH_AF[NY];
 hogcons[NY]:=sevenph[NY]*hognum[NY];
 end
 else hogcons[ny]:=sevenph_or[ny]*hognum[ny];

{ OUNCON = f ( TIME, REAL CORN FARM PRICE, REAL WHEAT PRICE, DV82, DV84 ) }

if other_unalloc_or[ny]=0.0 then
othercons[NY]:= -412.497 + 6.84 * YEAR[NY] - 63.265 * (CORNPT[NY] / CPIT[NY])
           + 26.481 * (WHTPT[NY] / CPIT[NY]) + 326.601 * DV82
           + 189.265 * DV84 + EIGHT_AF[NY]
else othercons[ny]:=other_unalloc_or[ny];

IF FEEDT_FLAG='N' THEN
 FEEDT[NY]:= dairycons[NY] + catonfed[NY] + otherbeefcat[NY] + chickencons[NY]
 + broilercons[NY] + turkeycons[NY] + hogcons[NY] + othercons[NY]
 ELSE FEEDT[NY]:= FEEDT_OR[NY];

IF DOMESTIC_USE_FLAG='N' THEN
 DOMESTIC_USE[NY]:=FEEDT[NY] + FSI_USE[NY]
 ELSE DOMESTIC_USE[NY]:=DOMESTIC_USE_OR[NY];

        {  EXPORT COMPONENT  }

{ INIPC uses generalized differencing and is adjusted below. }

{ THE SIMPLE LINKAGE BETWEEN FEEDGRAIN PRICE ($/BU.) AND
  FARM PRICE IS AN ADDITIONAL $0.42 }

FOR I:= 1 TO NY DO
   BEGIN
      FP[I]:= CORNPT[I] + 0.42;
   END;

IFP[NY]:= (( (FP[NY]/0.0254) / CPIT[NY] ) * NMXRF[NY] );
```

```
   { THIS TRANSFORMATION PUTS FP IN $/MT }

   IWP[NY]:= (( WP[NY] / CPIT[NY] ) * NMXRW[NY] );

   IYIELD[NY]:=-0.8943774 + 0.0328114 * YEAR[NY] + IYIELD_AF[NY];

          { Where: IYIELD[NY]* = IYIELD[4] - 0.3820334 * IYIELD[3],
          YEAR[4]* = YEAR[4] - 0.3820334 * YEAR[3],
          In this case, Rho is equal to 0.3820334. }

   IPRO[NY]:=(IYIELD[NY] * IHA[NY]);

    INIPC[NY]:= 0.0028 - 0.0000425 * IFP[NY] + 0.0000478 * IWP[NY]
    + 9209967.0 * (IGDP[NY]/IPOP[NY]) - 0.143 * (IPRO[NY]/IPOP[NY]) + INIPC_AF[NY];

          { Where: INIPC[NY]* = INIPC[4] - 0.7277196 * INIPC[3],
          IFP[4]* = IFP[4] - 0.7277196 * IFP[3],
          IWP[4]* = IWP[4] - 0.7277196 * IWP[3],
          IGDPPC[4]* = IGDPPC[4] - 0.7277196 * IGDPPC[3],
          and IPROPC[4]* = IPROPC[4] - 0.7277196 * IPROPC[3].
          In this case, Rho is equal to 0.7277196. }


    IF INI_IN_BU_FLAG='N' THEN
    INI_in_bu[NY]:= ((((INIPC[NY] * IPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
    ELSE INI_IN_BU[NY]:=INI_IN_BU_OR[NY];

    EYIELD[NY]:= -1.26 + 0.0460 * YEAR[NY] + EYIELD_AF[NY];  { EXPORTERS FEEDGRAIN YIELD }

    EWYIELD[NY]:= 0.2247 + 0.0175 * YEAR[NY] + EWYIELD_AF[NY];  {  EXPORTERS WHEAT YIELD  }


   { EXPORTERS HARVESTED ACRES = f ( LAGGED EXPORTERS FEEDGRAIN REVENUE, LAGGED
         EXPORTERS WHEAT REVENUE, LAGGED HARVESTED ACRES, LAGGED ENDING STOCKS }

   EFREV[NY-1]:= ((( (FP[NY-1]/0.0254) / CPIT[NY-1] ) * XRAR[NY-1] * FYAR[NY-1])/CPIAR[NY-1]
              + (( (FP[NY-1]/0.0254) / CPIT[NY-1] ) * XRAU[NY-1] * FYAU[NY-1])/CPIAU[NY-1]
              + (( (FP[NY-1]/0.0254) / CPIT[NY-1] ) * XRCA[NY-1] * FYCA[NY-1])/CPICA[NY-1]);

   EWREV[NY-1]:= ((( WP[NY-1] / CPIT[NY-1] ) * XRAR[NY-1] * WYAR[NY-1])/CPIAR[NY-1]
              + (( WP[NY-1] / CPIT[NY-1] ) * XRAU[NY-1] * WYAU[NY-1])/CPIAU[NY-1]
              + (( WP[NY-1] / CPIT[NY-1] ) * XRCA[NY-1] * WYCA[NY-1])/CPICA[NY-1]);

    EHA[NY]:= 8508.919 + 204.545 * EFREV[NY-1] - 343.169 * EWREV[NY-1]
    + 0.698 * EHA[NY-1] - 0.300 * EES[NY-1] + EHA_AF[NY];

           { EXPORTER PRODUCTION PER HEAD }

    EPROPC[NY]:= ( (EHA[NY] * EYIELD[NY]) / EPOP[NY] );
    EPRO[NY]:= ( EYIELD[NY] * EHA[NY] );

   ESPLYPC[NY]:=EPROPC[NY]+EESPC[NY-1];

{ EXPORTERS CONSUMPTION/HEAD = f ( EXPORTERS GROSS DOMESTIC PRODUCT, EXPORTERS
     FEEDGRAIN PRICE, EXPORTERS WHEAT PRICE, EXPORTERS SUPPLY PER CAP., SPLINE
     FROM 1974 ON ) }

 ECONPC[NY]:= 0.0578 + 1222063.5 * (EGDP[NY]/EPOP[NY]) - 0.001
 * (( (FP[NY]/0.0254) / CPIT[NY] ) * NXXRF[NY]) + 0.00083
 * (( WP[NY] / CPIT[NY] ) * NXXRW[NY]) + 0.314 * ESPLYPC[NY]
 - 0.0793 * DV74ON + ECONPC_AF[NY];

        {  EXPORTER ENDING STOCKS PER HEAD  }

 EESPC[NY]:= 0.02932 + 0.0728 * EPROPC[NY] + 0.3516 * (EES[NY-1]/EPOP[NY-1])
 - 0.1338 * EUDPC[NY-1] - 0.0932 * DV83 + EESPC_AF[NY];

 EES[NY]:=EESPC[NY] * EPOP[NY];

 EUDPC[NY]:= ESPLYPC[NY] - EESPC[NY];

 IF ECON_IN_BU_FLAG='N' THEN
 ECON_IN_BU[NY]:=((((ECONPC[NY] * EPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
 ELSE ECON_IN_BU[NY]:=ECON_IN_BU_OR[NY];

 IF EES_IN_BU_FLAG='N' THEN
 EES_IN_BU[NY]:=((((EESPC[NY] * EPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
 ELSE EES_IN_BU[NY]:=EES_IN_BU_OR[NY];

 IF ESPLY_IN_BU_FLAG='N' THEN
 esply_in_bu[NY]:=((((esplypc[NY]*EPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
 ELSE ESPLY_IN_BU[NY]:=ESPLY_IN_BU_OR[NY];

 IF  (ENE_IN_BU_FLAG='N') THEN
 ENE_IN_BU[NY]:= ESPLY_IN_BU[NY] - ECON_IN_BU[NY] - EES_IN_BU[NY]
 ELSE ENE_IN_BU[NY]:=ENE_IN_BU_OR[NY];

 if ene_in_bu[ny] < 0.0 then ene_in_bu[ny]:=0.0;

IF EXPORTS_FLAG='N' THEN
 EXPORTS[NY]:= ( INI_IN_BU[NY] - ENE_IN_BU[NY] )
 ELSE EXPORTS[NY]:= EXPORTS_OR[NY];

 TOTAL_USE[NY]:= DOMESTIC_USE[NY] + EXPORTS[NY];

        { STOCKS IN GOVERNMENT PROGRAM }

FPLR:=  FP[NY] / CORN_LOAN_RATE[NY];
FPLR2:= FPLR * FPLR;
FPLR3:= FPLR2 * FPLR;

{ CCC INVENTORY (t) =  CCC INVENTORY (t-1) + CHANGE IN CCC INVENTORY }

DCCCSTOCKS[NY]:= 34330.230 - 68153.126 * FPLR + 45967.279 * FPLR2
                 - 10569.075 * FPLR3;

CCCSTOCKS[NY]:= CCCSTOCKS[NY-1] + DCCCSTOCKS[NY];
```

```
   IF CCCSTOCKS[NY] < 0.0 THEN CCCSTOCKS[NY]:= 0.0;
   IF CCCSTOCKS_FLAG='Y' THEN CCCSTOCKS[NY]:= CCCSTOCKS_OR[NY];

{   { FOR INVENTORY (t) = FOR INVENTORY (t-1) + CHANGE IN FOR INVENTORY }

   DFORSTOCKS[NY]:= 28637.804 - 53497.38 * FPLR + 35532.478 * FPLR2
                   - 8431.0044 * FPLR3;

   FORSTOCKS[NY]:= FORSTOCKS[NY-1] + DFORSTOCKS[NY];
   IF FORSTOCKS[NY] >= ( ( TSUPPLY[NY] - TOTAL_USE[NY] ) - ( MIN_FREE_STOCKS[NY] +
                       CCCSTOCKS[NY] ) ) THEN FORSTOCKS[NY]:=
                       ( ( TSUPPLY[NY] - TOTAL_USE[NY] ) - ( MIN_FREE_STOCKS[NY] +
                       CCCSTOCKS[NY] ) );

   IF FORSTOCKS[NY] < 0.0 THEN FORSTOCKS[NY]:= 0.0;
   IF FORSTOCKS_FLAG='Y' THEN FORSTOCKS[NY]:= FORSTOCKS_OR[NY];

   FREESTOCK_PROC;

   IF FREESTOCKS_FLAG='Y' THEN FREESTOCKS[NY]:= FREESTOCKS_OR[NY];

   IF ((TOTAL_USE[NY]+CCCSTOCKS[NY]+FORSTOCKS[NY]+FREESTOCKS[NY])>TSUPPLY[NY]) THEN
    FORSTOCKS[NY]:=(TSUPPLY[NY]-(CCCSTOCKS[NY]+FREESTOCKS[NY]+TOTAL_USE[NY]));

           { TOTAL ENDING STOCKS EQUATION }

   END_STOCKS[NY]:= ( CCCSTOCKS[NY] + FORSTOCKS[NY] + FREESTOCKS[NY] );

           { TOTAL, PERCENT OF USE }

   PERCENT_USE[NY]:= (END_STOCKS[NY] / TOTAL_USE[NY]) * 100;

END; { PROCEDURE RE CALCULATE }
        ***** End of Include file RECALC.INC *****
        ***** Beginning of Include file CACULATE.INC *****
{$I CACULATE.INC}
overlay procedure CALCULATIONS(corn_price_in_year_t: real);
{ THE FORECAST CALCULATIONS USING RANDOM SHOCKS }

var conserved_acres:real;   { A LOCAL VARIABLE USED FOR PERCENT OF CONSERVATION ACRES
                                 ( ARP + DIVERSION) }

procedure FREESTOCK_PROC;
        { A PROCEDURE TO CALCULATE THE FREE STOCKS BASED
        ON WEIGHTS USED FOR POLICY }

VAR
CCCWT, FORWT: REAL;

   BEGIN

{        RULE                               CCC WEIGHT       FOR WEIGHT }
{        ----                               ----------       ---------- }
{     1) YEAR <= 76                           0.10              0.0     }
{     2) YEAR >= 77                           0.80              0.75    }
{     3) YEAR >= 77, CCCt <= CCCt-1,                                     }
{        TRIGGER PRICE > $1.55                0.10              0.75    }
{     4) YEAR >= 77, FORt <= FORt-1,                                     }
{        TRIGGER PRICE > $1.55                0.80              0.0     }

      IF ( year[NY] <= 76 ) THEN CCCWT:= 0.1 ELSE CCCWT:= 0.8;

      IF ( year[NY] >= 77 ) AND ( CCCstocks[NY] <= CCCstocks[NY-1] )
          AND ( fplr > 1.55 ) THEN CCCWT:= 0.1;

      IF ( year[NY] <= 76 ) THEN FORWT:= 0.0 ELSE FORWT:= 0.75;

      IF ( FORstocks[NY] <= FORstocks[NY-1] ) AND ( fplr > 1.55 ) THEN
      FORWT:= 0.0;

     FREESTOCKS[NY]:=1127.4231 - 1873.6898 * (( CCCWT * CCCSTOCKS[NY]
                     + FORWT * FORSTOCKS[NY] ) / TOTAL_USE[NY] )
                     - 1301390.2 * ( FPLR / TOTAL_USE[NY] ) + FREESTOCKS_AF[NY];

      IF FREESTOCKS[NY] < MIN_FREF_STOCKS[NY] THEN FREESTOCKS[NY]:=MIN_FREE_STOCKS[NY];

   END;  { AREA WHICH CALCULATES FUNCTION Fx FOR GIVEN WEIGHTS }

BEGIN
            {  DOMESTIC  SUPPLY  }

   { APCT = f ( DV83, LAGGED CORN REVENUE(NON-PARTICIPANTS), WHEAT REVENUE,
              LAGGED CORN REVENUE(PARTICIPANTS) }

   CONSERVED_ACRES:=ARP[NY]+DIVERSION[NY];

   IF APCT_FLAG='N' THEN

  APCT[NY]:= 90.993360 {+4.044 * dv79 - 6.54 * dv73} - 0.1262719 *
  (((WHTPT[NY-1] * WHTYT[NY-1]) - VCHW[NY-1]) / CPIT[NY-1]) - 0.0281170 *
  (((PARREV[NY-1] + DIVPAY[NY-1] + DEFPAY[NY-1]) - VCCN[NY-1]) / CPIT[NY-1])
  - 60.333270 * CONSERVED_ACRES + APCT_AF[NY]

    ELSE APCT[NY]:=APCT_OR[NY];

   IF AHCT_FLAG='N' THEN
   AHCT[NY]:= -8.412272 + 0.8729 * APCT[NY] + 0.1078 * YEAR[NY] + AHCT_AF[NY]
   ELSE AHCT[NY]:=AHCT_OR[NY];

   IF CORNYT_FLAG='N' THEN
   CORNYT[NY]:= -71.3758 + 2.153 * YEAR[NY] + CORNYT_AF[NY]
   ELSE CORNYT[NY]:=CORNYT_OR[NY];

   if year[ny]=85 then beginstocks[NY]:=1648.0
     else BEGINSTOCKS[NY]:=END_STOCKS[NY-1];

   YIELD:=7.661092 * Z + CORNYT[NY];
```

```
IF PRODUCT_FLAG='N' THEN
PRODUCT[NY]:= AHCT[NY] * YIELD
ELSE PRODUCT[NY]:=PRODUCT_OR[NY];

IF IMPORTS_FLAG='Y' THEN IMPORTS[NY]:=IMPORTS_OR[NY];

IF TSUPPLY_FLAG='N' THEN
TSUPPLY[NY]:= BEGINSTOCKS[NY] + PRODUCT[NY] + IMPORTS[NY]
ELSE TSUPPLY[NY]:=TSUPPLY_OR[NY];

        {  FEED USE  }

{ DACON/HEAD = f ( TIME, REAL MILK PRICE, REAL CORN FARM PRICE, REAL WHEAT PRICE ) }

 if oneph_or[ny]=0.0 then
 begin
 ONEPH[NY]:= -86.515 + 1.832 * YEAR[NY] + 3.415 * (MILKP[NY] / CPIT[NY])
 - 25.61 * (CORNPT[NY] / CPIT[NY]) + 10.746 * (WHTPT[NY] / CPIT[NY]) + ONEPH_AF[NY];
 dairycons[NY]:=oneph[NY]*danum[NY];
 end
 else dairycons[ny]:=oneph_or[ny]*danum[ny];

{ COFCON/HEAD = f ( TIME, BEEF-STEER/CORN RATIO, LAGGED DEPENDENT ) }

 if twoph_or[ny]=0.0 then
 begin
 TWOPH[NY]:= 240.119 - 2.126 * YEAR[NY] + 1.665 * (BEFSTRP[NY] / CORNPT[NY])
 - 0.659 * TWOPH[NY-1] + TWOPH_AF[NY];
 catonfed[NY]:=twoph[NY]*cofnum[NY];
 end
 else catonfed[ny]:=twoph_or[ny]*cofnum[ny];

{ OBCCON/HEAD = f ( BEEF COW/CORN RATIO, LAGGED REAL SOYBEAN MEAL PRICE ) }

 if threeph_or[ny]=0.0 then
 begin              .
 THREEPH[NY]:= 2.875 + 0.115 * (BCOWP[NY] / CORNPT[NY])
 + 0.00487 * (SMPT[NY-1] / CPIT[NY-1]) + THREEPH_AF[NY];
 otherbeefcat[NY]:=threeph[NY]*obcnum[NY];
 end
 else otherbeefcat[ny]:=threeph_or[ny]*obcnum[ny];

{ HPCCON/HEAD = f ( REAL CORN FARM PRICE, REAL SOYBEAN MEAL PRICE, REAL WHEAT PRICE,
                    LAGGED DEPENDENT }

 if fourph_or[ny]=0.0 then
 begin
 FOURPH[NY]:= 0.9349 - 0.724 * (CORNPT[NY] / CPIT[NY]) + 0.00121
 * (SMPT[NY] / CPIT[NY]) + 0.3409 * (WHTPT[NY] / CPIT[NY]) + 0.222 * FOURPH[NY-1]
 + FOURPH_AF[NY];
 chickencons[NY]:=fourph[NY]*hpcnum[NY];
 end
 else chickencons[ny]:=fourph_or[ny]*hpcnum[ny];
```

```
{BROCON/HEAD = f ( TIME, REAL CORN FARM PRICE, REAL WHEAT PRICE ) }

 if fiveph_or[ny]=0.0 then
 begin
 FIVEPH[NY]:= 0.00245 + 0.00143 * YEAR[NY] - 0.0431 * (CORNPT[NY] / CPIT[NY])
 + 0.0162 * (WHTPT[NY] / CPIT[NY]) + FIVEPH_AF[NY];
 broilercons[NY]:=fiveph[NY]*bronum[NY];
 end
 else broilercons[ny]:=fiveph_or[ny]*bronum[ny];

{ TURCON/HEAD = f ( REAL CORN FARM PRICE, REAL WHEAT PRICE, LAGGED DEPENDENT ) }

 if sixph_or[ny]=0.0 then
 begin
 SIXPH[NY]:= 0.709 - 0.3428 *  (CORNPT[NY] / CPIT[NY]) + 0.126 * (WHTPT[NY] / CPIT[NY])
 + 0.104 * SIXPH[NY-1] + SIXPH_AF[NY];
 turkeycons[NY]:=sixph[NY]*turnum[NY];
 end
 else turkeycons[ny]:=sixph_or[ny]*turnum[ny];

{ HOGCON/HEAD = f ( TIME, REAL HOG PRICE, REAL CORN FARM PRICE, ) }

 if sevenph_or[ny]=0.0 then
 begin
 SEVENPH[NY]:= -14.624 + 0.5 * YEAR[NY] + 0.394 * (HOGP[NY] / CPIT[NY])
 - 7.612 * (CORNPT[NY] / CPIT[NY]) + SEVENPH_AF[NY];
 hogcons[NY]:=sevenph[NY]*hognum[NY];
 end
 else hogcons[ny]:=sevenph_or[ny]*hognum[ny];

{ OUNCON = f ( TIME, REAL CORN FARM PRICE, REAL WHEAT PRICE, DV82, DV84 ) }

if other_unalloc_or[ny]=0.0 then
othercons[NY]:= -412.497 + 6.84 * YEAR[NY] - 63.265 * (CORNPT[NY] / CPIT[NY])
           + 26.481 * (WHTPT[NY] / CPIT[NY]) + 326.601 * DV82
           + 189.265 * DV84 + EIGHT_AF[NY]
else othercons[ny]:=other_unalloc_or[ny];

IF FEEDT_FLAG='N' THEN
FEEDT[NY]:= dairycons[NY] + catonfed[NY] + otherbeefcat[NY] + chickencons[NY]
+ broilercons[NY] + turkeycons[NY] + hogcons[NY] + othercons[NY]
ELSE FEEDT[NY]:=FEEDT_OR[NY];

IF DOMESTIC_USE_FLAG='N' THEN
DOMESTIC_USE[NY]:=FEEDT[NY] + FSI_USE[NY]
ELSE DOMESTIC_USE[NY]:=DOMESTIC_USE_OR[NY];

