U.S. Department of Transportation

Decision Tools for Transportation Infrastructure Reinvestment

User Guidelines for Microcomputer Decision Support System (DSS)

July 1988

T
58.62
.S36
1988a

Inquiries about the availability of the software described in this report can be directed to the Principal Investigator. The Decision Support System (DSS) software is also available through PC-TRANS, the microcomputer support component to the Kansas University Transportation Center's Technology Transfer Program. For further information on software availability through PC-TRANS, call (913) 864-5655 or write to PC-TRANS, Kansas University Transportation Center, 2011 Learned Hall, Lawrence, Kansas 66045. Their March 1990 catalog indicated a cost of $22.50 for DSS through that source.

Decision Tools for Transportation Infrastructure Reinvestment

User Guidelines for Microcomputer Decision Support System (DSS)

Final Report
July 1988

Prepared by
Joseph L. Schofer
Transportation Center
Northwestern University
Evanston, Illinois 60208

Prepared for
University Research Program
Office of the Secretary of Transportation
U.S. Department of Transporatation
Washington, D.C. 20590

Distributed in Cooperation with
Technology Sharing Program
U.S. Department of Transportation
Washington, D.C. 20590

DOT-T-89-16

T
58.62
.S36
1988a

19988

[illegible] 1995

EXECUTIVE SUMMARY
DECISION TOOLS FOR TRANSPORTATION INFRASTRUCTURE REINVESTMENT
USER GUIDELINES
DSS: MICROCOMPUTER DECISION SUPPORT SYSTEM

This report is intended to improve the quality of decisions about reinvestments, and modest new investments, in highway transportation infrastructure. Decisions of this type comprise the majority of planning actions taken in the field of public sector transportation management. Our aging highway system, faced with changing demands which result from demographic trends and market shifts, stresses the limited resources available for reinvestment. Thus, it is particularly important that economically sound reinvestment choices be made.

To improve the quality of such decisions, a microcomputer package has been developed to support the routine use of economic evaluation as a part of the highway reinvestment process. Observation of professional practice, and consultation with a technical advisory panel, suggested that the use of economic evaluation methods is not widespread in the highway transportation field. Among the factors limiting the application of such tools are time pressures on technical professionals, their lack of available data, and in some cases, lack of timely and relevant analysis skills.

To contribute to improving this situation, a user-oriented microcomputer software package was developed which can assist the highway professional in the performance of an economic evaluation of proposed reinvestment projects. The program package is screen oriented, that is, it presents a series of data input forms which invite the user to supply logical and generally available data needed to perform the economic analysis. Where data may not be available, the software supports judgmental estimation by immediately computing and displaying the quantitative implications of judgments, supporting and efficient judgment-results-evaluation-revision cycle.

The software package, known as DSS for Decision Support System, is highly structure to step the user through logical consideration of a broad set of project cost and benefit categories. The overall design philosophy calls for presenting the user with opportunities to consider each category. This is intended to support comprehensiveness in the analysis. Support of judgmental estimation helps break down the barrier presented by absence of objective data. Of course, if objective data are available, they can and should be used in the analysis.

The DSS system comprises five modules. The main program, MAINDSS, supports detailed economic analysis of project provider costs, including annualized and present worth costs by expenditure category, inflation-sensitive future cash flows,

fixed percentage "local" share of costs, and costs allocated to vehicle miles of travel by three vehicle types.

Benefits which are offered for consideration are vehicle travel time savings, fuel cost savings, and accident reduction savings. Each of these is estimated based on project descriptions and data on current operations and accident rates supplied by the user. Numerous opportunities are presented to recycle, review, and revise the data input and analysis. A benefit cost analysis, including benefit-cost ratio and net annualized and present worth computations, is performed.

Detailed hard copy reports, including all input data and analytic results, may be printed by a separate program, PRINTDSS. Project data may be saved to and retrieved from disk files.

A supporting program, INSTALL, may be used to install user-specified parameters, such as fleet average fuel consumption, values for travel time, and accident costs, to be used in every analysis.

Another supporting program, VIEW, may be used to review filed data and print brief reports without entering MAINDSS. A final supporting program uses dynamic programming to determine the optimal investment program from a series of proposed projects and a user-supplied budget level.

The package has been tested with a wide variety of technical professionals in training settings. It requires an IBM-PC, XT, or AT or compatible computer, one disk drive (fixed or floppy), DOS 2.10 or higher, and at least 256k bytes of random access memory.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

Many people provided important support in the development of this report and the software package it supports. We worked with a small technical advisory committee which provided guidance on the overall approach. On this committee were Mr. Ned Becker, Director of Transportation for Naperville, Illinois; Mr. Richard Hansen, Director of Public Works for Wilmette, Illinois; and Mr. David Jennings, Traffic Engineer for Evanston Illinois.

Professor Paul P. Jovanis provided thoughtful advice on the treatment of traffic operations and modeling fuel consumption.

Dr. Koji Tsunokawa developed the fuel consumption models along with Sompong Sirisoponsilp, who also wrote the program VIEW. Yen Jen Hung wrote the original version of the investment programming module, PROG.

This program package has been tested over several years in short courses offered by the Northwestern University Traffic Institute, under the management of Mr. Alex Sorton, who provided encouragement and feedback.

CHAPTER ONE
INTRODUCTION

Purpose of DSS

DSS stands for Decision Support System. The DSS software system was developed to encourage and support the use of more systematic and objective evaluation procedures in the planning and selection of highway improvement projects. Its particular focus is on highway reinvestment or rehabilitation projects of various sizes.

Through the integration of computer software and hardware, the DSS System is intended to make it easy, perhaps even pleasant, for the user to apply methods of economic evaluation, sometimes known as engineering economy, to the evaluation of proposed (not in-place) highway improvement projects. The system allows the user -- working interactively with the computer -- to describe a project, define its costs, specify its past (or existing) performance characteristics, and describe its expected performance when (and if) it is implemented.

Based on this user-provided information, the DSS system estimates benefits from various aspects of the proposed improvement, computes the economic worth of benefits and costs, and compares these to provide an overall economic evaluation. The system is rather highly structured, in that -- unlike a general analysis tool such as a spreadsheet routine -- it defines the structure of the analysis, including the items considered and the order in which they are treated. Some users may find this a bit constraining initially, but there is a reason for this. In the development process, we attempted to balance a number of important issues.

Issues in the Development of DSS

First, the development of the DSS system was guided by a desire to assure the reasonable use of economic evaluation tools in highway reinvestment planning. The nature of these tools imposes a useful structure on the analysis.