       {  EXPORT COMPONENT  }

{ INIPC uses generalized differencing and is adjusted below. }

{ THE SIMPLE LINKAGE BETWEEN FEEDGRAIN PRICE ($/BU.) AND
  FARM PRICE IS AN ADDITIONAL $0.42 }
```

```
FOR I:= 1 TO NY DO
   BEGIN
      FP[I]:= CORN_PRICE_IN_YEAR_T + 0.42;
   END;

IFP[NY]:= (( (FP[NY]/0.0254) / CPIT[NY] ) * NMXRF[NY] );
{ THIS TRANSFORMATION PUTS FP IN $/MT }

IWP[NY]:= (( WP[NY] / CPIT[NY] ) * NMXRW[NY] );

IYIELD[NY]:=-0.8943774 + 0.0328114 * YEAR[NY] + IYIELD_AF[NY];

    { Where: IYIELD[NY]* = IYIELD[4] - 0.3820334 * IYIELD[3],
    YEAR[4]* = YEAR[4] - 0.3820334 * YEAR[3],
    In this case, Rho is equal to 0.3820334. }

YIELD2:= 0.045231 * Z2 + IYIELD[NY];

IPRO[NY]:=( YIELD2 * IHA[NY] );

INIPC[NY]:= 0.0028 - 0.0000425 * IFP[NY] + 0.0000478 * IWP[NY]
 + 9209967.0 * (IGDP[NY]/IPOP[NY]) - 0.143 * (IPRO[NY]/IPOP[NY]) + INIPC_AF[NY];

       { Where: INIPC[NY]* = INIPC[NY] - 0.7277196 * INIPC[NY-1],
       IFP[NY]* = IFP[NY] - 0.7277196 * IFP[NY-1],
       IWP[NY]* = IWP[NY] - 0.7277196 * IWP[NY-1],
       IGDPPC[NY]* = IGDPPC[NY] - 0.7277196 * IGDPPC[NY-1],
       and IPROPC[NY]* = IPROPC[NY] - 0.7277196 * IPROPC[NY-1].
       In this case, Rho is equal to 0.7277196. }


IF INI_IN_BU_FLAG='N' THEN
      { corn is about 0.86 % of exports }
INI_IN_BU[NY]:= ((((INIPC[NY]*IPOP[NY])* 39.368)/1000.0)*PCNT_OF_XPORTS[NY])
ELSE INI_IN_BU[NY]:=INI_IN_BU_OR[NY];

EYIELD[NY]:= -1.26 + 0.0460 * YEAR[NY] + EYIELD_AF[NY];  { EXPORTERS FEEDGRAIN YIELD }

YIELD1:= 0.113202 * Z1 + EYIELD[NY];

EWYIELD[NY]:= 0.2247 + 0.0175 * YEAR[NY] + EWYIELD_AF[NY];  {  EXPORTERS WHEAT YIELD  }

{ EXPORTERS HARVESTED ACRES = f ( LAGGED EXPORTERS FEEDGRAIN REVENUE, LAGGED
      EXPORTERS WHEAT REVENUE, LAGGED HARVESTED ACRES, LAGGED ENDING STOCKS }

EFREV[NY-1]:= ((( (FP[NY-1]/0.0254) / CPIT[NY-1] ) * XRAR[NY-1] * FYAR[NY-1])/CPIAR[NY-1]
           + (( (FP[NY-1]/0.0254) / CPIT[NY-1] ) * XRAU[NY-1] * FYAU[NY-1])/CPIAU[NY-1]
           + (( (FP[NY-1]/0.0254) / CPIT[NY-1] ) * XRCA[NY-1] * FYCA[NY-1])/CPICA[NY-1]);
EWREV[NY-1]:= ((( WP[NY-1] / CPIT[NY-1] ) * XRAR[NY-1] * WYAR[NY-1])/CPIAR[NY-1]
           + (( WP[NY-1] / CPIT[NY-1] ) * XRAU[NY-1] * WYAU[NY-1])/CPIAU[NY-1]
           + (( WP[NY-1] / CPIT[NY-1] ) * XRCA[NY-1] * WYCA[NY-1])/CPICA[NY-1]);

EHA[NY]:= 8508.919 + 204.545 * EFREV[NY-1] - 343.169 * EWREV[NY-1]
 + 0.698 * EHA[NY-1] - 0.300 * EES[NY-1] + EHA_AF[NY];
```

```
        { EXPORTER PRODUCTION PER HEAD }

EPROPC[NY]:= ( (EHA[NY] * YIELD1) / EPOP[NY] );
EPRO[NY]:= ( YIELD1 * EHA[NY] );

ESPLYPC[NY]:= EPROPC[NY]+EESPC[NY-1];

{ EXPORTERS CONSUMPTION/HEAD = f ( EXPORTERS GROSS DOMESTIC PRODUCT, EXPORTERS
     FEEDGRAIN PRICE, EXPORTERS WHEAT PRICE, EXPORTERS SUPPLY PER CAP., SPLINE
     FROM 1974 ON ) }

ECONPC[NY]:= 0.0578 + 1222063.5 * (EGDP[NY]/EPOP[NY]) - 0.001
 * (( (FP[NY]/0.0254) / CPIT[NY] ) * NXXRF[NY]) + 0.00083
 * (( WP[NY] / CPIT[NY] ) * NXXRW[NY]) + 0.314 * esplypc[NY]
 - 0.0793 * DV74ON + ECONPC_AF[NY];

        {  EXPORTER ENDING STOCKS PER HEAD  }

EESPC[NY]:= 0.02932 + 0.0728 * EPROPC[NY] + 0.3516 * (EES[NY-1]/EPOP[NY-1])
 - 0.1338 * EUDPC[NY-1] - 0.0932 * DV83 + EESPC_AF[NY];

EES[NY]:= EESPC[NY] * EPOP[NY];

EUDPC[NY]:=ESPLYPC[NY] - EESPC[NY];

IF ECON_IN_BU_FLAG='N' THEN
ECON_IN_BU[NY]:= ((((ECONPC[NY] * EPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
ELSE ECON_IN_BU[NY]:=ECON_IN_BU_OR[NY];

IF EES_IN_BU_FLAG='N' THEN
EES_IN_BU[NY]:= ((((EESPC[NY] * EPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
ELSE EES_IN_BU[NY]:=EES_IN_BU_OR[NY];

IF ESPLY_IN_BU_FLAG='N' THEN
ESPLY_IN_BU[NY]:=((((esplypc[NY]*EPOP[NY])*39.368)/1000.0)*PCNT_OF_XPORTS[NY])
ELSE ESPLY_IN_BU[NY]:=ESPLY_IN_BU_OR[NY];

IF ENE_IN_BU_FLAG='N' THEN
ENE_IN_BU[NY]:= ESPLY_IN_BU[NY] - ECON_IN_BU[NY] - EES_IN_BU[NY]
ELSE ENE_IN_BU[NY]:=ENE_IN_BU_OR[NY];

if ene_in_bu[ny] < 0.0 then ene_in_bu[ny]:=0.0;

IF EXPORTS_FLAG='N' THEN
EXPORTS[NY]:= ( INI_IN_BU[NY] - ENE_IN_BU[NY] )
ELSE EXPORTS[NY]:=EXPORTS_OR[NY];

TOTAL_USE[NY]:= DOMESTIC_USE[NY] + EXPORTS[NY];
        { STOCKS IN GOVERNMENT PROGRAM }

FPLR:=  FP[NY] / CORN_LOAN_RATE[NY];
FPLR2:= FPLR * FPLR;
FPLR3:= FPLR2 * FPLR;
```

```
   { CCC INVENTORY (t) =  CCC INVENTORY (t-1) + CHANGE IN CCC INVENTORY }

   DCCCSTOCKS[NY]:= 34330.230 - 68153.126 * FPLR + 45967.279 * FPLR2
                  - 10569.075 * FPLR3;

   CCCSTOCKS[NY]:= CCCSTOCKS[NY-1] + DCCCSTOCKS[NY];

   IF CCCSTOCKS[NY] < 0.0 THEN CCCSTOCKS[NY]:= 0.0;
   IF CCCSTOCKS_FLAG='Y' THEN CCCSTOCKS[NY]:=CCCSTOCKS_OR[NY];

   { FOR INVENTORY (t) = FOR INVENTORY (t-1) + CHANGE IN FOR INVENTORY }

   DFORSTOCKS[NY]:= 28637.804 - 53497.38 * FPLR + 35532.478 * FPLR2
                  - 8431.0044 * FPLR3;

   FORSTOCKS[NY]:= FORSTOCKS[NY-1] + DFORSTOCKS[NY];

   IF FORSTOCKS[NY] < 0.0 THEN FORSTOCKS[NY]:= 0.0;
   IF FORSTOCKS_FLAG='Y' THEN FORSTOCKS[NY]:=FORSTOCKS_OR[NY];

   FREESTOCK_PROC;

   IF FREESTOCKS_FLAG='Y' THEN FREESTOCKS[NY]:=FREESTOCKS_OR[NY];

           { TOTAL ENDING STOCKS EQUATION }


   END_STOCKS[NY]:= ( CCCSTOCKS[NY] + FORSTOCKS[NY] + FREESTOCKS[NY] );

           { EQUILIBRIUM CONDITION }

   EQUILIBRIUM:=(TOTAL_USE[NY] + END_STOCKS[NY]) - TSUPPLY[NY];

END; { PROCEDURE CALCULATIONS }
        ***** End of Include file CACULATE.INC *****
        ***** Beginning of Include file SOLUTION.INC *****
{$I SOLUTION.INC}
OVERLAY PROCEDURE SOLUTION_SCREEN; { A PROCEDURE TO RUN 'RE_CALCULATE' }
BEGIN

  screenSS:=1; screen_fieldSS:=3; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('SOLVESCR.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen SOLVESCR') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(68,5,4,'N',CORNPT[NY],
     '#.##',0.00,9.99,retSS,retrieveSS,15,0);
2:  GETNUM(59,7,4,'N',REDUCE_PRICE,
     '#.##',0.00,0.99,retSS,retrieveSS,15,0);
3:  BEGIN
      IF RETRIEVESS=FALSE THEN
        FG:=0
       ELSE
        FG:=15;
      GETITEM(36,14,7,'C',INFO,
     'XXXXXXX','','',retSS,FALSE,FG,0);
    END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE SCREEN_1 }

        ***** End of Include file SOLUTION.INC *****
        ***** Beginning of Include file REMAIN.INC *****
{$I remain.INC}
procedure remain;  { A PROCEDURE TO DETERMINE MARKET CLEARING PRICE
                     FOR THE 'RE_CALCULATE' PROCEDURE }
begin
   SOLUTION_SCREEN;
   repeat
      re_calculate;
      cornpt[NY]:=cornpt[NY]-REDUCE_PRICE;
   until((total_use[NY] + end_stocks[NY]) >= tsupply[NY]) or (cornpt[ny] <= 0.10);
   cornpt[NY]:=cornpt[NY]+REDUCE_PRICE;
   beep(true);
   beeponss:=false;
end;      { procedure remain }
        ***** End of Include file REMAIN.INC *****
        ***** Beginning of Include file HOTT.INC *****
{$I HOTT.INC}           {  add factors  }
OVERLAY PROCEDURE ADD_FACTOR_SCREEN_1; { B:AFACTOR1 }
BEGIN

  screenSS:=2; screen_fieldSS:=20; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('AFACTOR1.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen AFACTOR1') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
```

```
  CASE varSS of
1:  GETNUM(34,3,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  BEGIN
       YEARB[NY]:=YEAR[NY]+1;
       GETNUM(37,3,2,'N',YEARB[NY],
        '##',0.0,99.0,retSS,FALSE,15,0);
    END;
3:  GETNUM(73,3,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(76,3,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(32,6,7,'N',APCT_AF[NY],
     '###.###',-99.000,999.000,retSS,retrieveSS,15,0);
6:  GETNUM(32,8,7,'N',AHCT_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
7:  GETNUM(32,10,7,'N',CORNYT_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
8:  GETNUM(32,15,7,'N',ONEPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
9:  GETNUM(32,17,7,'N',TWOPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
10:  GETNUM(32,19,7,'N',THREEPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
11:  GETNUM(32,21,7,'N',FOURPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
12:  GETNUM(32,23,7,'N',FIVEPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
13:  GETNUM(71,6,7,'N',SIXPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
14:  GETNUM(71,8,7,'N',SEVENPH_AF[NY],
     '###.###',-99.999,999.999,retSS,retrieveSS,15,0);
15:  GETNUM(71,10,7,'N',EIGHT_AF[NY],
     '####.##',-999.99,9999.99,retSS,retrieveSS,15,0);
16:  GETNUM(70,15,8,'N',INIPC_AF[NY],
     '##.#####',-9.99999,99.99999,retSS,retrieveSS,15,0);
17:  GETNUM(73,17,5,'N',IYIELD_AF[NY],
     '##.##',-9.99,99.99,retSS,retrieveSS,15,0);
18:  GETNUM(73,19,5,'N',EYIELD_AF[NY],
     '##.##',-9.99,99.99,retSS,retrieveSS,15,0);
19:  GETNUM(71,21,7,'N',EHA_AF[NY],
     '#####.#',-9999.9,99999.9,retSS,retrieveSS,15,0);
20:  GETNUM(70,23,8,'N',ECONPC_AF[NY],
     '##.#####',-9.99999,99.99999,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
```

```
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE ADD_FACTOR_SCREEN_1 }

OVERLAY PROCEDURE ADD_FACTOR_SCREEN_2; { B:AFACTOR2 }
BEGIN

  screenSS:=3; screen_fieldSS:=6; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('AFACTOR2.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen AFACTOR2') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(57,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(60,6,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(55,10,7,'N',EESPC_AF[NY],
     '##.#####',-0.99999,9.99999,retSS,retrieveSS,15,0);
4:  GETNUM(55,18,8,'N',DCCCSTOCKS_AF[NY],
     '#####.##',-9999.99,99999.99,retSS,retrieveSS,15,0);
5:  GETNUM(55,20,8,'N',DFORSTOCKS_AF[NY],
     '#####.##',-9999.99,99999.99,retSS,retrieveSS,15,0);
6:  GETNUM(55,22,8,'N',FREESTOCKS_AF[NY],
     '#####.##',-9999.99,99999.99,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE ADD_FACTOR_SCREEN_2 }
        ***** End of Include file HOTT.INC *****
        ***** Beginning of Include file INTRO.INC *****
{$I intro.inc}          { ALLOWS THE LOAD AND SAVE PROCESS }
overlay procedure LOAD;
BEGIN
    j:=1;
    screen_fieldSS:=5;
    varSS:=1;
    retrieveSS:=FALSE;
    last_fieldSS:=FALSE;
    DISPLAY_SCREEN('LOAD.SCR',file_existSS);  { Display Screen }
    if not file_existSS then
      BEGIN
      gotoxy(1,1);
      write('Missing Screen LOAD')
    END;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
    1: BEGIN
         GetItem(27,4,47,'C',DefaultDrive,
         'U:UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU','','',retSS,retrieveSS,0,7);
          if (length(defaultdrive) =3) then delete(defaultdrive, 3, 1);
         Checkdisk(defaultdrive);
       END;
    2: BEGIN
         LoadYn:='Y';
         GetItem(60,20,1,'Y',LoadYn,
          'U','','',retSS,retrieveSS,0,7);
         If (retrieveSS) and (upcase(LoadYn)='N') then varSS:=screen_fieldSS;
       END;
    3: begin
         GETNUM(46,21,1,'N',J,
       '#',1,4,retSS,retrieveSS,0,7);
       S:=TRUNC(J);
       end;
    4: BEGIN

        GetItem(26,23,8,'C',FileName,
         'UUUUUUUU','','',retSS,retrieveSS,0,7);
         If (retrievess) then Loadbev(FileName,DefaultDrive);
       END;
    5: begin
         GetItem(35,23,3,'C',FileExt,
         'UUU','','',retSS,FALSE,15,0);
         if (retrieveSS=true) and (S<>3) then varSS:=1;
       end;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE LOAD }
```

```
 overlay procedure SAVE;
BEGIN
    j:=4;
    screen_fieldSS:=5;
    varSS:=1;
    retrieveSS:=FALSE; last_fieldSS:=FALSE;
    DISPLAY_SCREEN('SAVE.SCR',file_existSS);  { Display Screen }
    if not file_existSS then
      BEGIN
        gotoxy(1,1);
        write('Missing Screen SAVE')
      END;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1: BEGIN
     GetItem(65,4,1,'Y',SAVEYN,
     'U','','',retSS,retrieveSS,0,7);
      If (retrieveSS) AND (upcase(Saveyn)='N')
        Then varSS := screen_fieldSS;
   END;
2: BEGIN
     GetItem(32,5,47,'C',DefaultDrive,
     'U:XUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU','','',retSS,retrieveSS,0,7);
      if (length(defaultdrive) =3) then delete(defaultdrive, 3, 1);
     Checkdisk(defaultdrive);
   END;
3: BEGIN

       GETNUM(48,6,1,'N',J,
       '#',1,4,retSS,retrieveSS,0,7);
       S:=TRUNC(J);
    END;
4: begin
{    on_line_help('SAVEFILE');      }
    GetItem(26,23,8,'C',FileName,
     'UUUUUUUU','','',retSS,retrieveSS,0,7);
   end;
5: BEGIN
     GetItem(35,23,3,'C',FileExt,
     'UUU','','',retSS,FALSE,15,0);
     If retrieveSS
      Then Savebev(FileName,DefaultDrive);
   END;
END; { CASE }


  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
```

```
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE LOAD }

        ***** End of Include file INTRO.INC *****
        ***** Beginning of Include file BALSHEET.INC *****
{$I BALSHEET.INC}
overlay PROCEDURE SCREEN_12; { B:SDBSCUS }
BEGIN

  screenSS:=4; screen_fieldSS:=30; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('SDBSCUS.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen SDBSCUS') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(25,7,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  begin
      yearb[NY-3]:=year[NY-3]+1;
      GETNUM(28,7,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
3:  GETNUM(36,7,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  begin
      yearb[NY-2]:=year[NY-2]+1;
      GETNUM(39,7,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
5:  GETNUM(46,7,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  begin
      yearb[NY-1]:=year[NY-1]+1;
      GETNUM(49,7,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
7:  GETNUM(56,7,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  begin
      yearb[NY]:=year[NY]+1;
      GETNUM(59,7,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
9:  GETNUM(26,11,4,'N',SETASIDE_ACRES[NY-3],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
10: GETNUM(26,14,4,'N',APCT[NY-3],
    '##.#',0.0,99.0,retSS,FALSE,15,0);
11: GETNUM(26,17,4,'N',AHCT[NY-3],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
12: GETNUM(25,20,5,'N',CORNYT[NY-3],
    '###.#',0.0,999.9,retSS,FALSE,15,0);
13: GETNUM(37,11,4,'N',SETASIDE_ACRES[NY-2],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
14: GETNUM(37,14,4,'N',APCT[NY-2],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
15: GETNUM(37,17,4,'N',AHCT[NY-2],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
16: GETNUM(36,20,5,'N',CORNYT[NY-2],
    '###.#',0.0,999.9,retSS,FALSE,15,0);
17: GETNUM(47,11,4,'N',SETASIDE_ACRES[NY-1],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
18: GETNUM(47,14,4,'N',APCT[NY-1],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
19: GETNUM(47,17,4,'N',AHCT[NY-1],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
20: GETNUM(46,20,5,'N',CORNYT[NY-1],
    '###.#',0.0,999.9,retSS,FALSE,15,0);
21: GETNUM(57,11,4,'N',SETASIDE_ACRES[NY],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
22: GETNUM(57,14,4,'N',APCT[NY],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
23: GETNUM(57,17,4,'N',AHCT[NY],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
24: GETNUM(56,20,5,'N',CORNYT[NY],
    '###.#',0.0,999.9,retSS,FALSE,15,0);
25: GETITEM(66,14,1,'Y',APCT_FLAG,
    'U','','',retSS,retrieveSS,15,0);
26: BEGIN
     if (retrieveSS=true) and (apct_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(72,14,6,'N',APCT_OR[NY],
        '##.###',0.000,99.999,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(72,14,6,'N',APCT_OR[NY],
        '##.###',0.000,99.999,retSS,false,fg,bg);
      end;
    END;
27: GETITEM(66,17,1,'Y',AHCT_FLAG,
    'U','','',retSS,retrieveSS,15,0);
28: BEGIN
    if (retrieveSS=true) and (ahct_flag='Y') then
     begin
```