Second, we wanted to work with a reasonably comprehensive set of benefit types which would encompass the major economic outcomes of the most typical highway improvement projects. Some users will discover that the DSS system calls for examination of benefits which do not result from a particular project they are considering. DSS asks the user to look for such benefits, but allows each benefit category to be skipped if it is not appropriate. But as a user, you will always be asked to consider each benefit category. Consider this a reminder to be comprehensive.

In other cases, users will find that there is no room in the DSS structure to consider a benefit unique to a given project. Here, the user should (really, must) consider that benefit category outside the framework of the DSS system.

An important (and third) issue in the development of the DSS system was the effort to ensure usefulness of the system by requiring only, or at least primarily, input data which a typical user is likely to have available. As a consequence, in some cases the DSS system uses a simple way to estimate a benefit, rather than a complicated way, in order to make the users' task feasible.

We worked with a small advisory committee of transportation professionals regularly engaged in highway project evaluation to get a better sense of project types, user needs, and typical data availability. Thus, the simplifications made in program development were based on recent and typical experience and professional practice.

Fourth, we have developed a system which does not substitute for other, advanced forecasting models or computer programs. Thus, for example, the DSS system does not perform a capacity analysis, because we did not want to replicate the Highway Capacity Manual. Instead, it asks the user to estimate by whatever means (e.g., judgment, capacity analysis, etc.) certain traffic performance inputs.

In most cases where the user is asked for data which may well be supplied through judgment, the immediate consequences of those judgments or assumptions for the evaluation are displayed on the screen. Then an opportunity is provided for the user to try different values of the input parameters, the computational results of which are again shown. The intent of this style is to allow the user to test the sensitivity of judgments as they are made.

The DSS system which accompanies this manual is the result of compromises among the above, and other, factors. One of those other factors was the computer environment. The system was developed to function on a simply-configured IBM-PC or compatible system. This imposed some constraints, as did the choice of Microsoft Basic and Basic Compiler as software environments. While this form of the Basic language is straightforward and widely supported, in the version used here it is significantly constrained in terms of program length (i.e., < 64K bytes). This necessitated some major structural features of the DSS system. The compiler, the first and until recently the only version of this compiler, suffers from a number of quirks which influenced the structure and style of the DSS system.

These constraints were dealt with in program development; the software is effective and reliable, though some operating features reflecting these constraints remain. They should not, we think, make the system less useful.

Who Should Use the DSS System

The DSS system is intended to be used by professionals engaged in planning and evaluating proposed highway improvement projects. DSS is useless for before-after evaluation of improvements, and it should not be applied for this purpose. It is also suitable for use by technicians working with such professionals. No knowledge of computer programming is necessary, though it will be important for the user to be thoroughly familiar with his or her computer and its operating system.

Professionals <u>routinely</u> engaged in the analysis of highway improvement projects will be the most logical users, since the DSS system can make repeated evaluation efforts efficient and less tedious.

We think it is particularly important that the user have a reasonable knowledge of economic analysis methods. While the DSS system does this analysis <u>for the user</u>, appropriate application of the system requires intelligent application: using the system without fully understanding the methods and assumptions of economic evaluation is risky at best and dangerous at worst. This users' manual will document the procedures used in the DSS system, and it will provide some important warnings. It is not written to be a substitute for a textbook on economic evaluation.

Traveling at Your Own Risk: Some Initial Caveats

No computer program or system can make its users "smart" or protect them from all pitfalls.

The most important warning for users to understand is that, like any computer program, the DSS system is dumb: it will take any input data you give it (as long as it is within some reasonable mathematical range) and process it to produce outputs. It has no way to know if those inputs and outputs are nonsense. It is up to the user to guard against such errors.

The DSS system does not supply data. The user supplies the data. Thus, project attributes, the value of travel time, the costs of accidents, the level of improvement expected from projects, etc., all must come from the user. In some cases this will present a challenge, in that data will not always be readily available. But don´t expect the program to supply it for you.

Computer Requirements

The DSS system is written for IBM-PC and compatible computers. Delivered in compiled Basic, it requires no particular run-time support software other than your computer´s MS-DOS operating system. That is, with the DSS system alone, the user should be able to accomplish the procedures described in this manual. The operating system will be required to prepare (format) diskettes, to create a working version of the DSS system and to make diskettes ready for file storage.

The operating system must be equivalent to the IBM MS DOS 2.10 (or higher), since the DSS system uses certain features of that operating system, in particular, the use of configuration files (CONFIG.SYS).

The computer itself must have at least 256K bytes of random access memory available when the DSS system is called. It should have at least two disk storage devices, either two floppy disk drives or one floppy drive and one hard disk. When it is run on a dual floppy configuration, the main project evaluation routine, called MAINDSS, requires the use of a ramdisk, a virtual disk drive which the program creates in memory when it is booted (started from scratch), but not when it is called from the operating system. Thus, to use this program with a dual floppy system, you must reboot the machine, either by simultaneously pushing Ctrl-Alt-Del or by turning the machine (off and then) on.

The required ramdisk will be named as either drive C or D, depending upon which designation is available: if you have a hard disk (normally drive C), the DSS Ramdisk will become drive D. If you have no drive C, the DSS ramdisk takes that designation. You may run DSS on a computer with a fixed disk without rebooting <u>as long</u> as there are about 2500 bytes free on the active fixed disk directory. This will be used by MAINDSS in place of the ramdisk.

A dot-matrix printer is also required if you want printed reports of project evaluations. It can be an 80 or 132 column printer. The DSS system does not utilize graphics, and so a simple monochrome monitor is sufficient. It will also display text on a color monitor if that is the only screen output device.

DSS should function on IBM compatibles to the PC, XT, and AT, as well as 80286 and 80386 computers.

CHAPTER TWO
DSS SYSTEM STRUCTURE

DSS System Overview

The DSS system has 4 components, illustrated in Figure 1. The principal component, the project evaluation routine, is divided into two parts because of programming software memory limitations. These are called MAINDSS and PRINTDSS. MAINDSS is highly interactive: the user supplies data describing the project, its costs, and effects. The program computes various measures of the project, including economic worth of costs, cash flows by year, and benefit estimates. Each computation is done and presented to the user for review and recycling for changing inputs. A summary report is presented on the CRT screen.

The second part of the principal component, PRINTDSS, is entered only if the user wants to print a hard copy report on a project. PRINTDSS is hardly interactive at all: it is a printing "engine", and only allows the user to interrupt, restart, or terminate printing. When the print task is completed, control returns to the beginning of MAINDSS. The user will not be affected by this switch between MAINDSS and PRINTDSS, except for a slight time delay while the switch takes place.