```
        fg:=15;
        bg:=0;
        GETNUM(72,17,6,'N',AHCT_OR[NY],
        '##.###',0.000,99.999,retSS,retrieveSS,fg,bg);
       end
      else
       begin
         fg:=0;
         bg:=0;
         GETNUM(72,17,6,'N',AHCT_OR[NY],
         '##.###',0.000,99.999,retSS,false,fg,bg);
       end;
     END;
29:  GETITEM(66,20,1,'Y',CORNYT_FLAG,
     'U','','',retSS,retrieveSS,15,0);
30:  BEGIN
     if (retrieveSS=true) and (cornyt_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(73,20,5,'N',CORNYT_OR[NY],
        '###.#',0.0,999.9,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(73,20,5,'N',CORNYT_OR[NY],
        '###.#',0.0,999.9,retSS,false,fg,bg);
      end;
     END;
  END; { CASE }
  color(white,black);

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE SCREEN_12 }

overlay PROCEDURE SCREEN_13; { B:SDBSCUSA }
BEGIN
  screenSS:=5; screen_fieldSS:=64; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('SDBSCUSA.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen SDBSCUSA') end;
```

```
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(30,4,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(33,4,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(40,4,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(43,4,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(50,4,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(53,4,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(60,4,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(63,4,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(31,7,4,'N',BEGINSTOCKS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
10:  GETNUM(31,8,4,'N',PRODUCT[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
11:  GETNUM(33,9,2,'N',IMPORTS[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
12:  GETNUM(30,11,5,'N',TSUPPLY[NY-3],
     '#####',0.0,99999.0,retSS,FALSE,15,0);
13:  GETNUM(31,13,4,'N',FEEDT[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
14:  GETNUM(31,14,4,'N',FSI_USE[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
15:  GETNUM(31,16,4,'N',DOMESTIC_USE[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
16:  GETNUM(31,17,4,'N',EXPORTS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
17:  GETNUM(31,19,4,'N',TOTAL_USE[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
18:  GETNUM(31,20,4,'N',END_STOCKS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
19:  GETNUM(30,21,5,'N',PERCENT_USE[NY-3],
     '###.#',0.0,100.0,retSS,FALSE,15,0);
20:  GETNUM(31,23,4,'N',CORNPT[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
21:  GETNUM(41,7,4,'N',BEGINSTOCKS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
22:  GETNUM(41,8,4,'N',PRODUCT[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
23:  GETNUM(43,9,2,'N',IMPORTS[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
24:  GETNUM(40,11,5,'N',TSUPPLY[NY-2],
     '#####',0.0,99999.0,retSS,FALSE,15,0);
25:  GETNUM(41,13,4,'N',FEEDT[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
26:  GETNUM(41,14,4,'N',FSI_USE[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
27:  GETNUM(41,16,4,'N',DOMESTIC_USE[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
28:  GETNUM(41,17,4,'N',EXPORTS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
29:  GETNUM(41,19,4,'N',TOTAL_USE[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
30:  GETNUM(41,20,4,'N',END_STOCKS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
31:  GETNUM(40,21,5,'N',PERCENT_USE[NY-2],
     '###.#',0.0,100.0,retSS,FALSE,15,0);
32:  GETNUM(41,23,4,'N',CORNPT[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
33:  GETNUM(51,7,4,'N',BEGINSTOCKS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
34:  GETNUM(51,8,4,'N',PRODUCT[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
35:  GETNUM(53,9,2,'N',IMPORTS[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
36:  GETNUM(50,11,5,'N',TSUPPLY[NY-1],
     '#####',0.0,99999.0,retSS,FALSE,15,0);
37:  GETNUM(51,13,4,'N',FEEDT[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
38:  GETNUM(51,14,4,'N',FSI_USE[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
39:  GETNUM(51,16,4,'N',DOMESTIC_USE[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
40:  GETNUM(51,17,4,'N',EXPORTS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
41:  GETNUM(51,19,4,'N',TOTAL_USE[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
42:  GETNUM(51,20,4,'N',END_STOCKS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
43:  GETNUM(50,21,5,'N',PERCENT_USE[NY-1],
     '###.#',0.0,100.0,retSS,FALSE,15,0);
44:  GETNUM(51,23,4,'N',CORNPT[NY-1],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
45:  GETNUM(61,7,4,'N',BEGINSTOCKS[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
46:  GETNUM(60,8,5,'N',PRODUCT[NY],
     '#####',0.0,99999.0,retSS,FALSE,15,0);
47:  GETNUM(63,9,2,'N',IMPORTS[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
48:  GETNUM(60,11,5,'N',TSUPPLY[NY],
     '#####',0.0,99999.0,retSS,FALSE,15,0);
49:  GETNUM(61,13,4,'N',FEEDT[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
50:  GETNUM(61,14,4,'N',FSI_USE[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
51:  GETNUM(61,16,4,'N',DOMESTIC_USE[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
52:  GETNUM(61,17,4,'N',EXPORTS[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
53:  GETNUM(61,19,4,'N',TOTAL_USE[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
54:  GETNUM(61,20,4,'N',END_STOCKS[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
55:  GETNUM(60,21,5,'N',PERCENT_USE[NY],
     '###.#',0.0,100.0,retSS,FALSE,15,0);
56:  GETNUM(61,23,4,'N',CORNPT[NY],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
57:  GETITEM(69,8,1,'Y',PRODUCT_FLAG,
     'U','','',retSS,retrieveSS,15,0);
58:  BEGIN
     IF (retrieveSS=true) AND (product_flag='Y') then
     begin
        fg:=15;
        bg:=0;
        GETNUM(73,8,5,'N',PRODUCT_OR[NY],
        '#####',0.0,99999.9,retSS,retrieveSS,fg,bg);
     end
     else
     begin
        fg:=0;
        bg:=0;
        GETNUM(73,8,5,'N',PRODUCT_OR[NY],
        '#####',0.0,99999.9,retSS,false,fg,bg);
     end;
     END;
59:  GETITEM(69,11,1,'Y',TSUPPLY_FLAG,
     'U','','',retSS,retrieveSS,15,0);
60:  BEGIN
     if (retrieveSS=true) and (TSUPPLY_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(73,11,5,'N',TSUPPLY_OR[NY],
        '#####',0.0,99999.0,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(73,11,5,'N',TSUPPLY_OR[NY],
        '#####',0.0,99999.0,retSS,false,fg,bg);
      end;
     END;
61:  GETITEM(69,13,1,'Y',FEEDT_FLAG,
     'U','','',retSS,retrieveSS,15,0);
62:  BEGIN
     if (retrieveSS=true) and (FEEDT_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(74,13,4,'N',FEEDT_OR[NY],
        '####',0.0,9999.0,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(74,13,4,'N',FEEDT_OR[NY],
        '####',0.0,9999.0,retSS,false,fg,bg);
      end;
     END;
63:  GETITEM(69,17,1,'Y',EXPORTS_FLAG,
     'U','','',retSS,retrieveSS,15,0);
64:  BEGIN
     if (retrieveSS=true) and (EXPORTS_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(74,17,4,'N',EXPORTS_OR[NY],
        '####',0.0,9999.0,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(74,17,4,'N',EXPORTS_OR[NY],
        '####',0.0,9999.0,retSS,false,fg,bg);
      end;
     END;
  END; { CASE }
  color(white,black);

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE SCREEN_13 }

overlay PROCEDURE SCREEN_14; { B:SDBSCUSB }
BEGIN

  screenSS:=6; screen_fieldSS:=34; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('SDBSCUSB.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen SDBSCUSB') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
```

```
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(30,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(33,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(40,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(43,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(50,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(53,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(60,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(63,5,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(31,11,4,'N',CCCSTOCKS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
10:  GETNUM(31,13,4,'N',FORSTOCKS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
11:  GETNUM(31,15,4,'N',FREESTOCKS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
12:  GETNUM(31,20,4,'N',GOVTSTOCKS[NY-3],
     '####',0.0,9999.0,retSS,FALSE,15,0);
13:  GETNUM(31,22,4,'N',CORN_LOAN_RATE[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
14:  GETNUM(41,11,4,'N',CCCSTOCKS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
15:  GETNUM(41,13,4,'N',FORSTOCKS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
16:  GETNUM(41,15,4,'N',FREESTOCKS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
17:  GETNUM(41,20,4,'N',GOVTSTOCKS[NY-2],
     '####',0.0,9999.0,retSS,FALSE,15,0);
18:  GETNUM(41,22,4,'N',CORN_LOAN_RATE[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
19:  GETNUM(51,11,4,'N',CCCSTOCKS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
20:  GETNUM(51,13,4,'N',FORSTOCKS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
21:  GETNUM(51,15,4,'N',FREESTOCKS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
22:  GETNUM(51,20,4,'N',GOVTSTOCKS[NY-1],
     '####',0.0,9999.0,retSS,FALSE,15,0);
23:  GETNUM(51,22,4,'N',CORN_LOAN_RATE[NY-1],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
24:  GETNUM(60,11,4,'N',CCCSTOCKS[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
25:  GETNUM(60,13,4,'N',FORSTOCKS[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
26:  GETNUM(60,15,4,'N',FREESTOCKS[NY],
     '####',0.0,9999.0,retSS,FALSE,15,0);
```

```
27:  GETNUM(60,22,4,'N',CORN_LOAN_RATE[NY],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
28:  GETITEM(70,11,1,'Y',CCCSTOCKS_FLAG,
     'U','','',retSS,retrieveSS,15,0);
29:  BEGIN
     if (retrieveSS=true) and (CCCSTOCKS_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(74,11,4,'N',CCCSTOCKS_OR[NY],
        '####',0.0,9999.0,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(74,11,4,'N',CCCSTOCKS_OR[NY],
        '####',0.0,9999.0,retSS,false,fg,bg);
      end;
     END;
30:  GETITEM(70,13,1,'Y',FORSTOCKS_FLAG,
     'U','','',retSS,retrieveSS,15,0);
31:  BEGIN
     if (retrieveSS=true) and (FORSTOCKS_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(74,13,4,'N',FORSTOCKS_OR[NY],
        '####',0.0,9999.0,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(74,13,4,'N',FORSTOCKS_OR[NY],
        '####',0.0,9999.0,retSS,false,fg,bg);
      end;
     END;
32:  GETITEM(70,15,1,'Y',FREESTOCKS_FLAG,
     'U','','',retSS,retrieveSS,15,0);
33:  BEGIN
     if (retrieveSS=true) and (FREESTOCKS_flag='Y') then
      begin
        fg:=15;
        bg:=0;
        GETNUM(74,15,4,'N',FREESTOCKS_OR[NY],
        '####',0.0,9999.0,retSS,retrieveSS,fg,bg);
      end
     else
      begin
        fg:=0;
        bg:=0;
        GETNUM(74,15,4,'N',FREESTOCKS_OR[NY],
        '####',0.0,9999.0,retSS,false,fg,bg);
      end;
     END;
34:  GETNUM(74,20,4,'N',GOVTSTOCKS[NY],
     '####',0.0,9999.0,retSS,retrieveSS,15,0);


  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE SCREEN_14 }

        ***** End of Include file BALSHEET.INC *****
        ***** Beginning of Include file ZEBRAC.INC *****
{$I ZEBRAC.INC}
PROCEDURE zebrac; { A PROCEDURE WHICH USES THE EQUATION BI-SECTION
                    METHOD TO DETERMINE MARKET CLEARING PRICE FOR THE
                    'CALCULATE' PROCEDURE }
var
  y0,x,xl,xu,es,xr,xn,aa,ea, temp1,temp2,temp3: real;
  ni,im: integer;

begin
  XL:=0.5; XU:=5.00; ES:=0.1;
  im:=20;
  xr:=(xl+xu)/2.0;
  ni:=0;
  repeat
    ni:=ni+1;
    calculations(xl);
    temp1:=equilibrium;
    calculations(xr);
    temp2:=equilibrium;
    aa:=temp1 * temp2;
    if aa < 0.0 then xu:=xr;
    if aa > 0.0 then xl:=xr;
    xn:=(xl+xu)/2.0;
    if (xn <> 0.0) then ea:=abs((xn-xr)/xn)*100.0 else ea:=999.99;
    xr:=xn;
    uscornpt[f]:=(xr+xn)/2;
  until ((ea < es) or (ni > im));
end;

        ***** End of Include file ZEBRAC.INC *****
        ***** Beginning of Include file RANDOMIZE.INC *****
```

```
{$I RANDOMIZE.INC}
overlay PROCEDURE RANDOMIZE_YIELD_SCREEN_1; { RAND }
VAR { Variables Section For RAND }
   jump,m,n : integer;
   temp : real;
   alldone : boolean;
   a: real;
   l,kount: integer;

BEGIN
    RANDOM_PULLS:=25.0; INFO:='SOLVING'; PAUSE:='PLEASE WAIT';
  screenSS:=7; screen_fieldSS:=3; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('RAND.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen RAND') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  BEGIN
       GETNUM(70,8,3,'N',RANDOM_PULLS,
        '###',1.0,999.0,retSS,retrieveSS,15,0);
       wa:=TRUNC(RANDOM_PULLS);
    END;
2:  BEGIN
     IF RETRIEVESS=FALSE THEN
     FG:=0
     ELSE FG:=15;
     GETITEM(36,12,7,'C',INFO,
     'XXXXXXX','','',retSS,FALSE,FG,0);
    END;
3:  BEGIN
     IF RETRIEVESS=FALSE THEN
     FG:=0
     ELSE FG:=15;
     GETITEM(34,15,11,'C',PAUSE,
     'XXXXXXXXXXX','','',retSS,FALSE,FG,0);
    END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS;
```

```
BEGIN
 FOR f:= 1 TO wa DO
  BEGIN
{    CORNPT[NY]:=2.50;}
   BETA:=0.0;       { THE NAME IS THE US YIELD }
   BETA1:=0.0;      { THE NAME1 IS THE EXPORTERS FEEDGRAIN YIELD }
   BETA2:=0.0;      { THE NAME2 IS THE IMPORTERS FEEDGRAIN YIELD }
   RANDOMIZE;
   FOR I:=1 TO 30 DO
    BEGIN
     ALPHA[I]:=RANDOM;
     ALPHA1[I]:=RANDOM;
     ALPHA2[I]:=RANDOM;
     BETA:=BETA+ALPHA[I];
     BETA1:=BETA1+ALPHA1[I];
     BETA2:=BETA2+ALPHA2[I];
    END;
   GAMMA:=BETA/30;
   GAMMA1:=BETA1/30;
   GAMMA2:=BETA2/30;
   Z:=((GAMMA-0.5)/0.052705);  { Z=(rnd-mu)/(sigma/sqrt(n)) by the central limit therom }
   Z1:=((GAMMA1-0.5)/0.052705);
   Z2:=((GAMMA2-0.5)/0.052705);

ZEBRAC;

  END;
END; { PROCEDURE RANDOMIZE_YIELD_SCREEN_1 }
{************************************************************************
*                          PROCEDURE SORT                               *
*  This procedure sorts an array, T, of length L using the diminishing *
*  increment sorting procedure developed by Shell.  This code is a      *
*  modification of that presented on pp. 158-63 of the revised edition *
*  "Programming in PASCAL" by Peter Grogono.                            *
************************************************************************}
begin {procedure Sort}
 l:=wa;
 jump:=l;
 while (jump>1) do
  begin
   jump:=jump div 2;
   repeat
    alldone:=true;
    for m:=1 to l-jump do
     begin
      n:=m+jump;
      if (uscornpt[m]>uscornpt[n])
       then
        begin
         temp:=uscornpt[m];
         uscornpt[m]:=uscornpt[n];
         uscornpt[n]:=temp;
         alldone:=false;
        end;
```

```
          end;
      until alldone;
    end;
end; {procedure Sort}

for i:=1 to 3 do
 begin
   beep(true);
 end;
beeponss:=false;

END; {PROCEDURE RANDOMIZE_YIELD_SCREEN_1}

overlay procedure DISPLAY_RANDOMIZE_SCREEN;
VAR { Variables Section For B:PROBDISP }
   USCPRICE1: REAL; USCPRICE2: REAL; USCPRICE3: REAL; USCPRICE4: REAL;
   USCPRICE5: REAL; USCPRICE6: REAL; USCPRICE7: REAL; USCPRICE8: REAL;
   USCPRICE9: REAL; USCPRICE10: REAL; USCPRICE11: REAL; USCPRICE12: REAL;
   temp, hold1, mean, std_dev: real;

{************************************************************************
*                  PROCEDURE FIND_PERCENTILES                           *
*     A PROCEDURE WHICH USES LINEAR INTERPOLATION TO CONSTRUCT THE      *
*     FREQUENCY DISTRIBUTION.  IT USES THE METHOD SIMULAR TO THAT       *
*     DESCRIBED IN "STATISTICAL CONCEPTS AND METHODS" PAGES 14-24       *
*     BY BHATTACHARYYA AND JOHNSON.                                     *
************************************************************************}

PROCEDURE FIND_PERCENTILES;
VAR    cum_freq, frequency,interval:array[1..10] of real;
   lowprice, highprice, spread, classlength:real;
   counter,obs,fraction:real;
   q,i,l,j:integer;
begin
   lowprice:=uscornpt[1]-0.005;
   highprice:=uscornpt[wa]+0.005;
   spread:=highprice-lowprice;
   classlength:=spread/10;
   percentile_answer:=0.0;
   for i:=1 to 10 do
      begin
         interval[i]:=lowprice+classlength;
         counter:=0.0;
         for j:=1 to wa do
         if (uscornpt[j] > lowprice) and (uscornpt[j] < interval[i]) then
            begin
               counter:=counter+1;
            end;
         frequency[i]:=counter;
         if j=1 then cum_freq[i]:=frequency[i]+0.0
         else cum_freq[i]:=frequency[i]+cum_freq[i-1];
         lowprice:=interval[i];
      end;
         obs:=wa*(p/100);
         i:= 1;
         while i<11 do
            begin
               if cum_freq[i]>=obs then
                  begin
                     q:=i;
                     i:=10;
                  end;
               i:= i+1;
            end;
         fraction:=((obs-cum_freq[q-1])/frequency[q]);
         percentile_answer:=(interval[q]-classlength)+(fraction*classlength);
end;


BEGIN
  temp:=0.0;
  for f:=1 to wa do
    begin
      temp:=uscornpt[f]+temp;
    end;
  mean:=temp/wa;

  temp:=0.0;
  for f:=1 to wa do
    begin
      temp:=(uscornpt[f]-mean)*(uscornpt[f]-mean) + temp;
    end;
 std_dev:=sqrt(temp/(wa-1));

  screenSS:=8; screen_fieldSS:=15; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('PROBDISP.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen PROBDISP') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  BEGIN
    P:=1;
    FIND_PERCENTILES;
    USCPRICE1:=PERCENTILE_ANSWER;
    GETNUM(60,7,4,'N',USCPRICE1,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
    END;
2:  BEGIN
    P:=5;
    FIND_PERCENTILES;
    USCPRICE2:=PERCENTILE_ANSWER;
    GETNUM(60,8,4,'N',USCPRICE2,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
    END;
```

```
3:  BEGIN
    P:=10;
    FIND_PERCENTILES;
    USCPRICE3:=PERCENTILE_ANSWER;
    GETNUM(60,9,4,'N',USCPRICE3,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
4:  BEGIN
    P:=20;
    FIND_PERCENTILES;
    USCPRICE4:=PERCENTILE_ANSWER;
    GETNUM(60,10,4,'N',USCPRICE4,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
5:  BEGIN
    P:=30;
    FIND_PERCENTILES;
    USCPRICE5:=PERCENTILE_ANSWER;
    GETNUM(60,11,4,'N',USCPRICE5,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
6:  BEGIN
    P:=40;
    FIND_PERCENTILES;
    USCPRICE6:=PERCENTILE_ANSWER;
    GETNUM(60,12,4,'N',USCPRICE6,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
7:  BEGIN
    P:=50;
    FIND_PERCENTILES;
    USCPRICE7:=PERCENTILE_ANSWER;
    GETNUM(60,13,4,'N',USCPRICE7,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
8:  BEGIN
    P:=60;
    FIND_PERCENTILES;
    USCPRICE8:=PERCENTILE_ANSWER;
    GETNUM(60,14,4,'N',USCPRICE8,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
9:  BEGIN
    P:=70;
    FIND_PERCENTILES;
    USCPRICE9:=PERCENTILE_ANSWER;
    GETNUM(60,15,4,'N',USCPRICE9,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
10:  BEGIN
    P:=80;
    FIND_PERCENTILES;
    USCPRICE10:=PERCENTILE_ANSWER;
    GETNUM(60,16,4,'N',USCPRICE10,
    '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
11:  BEGIN
    P:=90;
    FIND_PERCENTILES;
    USCPRICE11:=PERCENTILE_ANSWER;
    GETNUM(60,17,4,'N',USCPRICE11,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
12:  BEGIN
    P:=95;
    FIND_PERCENTILES;
    USCPRICE12:=PERCENTILE_ANSWER;
    GETNUM(60,18,4,'N',USCPRICE12,
     '#.##',0.00,9.99,retSS,FALSE,15,0);
   END
13: GETNUM(60,19,4,'N',uscornpt[wa],
    '#.##',0.00,9.99,retSS,FALSE,15,0);
14: GETNUM(45,20,4,'N',MEAN,
    '#.##',0.00,9.99,retSS,FALSE,15,0);
15: GETNUM(45,21,6,'N',STD_DEV,
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE SCREEN_1 }
        ***** End of Include file RANDOMIZE.INC *****

overlay procedure EXOGENOUS_VARIABLE_SUB_MENU;  {  A SUB MENU UNDER THE MAIN MENU  }

        ***** Beginning of Include file MAGIC1.INC *****
{$I MAGIC1.INC}       { EXOGENOUS VARIABLE SUMMARY }
OVERLAY PROCEDURE FSI_USE_SCREEN; { B:FSIUSE }
BEGIN
  screenSS:=9; screen_fieldSS:=28; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('FSIUSE.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen FSIUSE') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
```

```
1:  GETNUM(41,6,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,6,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(51,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(54,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(61,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(64,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(71,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(74,6,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(41,11,5,'N',HFCS[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
10:  GETNUM(41,13,5,'N',RESIDUAL[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
11:  GETNUM(42,15,4,'N',SEED_USE[NY-3],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
12:  GETNUM(41,19,5,'N',FUEL_USE[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
13:  GETNUM(40,21,6,'N',FSI_USE[NY-3],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
14:  GETNUM(51,11,5,'N',HFCS[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
15:  GETNUM(51,13,5,'N',RESIDUAL[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
16:  GETNUM(52,15,4,'N',SEED_USE[NY-2],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
17:  GETNUM(51,19,5,'N',FUEL_USE[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
18:  GETNUM(50,21,6,'N',FSI_USE[NY-2],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
19:  GETNUM(61,11,5,'N',HFCS[NY-1],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
20:  GETNUM(61,13,5,'N',RESIDUAL[NY-1],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
21:  GETNUM(62,15,4,'N',SEED_USE[NY-1],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
22:  GETNUM(61,19,5,'N',FUEL_USE[NY-1],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
23:  GETNUM(60,21,6,'N',FSI_USE[NY-1],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
24:  GETNUM(71,11,5,'N',HFCS[NY],
     '###.#',0.0,999.9,retSS,retrieveSS,15,0);
25:  GETNUM(71,13,5,'N',RESIDUAL[NY],
     '###.#',0.0,999.9,retSS,retrieveSS,15,0);
26:  GETNUM(72,15,4,'N',SEED_USE[NY],
     '##.#',0.0,99.9,retSS,retrieveSS,15,0);
27:  GETNUM(71,19,5,'N',FUEL_USE[NY],
     '###.#',0.0,999.9,retSS,retrieveSS,15,0);
```

```
28:  BEGIN
       FSI_USE[NY]:=HFCS[NY]+RESIDUAL[NY]+SEED_USE[NY]+FUEL_USE[NY];
       GETNUM(70,21,6,'N',FSI_USE[NY],
     '####.#',0.0,9999.9,retSS,retrieveSS,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE FSI USE SCREEN_1 }

OVERLAY PROCEDURE SCREEN_1; { EXOGENOUS VARIABLE SUMMARY SCREEN 1 }
BEGIN

  screenSS:=10; screen_fieldSS:=48; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR1.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR1') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(43,3,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:   BEGIN
        YEARB[NY-3]:=YEAR[NY-3]+1;
        GETNUM(46,3,2,'N',YEARB[NY-3],
        '##',0.0,99.0,retSS,FALSE,15,0);
     END;
3:  GETNUM(54,3,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:   BEGIN
        YEARB[NY-2]:=YEAR[NY-2]+1;
        GETNUM(57,3,2,'N',YEARB[NY-2],
        '##',0.0,99.0,retSS,FALSE,15,0);
     END;
5:  GETNUM(64,3,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:   BEGIN
        YEARB[NY-1]:=YEAR[NY-1]+1;
        GETNUM(67,3,2,'N',YEARB[NY-1],
        '##',0.0,99.0,retSS,FALSE,15,0);
     END;
```

```
7:  GETNUM(74,3,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,retrieveSS,15,0);
8:   BEGIN
        YEARB[NY]:=YEAR[NY]+1;
        GETNUM(77,3,2,'N',YEARB[NY],
        '##',0.0,99.0,retSS,FALSE,15,0);
     END;
9:  GETNUM(44,5,4,'N',IMPORTS[NY-3],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
10:  GETNUM(43,7,5,'N',CPIT[NY-3],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
11:  GETNUM(43,9,5,'N',VCHW[NY-3],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
12:  GETNUM(42,11,6,'N',VCCN[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
13:  GETNUM(42,13,6,'N',PARREV[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
14:  GETNUM(42,15,6,'N',DIVPAY[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
15:  GETNUM(42,17,6,'N',DEFPAY[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
16:  GETNUM(43,19,5,'N',ARP[NY-3],
     '#.###',0.0,1.0,retSS,FALSE,15,0);
17:  GETNUM(43,21,5,'N',DIVERSION[NY-3],
     '#.###',0.0,1.0,retSS,FALSE,15,0);
18:  GETNUM(44,23,4,'N',CORN_LOAN_RATE[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
19:  GETNUM(55,5,4,'N',IMPORTS[NY-2],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
20:  GETNUM(54,7,5,'N',CPIT[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
21:  GETNUM(54,9,5,'N',VCHW[NY-2],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
22:  GETNUM(53,11,6,'N',VCCN[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
23:  GETNUM(53,13,6,'N',PARREV[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
24:  GETNUM(53,15,6,'N',DIVPAY[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
25:  GETNUM(53,17,6,'N',DEFPAY[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
26:  GETNUM(54,19,5,'N',ARP[NY-2],
     '#.###',0.0,1.0,retSS,FALSE,15,0);
27:  GETNUM(54,21,5,'N',DIVERSION[NY-2],
     '#.###',0.0,1.0,retSS,FALSE,15,0);
28:  GETNUM(55,23,4,'N',CORN_LOAN_RATE[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
29:  GETNUM(65,5,4,'N',IMPORTS[NY-1],
    '##.#',0.0,99.9,retSS,FALSE,15,0);
30:  GETNUM(64,7,5,'N',CPIT[NY-1],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
31:  GETNUM(64,9,5,'N',VCHW[NY-1],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
32:  GETNUM(63,11,6,'N',VCCN[NY-1],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
33:   GETNUM(63,13,6,'N',PARREV[NY-1],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
34:   GETNUM(63,15,6,'N',DIVPAY[NY-1],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
35:   GETNUM(63,17,6,'N',DEFPAY[NY-1],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
36:   GETNUM(64,19,5,'N',ARP[NY-1],
      '#.###',0.0,1.0,retSS,FALSE,15,0);
37:   GETNUM(64,21,5,'N',DIVERSION[NY-1],
      '#.###',0.0,1.0,retSS,FALSE,15,0);
38:   GETNUM(65,23,4,'N',CORN_LOAN_RATE[NY-1],
      '#.##',0.00,9.99,retSS,FALSE,15,0);
39:   GETNUM(75,5,4,'N',IMPORTS[NY],
     '##.#',0.0,99.9,retSS,retrieveSS,15,0);
40:   GETNUM(74,7,5,'N',CPIT[NY],
      '#.###',0.000,9.999,retSS,retrieveSS,15,0);
41:   GETNUM(74,9,5,'N',VCHW[NY],
      '##.##',0.00,99.99,retSS,retrieveSS,15,0);
42:   GETNUM(73,11,6,'N',VCCN[NY],
      '###.##',0.00,999.99,retSS,retrieveSS,15,0);
43:   GETNUM(73,13,6,'N',PARREV[NY],
      '###.##',0.00,999.99,retSS,retrieveSS,15,0);
44:   GETNUM(73,15,6,'N',DIVPAY[NY],
      '###.##',0.00,999.99,retSS,retrieveSS,15,0);
45:   GETNUM(73,17,6,'N',DEFPAY[NY],
      '###.##',0.00,999.99,retSS,retrieveSS,15,0);
46:   GETNUM(74,19,5,'N',ARP[NY],
      '#.###',0.0,1.0,retSS,retrieveSS,15,0);
47:   GETNUM(74,21,5,'N',DIVERSION[NY],
      '#.###',0.0,1.0,retSS,retrieveSS,15,0);
48:   GETNUM(75,23,4,'N',CORN_LOAN_RATE[NY],
      '#.##',0.00,9.99,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_1 }

OVERLAY PROCEDURE SCREEN_2; { EXOGENOUS VARIABLE SUMMARY SCREEN 2 }
BEGIN

  screenSS:=11; screen_fieldSS:=36; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR2.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR2') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(43,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(46,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(54,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(57,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(64,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(67,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(42,10,6,'N',NMXRF[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
10:  GETNUM(42,12,6,'N',NMXRW[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
11:  GETNUM(42,14,6,'N',NXXRF[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
12:  GETNUM(42,16,6,'N',NXXRW[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
13:  GETNUM(41,18,7,'N',WP[NY-3],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
14:  GETNUM(44,20,4,'N',WHTPT[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
15:  GETNUM(44,22,4,'N',WHTYT[NY-3],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
16:  GETNUM(53,10,6,'N',NMXRF[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
17:  GETNUM(53,12,6,'N',NMXRW[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
18:  GETNUM(53,14,6,'N',NXXRF[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
19:  GETNUM(53,16,6,'N',NXXRW[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
20:  GETNUM(52,18,7,'N',WP[NY-2],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
21:  GETNUM(55,20,4,'N',WHTPT[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
22:  GETNUM(55,22,4,'N',WHTYT[NY-2],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
23:  GETNUM(63,10,6,'N',NMXRF[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
24:  GETNUM(63,12,6,'N',NMXRW[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
25:  GETNUM(63,14,6,'N',NXXRF[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
26:  GETNUM(63,16,6,'N',NXXRW[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
27:  GETNUM(62,18,7,'N',WP[NY-1],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
28:  GETNUM(65,20,4,'N',WHTPT[NY-1],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
29:  GETNUM(65,22,4,'N',WHTYT[NY-1],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
30:  GETNUM(73,10,6,'N',NMXRF[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
31:  GETNUM(73,12,6,'N',NMXRW[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
32:  GETNUM(73,14,6,'N',NXXRF[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
33:  GETNUM(73,16,6,'N',NXXRW[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
34:  GETNUM(72,18,7,'N',WP[NY],
     '####.##',0000.00,9999.99,retSS,retrieveSS,15,0);
35:  GETNUM(75,20,4,'N',WHTPT[NY],
     '#.##',0.00,9.99,retSS,retrieveSS,15,0);
36:  GETNUM(75,22,4,'N',WHTYT[NY],
     '##.#',0.0,99.9,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_2 }

OVERLAY PROCEDURE SCREEN_3; { EXOGENOUS VARIABLE SUMMARY SCREEN 3 }
BEGIN

  screenSS:=12; screen_fieldSS:=36; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR3.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR3') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(43,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(46,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(54,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(57,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(64,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(67,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9: GETNUM(42,10,6,'N',DANUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
10:  GETNUM(42,12,6,'N',COFNUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
11:  GETNUM(42,14,6,'N',OBCNUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
12:  GETNUM(41,16,7,'N',HPCNUM[NY-3],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
13:  GETNUM(41,18,7,'N',BRONUM[NY-3],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
14:  GETNUM(41,20,7,'N',TURNUM[NY-3],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
15:  GETNUM(42,22,6,'N',HOGNUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
16:  GETNUM(53,10,6,'N',DANUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
17:  GETNUM(53,12,6,'N',COFNUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
18:  GETNUM(53,14,6,'N',OBCNUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
19:  GETNUM(52,16,7,'N',HPCNUM[NY-2],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
20:  GETNUM(52,18,7,'N',BRONUM[NY-2],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
21:  GETNUM(52,20,7,'N',TURNUM[NY-2],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
22:  GETNUM(53,22,6,'N',HOGNUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
23:  GETNUM(63,10,6,'N',DANUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
24:  GETNUM(63,12,6,'N',COFNUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
25:  GETNUM(63,14,6,'N',OBCNUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
26:  GETNUM(62,16,7,'N',HPCNUM[NY-1],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
27:  GETNUM(62,18,7,'N',BRONUM[NY-1],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
28:  GETNUM(62,20,7,'N',TURNUM[NY-1],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
29:  GETNUM(63,22,6,'N',HOGNUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
30:  GETNUM(73,10,6,'N',DANUM[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
31:  GETNUM(73,12,6,'N',COFNUM[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
32:  GETNUM(73,14,6,'N',OBCNUM[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
33:  GETNUM(72,16,7,'N',HPCNUM[NY],
     '###.###',0.000,999.999,retSS,retrieveSS,15,0);
34:  GETNUM(72,18,7,'N',BRONUM[NY],
     '####.##',0.00,9999.99,retSS,retrieveSS,15,0);
35:  GETNUM(72,20,7,'N',TURNUM[NY],
     '###.###',0.000,999.999,retSS,retrieveSS,15,0);
36:  GETNUM(73,22,6,'N',HOGNUM[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_3 }

OVERLAY PROCEDURE SCREEN_3A; {B:EXOGVAR 3A }
BEGIN

  screenSS:=13; screen_fieldSS:=44; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOVAR3A.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOVAR3A') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(43,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(46,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(54,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(57,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(64,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(67,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(42,7,6,'N',SMPT[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
10:  GETNUM(43,9,5,'N',HOGP[NY-3],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
11:  GETNUM(43,11,5,'N',BEFSTRP[NY-3],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
12:  GETNUM(43,13,5,'N',MILKP[NY-3],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
13: GETNUM(43,15,5,'N',BCOWP[NY-3],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
14: GETNUM(44,17,4,'N',SETASIDE_ACRES[NY-3],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
15: GETNUM(43,19,5,'N',MIN_FREE_STOCKS[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
16: GETNUM(42,21,6,'N',FSI_USE[NY-3],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
17:  GETNUM(41,23,7,'N',PCNT_OF_XPORTS[NY-3],
     '#.#####',0.0,1.0,retSS,FALSE,15,0);
18:  GETNUM(53,7,6,'N',SMPT[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
19:  GETNUM(54,9,5,'N',HOGP[NY-2],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
20:  GETNUM(54,11,5,'N',BEFSTRP[NY-2],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
21:  GETNUM(54,13,5,'N',MILKP[NY-2],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
22:  GETNUM(54,15,5,'N',BCOWP[NY-2],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
23:  GETNUM(55,17,4,'N',SETASIDE_ACRES[NY-2],
     '##.#',0.0,99.9,retSS,FALSE,15,0);
24:  GETNUM(54,19,5,'N',MIN_FREE_STOCKS[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
25:  GETNUM(53,21,6,'N',FSI_USE[NY-2],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
26:  GETNUM(52,23,7,'N',PCNT_OF_XPORTS[NY-2],
     '#.#####',0.0,1.0,retSS,FALSE,15,0);
27:  GETNUM(63,7,6,'N',SMPT[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
28:  GETNUM(64,9,5,'N',HOGP[NY-1],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
29:  GETNUM(64,11,5,'N',BEFSTRP[NY-1],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
30:  GETNUM(64,13,5,'N',MILKP[NY-1],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
31:  GETNUM(64,15,5,'N',BCOWP[NY-1],
     '##.##',0.00,99.99,retSS,FALSE,15,0);
32:  GETNUM(65,17,4,'N',SETASIDE_ACRES[NY-1],
      '##.#',0.0,99.9,retSS,FALSE,15,0);
33:  GETNUM(64,19,5,'N',MIN_FREE_STOCKS[NY-1],
      '###.#',0.0,999.9,retSS,FALSE,15,0);
34:  GETNUM(63,21,6,'N',FSI_USE[NY-1],
      '####.#',0.0,9999.9,retSS,FALSE,15,0);
35:  GETNUM(62,23,7,'N',PCNT_OF_XPORTS[NY-1],
      '#.#####',0.0,1.0,retSS,FALSE,15,0);
36:  GETNUM(73,7,6,'N',SMPT[NY],
      '###.##',0.00,999.99,retSS,retrieveSS,15,0);
37:  GETNUM(74,9,5,'N',HOGP[NY],
      '##.##',0.00,99.99,retSS,retrieveSS,15,0);
38:  GETNUM(74,11,5,'N',BEFSTRP[NY],
      '##.##',0.00,99.99,retSS,retrieveSS,15,0);
39:  GETNUM(74,13,5,'N',MILKP[NY],
      '##.##',0.00,99.99,retSS,retrieveSS,15,0);
40:  GETNUM(74,15,5,'N',BCOWP[NY],
      '##.##',0.00,99.99,retSS,retrieveSS,15,0);
41:  GETNUM(75,17,4,'N',SETASIDE_ACRES[NY],
      '##.#',0.0,99.9,retSS,retrieveSS,15,0);
42:  GETNUM(74,19,5,'N',MIN_FREE_STOCKS[NY],
      '###.#',0.0,999.9,retSS,retrieveSS,15,0);
43:  GETNUM(73,21,6,'N',FSI_USE[NY],
      '####.#',0.0,9999.9,retSS,retrieveSS,15,0);
44:  GETNUM(72,23,7,'N',PCNT_OF_XPORTS[NY],
      '#.#####',0.0,1.0,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_3A }

OVERLAY PROCEDURE SCREEN_4; { EXOGENOUS VARIABLE SUMMARY SCREEN 4 }
BEGIN

  screenSS:=14; screen_fieldSS:=34; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR4.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR4') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(44,5,2,'N',YEAR[NY-3],
      '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(47,5,2,'N',YEARB[NY-3],
      '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(54,5,2,'N',YEAR[NY-2],
      '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(57,5,2,'N',YEARB[NY-2],
      '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(64,5,2,'N',YEAR[NY-1],
      '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(67,5,2,'N',YEARB[NY-1],
      '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,5,2,'N',YEAR[NY],
      '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
      '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(40,7,9,'N',IPOP[NY-3],
      '#######.#',0.0,9999999.9,retSS,FALSE,15,0);
10:  GETNUM(41,9,8,'N',IGDP[NY-3],
      '#.######',0.000000,9.999999,retSS,FALSE,15,0);
11:  GETNUM(41,11,8,'N',IHA[NY-3],
      '######.#',0.0,999999.9,retSS,FALSE,15,0);
12:  GETNUM(43,14,6,'N',EFREV[NY-3],
      '#.####',0.0000,9.9999,retSS,FALSE,15,0);
13:  GETNUM(43,16,6,'N',EWREV[NY-3],
      '#.####',0.0000,9.9999,retSS,FALSE,15,0);
14:  GETNUM(42,18,7,'N',EPOP[NY-3],
      '#####.#',0.0,99999.9,retSS,FALSE,15,0);
15:  GETNUM(41,20,8,'N',EGDP[NY-3],
      '#.######',0.000000,9.999999,retSS,FALSE,15,0);
16:  GETNUM(50,7,9,'N',IPOP[NY-2],
      '#######.#',0.0,9999999.9,retSS,FALSE,15,0);
17:  GETNUM(51,9,8,'N',IGDP[NY-2],
      '#.######',0.000000,9.999999,retSS,FALSE,15,0);
18:  GETNUM(51,11,8,'N',IHA[NY-2],
      '######.#',0.0,999999.9,retSS,FALSE,15,0);
19:  GETNUM(53,14,6,'N',EFREV[NY-2],
      '#.####',0.0000,9.9999,retSS,FALSE,15,0);
20:  GETNUM(53,16,6,'N',EWREV[NY-2],
      '#.####',0.9999,9.9999,retSS,FALSE,15,0);
21:  GETNUM(52,18,7,'N',EPOP[NY-2],
      '#####.#',0.0,99999.9,retSS,FALSE,15,0);
22:  GETNUM(51,20,8,'N',EGDP[NY-2],
      '#.######',0.000000,9.999999,retSS,FALSE,15,0);
23:  GETNUM(60,7,9,'N',IPOP[NY-1],
      '#######.#',0.0,9999999.9,retSS,FALSE,15,0);
24:  GETNUM(61,9,8,'N',IGDP[NY-1],
      '#.######',0.000000,9.999999,retSS,FALSE,15,0);
25:  GETNUM(61,11,8,'N',IHA[NY-1],
      '######.#',0.0,999999.9,retSS,FALSE,15,0);
26:  GETNUM(63,14,6,'N',EFREV[NY-1],
      '#.####',0.0000,9.9999,retSS,FALSE,15,0);
27:  GETNUM(63,16,6,'N',EWREV[NY-1],
      '#.####',0.0000,9.9999,retSS,FALSE,15,0);
28:  GETNUM(62,18,7,'N',EPOP[NY-1],
     '#####.#',0.0,99999.9,retSS,FALSE,15,0);
29:  GETNUM(61,20,8,'N',EGDP[NY-1],
     '#.######',0.000000,9.999999,retSS,FALSE,15,0);
30:  GETNUM(70,7,9,'N',IPOP[NY],
     '#######.#',0.0,9999999.9,retSS,retrieveSS,15,0);
31:  GETNUM(71,9,8,'N',IGDP[NY],
     '#.######',0.000000,9.999999,retSS,retrieveSS,15,0);
32:  GETNUM(71,11,8,'N',IHA[NY],
     '######.#',0.0,999999.9,retSS,retrieveSS,15,0);
33:  GETNUM(72,18,7,'N',EPOP[NY],
     '#####.#',0.0,99999.9,retSS,retrieveSS,15,0);
34:  GETNUM(71,20,8,'N',EGDP[NY],
     '#.######',0.000000,9.999999,retSS,retrieveSS,15,0);