Functionally, what happens is as follows: when the user selects the hard copy print option, all of the project data (supplied by the user and computed by the software) are written to a ramdisk file -- a small simulated diskette in the random access computer memory -- or to the active directory on your fixed disk. Then, PRINTDSS is called from the active disk drive

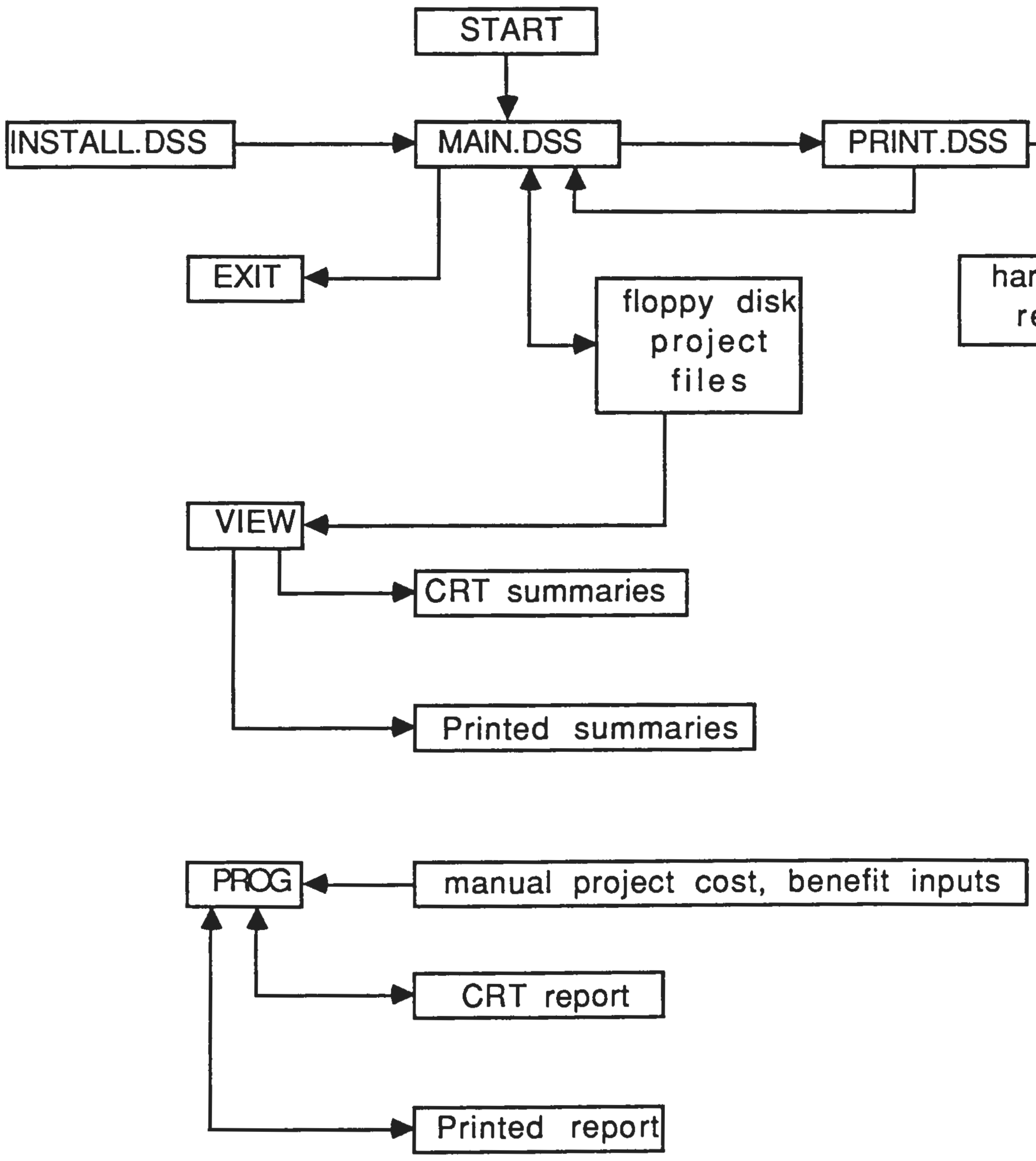

FIGURE 1: DSS SYSTEM COMPONENTS

(and directory), replacing MAINDSS in memory. PRINTDSS takes control, reads the file to be printed from the ramdisk (or fixed disk), and does the print job. Then MAINDSS is called from the active drive, replacing PRINTDSS in memory.

A program called VIEW has been written to allow the user to review project files saved by MAINDSS on floppy diskette (or fixed disk) files. Thus, without running MAIN.DSS, the user can invoke VIEW to look a project files, review a summary of results, and print hard copies of those summaries.

PROG is a program which takes data on <u>a series of non-exclusive projects</u> provided through keyboard input by the user, and <u>finds the optimal investment program for a given budget level</u>. PROG uses dynamic programming to do this, permitting users to select some projects as <u>mandatory</u>: that is, to choose some projects as required to be in the investment program.

PROG can accommodate up to 50 projects. However, the dynamic programming routine at its core is inherently slow, and so even in its machine language form, PROG will run quite slowly when the number of projects gets large.

The final program in the system is INSTALL.DSS, which allows the user to set default values of some key parameters used by the system. These include fuel consumption rates, accidents costs, and values for travel time. INSTALL.DSS is run once -- or occasionally -- by the user to establish default parameter values, which are written to a disk file in the default drive. This file is called STARTUP.DSS. Whenever MAIN.DSS is started, it looks for and tries to read STARTUP.DSS to get these parameter values. If it doesn't find STARTUP.DSS, it lets the user know and proceeds, using some built in default parameter values and starting with zeroes for others. The DSS system is shipped with a valid STARTUP.DSS file, but the user is urged to set his or her own default parameters before using the DSS system on a regular basis.

Program Flexibility

The program accommodates up to three pre-specified benefit types: travel time savings; fuel cost savings; and accident reduction cost savings. One, two, or all three benefit types can be examined for a particular project, although fuel cost savings can only be computed if time savings have been estimated. Benefits can be either positive or negative (disbenefits).

The DSS system analyses either of two improvement site types, intersections ("spot" locations) having zero length, or sections, with lengths greater than zero miles.

Three vehicle types are recognized and carried separately through the analysis. These are predefined as automobiles, light (single unit) trucks, and heavy (combination) trucks. The user may, by varying the data, modify these vehicle definitions, but the data labels may not be changed.

Time and fuel consumption benefits may be treated separately for peak and offpeak periods of the day. Cross street traffic is treated separately from mainline flows (for both intersections and sections) for these two benefits.