  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_4 }
        ***** End of Include file MAGIC1.INC *****

OVERLAY procedure US_COMPOSITE_VARIABLES;  { A SUB MENU UNDER EXOGENOUS VARIABLE SUB MENU }

        ***** Beginning of Include file MAGIC2.INC *****
{$I MAGIC2.INC}        { US COMPOSITES }
OVERLAY PROCEDURE US_COMPOSITES_SCREEN_1; { B:EXOGVAR5 }
BEGIN

  screenSS:=15; screen_fieldSS:=32; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR5.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR5') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,4,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  BEGIN
       YEARB[NY-3]:=YEAR[NY-3]+1;
       GETNUM(44,4,2,'N',YEARB[NY-3],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
```

```
3:  GETNUM(52,4,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  BEGIN
       YEARB[NY-2]:=YEAR[NY-2]+1;
       GETNUM(55,4,2,'N',YEARB[NY-2],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
5:  GETNUM(63,4,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  BEGIN
       YEARB[NY-1]:=YEAR[NY-1]+1;
       GETNUM(66,4,2,'N',YEARB[NY-1],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
7:  GETNUM(74,4,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  BEGIN
       YEARB[NY]:=YEAR[NY]+1;
       GETNUM(77,4,2,'N',YEARB[NY],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
9:  GETNUM(41,11,5,'N',ARP[NY-3],
    '#.###',0.000,9.999,retSS,FALSE,15,0);
10: GETNUM(41,13,5,'N',DIVERSION[NY-3],
    '#.###',0.000,9.999,retSS,FALSE,15,0);
11:  GETNUM(42,15,4,'N',GROSS[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
12:  BEGIN
        BAL[NY-3]:=(1.0 - (ARP[NY-3] + DIVERSION[NY-3]));
        GETNUM(41,18,5,'N',BAL[NY-3],
        '#.###',0.000,9.999,retSS,FALSE,15,0);
     END;
13:  GETNUM(41,20,5,'N',PYIELD[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
14:  BEGIN
        PARREV[NY-3]:=GROSS[NY-3]*BAL[NY-3]*PYIELD[NY-3];
        GETNUM(40,22,6,'N',PARREV[NY-3],
        '###.##',000.00,999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(52,11,5,'N',ARP[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
16:  GETNUM(52,13,5,'N',DIVERSION[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
17:  GETNUM(53,15,4,'N',GROSS[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
18:  BEGIN
        BAL[NY-2]:=(1.0 - (ARP[NY-2] + DIVERSION[NY-2]));
        GETNUM(52,18,5,'N',BAL[NY-2],
        '#.###',0.000,9.999,retSS,FALSE,15,0);
     END;
19:  GETNUM(52,20,5,'N',PYIELD[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
20:  BEGIN
        PARREV[NY-2]:=GROSS[NY-2]*BAL[NY-2]*PYIELD[NY-2];
```

```
      GETNUM(51,22,6,'N',PARREV[NY-2],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
    END;
21: GETNUM(63,11,5,'N',ARP[NY-1],
    '#.###',0.000,9.999,retSS,FALSE,15,0);
22: GETNUM(63,13,5,'N',DIVERSION[NY-1],
    '#.###',0.000,9.999,retSS,FALSE,15,0);
23: GETNUM(64,15,4,'N',GROSS[NY-1],
    '#.##',0.00,9.99,retSS,FALSE,15,0);
24: BEGIN
      BAL[NY-1]:=(1.0 - (ARP[NY-1] + DIVERSION[NY-1]));
      GETNUM(63,18,5,'N',BAL[NY-1],
      '#.###',0.000,9.999,retSS,FALSE,15,0);
    END;
25: GETNUM(63,20,5,'N',PYIELD[NY-1],
    '###.#',0.0,999.9,retSS,FALSE,15,0);
26: BEGIN
      PARREV[NY-1]:=GROSS[NY-1]*BAL[NY-1]*PYIELD[NY-1];
      GETNUM(62,22,6,'N',PARREV[NY-1],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
    END;
27: GETNUM(74,11,5,'N',ARP[NY],
    '#.###',0.000,9.999,retSS,retrieveSS,15,0);
28: GETNUM(74,13,5,'N',DIVERSION[NY],
    '#.###',0.000,9.999,retSS,retrieveSS,15,0);
29: GETNUM(75,15,4,'N',GROSS[NY],
    '#.##',0.00,9.99,retSS,retrieveSS,15,0);
30: BEGIN
      BAL[NY]:=(1.0 - (ARP[NY] + DIVERSION[NY]));
      GETNUM(74,18,5,'N',BAL[NY],
      '#.###',0.000,9.999,retSS,FALSE,15,0);
    END;
31: GETNUM(74,20,5,'N',PYIELD[NY],
    '###.#',0.0,999.9,retSS,retrieveSS,15,0);
32: BEGIN
      PARREV[NY]:=GROSS[NY]*BAL[NY]*PYIELD[NY];
      GETNUM(73,22,6,'N',PARREV[NY],
      '###.##',0.00,999.99,retSS,FALSE,15,0);
    END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE US_COMPOSITES_SCREEN_1 }
```

```
OVERLAY PROCEDURE US_COMPOSITES_SCREEN_2; { B:EXOVAR5A }
BEGIN

  screenSS:=16; screen_fieldSS:=24; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOVAR5A.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOVAR5A') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(52,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(55,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(63,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(66,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(42,13,4,'N',DVPCN[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
10:  GETNUM(41,16,5,'N',DIVERSION[NY-3],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
11:  GETNUM(41,19,5,'N',PYIELD[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
12:  BEGIN
        DIVPAY[NY-3]:=DVPCN[NY-3]*DIVERSION[NY-3]*PYIELD[NY-3];
        GETNUM(40,21,6,'N',DIVPAY[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
13:  GETNUM(53,13,4,'N',DVPCN[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
14:  GETNUM(52,16,5,'N',DIVERSION[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
15:  GETNUM(52,19,5,'N',PYIELD[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
16:  BEGIN
        DIVPAY[NY-2]:=DVPCN[NY-2]*DIVERSION[NY-2]*PYIELD[NY-2];
        GETNUM(51,21,6,'N',DIVPAY[NY-2],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
17:  GETNUM(64,13,4,'N',DVPCN[NY-1],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
18:  GETNUM(63,16,5,'N',DIVERSION[NY-1],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
19:  GETNUM(63,19,5,'N',PYIELD[NY-1],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
20:  BEGIN
        DIVPAY[NY-1]:=DVPCN[NY-1]*DIVERSION[NY-1]*PYIELD[NY-1];
        GETNUM(62,21,6,'N',DIVPAY[NY-1],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(75,13,4,'N',DVPCN[NY],
     '#.##',0.00,9.99,retSS,retrieveSS,15,0);
22:  GETNUM(74,16,5,'N',DIVERSION[NY],
     '#.###',0.000,9.999,retSS,retrieveSS,15,0);
23:  GETNUM(74,19,5,'N',PYIELD[NY],
     '###.#',0.0,999.9,retSS,retrieveSS,15,0);
24:  BEGIN
        DIVPAY[NY]:=DVPCN[NY]*DIVERSION[NY]*PYIELD[NY];
        GETNUM(73,21,6,'N',DIVPAY[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE US_COMPOSITES_SCREEN_2 }



OVERLAY PROCEDURE US_COMPOSITES_SCREEN_3; { B:EXOVAR5B }
BEGIN

  screenSS:=17; screen_fieldSS:=40; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOVAR5B.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOVAR5B') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,4,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,4,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(52,4,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(55,4,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(63,4,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(66,4,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,4,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,4,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(41,8,5,'N',ARP[NY-3],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
10:  GETNUM(41,10,5,'N',DIVERSION[NY-3],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
11:  GETNUM(42,12,4,'N',C_TARGETP[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
12:  GETNUM(42,14,4,'N',GROSS[NY-3],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
13:  BEGIN
        DPAY[NY-3]:=C_TARGETP[NY-3]-GROSS[NY-3];
        IF DPAY[NY-3] < 0.0 THEN DPAY[NY-3]:=0.0;
        GETNUM(42,17,4,'N',DPAY[NY-3],
        '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
14:  BEGIN
        BAL[NY-3]:=(1.0 - (ARP[NY-3] + DIVERSION[NY-3]));
        GETNUM(41,19,5,'N',BAL[NY-3],
        '#.###',0.000,9.999,retSS,FALSE,15,0);
     END;
15:  GETNUM(41,21,5,'N',PYIELD[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
16:  BEGIN
        DEFPAY[NY-3]:=DPAY[NY-3]*BAL[NY-3]*PYIELD[NY-3];
        GETNUM(40,23,6,'N',DEFPAY[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
17:  GETNUM(52,8,5,'N',ARP[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
18:  GETNUM(52,10,5,'N',DIVERSION[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
19:  GETNUM(53,12,4,'N',C_TARGETP[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
20:  GETNUM(53,14,4,'N',GROSS[NY-2],
     '#.##',0.00,9.99,retSS,FALSE,15,0);
21:  BEGIN
        DPAY[NY-2]:=C_TARGETP[NY-2]-GROSS[NY-2];
        IF DPAY[NY-2] < 0.0 THEN DPAY[NY-2]:=0.0;
        GETNUM(53,17,4,'N',DPAY[NY-2],
        '#.##',0.00,9.99,retSS,FALSE,15,0);
     END;
22:  BEGIN
```

```
         BAL[NY-2]:=(1.0 - (ARP[NY-2] + DIVERSION[NY-2]));
         GETNUM(52,19,5,'N',BAL[NY-2],
         '#.###',0.000,9.999,retSS,FALSE,15,0);
      END;
23:   GETNUM(52,21,5,'N',PYIELD[NY-2],
      '###.#',0.0,999.9,retSS,FALSE,15,0);
24:   BEGIN
         DEFPAY[NY-2]:=DPAY[NY-2]*BAL[NY-2]*PYIELD[NY-2];
         GETNUM(51,23,6,'N',DEFPAY[NY-2],
         '###.##',0.00,999.99,retSS,FALSE,15,0);
      END;
25:   GETNUM(63,8,5,'N',ARP[NY-1],
      '#.###',0.000,9.999,retSS,FALSE,15,0);
26:   GETNUM(63,10,5,'N',DIVERSION[NY-1],
      '#.###',0.000,9.999,retSS,FALSE,15,0);
27:   GETNUM(64,12,4,'N',C_TARGETP[NY-1],
      '#.##',0.00,9.99,retSS,FALSE,15,0);
28:   GETNUM(64,14,4,'N',GROSS[NY-1],
      '#.##',0.00,9.99,retSS,FALSE,15,0);
29:   BEGIN
         DPAY[NY-1]:=C_TARGETP[NY-1]-GROSS[NY-1];
         IF DPAY[NY-1] < 0.0 THEN DPAY[NY-1]:=0.0;
         GETNUM(64,17,4,'N',DPAY[NY-1],
         '#.##',0.00,9.99,retSS,FALSE,15,0);
      END;
30:   BEGIN
         BAL[NY-1]:=(1.0 - (ARP[NY-1] + DIVERSION[NY-1]));
         GETNUM(63,19,5,'N',BAL[NY-1],
         '#.###',0.000,9.999,retSS,FALSE,15,0);
      END;
31:   GETNUM(63,21,5,'N',PYIELD[NY-1],
      '###.#',0.0,999.9,retSS,FALSE,15,0);
32:   BEGIN
         DEFPAY[NY-1]:=DPAY[NY-1]*BAL[NY-1]*PYIELD[NY-1];
         GETNUM(62,23,6,'N',DEFPAY[NY-1],
         '###.##',0.00,999.99,retSS,FALSE,15,0);
      END;
33:   GETNUM(74,8,5,'N',ARP[NY],
      '#.###',0.000,9.999,retSS,retrieveSS,15,0);
34:   GETNUM(74,10,5,'N',DIVERSION[NY],
      '#.###',0.000,9.999,retSS,retrieveSS,15,0);
35:   GETNUM(75,12,4,'N',C_TARGETP[NY],
      '#.##',0.00,9.99,retSS,retrieveSS,15,0);
36:   GETNUM(75,14,4,'N',GROSS[NY],
      '#.##',0.00,9.99,retSS,retrieveSS,15,0);
37:   BEGIN
         DPAY[NY]:=C_TARGETP[NY]-GROSS[NY];
         IF DPAY[NY] < 0.0 THEN DPAY[NY]:=0.0;
         GETNUM(75,17,4,'N',DPAY[NY],
         '#.##',0.00,9.99,retSS,FALSE,15,0);
      END;
38:   BEGIN
         BAL[NY]:=(1.0 - (ARP[NY] + DIVERSION[NY]));
         GETNUM(74,19,5,'N',BAL[NY],
        '#.###',0.000,9.999,retSS,FALSE,15,0);
     END;
39:  GETNUM(74,21,5,'N',PYIELD[NY],
     '###.#',0.0,999.9,retSS,retrieveSS,15,0);
40:  BEGIN
        DEFPAY[NY]:=DPAY[NY]*BAL[NY]*PYIELD[NY];
        GETNUM(73,23,6,'N',DEFPAY[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE US COMPOSITES SCREEN 3 }
        ***** End of Include file MAGIC2.INC *****

BEGIN
REPEAT
      IF (EDIT=FALSE) AND (BACTIVE=FALSE)
      THEN BEGIN
           COLOR(WHITE,BLACK);
           CLRSCR;
           FLASHUP('W=USCOMPOS,NOESC,NOMOVE');
           GOTOXY(1,25);
           READ(RESPONSE);
           CLREOL;
           FLASHUP('C=ALL');
           END;
           CASE (RESPONSE) OF
                '1': US_COMPOSITES_SCREEN_1;     { GOVT PARTICIPANTS REV. }
                '2': US_COMPOSITES_SCREEN_2;     { DIVERSION PAYMENTS }
                '3': US_COMPOSITES_SCREEN_3;     { DEFFICIENCY PAYMENTS }
                '4': FSI_USE_SCREEN;
                '5':;
           END;  {CASE}
      IF (RESPONSE <> '5') THEN
      WHATMENU(BSCREEN,BACTIVE,UB);
UNTIL (RESPONSE = '5');
END;   {  PROCEDURE US COMPOSITES   }

        ***** Beginning of Include file MAGIC3.INC *****
{$I MAGIC3.INC}       { IMPORTERS GDP }
OVERLAY PROCEDURE IMPORTERS_GDP_SCREEN_1; { B:EXOGVAR6 }
BEGIN
```

```
  screenSS:=18; screen_fieldSS:=31; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR6.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR6') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(31,6,5,'N',CHWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
2:  GETNUM(31,7,5,'N',SBWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
3:  GETNUM(31,8,5,'N',DMWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
4:  GETNUM(31,9,5,'N',LOWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
5:  GETNUM(31,10,5,'N',NIWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
6:  GETNUM(31,11,5,'N',LDWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
7:  GETNUM(31,12,5,'N',BRWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
8:  GETNUM(41,13,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(44,13,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
10:  GETNUM(52,13,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
11:  GETNUM(55,13,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
12:  GETNUM(63,13,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
13:  GETNUM(66,13,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
14:  GETNUM(74,13,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
15:  GETNUM(77,13,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
16:  GETNUM(41,17,5,'N',CPICH[NY-3],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
17:  GETNUM(40,19,6,'N',CPISB[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
18:  GETNUM(40,21,6,'N',CPIDM[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
19:  GETNUM(40,23,6,'N',CPILO[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
20:  GETNUM(52,17,5,'N',CPICH[NY-2],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
21:  GETNUM(51,19,6,'N',CPISB[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
22:  GETNUM(51,21,6,'N',CPIDM[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
```

```
23:  GETNUM(51,23,6,'N',CPILO[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
24:  GETNUM(63,17,5,'N',CPICH[NY-1],
     '###.#',0.0,999.9,retSS,FALSE,15,0);
25:  GETNUM(62,19,6,'N',CPISB[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
26:  GETNUM(62,21,6,'N',CPIDM[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
27:  GETNUM(62,23,6,'N',CPILO[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
28:  GETNUM(74,17,5,'N',CPICH[NY],
     '###.#',0.0,999.9,retSS,retrieveSS,15,0);
29:  GETNUM(73,19,6,'N',CPISB[NY],
     '###.##',0.00,999.99,retSS,retrieveSS,15,0);
30:  GETNUM(73,21,6,'N',CPIDM[NY],
     '###.##',0.00,999.99,retSS,retrieveSS,15,0);
31:  GETNUM(73,23,6,'N',CPILO[NY],
     '###.##',0.00,999.99,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_6 }



OVERLAY PROCEDURE IMPORTERS_GDP_SCREEN_2; { B:EXOGVAR7 }
BEGIN

  screenSS:=19; screen_fieldSS:=36; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR7.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR7') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,4,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,4,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(52,4,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
```

```
4:  GETNUM(55,4,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(63,4,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(66,4,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,4,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,4,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(40,8,6,'N',CPINI[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
10:  GETNUM(40,10,6,'N',CPILD[NY-3],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
11:  GETNUM(40,12,6,'N',CPIBR[NY-3],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
12:  GETNUM(40,17,6,'N',GDPCH[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
13:  GETNUM(40,19,6,'N',GDPSB[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
14:  GETNUM(40,21,6,'N',GDPDM[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
15:  GETNUM(40,23,6,'N',GDPLO[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
16:  GETNUM(51,8,6,'N',CPINI[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
17:  GETNUM(51,10,6,'N',CPILD[NY-2],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
18:  GETNUM(51,12,6,'N',CPIBR[NY-2],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
19:  GETNUM(51,17,6,'N',GDPCH[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
20:  GETNUM(51,19,6,'N',GDPSB[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
21:  GETNUM(51,21,6,'N',GDPDM[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
22:  GETNUM(51,23,6,'N',GDPLO[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
23:  GETNUM(62,8,6,'N',CPINI[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
24:  GETNUM(62,10,6,'N',CPILD[NY-1],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
25:  GETNUM(62,12,6,'N',CPIBR[NY-1],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
26:  GETNUM(62,17,6,'N',GDPCH[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
27:  GETNUM(62,19,6,'N',GDPSB[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
28:  GETNUM(62,21,6,'N',GDPDM[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
29:  GETNUM(62,23,6,'N',GDPLO[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
30:  GETNUM(73,8,6,'N',CPINI[NY],
     '###.##',0.00,999.99,retSS,retrieveSS,15,0);
```

```
31:  GETNUM(73,10,6,'N',CPILD[NY],
     '####.#',0.0,9999.9,retSS,retrieveSS,15,0);
32:  GETNUM(73,12,6,'N',CPIBR[NY],
     '####.#',0.0,9999.9,retSS,retrieveSS,15,0);
33:  GETNUM(73,17,6,'N',GDPCH[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
34:  GETNUM(73,19,6,'N',GDPSB[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
35:  GETNUM(73,21,6,'N',GDPDM[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
36:  GETNUM(73,23,6,'N',GDPLO[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_7 }



OVERLAY PROCEDURE IMPORTERS_GDP_SCREEN_3; { B:EXOGVAR8 }
BEGIN

  screenSS:=20; screen_fieldSS:=24; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR8.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR8') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(52,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(55,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(63,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(66,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
```

```
7:  GETNUM(74,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(40,9,6,'N',GDPNI[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
10:  GETNUM(40,11,6,'N',GDPLD[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
11:  GETNUM(40,13,6,'N',GDPBR[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
12:  GETNUM(38,21,8,'N',IGDP[NY-3],
     '#.######',0.000000,9.999999,retSS,FALSE,15,0);
13:  GETNUM(51,9,6,'N',GDPNI[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
14:  GETNUM(51,11,6,'N',GDPLD[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
15:  GETNUM(51,13,6,'N',GDPBR[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
16:  GETNUM(49,21,8,'N',IGDP[NY-2],
     '#.######',0.000000,9.999999,retSS,FALSE,15,0);
17:  GETNUM(62,9,6,'N',GDPNI[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
18:  GETNUM(62,11,6,'N',GDPLD[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
19:  GETNUM(62,13,6,'N',GDPBR[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
20:  GETNUM(60,21,8,'N',IGDP[NY-1],
     '#.######',0.000000,9.999999,retSS,FALSE,15,0);
21:  GETNUM(73,9,6,'N',GDPNI[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
22:  GETNUM(73,11,6,'N',GDPLD[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
23:  GETNUM(73,13,6,'N',GDPBR[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
24:  BEGIN
        if retrieveSS=true then begin
        IGDP[NY]:=((CHWT*GDPCH[NY])/CPICH[NY])
        + ((SBWT*GDPSB[NY])/CPISB[NY]) + ((DMWT*GDPDM[NY])/CPIDM[NY])
        + ((LOWT*GDPLO[NY])/CPILO[NY]) + ((NIWT*GDPNI[NY])/CPINI[NY])
        + ((LDWT*GDPLD[NY])/CPILD[NY]) + ((BRWT*GDPBR[NY])/CPIBR[NY]);
        GETNUM(71,21,8,'N',IGDP[NY],
        '#.######',0.000000,9.999999,retSS,retrieveSS,15,0);
     end;
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
```

```
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_8 }

        ***** End of Include file MAGIC3.INC *****

OVERLAY procedure EXPORTERS_COMPOSITES;  { A SUB MENU UNDER EXOGENOUS VARIABLE SUB MENU }

        ***** Beginning of Include file MAGIC4.INC *****
{$I MAGIC4.INC}        { EXPORTERS GDP }
OVERLAY PROCEDURE EXPORTERS_GDP_SCREEN_1; { B:EXOGVAR9 }
BEGIN

  screenSS:=21; screen_fieldSS:=23; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOGVAR9.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOGVAR9') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(31,7,5,'N',ARWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
2:  GETNUM(31,9,5,'N',AUWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
3:  GETNUM(31,11,5,'N',CAWT,
     '#.###',0.000,1.000,retSS,retrieveSS,15,0);
4:  GETNUM(41,14,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(44,14,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(52,14,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(55,14,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(63,14,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(66,14,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
10:  GETNUM(74,14,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
11:  GETNUM(77,14,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
12:  GETNUM(39,18,7,'N',CPIAR[NY-3],
     '#####.#',0.0,99999.9,retSS,FALSE,15,0);
13:  GETNUM(40,20,6,'N',CPIAU[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
14:  GETNUM(40,22,6,'N',CPICA[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
15:  GETNUM(50,18,7,'N',CPIAR[NY-2],
     '#####.#',0.0,99999.9,retSS,FALSE,15,0);
```

```
16:  GETNUM(51,20,6,'N',CPIAU[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
17:  GETNUM(51,22,6,'N',CPICA[NY-2],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
18:  GETNUM(61,18,7,'N',CPIAR[NY-1],
     '#####.#',0.0,99999.9,retSS,FALSE,15,0);
19:  GETNUM(62,20,6,'N',CPIAU[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
20:  GETNUM(62,22,6,'N',CPICA[NY-1],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
21:  GETNUM(72,18,7,'N',CPIAR[NY],
     '#####.#',0.0,99999.9,retSS,retrieveSS,15,0);
22:  GETNUM(73,20,6,'N',CPIAU[NY],
     '###.##',0.00,999.99,retSS,retrieveSS,15,0);
23:  GETNUM(73,22,6,'N',CPICA[NY],
     '###.##',0.00,999.99,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_9 }


OVERLAY PROCEDURE EXPORTERS_GDP_SCREEN_2; { B:EXOVAR10 }
BEGIN

  screenSS:=22; screen_fieldSS:=24; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOVAR10.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOVAR10') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(52,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(55,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(63,5,2,'N',YEAR[NY-1],
    '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(66,5,2,'N',YEAR[NY-1],
    '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,5,2,'N',YEAR[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,5,2,'N',YEARB[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(40,9,6,'N',GDPAR[NY-3],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
10:  GETNUM(40,11,6,'N',GDPAU[NY-3],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
11:  GETNUM(40,13,6,'N',GDPCA[NY-3],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
12:  GETNUM(38,21,8,'N',EGDP[NY-3],
    '#.######',0.000000,9.999999,retSS,FALSE,15,0);
13:  GETNUM(51,9,6,'N',GDPAR[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
14:  GETNUM(51,11,6,'N',GDPAU[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
15:  GETNUM(51,13,6,'N',GDPCA[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
16:  GETNUM(49,21,8,'N',EGDP[NY-2],
    '#.######',0.000000,9.999999,retSS,FALSE,15,0);
17:  GETNUM(62,9,6,'N',GDPAR[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
18:  GETNUM(62,11,6,'N',GDPAU[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
19:  GETNUM(62,13,6,'N',GDPCA[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
20:  GETNUM(60,21,8,'N',EGDP[NY-1],
    '#.######',0.000000,9.999999,retSS,FALSE,15,0);
21:  GETNUM(73,9,6,'N',GDPAR[NY],
    '###.##',0.00,999.99,retSS,retrieveSS,15,0);
22:  GETNUM(73,11,6,'N',GDPAU[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
23:  GETNUM(73,13,6,'N',GDPCA[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
24:  BEGIN
        if retrieveSS=true then begin
        EGDP[NY]:=((ARWT*GDPAR[NY])/CPIAR[NY])
        + ((AUWT*GDPAU[NY])/CPIAU[NY]) + ((CAWT*GDPCA[NY])/CPICA[NY]);
        GETNUM(71,21,8,'N',EGDP[NY],
        '#.######',0.000000,9.999999,retSS,retrieveSS,15,0);
     end;
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
```

```
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_10 }

        ***** End of Include file MAGIC4.INC *****
        ***** Beginning of Include file MAGIC5.INC *****
{$I MAGIC5.INC}       { EXPORTERS REVENUE }
OVERLAY PROCEDURE EXPORTER_CROP_REVENUE; { B:EXOVAR11 }
BEGIN

  screenSS:=23; screen_fieldSS:=44; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('EXOVAR11.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen EXOVAR11') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(41,3,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(44,3,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(52,3,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(55,3,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(63,3,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(66,3,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(74,3,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(77,3,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(40,7,6,'N',XRAR[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
10:  GETNUM(40,9,6,'N',XRAU[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
11:  GETNUM(40,11,6,'N',XRCA[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
12:  GETNUM(40,13,6,'N',FYAR[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
13:  GETNUM(40,15,6,'N',FYAU[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
14:  GETNUM(40,17,6,'N',FYCA[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
15:  GETNUM(40,19,6,'N',WYAR[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
16:  GETNUM(40,21,6,'N',WYAU[NY-3],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
```

```
GETNUM(40,23,6,'N',WYCA[NY-3],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,7,6,'N',XRAR[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,9,6,'N',XRAU[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,11,6,'N',XRCA[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,13,6,'N',FYAR[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,15,6,'N',FYAU[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,17,6,'N',FYCA[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,19,6,'N',WYAR[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,21,6,'N',WYAU[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(51,23,6,'N',WYCA[NY-2],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,7,6,'N',XRAR[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,9,6,'N',XRAU[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,11,6,'N',XRCA[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,13,6,'N',FYAR[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,15,6,'N',FYAU[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,17,6,'N',FYCA[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,19,6,'N',WYAR[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,21,6,'N',WYAU[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(62,23,6,'N',WYCA[NY-1],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
GETNUM(73,7,6,'N',XRAR[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,9,6,'N',XRAU[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,11,6,'N',XRCA[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,13,6,'N',FYAR[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,15,6,'N',FYAU[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,17,6,'N',FYCA[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,19,6,'N',WYAR[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
GETNUM(73,21,6,'N',WYAU[NY],
    '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
```

```
44:  GETNUM(73,23,6,'N',WYCA[NY],
       '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
   END; { CASE }

   if varSS=screen_fieldSS then last_fieldSS:=TRUE;
   RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
   retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
   last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_11 }

         ***** End of Include file MAGIC5.INC *****

BEGIN

REPEAT
      IF (EDIT=FALSE) AND (BACTIVE=FALSE)
      THEN BEGIN
           COLOR(WHITE,BLACK);
           CLRSCR;
           FLASHUP('W=EXPORTER,NOESC,NOMOVE');
           GOTOXY(1,25);
           READ(RESPONSE);
           CLREOL;
           FLASHUP('C=ALL');
           END;
           CASE (RESPONSE) OF
                '1':EXPORTER_CROP_REVENUE;
                '2':BEGIN      { EXPORTERS_GDP }
                      UB:=2;
                       BACTIVE:=TRUE;
                       CASE (BSCREEN) OF
                          1:  EXPORTERS_GDP_SCREEN_1;
                          2:  EXPORTERS_GDP_SCREEN_2;
                       END;  {CASE BSCREEN OF}
                    END;
                '3'::
           END;  {CASE}
      IF (RESPONSE <> '3') THEN
      WHATMENU(BSCREEN,BACTIVE,UB);
UNTIL (RESPONSE = '3');
END;   {  PROCEDURE EXPORTERS COMPOSITES   }


BEGIN
REPEAT
   IF (EDIT=FALSE) AND (BACTIVE=FALSE) THEN  .
      BEGIN
```

```
            COLOR(WHITE,BLACK);
            CLRSCR;
            FLASHUP('W=EXOGSUMM,NOESC,NOMOVE');
            GOTOXY(1,25);
            READ(RESPONSE);
            CLREOL;
            FLASHUP('C=ALL');
            END;
            CASE (RESPONSE) OF
                 '1':US_COMPOSITE_VARIABLES;
                 '2':BEGIN            { IMPORTERS_GDP }
                        UB:=3;
                        BACTIVE:=TRUE;
                        CASE (BSCREEN) OF
                           1: IMPORTERS_GDP_SCREEN_1;
                           2: IMPORTERS_GDP_SCREEN_2;
                           3: IMPORTERS_GDP_SCREEN_3;
                        END; { CASE BSCREEN OF }
                     END;
                 '3':EXPORTERS_COMPOSITES;
                 '4':BEGIN            { EXOGENOUS_VARIABLE_SUMMARY }
                        UB:=5;
                        BACTIVE:=TRUE;
                        CASE (BSCREEN) OF
                           1: SCREEN_1;
                           2: SCREEN_3A;
                           3: SCREEN_2;
                           4: SCREEN_3;
                           5: SCREEN_4;
                        END; { CASE BSCREEN OF }
                     END;
                 '5':;
            END;  {CASE}
       IF (RESPONSE <> '5') THEN
       WHATMENU(BSCREEN,BACTIVE,UB);

UNTIL (RESPONSE = '5');
END;  {  PROCEDURE EXOGENOUS VARIABLE SUB MENU  }

overlay procedure EXPORT_VARIABLE_SUB_MENU;

        ***** Beginning of Include file WETT.INC *****
{$I WETT.INC}           {  exporters net exports and us exports }
(* ********************************************************************
   { Initialize Variables To Default Values }
    BEGIN           { formula (((x*39.368)/1000)*0.85995 }
                    { corn = 0.85995 as a ' % ' of exports }
  esply_in_bu[1]:=1890.21;             {   actual values     }
  econ_in_bu[1]:=914.68;               { ECON[1]:=27017.96;    }
  ees_in_bu[1]:=294.47;                { EES[1]:=8698.01;      }
  ene_in_bu[1]:=681.06;                { ENE[1]:=19112.01;     }
  eha[1]:=19038.0;

  esply_in_bu[2]:=1824.19;
  econ_in_bu[2]:=986.76;                { ECON[2]:=29146.99;    }
  ees_in_bu[2]:=168.59;                 { EES[2]:=4979.98;      }
  ene_in_bu[2]:=668.84;                 { ENE[2]:=21370.00;     }
  eha[2]:=19065.0;

  esply_in_bu[3]:=1840.10;
  econ_in_bu[3]:=961.13;                { ECON[3]:=28390.02;    }
  ees_in_bu[3]:=171.34;                 { EES[3]:=5060.98;      }
  ene_in_bu[3]:=707.63;                 { ENE[3]:=20425.00;     }
  eha[3]:=19424.0;

    INI_IN_BU[1]:=2552.03;
    INI_IN_BU[2]:=2532.85;
    INI_IN_BU[3]:=2545.63;

********************************************************************* *)
procedure EXPORTERS_NET_EXPORTS_SCREEN; { B:ENEXPORT }
BEGIN

  screenSS:=24; screen_fieldSS:=32; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('ENEXPORT.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen ENEXPORT') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(31,6,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  begin
      yearb[NY-3]:=year[NY-3]+1;
      GETNUM(34,6,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
3:  GETNUM(41,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  begin
      yearb[NY-2]:=year[NY-2]+1;
      GETNUM(44,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
5:  GETNUM(51,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  begin
      yearb[NY-1]:=year[NY-1]+1;
      GETNUM(54,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
7:  GETNUM(61,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  begin
      yearb[NY]:=year[NY]+1;
```

```
        GETNUM(64,6,2,'N',YEARB[NY],
       '##',0.0,99.0,retSS,FALSE,15,0);
       end;
9:        GETNUM(28,11,8,'N',ESPLY_IN_BU[NY-3],
          '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
10:     GETNUM(28,15,8,'N',ECON_IN_BU[NY-3],
           '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
11:     GETNUM(29,19,7,'N',EES_IN_BU[NY-3],
           '####.##',000.00,9999.99,retSS,FALSE,15,0);
12:     GETNUM(29,22,7,'N',ENE_IN_BU[NY-3],
           '####.##',0.00,9999.99,retSS,FALSE,15,0);
13:     GETNUM(38,11,8,'N',ESPLY_IN_BU[NY-2],
          '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
14:       GETNUM(38,15,8,'N',ECON_IN_BU[NY-2],
           '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
15:       GETNUM(39,19,7,'N',EES_IN_BU[NY-2],
           '####.##',000.00,9999.99,retSS,FALSE,15,0);
16:       GETNUM(39,22,7,'N',ENE_IN_BU[NY-2],
           '####.##',0.00,9999.99,retSS,FALSE,15,0);
17:     GETNUM(48,11,8,'N',ESPLY_IN_BU[NY-1],
          '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
18:       GETNUM(48,15,8,'N',ECON_IN_BU[NY-1],
           '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
19:       GETNUM(49,19,7,'N',EES_IN_BU[NY-1],
           '####.##',000.00,9999.99,retSS,FALSE,15,0);
20:       GETNUM(49,22,7,'N',ENE_IN_BU[NY-1],
           '####.##',0.00,9999.99,retSS,FALSE,15,0);
21:       GETNUM(58,11,8,'N',ESPLY_IN_BU[NY],
           '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
22:       GETNUM(58,15,8,'N',ECON_IN_BU[NY],
           '#####.##',0000.00,99999.99,retSS,FALSE,15,0);
23:       GETNUM(59,19,7,'N',EES_IN_BU[NY],
           '####.##',000.00,9999.99,retSS,FALSE,15,0);
24:       GETNUM(59,22,7,'N',ENE_IN_BU[NY],
           '####.##',0.00,9999.99,retSS,FALSE,15,0);
25:       GETITEM(70,11,1,'Y',ESPLY_IN_BU_FLAG,
          'U',' ',' ',retSS,retrieveSS,15,0);
26:       BEGIN
            IF (RETRIEVESS=TRUE) AND (ESPLY_IN_BU_FLAG='Y') THEN
            BEGIN
              FG:=15;
              BG:=0;
              GETNUM(74,11,5,'N',ESPLY_IN_BU_OR[NY],
               '#####',0.0,99999.9,retSS,RETRIEVESS,FG,BG);
            END
            ELSE
            BEGIN
              FG:=0;
              BG:=0;
              GETNUM(74,11,5,'N',ESPLY_IN_BU_OR[NY],
               '#####',0.0,99999.9,retSS,FALSE,FG,BG);
            END;
          END;
27:       GETITEM(70,15,1,'Y',ECON_IN_BU_FLAG,
       'U',' ',' ',retSS,retrieveSS,15,0);
28:    BEGIN
         IF (RETRIEVESS=TRUE) AND (ECON_IN_BU_FLAG='Y') THEN
         BEGIN
           FG:=15;
           BG:=0;
           GETNUM(74,15,5,'N',ECON_IN_BU_OR[NY],
           '#####',0.0,99999.9,retSS,RETRIEVESS,FG,BG);
         END
         ELSE
         BEGIN
           FG:=0;
           BG:=0;
           GETNUM(74,15,5,'N',ECON_IN_BU_OR[NY],
           '#####',0.0,99999.9,retSS,FALSE,FG,BG);
         END;
       END;
29:    GETITEM(70,19,1,'Y',EES_IN_BU_FLAG,
       'U',' ',' ',retSS,retrieveSS,15,0);
30:    BEGIN
         IF (RETRIEVESS=TRUE) AND (EES_IN_BU_FLAG='Y') THEN
         BEGIN
           FG:=15;
           BG:=0;
           GETNUM(74,19,5,'N',EES_IN_BU_OR[NY],
           '#####',0.0,99999.9,retSS,RETRIEVESS,FG,BG);
         END
         ELSE
         BEGIN
           FG:=0;
           BG:=0;
           GETNUM(74,19,5,'N',EES_IN_BU_OR[NY],
           '#####',0.0,99999.9,retSS,FALSE,FG,BG);
         END;
       END;
31:    GETITEM(70,22,1,'Y',ENE_IN_BU_FLAG,
       'U',' ',' ',retSS,retrieveSS,15,0);
32:    BEGIN
         IF (RETRIEVESS=TRUE) AND (ENE_IN_BU_FLAG='Y') THEN
         BEGIN
           FG:=15;
           BG:=0;
           GETNUM(74,22,5,'N',ENE_IN_BU_OR[NY],
           '#####',0.0,99999.9,retSS,RETRIEVESS,FG,BG);
         END
         ELSE
         BEGIN
           FG:=0;
           BG:=0;
           GETNUM(74,22,5,'N',ENE_IN_BU_OR[NY],
           '#####',0.0,99999.9,retSS,FALSE,FG,BG);
         END;
       END;
  END; { CASE }
  COLOR(WHITE,BLACK);

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }


{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE EXPORTERS NET EXPORTS SCREEN_1 }



procedure US_EXPORTS_SCREEN; { B:XXPORT }
BEGIN

  screenSS:=25; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('XXPORT.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen XXPORT') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(30,5,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  begin
      yearb[NY-3]:=year[NY-3]+1;
      GETNUM(33,5,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
3:  GETNUM(40,5,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  begin
      yearb[NY-2]:=year[NY-2]+1;
      GETNUM(43,5,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
5:  GETNUM(50,5,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  begin
      yearb[NY-1]:=year[NY-1]+1;
      GETNUM(53,5,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
7:  GETNUM(60,5,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
```

```
8: begin
     yearb[NY]:=year[NY]+1;
     GETNUM(63,5,2,'N',YEARB[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
    end;
9: GETNUM(28,11,7,'N',INI_IN_BU[NY-3],
   '####.##',0000.00,9999.99,retSS,FALSE,15,0);
10:  GETNUM(28,15,7,'N',ENE_IN_BU[NY-3],
    '####.##',0000.00,9999.99,retSS,FALSE,15,0);
11:  begin
       exports[NY-3]:=ini_in_bu[NY-3]-ene_in_bu[NY-3];
       GETNUM(29,19,6,'N',EXPORTS[NY-3],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
     end;
12:  GETNUM(38,11,7,'N',INI_IN_BU[NY-2],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
13:  GETNUM(38,15,7,'N',ENE_IN_BU[NY-2],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
14:  begin
       exports[NY-2]:=ini_in_bu[NY-2]-ene_in_bu[NY-2];
       GETNUM(39,19,6,'N',EXPORTS[NY-2],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
     end;
15:  GETNUM(48,11,7,'N',INI_IN_BU[NY-1],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
16:  GETNUM(48,15,7,'N',ENE_IN_BU[NY-1],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
17:  begin

       exports[NY-1]:=ini_in_bu[NY-1]-ene_in_bu[NY-1];
       GETNUM(49,19,6,'N',EXPORTS[NY-1],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
     end;
18:  GETNUM(58,11,7,'N',INI_IN_BU[NY],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
19:  GETNUM(58,15,7,'N',ENE_IN_BU[NY],
     '####.##',0000.00,9999.99,retSS,FALSE,15,0);
20:  begin
       exports[NY]:=ini_in_bu[NY]-ene_in_bu[NY];
       GETNUM(59,19,6,'N',EXPORTS[NY],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
     end;
21:    GETITEM(70,11,1,'Y',INI_IN_BU_FLAG,
       'U',' ',' ',retSS,retrieveSS,15,0);
22:    BEGIN
         IF (RETRIEVESS=TRUE) AND (INI_IN_BU_FLAG='Y') THEN
         BEGIN
           FG:=15;
           BG:=0;
           GETNUM(74,11,5,'N',INI_IN_BU_OR[NY],
            '#####',0.0,99999.9,retSS,RETRIEVESS,FG,BG);
         END
         ELSE
         BEGIN
```

```
            FG:=0;
            BG:=0;
            GETNUM(74,11,5,'N',INI_IN_BU_OR[NY],
            '#####',0.0,99999.9,retSS,FALSE,FG,BG);
          END;
        END;

  END; { CASE }
  COLOR(WHITE,BLACK);

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { PROCEDURE US EXPORTS SCREEN }