Accident reductions are disaggregated by severity class--fatal, injury, and property damage only accidents. They may be further disaggregated into two additional accident type classes (e.g. night and day accidents).

MAINDSS Program Flow

MAINDSS is made up of a series of program or procedure blocks. The program flows smoothly between these without user control, except that the user will have a series of branching options.

Entry routine: When it is started, MAINDSS allows the user to choose (1) to read an existing project file for further processing or modification or (2) to define and evaluate a new project. The user is given the option at this point only to save the new or revised project data on a disk file.

Project Description Entry: In this section of the program, the user enters the project name, description, life, opening year, and traffic data including average daily volumes, vehicle type mix, and projected traffic growth rate.

Project Cost Data Entry: Here the user enters project costs, giving the year incurred, the amount (in total dollars or dollars per mile), and the life of the investment by project component. Thus, costs for design, right-of-way, construction, maintenance and operations can be entered separately. Two "other" cost categories are available for additional items or to reflect, for example, construction costs incurred over several years. The user chooses a discount rate and the program computes present value and annualized costs.

Cash Flow Analysis: At the user's option, the program will compute cost cash flows by year over the project life based on a user-defined inflation rate. A local and nonlocal share cost breakdown may also be computed here.

Cost Allocation: In lieu of, or as a supplement to, a full-blown analysis of project benefits, the project costs may be allocated over the vehicle miles of travel (or entering vehicles for spot locations) to get a cost per vehicle mile by vehicle type. Particularly if the user is unwilling or unable to provide information to estimate benefits, the cost per vehicle mile offers a useful perspective on the worthiness of the proposed projects.

Benefits Analysis: If the user chooses to engage in an evaluation of benefits, the program allows treatment of peak period and nonpeak period traffic differently (because, for example, congestion would probably differ between these periods). At the start of a benefits analysis, the user specifies the fraction of daily traffic in the peak period. Cross street traffic is also entered at this point. Then the program calculates future traffic by time period and vehicle type over the project life, given the traffic growth rate or horizon year volume estimate.

Travel Time Savings: At the user's option, travel time benefits may be evaluated, based either on user-provided before-

and-after speeds or before-and-after travel times. Speeds may only be used if the section has a non-zero length (i.e., not for intersections). This may be accomplished by time period (peak/offpeak) and mainline and cross street flows will be treated separately for both intersections and longer sections.

Valuing Time Savings: At three points in the analysis (once for each major benefit type), MAINDSS asks the user to place monetary values on outcomes. In the travel time benefits section, the task is to put a money value on an hour of travel time for each of the three vehicle types. Default values for travel time can be entered in INSTALLDSS and then read from STARTUPDSS. These values can also be entered or modified at this point in the program.

Fuel Cost Savings: Fuel cost savings are strictly based on travel time and speed changes and thus the travel time benefits section must have been completed before proceeding with this section. Values for fleet average fuel economy and idling fuel consumption are required by vehicle type; these may come from INSTALLDSS via the STARTUPDSS file, from internal software defaults, or from user keyboard input at this point. One of the default/inputs is fuel price, so that the money value of fuel consumption changes can be estimated at this point.

Accident Reduction Benefits: Here the software allows the user to define up to two classes of accidents which will receive separate treatment in the benefits analysis. For example, suppose a rehabilitation project involves realignment and new street lighting. Night and day accidents might logically be treated differently -- the street lights might eliminate a share of night accidents but no day accidents. The number of classes and their names must be selected by the user at this point.

Then up to 3 years of accident data, the number of accidents by severity class, may be entered, after which the historical accident rates are computed and presented.

Valuing Accident Reductions: Money values are placed on accident reductions just as they were for travel savings -- from the default values and/or through direct entry. The total money value of accident savings due to the proposed project is then computed.

Summary Evaluation: The next program step computes and displays summary economic evaluation measures, including annualized and present values of costs, benefits by type, total benefits, and net project worths (benefits - costs). A benefit/cost ratio is computed and a simple sensitivity analysis is conducted.

Money values of benefits may be revised at this point, and thus the user can cycle between the summary evaluation and the benefit evaluation to explore the effects of different benefit values. Then the automatic filing routine writes project data to disk file if the user requested disk filing at the start of the run.

After the summary evaluation, the user has several options: (1) print hard copy report; (2) revise analysis of current project; (3) analyse a new project; or (4) quit. Printing causes a data file to be written to ramdisk as stated above, after which a printed report is produced. Revising cycles through the entire program, displaying previously entered data and permitting modifications from the keyboard. Analysing a new project cycles the program to its starting point and wipes out the data in memory describing the previous project.

Figure 2 summarizes this overall program flow and clarifies branch and recycle points.

Numerous, though not unlimited, opportunities are provided for the user to correct data inputs and to modify assumptions throughout the program. Data input correction opportunities are normally presented immediately after data entry, or -- sometimes -- after the associated and primary computational results have been computed and displayed.

Wholesale changes in data may be made by recycling through the analysis process, during which previously entered and/or computed data are displayed and the opportunity to make changes is presented.

The DSS system does not benefit from a full screen editor. This means that the user can only cycle back to selected and limited prior points when making corrections or modifications. This inflexibility is the price paid for reducing programming complexity. It has not proved to be a major limitation, for it is easy for the user to learn the limits beyond which corrections to inputs become more difficult. And, it is easy to recycle

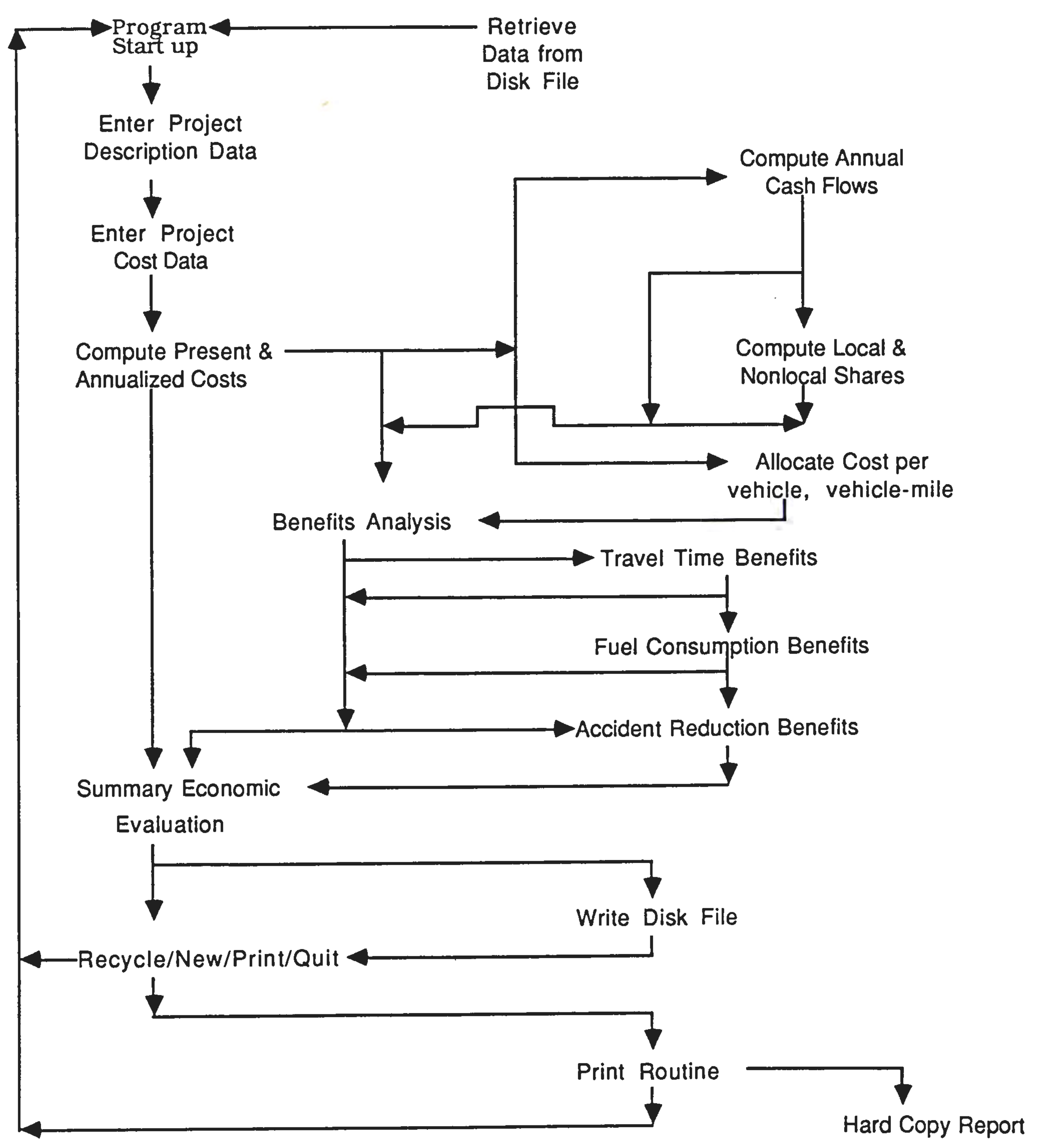
Program Start up
Retrieve Data from Disk File
Enter Project Description Data
Enter Project Cost Data
Compute Present & Annualized Costs
Compute Annual Cash Flows
Compute Local & Nonlocal Shares
Allocate Cost per vehicle, vehicle-mile
Benefits Analysis
Travel Time Benefits
Fuel Consumption Benefits
Accident Reduction Benefits
Summary Economic Evaluation
Write Disk File
Recycle/New/Print/Quit
Print Routine
Hard Copy Report

Figure 2: OVERALL FLOW OF MAINDSS

quickly through the entire program after the summary evaluation has been presented on the screen.

CHAPTER THREE
GETTING STARTED WITH DSS

Preparing System Diskettes

Floppy Disk Systems: The first thing the DSS user should do is to prepare two bootable DSS diskettes, each of which contains part of the DOS (2.10 or higher) operating system, one for the main program files and one for the capital budgeting routine, PROG. DSS is not shipped with the operating system on the diskette. Therefore, the user should prepare two appropriately formatted diskettes, as follows:

1. Enter DOS (e.g., by booting the computer with DOS in drive A); respond to the DATE and TIME prompts. You should see the Prompt "A>".
2. Type "FORMAT B:/S" <return>. The prompt should tell you to put a blank diskette in drive B and hit <return>. The blank diskette will be formatted with the bootable segment of the operating system on it.
3. Remove the newly formatted diskette and replace it with a blank one; format the second diskette as in step 2.

Now you must copy the DSS files from the delivery diskette to your formatted diskettes:

4. Put the DSS delivery diskette in drive A; put one of the newly formatted diskettes in drive B.
4. Type "T" <return>. This will copy the main program files on the delivery diskette (in A) onto the newly formatted system diskette (in B). The program will prompt you to remove the first diskette from drive B,

and replace it with the second blank, to which the remaining files will be copied.

Floppy diskette users may now <u>go on to the installation procedure</u> described below.

<u>Fixed Disk Systems</u>: You may install DSS on your fixed disk. First, be sure that you know how to use your fixed disk system. The instructions below are a bit less specific than those for floppy diskette systems, as fixed disk systems may differ.

1. With the computer on and the root directory (e.g., C:\) active, create a subdirectory for DSS: type "MD DSS" <return>. This creates a subdirectory called "DSS". (You may choose any valid directory name you wish).
2. Change the default directory to your new DSS directory: type "CD \DSS" <return>.
3. Put the DSS delivery diskette in the floppy drive (drive A) and type "COPY A:*.*." <return>. This will copy all of the DSS delivery file onto your fixed disk under subdirectory "DSS".
4. DSS is delivered with an autoexecution file, which will start DSS when you reboot your computer. Since you will not want this on your fixed disk, <u>with "DSS" as the default directory</u>, type "ERASE AUTOEXEC.BAT" <return> to delete this file from your fixed disk.

We suggest that you place the DSS delivery diskette in a safe place as your backup. Note that DSS is not copy protected. You may make additional copies using the procedure described above.

Installing DSS

You may now set the default values for key parameters which DSS uses. Each of these parameters will be automatically offered to the user with each pass through DSS. These parameters can be changed at run time, but it is more efficient to set as defaults the values you will most commonly use.

Make sure the DSS files are in the default drive. For a floppy drive installation, this (usually) means drive A. For a fixed disk system, make sure that the DSS subdirectory on the fixed disk is the active directory.

Then type "INSTALL" <return>. This loads and runs the installation program, which will prompt you for default parameter values. Enter each, in sequence, followed by <return>.

The parameters to be specified, along with their default values supplied in the file STARTUP.DSS, are:

- value of travel time for autos, $/hour ($2.50);
- value of travel time for light trucks, $/hour ($5.00);
- value of travel time for heavy trucks, $/hour ($7.50);
- cost per accident for fatal accidents, $/accident ($350,000);
- cost per accident for property damage accidents $/accident ($1,000);
- cost per accident for injury accidents, $/accident ($5,000);
- price of motor fuel, $/gallon ($1.00);

- fleet average running fuel consumption for autos, miles per gallon (16.30);
- fleet average running fuel consumption for light trucks, miles per gallon (12.90);
- fleet average running fuel consumption for heavy trucks, miles per gallon (5.70);
- fleet average idling fuel consumption for autos, gallons per hour (0.5)
- fleet average idling fuel consumption for light trucks, gallons per hour (0.70);
- fleet average idling fuel consumption for heavy trucks, gallons per hour (2.00).