        ***** End of Include file WETT.INC *****

BEGIN
REPEAT
   IF (EDIT=FALSE) AND (BACTIVE=FALSE) THEN
      BEGIN
           COLOR(WHITE,BLACK);
           CLRSCR;
           FLASHUP('W=XPORTSUM,NOESC,NOMOVE');
           GOTOXY(1,25);
           READ(RESPONSE);
           CLREOL;
           FLASHUP('C=ALL');
           END;
           CASE (RESPONSE) OF
                '1': EXPORTERS_NET_EXPORTS_SCREEN;     { EXPORTER NET EXPORT }
                '2': US_EXPORTS_SCREEN;                { US EXPORT SUMMARY   }
                '3':;
           END; { CASE }
      IF (RESPONSE <> '3') THEN
      WHATMENU(BSCREEN,BACTIVE,UB);

UNTIL (RESPONSE = '3');

END;  {  PROCEDURE EXPORTER VARIABLE SUB MENU  }

 overlay procedure feed_grain_demand;

        ***** Beginning of Include file FEEDTOR.INC *****
{$I FEEDTOR.INC}
OVERLAY PROCEDURE DAIRY_ANIMAL_CONS_PERHEAD; { B:DACON }
```

```
BEGIN

  screenSS:=26; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('DACON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen DACON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(31,6,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  begin
    yearb[NY-3]:=year[NY-3]+1;
    GETNUM(34,6,2,'N',YEARB[NY-3],
    '##',0.0,99.0,retSS,FALSE,15,0);
    end;
3:  GETNUM(41,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  begin
    YEARB[NY-2]:=year[NY-2]+1;
    GETNUM(44,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
5:  GETNUM(51,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  begin
    YEARB[NY-1]:=year[NY-1]+1;
    GETNUM(54,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
7:  GETNUM(61,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  begin
    YEARB[NY]:=year[NY]+1;
    GETNUM(64,6,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
     end;
9:  GETNUM(30,9,6,'N',ONEPH[NY-3],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
10:  GETNUM(30,15,6,'N',DANUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
11:  BEGIN
        DAIRYCONS[NY-3]:=ONEPH[NY-3]*DANUM[NY-3];
        GETNUM(30,20,6,'N',DAIRYCONS[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
12:  GETNUM(40,9,6,'N',ONEPH[NY-2],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
13:  GETNUM(40,15,6,'N',DANUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
14:  BEGIN
```

```
        DAIRYCONS[NY-2]:=ONEPH[NY-2]*DANUM[NY-2];
        GETNUM(40,20,6,'N',DAIRYCONS[NY-2],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(50,9,6,'N',ONEPH[NY-1],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
16:  GETNUM(50,15,6,'N',DANUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
17:  BEGIN
        DAIRYCONS[NY-1]:=ONEPH[NY-1]*DANUM[NY-1];
        GETNUM(50,20,6,'N',DAIRYCONS[NY-1],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
18:  GETNUM(60,9,6,'N',ONEPH[NY],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
19:  GETNUM(60,15,6,'N',DANUM[NY],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
20:  BEGIN
        DAIRYCONS[NY]:=ONEPH[NY]*DANUM[NY];
        GETNUM(60,20,6,'N',DAIRYCONS[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(71,9,6,'N',ONEPH_OR[NY],
     '##.###',00.000,99.999,retSS,retrieveSS,15,0);
22:  BEGIN
        IF ONEPH_OR[NY]<>0.0 THEN
        DAIRYCONS_OR[NY]:=ONEPH_OR[NY]*DANUM[NY]
        ELSE DAIRYCONS_OR[NY]:=0.0;
        GETNUM(72,20,6,'N',DAIRYCONS_OR[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE DAIRY ANIMAL CONS PERHEAD }


OVERLAY PROCEDURE CATTLE_ON_FEED_CONS_PERHEAD; { B:COFCON }
BEGIN

  screenSS:=27; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('COFCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen COFCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(33,6,2,'N',YEAR[NY-3],
     '#X',0.0,9.0,retSS,FALSE,15,0);
2:  GETNUM(36,6,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(43,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(46,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(53,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(56,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(63,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(66,6,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(31,9,7,'N',TWOPH[NY-3],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
10:  GETNUM(32,15,6,'N',COFNUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
11:  BEGIN
        CATONFED[NY-3]:=TWOPH[NY-3]*COFNUM[NY-3];
        GETNUM(32,20,6,'N',CATONFED[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
12:  GETNUM(41,9,7,'N',TWOPH[NY-2],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
13:  GETNUM(42,15,6,'N',COFNUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
14:  BEGIN
        CATONFED[NY-2]:=TWOPH[NY-2]*COFNUM[NY-2];
        GETNUM(42,20,6,'N',CATONFED[NY-2],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(51,9,7,'N',TWOPH[NY-1],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
16:  GETNUM(52,15,6,'N',COFNUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
17:  BEGIN
        CATONFED[NY-1]:=TWOPH[NY-1]*COFNUM[NY-1];
        GETNUM(52,20,6,'N',CATONFED[NY-1],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
18:  GETNUM(61,9,7,'N',TWOPH[NY],
     '##.###',00.000,99.999,retSS,FALSE,15,0);
19:  GETNUM(62,15,6,'N',COFNUM[NY],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
20:  BEGIN
```

```
        CATONFED[NY]:=TWOPH[NY]*COFNUM[NY];
        GETNUM(62,20,6,'N',CATONFED[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(72,9,7,'N',TWOPH_OR[NY],
     '##.###',00.000,99.999,retSS,retrieveSS,15,0);
22:  BEGIN
        IF TWOPH_OR[NY]<>0.0 THEN
        CATONFED_OR[NY]:=TWOPH_OR[NY]*COFNUM[NY]
        ELSE CATONFED_OR[NY]:=0.0;
        GETNUM(73,20,6,'N',CATONFED_OR[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE CATTLE_ON_FEED_CONS_PERHEAD }


OVERLAY PROCEDURE OTHER_BEEF_CATTLE_CONS_PERHEAD; { B:OBCCON }
BEGIN

  screenSS:=28; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('OBCCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen OBCCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(34,6,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(37,6,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(44,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(47,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(54,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(57,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
```

```
7:  GETNUM(64,6,2,'N',YEAR[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(67,6,2,'N',YEARB[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(32,9,7,'N',THREEPH[NY-3],
    '#.###',0.000,9.999,retSS,FALSE,15,0);
10:  GETNUM(33,15,6,'N',OBCNUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
11:  BEGIN
        OTHERBEEFCAT[NY-3]:=THREEPH[NY-3]*OBCNUM[NY-3];
        GETNUM(33,20,6,'N',OTHERBEEFCAT[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
12:  GETNUM(42,9,7,'N',THREEPH[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
13:  GETNUM(43,15,6,'N',OBCNUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
14:  BEGIN
        OTHERBEEFCAT[NY-2]:=THREEPH[NY-2]*OBCNUM[NY-2];
        GETNUM(43,20,6,'N',OTHERBEEFCAT[NY-2],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(52,9,7,'N',THREEPH[NY-1],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
16:  GETNUM(53,15,6,'N',OBCNUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
17:  BEGIN
        OTHERBEEFCAT[NY-1]:=THREEPH[NY-1]*OBCNUM[NY-1];
        GETNUM(53,20,6,'N',OTHERBEEFCAT[NY-1],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
18:  GETNUM(62,9,7,'N',THREEPH[NY],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
19:  GETNUM(63,15,6,'N',OBCNUM[NY],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
20:  BEGIN
        OTHERBEEFCAT[NY]:=THREEPH[NY]*OBCNUM[NY];
        GETNUM(63,20,6,'N',OTHERBEEFCAT[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(72,9,7,'N',THREEPH_OR[NY],
     '#.###',0.000,9.999,retSS,retrieveSS,15,0);
22:  BEGIN
        IF THREEPH_OR[NY]<>0.0 THEN
        OTHERBEEFCAT_OR[NY]:=THREEPH_OR[NY]*OBCNUM[NY]
        ELSE OTHERBEEFCAT_OR[NY]:=0.0;
        GETNUM(73,20,6,'N',OTHERBEEFCAT_OR[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }
```

```
{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE OTHER_BEEF_CATTLE_CONS_PERHEAD }


OVERLAY PROCEDURE CHICKEN_CONS_PERHEAD; { B:HPCCON }
BEGIN

  screenSS:=29; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('HPCCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen HPCCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(29,6,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(32,6,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(41,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(44,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(53,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(56,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(65,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(68,6,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(29,9,5,'N',FOURPH[NY-3],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
10:  GETNUM(27,15,7,'N',HPCNUM[NY-3],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
11:  BEGIN
        CHICKENCONS[NY-3]:=FOURPH[NY-3]*HPCNUM[NY-3];
        GETNUM(28,20,6,'N',CHICKENCONS[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
12:  GETNUM(41,9,5,'N',FOURPH[NY-2],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
13:  GETNUM(39,15,7,'N',HPCNUM[NY-2],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
14:  BEGIN
```

```
        CHICKENCONS[NY-2]:=FOURPH[NY-2]*HPCNUM[NY-2];
        GETNUM(40,20,6,'N',CHICKENCONS[NY-2],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(53,9,5,'N',FOURPH[NY-1],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
16:  GETNUM(51,15,7,'N',HPCNUM[NY-1],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
17:  BEGIN
        CHICKENCONS[NY-1]:=FOURPH[NY-1]*HPCNUM[NY-1];
        GETNUM(52,20,6,'N',CHICKENCONS[NY-1],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
18:  GETNUM(65,9,5,'N',FOURPH[NY],
     '#.###',0.000,9.999,retSS,FALSE,15,0);
19:  GETNUM(63,15,7,'N',HPCNUM[NY],
     '###.###',0.000,999.999,retSS,FALSE,15,0);
20:  BEGIN
        CHICKENCONS[NY]:=FOURPH[NY]*HPCNUM[NY];
        GETNUM(64,20,6,'N',CHICKENCONS[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(74,9,5,'N',FOURPH_OR[NY],
     '#.###',0.000,9.999,retSS,retrieveSS,15,0);
22:  BEGIN
        IF FOURPH_OR[NY]<>0.0 THEN
        CHICKENCONS_OR[NY]:=FOURPH_OR[NY]*HPCNUM[NY]
        ELSE CHICKENCONS_OR[NY]:=0.0;
        GETNUM(73,20,6,'N',CHICKENCONS_OR[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE CHICKEN CONS PERHEAD }



OVERLAY PROCEDURE BROILER_CONS_PERHEAD; { B:BROCON }
BEGIN

  screenSS:=30; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('BROCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen BROCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(29,6,2,'N',YEAR[NY-3],
    '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(32,6,2,'N',YEARB[NY-3],
    '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(41,6,2,'N',YEAR[NY-2],
    '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(44,6,2,'N',YEARB[NY-2],
    '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(53,6,2,'N',YEAR[NY-1],
    '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(56,6,2,'N',YEARB[NY-1],
    '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(64,6,2,'N',YEAR[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(67,6,2,'N',YEARB[NY],
    '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(28,9,6,'N',FIVEPH[NY-3],
    '#.####',0.0000,9.9999,retSS,FALSE,15,0);
10:  GETNUM(26,15,8,'N',BRONUM[NY-3],
     '####.###',0.000,9999.999,retSS,FALSE,15,0);
11:  BEGIN
        BROILERCONS[NY-3]:=FIVEPH[NY-3]*BRONUM[NY-3];
        GETNUM(28,20,6,'N',BROILERCONS[NY-3],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
12:  GETNUM(40,9,6,'N',FIVEPH[NY-2],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
13:  GETNUM(38,15,8,'N',BRONUM[NY-2],
     '####.###',0.000,9999.999,retSS,FALSE,15,0);
14:  BEGIN
        BROILERCONS[NY-2]:=FIVEPH[NY-2]*BRONUM[NY-2];
        GETNUM(40,20,6,'N',BROILERCONS[NY-2],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(52,9,6,'N',FIVEPH[NY-1],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
16:  GETNUM(50,15,8,'N',BRONUM[NY-1],
     '####.###',0.000,9999.999,retSS,FALSE,15,0);
17:  BEGIN
        BROILERCONS[NY-1]:=FIVEPH[NY-1]*BRONUM[NY-1];
        GETNUM(52,20,6,'N',BROILERCONS[NY-1],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
18:  GETNUM(63,9,6,'N',FIVEPH[NY],
     '#.####',0.0000,9.9999,retSS,FALSE,15,0);
19:  GETNUM(61,15,8,'N',BRONUM[NY],
     '####.###',0.000,9999.999,retSS,FALSE,15,0);
```

```
20:  BEGIN
        BROILERCONS[NY]:=FIVEPH[NY]*BRONUM[NY];
        GETNUM(63,20,6,'N',BROILERCONS[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(73,9,6,'N',FIVEPH_OR[NY],
     '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
22:  BEGIN
        IF FIVEPH_OR[NY]<>0.0 THEN
        BROILERCONS_OR[NY]:=FIVEPH_OR[NY]*BRONUM[NY]
        ELSE BROILERCONS_OR[NY]:=0.0;
        GETNUM(73,20,6,'N',BROILERCONS_OR[NY],
        '###.##',0.00,999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE BROILER CONS PERHEAD }


OVERLAY PROCEDURE TURKEY_CONS_PERHEAD; { B:TURCON }
BEGIN

  screenSS:=31; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('TURCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen TURCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(31,6,2,'N',YEAR[NY-3],
    '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(34,6,2,'N',YEARB[NY-3],
    '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(42,6,2,'N',YEAR[NY-2],
    '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(45,6,2,'N',YEARB[NY-2],
    '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(53,6,2,'N',YEAR[NY-1],
    '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(56,6,2,'N',YEARB[NY-1],
       '##',0.0,99.0,retSS,FALSE,15,0);
   7:  GETNUM(64,6,2,'N',YEAR[NY],
       '##',0.0,99.0,retSS,FALSE,15,0);
   8:  GETNUM(67,6,2,'N',YEARB[NY],
       '##',0.0,99.0,retSS,FALSE,15,0);
   9:  GETNUM(30,9,6,'N',SIXPH[NY-3],
       '#.####',0.0000,9.9999,retSS,FALSE,15,0);
   10:  GETNUM(29,15,7,'N',TURNUM[NY-3],
       '###.###',0.000,999.999,retSS,FALSE,15,0);
   11:  BEGIN
          TURKEYCONS[NY-3]:=SIXPH[NY-3]*TURNUM[NY-3];
          GETNUM(30,20,6,'N',TURKEYCONS[NY-3],
          '###.##',0.00,999.99,retSS,FALSE,15,0);
       END;
   12:  GETNUM(41,9,6,'N',SIXPH[NY-2],
       '#.####',0.0000,9.9999,retSS,FALSE,15,0);
   13:  GETNUM(40,15,7,'N',TURNUM[NY-2],
       '###.###',0.000,999.999,retSS,FALSE,15,0);
   14:  BEGIN
          TURKEYCONS[NY-2]:=SIXPH[NY-2]*TURNUM[NY-2];
          GETNUM(41,20,6,'N',TURKEYCONS[NY-2],
          '###.##',0.00,999.99,retSS,FALSE,15,0);
       END;
   15:  GETNUM(52,9,6,'N',SIXPH[NY-1],
       '#.####',0.0000,9.9999,retSS,FALSE,15,0);
   16:  GETNUM(51,15,7,'N',TURNUM[NY-1],
       '###.###',0.000,999.999,retSS,FALSE,15,0);
   17:  BEGIN
          TURKEYCONS[NY-1]:=SIXPH[NY-1]*TURNUM[NY-1];
          GETNUM(52,20,6,'N',TURKEYCONS[NY-1],
          '###.##',0.00,999.99,retSS,FALSE,15,0);
       END;
   18:  GETNUM(63,9,6,'N',SIXPH[NY],
       '#.####',0.0000,9.9999,retSS,FALSE,15,0);
   19:  GETNUM(62,15,7,'N',TURNUM[NY],
       '###.###',0.000,999.999,retSS,FALSE,15,0);
   20:  BEGIN
          TURKEYCONS[NY]:=SIXPH[NY]*TURNUM[NY];
          GETNUM(63,20,6,'N',TURKEYCONS[NY],
          '###.##',0.00,999.99,retSS,FALSE,15,0);
       END;
   21:  GETNUM(73,9,6,'N',SIXPH_OR[NY],
       '#.####',0.0000,9.9999,retSS,retrieveSS,15,0);
   22:  BEGIN
          IF SIXPH_OR[NY]<>0.0 THEN
          TURKEYCONS_OR[NY]:=SIXPH_OR[NY]*TURNUM[NY]
          ELSE TURKEYCONS_OR[NY]:=0.0;
          GETNUM(73,20,6,'N',TURKEYCONS_OR[NY],
          '###.##',0.00,999.99,retSS,FALSE,15,0);
       END;
    END; { CASE }

    if varSS=screen_fieldSS then last_fieldSS:=TRUE;
    RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }
```

```
{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE TURKEY CONS PERHEAD }


OVERLAY PROCEDURE HOG_CONS_PERHEAD; { B:HOGCON }
BEGIN

  screenSS:=32; screen_fieldSS:=22; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('HOGCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen HOGCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(31,6,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(34,6,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(42,6,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(45,6,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(53,6,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(56,6,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(64,6,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(67,6,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(30,9,6,'N',SEVENPH[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
10:  GETNUM(30,15,6,'N',HOGNUM[NY-3],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
11:  BEGIN
        HOGCONS[NY-3]:=SEVENPH[NY-3]*HOGNUM[NY-3];
        GETNUM(29,20,7,'N',HOGCONS[NY-3],
        '####.##',0.00,9999.99,retSS,FALSE,15,0);
     END;
12:  GETNUM(41,9,6,'N',SEVENPH[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
13:  GETNUM(41,15,6,'N',HOGNUM[NY-2],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
```

```
14:  BEGIN
         HOGCONS[NY-2]:=SEVENPH[NY-2]*HOGNUM[NY-2];
         GETNUM(40,20,7,'N',HOGCONS[NY-2],
         '####.##',0.00,9999.99,retSS,FALSE,15,0);
     END;
15:  GETNUM(52,9,6,'N',SEVENPH[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
16:  GETNUM(52,15,6,'N',HOGNUM[NY-1],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
17:  BEGIN
         HOGCONS[NY-1]:=SEVENPH[NY-1]*HOGNUM[NY-1];
         GETNUM(51,20,7,'N',HOGCONS[NY-1],
         '####.##',0.00,9999.99,retSS,FALSE,15,0);
     END;
18:  GETNUM(63,9,6,'N',SEVENPH[NY],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
19:  GETNUM(63,15,6,'N',HOGNUM[NY],
     '##.###',0.000,99.999,retSS,FALSE,15,0);
20:  BEGIN
         HOGCONS[NY]:=SEVENPH[NY]*HOGNUM[NY];
         GETNUM(62,20,7,'N',HOGCONS[NY],
         '####.##',0.00,9999.99,retSS,FALSE,15,0);
     END;
21:  GETNUM(73,9,6,'N',SEVENPH_OR[NY],
     '##.###',0.000,99.999,retSS,retrieveSS,15,0);
22:  BEGIN
         IF SEVENPH_OR[NY]<>0.0 THEN
         HOGCONS_OR[NY]:=SEVENPH_OR[NY]*HOGNUM[NY]
         ELSE HOGCONS_OR[NY]:=0.0;
         GETNUM(72,20,7,'N',HOGCONS_OR[NY],
         '####.##',0.00,9999.99,retSS,FALSE,15,0);
     END;
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE HOG CONS PERHEAD }



OVERLAY PROCEDURE OTHER_UNALLOCATED_CONS; { B:OUNCON }
BEGIN

  screenSS:=33; screen_fieldSS:=13; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('OUNCON.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen OUNCON') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(29,8,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  GETNUM(32,8,2,'N',YEARB[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
3:  GETNUM(40,8,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  GETNUM(43,8,2,'N',YEARB[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
5:  GETNUM(51,8,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  GETNUM(54,8,2,'N',YEARB[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
7:  GETNUM(62,8,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  GETNUM(65,8,2,'N',YEARB[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
9:  GETNUM(27,14,7,'N',OTHERCONS[NY-3],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
10:  GETNUM(38,14,7,'N',OTHERCONS[NY-2],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
11:  GETNUM(49,14,7,'N',OTHERCONS[NY-1],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
12:  GETNUM(60,14,7,'N',OTHERCONS[NY],
     '####.#',0.0,9999.9,retSS,FALSE,15,0);
13:  GETNUM(71,14,7,'N',OTHER_UNALLOC_OR[NY],
     '####.#',0.0,9999.9,retSS,retrieveSS,15,0);
  END; { CASE }

  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE OTHER UNALLOCATED CONS }


OVERLAY PROCEDURE SCREEN_15; { THE CORN CONSUMPTION SUMMARY SCREEN }
BEGIN

  screenSS:=34; screen_fieldSS:=44; varSS:=1;
  retrieveSS:=FALSE; last_fieldSS:=FALSE;
  DISPLAY_SCREEN('CORNCONS.SCR',file_existSS);  { Display Screen }
  if not file_existSS then
  begin gotoxy(1,1); write('Missing Screen CORNCONS') end;
retSS:='';