NOTE: The user should provide values for these parameters which reflect local conditions and which are suitable for use over the typical project time frame. This would allow you to introduce projected improvements in fuel economy if you wish.

The vehicle types are merely accounting categories: while you cannot change the names of these types in the program, you can set the values to represent any 3-part classification you wish.

In the development process, we have assumed "autos" to mean all types of automobiles, "heavy trucks" to mean all tractor trailer combination trucks, and "light trucks" to mean all other single unit trucks. Diesel and gasoline powered trucks are lumped together because the fuel price differences for these types are now small.

Note also that the accident costs should be cost per accident, not cost per fatality or injury.

When you have completed entry of these defaults (revising your entries as necessary) confirm the entries in response to the prompt. When you have done this, the installation program will prepare and write to your default directory a file called "STARTUP.DSS."

Each time you load DSS, the program will look for this file and, upon finding it, will load and use your default parameter values. If the STARTUP.DSS file is not present, the program will so notify you, and will proceed with execution with built-in default parameter values.

You may change the default parameter values as often as you like. Before you start the installation procedure, be sure that there is no write protect tab on your DSS system diskette, since INSTALL will try to write to this file.

FLOPPY DISKETTE USERS SHOULD NOW REMOVE THE DSS SYSTEM DISKETTE FROM DRIVE "A" AND PLACE A WRITE PROTECTION TAB ON IT. This will become your standard DSS diskette.

Starting DSS

BEFORE YOU RUN DSS, READ THE DETAILED PROGRAM DESCRIPTION IN CHAPTER 4. Lack of understanding of program operating characteristics may produce misleading results and/or cause program failures.

But, for familiar users anxious to get going with DSS, we present here instructions for starting the program.

With the Computer Off: From a "cold start" on a floppy drive system, put the DSS system diskette in drive A and turn on the machine. The computer should boot, ask for date and time (which you should always provide since floppy files and reports are always date/time stamped), and then load and run DSS.

Floppy diskette users should have a formatted diskette in drive B with sufficient space on which to store project files. These files are small -- around 2,500 bytes or less per project. The AUTOEXEC.BAT file supplied on the DSS system delivery diskette will automatically run DSS.

On a fixed disk system, boot the computer and enter the date and time. Then change to the DSS subdirectory (with "CD\DSS" <return>, and type "DSS" <return>. This will load and run DSS.

You may wish to create a batch file for automatically running DSS on your fixed disk system. Suppose the DSS system is on your fixed disk in a subdirectory called DSS, and you wish to provide quick access to it from the root directory. Make the root directory active (CD\ <return>) and respond to the prompt with:

```
COPY  CON:DSS.BAT  <return>   {accept batch file DSS.BAT
     from keyboard}

CD\DSS <return>  {change active subdirectory to DSS}

DSS <return>  {run DSS}

CD\ <return>  {when DSS is finished, return to root
     directory}

F6 <return>   {end batch file and copy to active (root)
     directory}
```

To run DSS from the root directory, simply type DSS <return>.

As long as there is sufficient space available on the fixed disk, project files may be stored on this (default) drive. Alternatively, fixed disk users may also store and retrieve project files from the floppy drived; a formatted disk should be provided if this is to be done.

NOTE!!! For floppy disk users starting with the Computer on: To transfer files from the main DSS routine to the hard copy print routine, DSS uses a RAM disk which it creates when the program boots from a "cold start". Thus, floppy drive users who may want to print hard copy must start each session by (re-) booting the computer from the DSS system diskette in order to allow DSS to create the required RAM disk!!

With the computer on, this can be done by simultaneously pressing control, alternate, and delete (Ctrl-Alt-Del) with the DSS system diskette in drive A.

DO NOT START DSS BY TYPING "MAINDSS" <return> FROM A "HOT START" (computer already on) with a floppy system! If you do, the RAM drive will not be available, and the print routine will fail. (The program should not crash in this case. A special file will be written to your floppy drive called "ERROR.FIL" which will have in it all project attributes. You can later retrieve this like any other project file).

On fixed disk system, with the computer already on, simply change to the DSS subdirectory ("CD\DSS" <return> and call DSS

"MAIN.DSS" <return>. In this case, the hard copy print file will be stored temporarily on your fixed disk.

Other programs in the DSS system (i.e., VIEW, PROG) can be run by calling them from DOS. That is, with the operating system "up" (e.g., with an A> or C> prompt), make sure the DSS diskette or subdirectory is the default drive, and then type in the program name (e.g., "VIEW") followed by <return>. To run VIEW, the project files to be viewed should be in an available (Not necessarily default) drive or subdirectory.

Since floppy drive system users were instructed to put PRO on a separate diskette, that diskette must be in the active drive to run PROG.

CHAPTER FOUR
PROGRAM OPERATION AND USE

Entering Data and Commands

DSS has special requirements for data and command entry. Almost all program commands -- which are usually presented in inverse video -- such as branching instructions, call for single key entry. This means that when DSS asks you to select a branching option from a numbered list, for example, you must only enter the number without following it by <return>.

In this case, DSS knows how long the instruction string will be (one character), and so it knows when you have completed entering that string. Where DSS does not look for <return>, an unnecessary <return> will be stored in the keyboard buffer and WILL BE USED AT THE NEXT DATA OR COMMAND ENTRY POINT. Thus, the unnecessary <return> will provide an incorrect input item, and it may cause you to lose a control opportunity at the next branch point.

Use <returns> to follow ONLY data string or numerical entries, or file names. Do not follow branching instructions with <return>!

Character inputs, such as project of file names, should be entered from the keyboard, corrected as necessary using the delete (DEL) and cursor keys or destructive backspace key, and followed by <return>.

Numerical inputs, such as project length, costs, accident history, etc., can be entered using the number keys at the top of the keyboard or the numeric keypad. The latter is faster and easier, and thus we recommend that, when you start DSS, you push "NUM LOCK" to activate the numeric keypad, or use the dedicated numeric key pad if your keyboard has one.

To correct numerical data inputs (in DSS only, not in VIEW or PROG), hit the backslash key (\). This will erase the number you are entering (before you hit <return>) from both the keyboard buffer and the screen display. Then, you can re-enter the correct numerical data. NEITHER THE DELETE KEY (DEL) NOR THE DESTRUCTIVE BACKSPACE WILL WORK IN DSS.

The reason for this is that a numeric data input trap has been written into the program which screens out any non-numeric keyboard inputs (e.g., letters, dollar signs, periods, commas, etc.). If you press any non-numeric key when numeric data is expected, the computer will "beep" and ignore your input. Thus, it also ignores "DELETE" and destructive backspace. it recognizes "\" as a command to erase data and receive corrected values.

While this takes some experience to get used to, it saves an immense amount of grief in routine applications; without this feature, entering non-numeric inputs where numbers are expected would interrupt the program with an error message and destroy the data entry screen format.

At several points in DSS, if you enter incorrect instructions (e.g., branch option 4 when only 3 branches are available; or a file name which cannot be found), the program will cause the computer to "beep", an error message will be displayed in inverse video, and a short time delay will be imposed so you can read the message. Then the program continues by giving you another chance to enter the correct instructions.

The standard procedure in DSS is to provide the user with an on-screen "form" to fill in. Use only the space provided on this "form". Do not go beyond this space or the screen display will be broken up and subsequent data entry will become difficult.

Remember, DSS does not have a full screen editor. This means you should try to correct data entries before hitting <return>. You cannot bounce freely around the screen to change inputs. You will be provided with periodic opportunities to correct the immediately previous collection of entries which have been followed by <return>, but once you pass the inverse video "change inputs (Y,N)" response, the only way to make a change is to recycle the entire analysis of the project you are studying.

Incidently, you should respond to the prompt "change inputs (Y,N)" with single letters, y, Y, n, or N, not followed by <returns>.

The Startup Screen

When DSS is started, you will be presented with three options:

1) Retrieve Data from File
2) Define New Project
3) Quit

Implement the path you wish to follow by pressing the SINGLE NUMBER associated with it on the menu.

Retrieving a File: Retrieving an existing file means pulling back into DSS a project file which has already been written and saved by DSS. If you choose option 1, you will see a new screen which give you an opportunity to view an abbreviated directory of the files in drive A:, B:, or the active directory of drive C:. This is useful if you have forgotten the name of the project file you want to review.

NOTE: Data describing several projects, or perhaps different versions of the same project, can be stored in a single data file. Thus, when you run DSS, you could define a file called, for example, KNOXCO, containing all current projects in Knox County. If you want to retrieve one of those projects, you do so by retrieving KNOXCO.

If you know the name of the file to be retrieved, you can skip the drive directory option by hitting <return>.

Whether or not you have called for the directory, when retrieving a file DSS will next ask you to name the file to be pulled in.

If that file is in the default drive (and directory), you need only enter the complete file name, including extensions, if any. Thus, you can respond to the file name prompt with something like: KNOXCO, or if there is a filename extension, KNOXCO.DOT.

If the desired file is on a drive other than the default, you must enter that drive name, a colon, and then the complete file name. If you are running DSS from your fixed drive, directory

DSS, and you wish to retrieve a file on drive A:, you might type A:KNOXCO.DOT <return>.

If DSS cannot find the file you name, perhaps because you typed the wrong name, it will recycle the file retrieval screen.

Once the file is found, DSS will show you the project location, proposed action, and date of record creation for each project in that file, in sequence. In each case, you will be asked if this is the file you wish to work on. If it is, type "y" or "Y"; if it is not (and thus you want to continue to flip through the projects in the named file), type "n" or "N".

> NOTE: DSS recognizes responses to prompts which are either lower or upper case characters; you may use either type. DSS accepts <return> as equivalent to "n" or "N" for responses to branching prompts.

If DSS runs out of projects in a file before you find the one you want, it will again recycle the file retrieve screen.

Once you select the file you want, DSS will ask whether you wish to save the changes you will make in it. What it really means is, "do you want to resave this project data set-- whether or not you change anything in it." If you respond to the question with "y" or "Y", the project data record will be <u>appended</u> to the file you have retrieved. Its a good idea to edit the project location identifier when you get to it to indicate "version 2" or something else to remind you that this is different from the original project data record.

Defining a New Project: Select option 2 from the startup screen, if you wish to analyze a project not now on file.

The program will then ask if you want to file the data record from this project, and if so, under what drive and file name. If the file name you select already exists in the active (or selected) directory, the program will warn you and ask if you wish to use a different file name, write over the old file (thus destroying it), or append the record for this new project analysis to the existing file.

IF YOU FAIL TO ASK FOR THE PROJECT DATA TO BE SAVED TO DISK FILE AT THIS POINT, YOU LOSE THE OPPORTUNITY TO DO SO. The data will be written to disk automatically after you have entered all items and DSS has completed its analysis.

Quitting: If you elect to quit DSS at this point, you will return to the operating system and the DSS directory will be active. If you wish to quit anywhere in the middle of a DSS analysis run WITHOUT SAVING YOUR WORK, press ctrl-break (control-break).

Entering Project Description

Now you are ready to enter data about the proposed project. NOTE: If you are reviewing an "old" (filed) project, the procedures described from here on are the same as for a new project, EXCEPT that the old data will be presented on the screen, and you will be given the option of making revisions at each point. You can choose to revise an item by typing "y" or "Y" when given the choice. Or you can skip through parts or all of the project record without making changes. To answer "no" to a branching question you may type "n", "N", or <return>.

In the data entry process, you will see a full-screen form; the cursor will automatically bounce to the first (or next) data entry position. Enter the data item requested, being careful not to go beyond the space on the allotted form. Follow each data item with <return>.

NOTE: Branching instructions should be answered with single key entries, without <return>.

Correct alphanumeric entry errors with the destructive backspace or delete keys; the cursor keys will be functional. Correct numeric data entries by using the backslash (\) key, which will erase the entry and allow you to rekey the item.

Every few lines, you will get a branching opportunity to revise the last series of input items. If you want to make a revision, type "y" or "Y", select the number of the item to be changed, and the old entry will be erased. You may then enter the revised item.

Initial Project Description Panel: In the first panel you will asked to define the PROJECT LOCATION and the PROPOSED ACTION. Use clear descriptions to facilitate retrieving the right file at a later time.

The SECTION LENGTH in miles will be used to evaluate vehicle miles of travel and the costs and benefits related to it. If you enter a zero (or just hit <return>, which is read as a blank, the program assumes you are analyzing a spot location (intersection). The PROJECT LIFE is the time horizon, in years, over which you want to amortize (and thus evaluate) the proposed project.