{ Display Items. Change retrieveSS to TRUE and INPUT items}
REPEAT { until actionSS = exitSS }
  CASE varSS of
1:  GETNUM(36,4,2,'N',YEAR[NY-3],
     '##',0.0,99.0,retSS,FALSE,15,0);
2:  BEGIN
       YEARB[NY-3]:=YEAR[NY-3]+1;
       GETNUM(39,4,2,'N',YEARB[NY-3],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
3:  GETNUM(47,4,2,'N',YEAR[NY-2],
     '##',0.0,99.0,retSS,FALSE,15,0);
4:  BEGIN
       YEARB[NY-2]:=YEAR[NY-2]+1;
       GETNUM(50,4,2,'N',YEARB[NY-2],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
5:  GETNUM(58,4,2,'N',YEAR[NY-1],
     '##',0.0,99.0,retSS,FALSE,15,0);
6:  BEGIN
       YEARB[NY-1]:=YEAR[NY-1]+1;
       GETNUM(61,4,2,'N',YEARB[NY-1],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
7:  GETNUM(69,4,2,'N',YEAR[NY],
     '##',0.0,99.0,retSS,FALSE,15,0);
8:  BEGIN
       YEARB[NY]:=YEAR[NY]+1;
       GETNUM(72,4,2,'N',YEARB[NY],
       '##',0.0,99.0,retSS,FALSE,15,0);
    END;
9:  GETNUM(35,6,6,'N',DAIRYCONS[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
10:  GETNUM(34,8,7,'N',CATONFED[NY-3],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
11:  GETNUM(35,10,6,'N',OTHERBEEFCAT[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
12:  GETNUM(35,13,6,'N',CHICKENCONS[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
13:  GETNUM(35,15,6,'N',BROILERCONS[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
14:  GETNUM(35,17,6,'N',TURKEYCONS[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
15:  GETNUM(34,19,7,'N',HOGCONS[NY-3],
     '####.##',0.00,9999.99,retSS,FALSE,15,0);
16:  GETNUM(35,21,6,'N',OTHERCONS[NY-3],
     '###.##',0.00,999.99,retSS,FALSE,15,0);
17:  BEGIN
```

```
       FEEDT[NY-3]:=DAIRYCONS[NY-3]+CATONFED[NY-3]+OTHERBEEFCAT[NY-3]
                  +CHICKENCONS[NY-3]+BROILERCONS[NY-3]+TURKEYCONS[NY-3]
                  +HOGCONS[NY-3]+OTHERCONS[NY-3];
       GETNUM(34,23,7,'N',FEEDT[NY-3],
       '####.##',0.00,9999.99,retSS,FALSE,15,0);
    END;
18: GETNUM(46,6,6,'N',DAIRYCONS[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
19: GETNUM(45,8,7,'N',CATONFED[NY-2],
    '####.##',0.00,9999.99,retSS,FALSE,15,0);
20: GETNUM(46,10,6,'N',OTHERBEEFCAT[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
21: GETNUM(46,13,6,'N',CHICKENCONS[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
22: GETNUM(46,15,6,'N',BROILERCONS[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
23: GETNUM(46,17,6,'N',TURKEYCONS[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
24: GETNUM(45,19,7,'N',HOGCONS[NY-2],
    '####.##',0.00,9999.99,retSS,FALSE,15,0);
25: GETNUM(46,21,6,'N',OTHERCONS[NY-2],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
26: BEGIN
       FEEDT[NY-2]:=DAIRYCONS[NY-2]+CATONFED[NY-2]+OTHERBEEFCAT[NY-2]
                  +CHICKENCONS[NY-2]+BROILERCONS[NY-2]+TURKEYCONS[NY-2]
                  +HOGCONS[NY-2]+OTHERCONS[NY-2];
       GETNUM(45,23,7,'N',FEEDT[NY-2],
       '####.##',0.00,9999.99,retSS,FALSE,15,0);
    END;
27: GETNUM(57,6,6,'N',DAIRYCONS[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
28: GETNUM(56,8,7,'N',CATONFED[NY-1],
    '####.##',0.00,9999.99,retSS,FALSE,15,0);
29: GETNUM(57,10,6,'N',OTHERBEEFCAT[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
30: GETNUM(57,13,6,'N',CHICKENCONS[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
31: GETNUM(57,15,6,'N',BROILERCONS[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
32: GETNUM(57,17,6,'N',TURKEYCONS[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
33: GETNUM(56,19,7,'N',HOGCONS[NY-1],
    '####.##',0.00,9999.99,retSS,FALSE,15,0);
34: GETNUM(57,21,6,'N',OTHERCONS[NY-1],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
35: BEGIN
       FEEDT[NY-1]:=DAIRYCONS[NY-1]+CATONFED[NY-1]+OTHERBEEFCAT[NY-1]
       +CHICKENCONS[NY-1]+BROILERCONS[NY-1]+TURKEYCONS[NY-1]+HOGCONS[NY-1]
       +OTHERCONS[NY-1];
       GETNUM(56,23,7,'N',FEEDT[NY-1],
       '####.##',0.00,9999.99,retSS,FALSE,15,0);
    END;
36: begin
      if dairycons_OR[NY]<>0.0 then
      dairycons[NY]:=dairycons_OR[NY];
      GETNUM(68,6,6,'N',DAIRYCONS[NY],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
    end;
37: begin
      if catonfed_OR[NY]<>0.0 then
      catonfed[NY]:=catonfed_OR[NY];
      GETNUM(67,8,7,'N',CATONFED[NY],
    '####.##',0.00,9999.99,retSS,FALSE,15,0);
    end;
38: begin
      if otherbeefcat_OR[NY]<>0.0 then
      otherbeefcat[NY]:=otherbeefcat_OR[NY];
      GETNUM(68,10,6,'N',OTHERBEEFCAT[NY],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
    end;
39: begin
      if chickencons_OR[NY]<>0.0 then
      chickencons[NY]:=chickencons_OR[NY];
      GETNUM(68,13,6,'N',CHICKENCONS[NY],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
    end;
40: begin
      if broilercons_OR[NY]<>0.0 then
      broilercons[NY]:=broilercons_OR[NY];
      GETNUM(68,15,6,'N',BROILERCONS[NY],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
    end;
41: begin
      if turkeycons_OR[NY]<>0.0 then
      turkeycons[NY]:=turkeycons_OR[NY];
      GETNUM(68,17,6,'N',TURKEYCONS[NY],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
    end;
42: begin
      if hogcons_OR[NY]<>0.0 then
      hogcons[NY]:=hogcons_OR[NY];
      GETNUM(67,19,7,'N',HOGCONS[NY],
    '####.##',0.00,9999.99,retSS,FALSE,15,0);
    end;
43: begin
      if other_unalloc_OR[NY]<>0.0 then
      othercons[NY]:=other_unalloc_OR[NY];
      GETNUM(68,21,6,'N',OTHERCONS[NY],
    '###.##',0.00,999.99,retSS,FALSE,15,0);
    end;
44: BEGIN
       FEEDT[NY]:=DAIRYCONS[NY]+CATONFED[NY]+OTHERBEEFCAT[NY]+CHICKENCONS[NY]
                  +BROILERCONS[NY]+TURKEYCONS[NY]+HOGCONS[NY]+OTHERCONS[NY];
       GETNUM(67,23,7,'N',FEEDT[NY],
       '####.##',0.00,9999.99,retSS,FALSE,15,0);
    END;
  END; { CASE }
  if varSS=screen_fieldSS then last_fieldSS:=TRUE;
  RET_STATUS; { Check the code in "retSS". Set  "varSS" and "actionSS" }

{ Check to see whether to switch retrieveSS to true }
if last_fieldSS and (not retrieveSS) then
begin
  retrieveSS:=TRUE; last_fieldSS:=FALSE; actionSS:=staySS; varSS:=1;
end else
  last_fieldSS:=FALSE;

UNTIL actionSS=exitSS
END; { OVERLAY PROCEDURE SCREEN_15 }
        ***** End of Include file FEEDTOR.INC *****

BEGIN
REPEAT
   IF (EDIT=FALSE) AND (BACTIVE=FALSE) THEN
      BEGIN
           color(White,Black);
           ClrScr;
           FLASHUP('W=feedt,NOESC,NOMOVE');
           GOTOXY(1,25);
           READ(RESPONSE);
           CLREOL;
           FLASHUP('C=ALL');
           END;
           CASE (RESPONSE) OF
                '1': screen_15;                     { feed grain summary }
                '2': dairy_animal_cons_perhead;
                '3': cattle_on_feed_cons_perhead;
                '4': other_beef_cattle_cons_perhead;
                '5': chicken_cons_perhead;
                '6': broiler_cons_perhead;
                '7': turkey_cons_perhead;
                '8': hog_cons_perhead;
                '9': other_unallocated_cons;
                '0': ;
           END; { CASE }
      IF (response <> '0') THEN
         WHATMENU(BSCREEN,BACTIVE,UB);

UNTIL (RESPONSE = '0');

END;

VAR RESPONSE1:CHAR;
BEGIN { Main Test Program }
  vtypeSS:=SET_MONITOR_TYPE; { 2=Color, 3=Mono }
  screenSS:=0; use_screenSS:=1; BeepOnSS:=FALSE;

  initial;
  INIT_VAR_UTIL;                    { USED IN THE LOAD AND SAVE PROCEDURES }
  FLASHUP('X:L=TLPSCR');
  FLASHUP('L=TLPLIB');
  FLASHUP('S=TITLE');
  WHATMENU(BSCREEN,BACTIVE,UB);
REPEAT
      IF (EDIT=FALSE) AND (BACTIVE=FALSE)
      THEN BEGIN
           COLOR(WHITE,BLACK);
           CLRSCR;
           FLASHUP('W=MAINMENU,NOESC,NOMOVE');
           GOTOXY(1,25);
           READ(RESPONSE);
           CLREOL;
           FLASHUP('C=ALL');
           END;

           CASE (RESPONSE) OF
                '1':LOAD;
                '2':SAVE;
                '3':EXOGENOUS_VARIABLE_SUB_MENU;
                '4':BEGIN                { ADD_FACTOR }
                      UB:=2;
                      BACTIVE:=TRUE;
                      CASE (BSCREEN) OF
                         1: ADD_FACTOR_SCREEN_1;
                         2: ADD_FACTOR_SCREEN_2;
                      END;  { CASE BSCREEN OF }
                    END;
                '5':BEGIN                        { BALANCE SHEET }
                      UB:=3;
                      BACTIVE:=TRUE;
                      CASE (BSCREEN) OF
                         1: SCREEN_12;
                         2: SCREEN_13;
                         3: SCREEN_14;
                      END; { CASE BSCREEN OF }
                    END;
                '6':feed_grain_demand;           { DOMESTIC FEED DEMAND }
                '7':EXPORT_VARIABLE_SUB_MENU;    { EXPORT_DEMAND }
                '8':begin
                    REPEAT
                      IF (EDIT=FALSE) AND (BACTIVE=FALSE) THEN
                      BEGIN
                        COLOR(WHITE,BLACK);
                        CLRSCR;
                        FLASHUP('W=SOLUTION,NOESC,NOMOVE');
                        GOTOXY(1,25);
                        READ(RESPONSE1);
                        CLREOL;
                        FLASHUP('C=ALL');
                        END;
                        CASE (RESPONSE1) OF
                             '1': remain;
                             '2': BEGIN
                                    UB:=2;
                                    BACTIVE:=TRUE;
                                    CASE (BSCREEN) OF
                                       1: RANDOMIZE_YIELD_SCREEN_1;   {SOLVES FOR A SINGLE PRICE}
                                       2: DISPLAY_RANDOMIZE_SCREEN;   {SOLVES WITH RANDOM SHOCKS}
                                    END;  { CASE BSCREEN OF }
                                  END;
                             '3':;
                        end;  { CASE }
                      IF (RESPONSE1 <> '3') THEN
                      WHATMENU(BSCREEN,BACTIVE,UB);
                    UNTIL(RESPONSE1='3');
                    END;
                '9':EXITSCREEN;
           END;  {CASE}
      IF (NOT EXIT)
      THEN WHATMENU(BSCREEN,BACTIVE,UB);
UNTIL (EXIT=TRUE);
END. { PROGRAM FIRST.PAS }
```

ALPHABETIC SORT OF PROCEDURES

PROCEDURE NAME	LISTING FILE
ADD_FACTOR_SCREEN_1	HOTT.INC
ADD_FACTOR_SCREEN_2	HOTT.INC
BROILER_CONS_PERHEAD	FEEDTOR.INC
CALCULATIONS	CACULATE.INC
CATTLE_ON_FEED_CONS_PERHEAD	FEEDTOR.INC
CHICKEN_CONS_PERHEAD	FEEDTOR.INC
CHECKDISK	MENU.UTL
CONTINUE	MENU.UTL
DAIRY_ANIMAL_CONS_PERHEAD	FEEDTOR.INC
DISPLAY_RANDOMIZE_SCREEN	RANDOMIZE.INC
DIRECTORY	MENU.UTL
EXOGENOUS_VARIABLE_SUB_MENU	FIRST.PAS
EXPORTERS_COMPOSITES	FIRST.PAS
EXPORTERS_GDP_SCREEN_1	MAGIC4.INC
EXPORTERS_GDP_SCREEN_2	MAGIC4.INC
EXPORTERS_NET_EXPORTS_SCREEN	WETT.INC
EXPORTER_CROP_REVENUE	MAGIC5.INC
EXPORT_VARIABLE_SUB_MENU	FIRST.PAS
EXITSCREEN	MENU.UTL
FIND_PERCENTILES	RANDOMIZE.INC
FREESTOCK_PROC	CACULATE.INC
FREESTOCK_PROC	RECALC.INC
FSI_USE_SCREEN	MAGIC1.INC
FEED_GRAIN_DEMAND	FIRST.PAS
FLASHUP	MENU.UTL
FUNCTION_KEY_PRESSED	MENU.UTL
GETNUM	FIRST.PAS
HOG_CONS_PERHEAD	FEEDTOR.INC
HELPSCREEN	MENU.UTL
IMPORTERS_GDP_SCREEN_1	MAGIC3.INC
IMPORTERS_GDP_SCREEN_2	MAGIC3.INC
IMPORTERS_GDP_SCREEN_3	MAGIC3.INC
INIT_VAR_UTIL	MENU.UTL
IOCHECK	MENU.UTL
INITIAL	INITIAL.INC
LOAD	INTRO.INC
LINE25	MENU.UTL
LOADBEV	MENU.UTL
OTHER_BEEF_CATTLE_CONS_PERHEAD	FEEDTOR.INC
OTHER_UNALLOCATED_CONS	FEEDTOR.INC
PRESSKEY	MENU.UTL
PRINT	MENU.UTL
RANDOMIZE_YIELD_SCREEN_1	RANDOMIZE.INC
RET_STATUS	FIRST.PAS
RE_CALCULATE	RECALC.INC
REMAIN	REMAIN.INC
SAVE	INTRO.INC
SCREEN_1	MAGIC1.INC
SCREEN_12	BALSHEET.INC
SCREEN_13	BALSHEET.INC

ALPHABETIC SORT OF PROCEDURES (Cont'd)

PROCEDURE NAME	LISTING FILE
SCREEN_14	BALSHEET.INC
SCREEN_15	FEEDTOR.INC
SCREEN_2	MAGIC1.INC
SCREEN_3	MAGIC1.INC
SCREEN_3A	MAGIC1.INC
SCREEN_4	MAGIC1.INC
SOLUTION_SCREEN	SOLUTION.INC
SAVEBEV	MENU.UTL
TURKEY_CONS_PERHEAD	FEEDTOR.INC
US_COMPOSITES_SCREEN_1	MAGIC2.INC
US_COMPOSITES_SCREEN_2	MAGIC2.INC
US_COMPOSITES_SCREEN_3	MAGIC2.INC
US_COMPOSITES_VARIABLES	FIRST.PAS
US_EXPORTS_SCREEN	WETT.INC
WHATMENU	MENU.UTL
ZEBRAC	ZEBRAC.INC

BIBLIOGRAPHY

BIBLIOGRAPHY

Anderson, J.R., J.L. Dillon, and J.B. Hardaker. Agricultural Decision Analysis. Ames: Iowa State University Press, 1977.

Bentham, Jeremy. Introduction to the Principles of Morals and Legislation. Hafner, London, 1948.

Bernoulli, Daniel. "Exposition of a New Theory on the Measurement of Risk." Econometrica 22, January 1954.

Bhattacharyya, Gouri K., and Richard A. Johnson. Statistical Concepts and Methods. Whiley & Sons, New York, 1977.

Doll, John P. Production Economics: Theory With Applications. John Wiley & Sons, New York, 1978.

Eisner, Robert. Expectations, Plans, and Capital Expenditures: A Synthesis of Ex Post and Ex Ante Data. presented at the Conference on Expectations, Uncertainty, and Business Behavior, October 1955.

Feed Situation and Outlook Report. United States Department of Agriculture, August 1986.

Ferris, John N. Forecasting Corn Producers' Response to the 1986 Feed Grain Program. Presented at AAEA 1986.

------, John. "Probability Forecasts on U.S. Corn Prices." Ag Econ Staff Paper #82-4, Michigan State University, December 1983.

------, J. "Feed Grain, Wheat, and Oilseed Supply and Demand." MSU Staff Paper, October 1985.

Gadson, Kenneth E., J. Michael Price, and Larry E. Salathe, U.S. Department of Agriculture, Staff Report AGES820506, May 1982.

Halter, A. N. and R. Mason. "Utility Measurement for Those Who Need to Know," Western Journal Of Agricultural Economics, Vol. 3, No. 2, December 1978, p. 99-109.

BIBLIOGRAPHY (cont'd)

Heady, Earl O. An Econometric Simulation Model of the U.S. Farm Sector and its Policies and Food Exports. Vandenhoeck & Ruprecht, 1978.

Hertzmark, Donald R. "Economic Feasibility of Agricultural Alcohol Production Within a Biomass System." American Journal of Agricultural Economics, December 1980.

Hilker, Jim and Mary Schultz. '"PIK and ROLL" - Is It For You.' Agricultural Economics Staff Paper No. 86-90, November 1986.

Houck, J.P. and M.E. Ryan. "Supply Analysis for Corn in the United States: The Impact of Changing Government Programs." American Journal of Agriculture Economics, May 1972.

Houck, J.P., M.E. Ryan, and A. Subotnik. Soybeans and Their Products. University on Minnesota Press, Minneapolis, 1972.

Knight, Frank H. Risk, Uncertainty, and Profit. Houghton-Mifflin, 1921.

Marshall, Alfred. Principles of Economics. Macmillan & Co., London, 1920.

Mc Keon, John. "Farm Commodity Programs: Their Effect On Plantings of Feed Grains and Soybeans." doctoral dissertation, Michigan State University, 1974.

Nerlove, Marc. The Dynamics of Supply: Estimation of Farmers' Response To Price. The John Hopkins Press, 1958.

Nicholson, Walter. Microeconomic Theory: Basic Principles and Extensions. The Dryden Press, Chicago, 1985.

Pindyck, Robert S. and Daniel L. Rubinfeld. Econometric Models and Economic Forecasts, McGraw-Hill, New York, 1981.

Raiffa, Howard. Decision Analysis. Addison-Wesley Publishing Co., Reading, Mass., 1968

Robison, L.J., and P.J. Barry. The Competitive Firm's Response to Risk. Macmillian, New York, 1987.

BIBLIOGRAPHY (cont'd)

Ryan, W.J.L. Price Theory. Macmillan and Company Limited, London, 1958.

Schlaifer, Robert. Introduction to Statistics for Business Decisions. Mc Graw-Hill Book Co., New York, 1961.

Shagam, Shayle. "Specification and Structure of an International Agricultural Trade Model With Linkage Capabilities." master's thesis, Michigan State University, 1986.

Tomek, W.G. and K.L. Robinson. Agricultural Product Prices. Cornell University Press, Ithaca, NY, 1972.

Tweeten, L.G. and C.L. Quance. "Positivistic Measures of Aggregate Supply Elasticities: Some New Approaches." American Journal of Agricultural Economics, 1969.

Tweeten, L.G. Foundations of Farm Policy. University of Nebraska Press, Lincoln, 1970.

Wailes, E. "A Simulation Model of United States Agriculture." doctoral dissertation, Michigan State University, 1983.

Walker, O.L. "Teaching Decision Making Under Uncertainty in the Farm Management Curriculum." Proceedings of Farm Management Teaching Workshop, Michigan State University, East Lansing, Michigan, April 14-15 1977, New York: Agricultural Development Council, Inc.

Walker, O.L. and A.G. Nelson. Agricultural Research and Education Related to Decision-Making Under Uncertainty: An Interpretive Review of Literature. Oklahoma State University Agricultural Experiment Station Research Report P-747, March 1977.