AADT is the most recent average annual daily traffic on the link or the total entering volume at the intersection. This is used to estimate benefits. Next you are asked for the YEAR of the AADT. This is used to project AADT to the opening year of the project, as well as to future years for benefit estimation.

NOTE: ALL YEARS SHOULD BE ENTERED IN FULL. That is, do not use "88" when you mean 1988. You MUST enter the full label, 1988!!

You will then be asked for the YEAR the project OPENS. This is to accommodate cases where the opening year is different from the AADT year or the year you are doing the analysis. Traffic volume will be projected from the AADT year to the opening and future years using a constant annual percentage growth (or decline) rate. Costs incurred before the opening year will be treated as if they were incurred in the opening year.

In the next panel you must supply the percentage breakdown of traffic at the project site by each of three vehicle types, autos, light (single unit) trucks,and heavy trucks. Travel time and fuel cost savings will be computed separately for each vehicle type. The percentages for each vehicle types must add to 100%; if they do not, DSS will warn you and recycle to allow you to re-enter these percentages.

DSS forecasts traffic growth (or decline) over the project life based on a constant, annual percentage growth rate. You are asked to supply this growth (or decline rate, entered as a negative value), or to hit <return> which signals the program that you have a prediction of design year (end of the project life) traffic from some other source. You will be asked for this prediction.

In either case, DSS computes (either) the end-year volume or the constant annual percentage growth rate to give you the opportunity to review these figures and revise your forecasts at this time. If you have a constant annual percentage growth rate to begin with, you can use the DSS- computed end-year volume as a reasonableness check to be sure that the projected volume does not exceed the capacity of the facility.

From this point on in the operation of DSS, the last line on the left side of the screen will show the proposed action, the project location, the year the project opens, and the life of the investment. This serves as a reminder of what project you are analyzing and its key parameters.

> NOTE: The standard printed report provided by DSS will include ALL of your input data as well as ALL computed results. However, if at any point in your use of DSS you see a screen which you wish to preserve, you may press shift-PrtSc (shift-print screen) with a printer online and the screen image will be printed without otherwise affecting the operation of the program.

Provider Cost Data Input

The next screen allows you to enter the provider's costs for the project, that is, the cost to implement, maintain, and operate the improvement. These costs are requested in 7 categories to allow you to account for the fact that the effective lives of these components may be different. The cost categories are:

Design (or engineering) costs;

R-O-W (Right-of-Way) Costs;

Construction Costs;

Maintenance Costs; ⎤ These categories assumed to <u>recur</u> at
Operating Costs; ⎦ intervals equal to their service lives.

Other; you may enter any other costs here.

Other; you may enter any other costs here.

The input panel permits you to enter different years of occurrence, and different service lives, for each cost component. Any of these costs may be assigned a zero value by pressing <return> for FIRST YEAR PAID. DSS supports easy analysis of relatively complex patterns of project costs, allowing you flexibility in cost representation, and computing equivalent uniform annualized costs, and the present worth of costs, for each component, as well as for total project costs. This is the first major analysis product of DSS.

For each cost component, you must enter:

The YEAR it is FIRST PAID (a <return> means you have no cost in this category); As before, you must enter all four digits of the year, i.e., "1988," not "88."

LUMP SUM COST value (a <return> here means that you wish to enter unit costs per project mile, which brings you to the next item...

PER MILE COST (entered only if lump sum cost is zero (signified by <return>).

(If you enter per mile costs, DSS computes the lump sum value and writes it to the screen.)

LIFE of the component, which defines the period over which it is amortized.

If the construction cost is incurred in several parts over two or three years, you can indicate this by assigning the first part to CONSTRUCTION COST, and the second and third to the two OTHER cost categories.

All of the costs should be INCREMENTAL, that is, the cost over and above those costs incurred if the project is not implemented. The program accommodates both negative and positive costs, with negative costs signified by entering a "-" sign before the number itself. Negative costs do not make sense for the one-time costs associated with design, right-of-way, and construction; they may be appropriate for recurring costs if the project results in reductions in maintenance and/or operating costs. Such reductions would be represented by negative cost values.

Once you have entered these costs, and have had a chance to modify them, DSS asks for a discount rate, representing the time value of money. This is used in the program to convert all of the cost items to an annualized basis, amortizing each over its unique life. A default discount rate of 10% is built into the software, which you can change at this point.

When it has the discount rate, DSS computes the annualized and present values of each cost item, as well as total annualized and present worth costs, and writes them to the screen. You may then revise the discount rate if you wish.

For example, suppose we are evaluating a four mile-long project for which the $125,000 design cost was paid in 1986, prior to the analysis year (1988). Right-of-way cost is $225,000 per mile, and is assumed to have a fifty year life. The Initial construction cost of $850,000 is a lump sum value with a 20 year life. The project results in a <u>savings</u> in biennial maintenance cost of $4,000 per mile, beginning in 1990, the first year maintenance must be applied. Annual operating cost is $1,000 more than for the current facility. The final construction cost increment of $98,000 per mile is applied in 1989. In this case, the provider cost data input screen would be filled in by the user as shown in Figure 3.

If the user selected a discount rate of 8%, DSS would compute the equivalent uniform annualized costs and present worth costs and produce the results shown in Figure 4. Note that the right-of-way life extends beyond the project life, and as a result, DSS

ITEM NAME	(FIRST) YEAR PAID	LUMP SUM	PER MILE	LIFE YEARS
Design	1986	$125,000		20
R-O-W	1988		$225,000	50
Construction	1988	$850,000		20
Maintenance	1990		-$4,000	2
Operation	1989		$1,000	1
Other	1989		$98,000	20
Other				

FIGURE 3
EXAMPLE OF PROVIDER COST INPUT

ITEM NAME	(FIRST) YEAR PAID	LUMP SUM	PER MILE	LIFE YEARS	ANN. COST	PRES. COST
Design	1986	$125,000		20	$12,732	$125,000
R-O-W	1988	$900,000	$225,000	50	$73,569	$722,307
Construction	1988	$850,000		20	$86,574	$850,000
Maintenance	1990	-$16,000	-$4,000	2	-$8,126	-$79,784
Operation	1989	$4,000	$1,000	1	$4,320	$42,414
Other	1989	$392,000	$98,000	20	$39,926	$392,000
Other						
	TOTAL COST				$208,994	$2,051,938

FIGURE 4
EXAMPLE OF PROVIDER COST INPUT AND CALCULATED RESULTS

only charges the project for the share "consumed" during the project life ($722,307 vs. $850,000). The difference is the salvage value, the worth remaining in the right-of-way.

The ability of DSS to manipulate complex cost time streams should be useful to transportation professionals even if they are unable or unwilling to evaluate project benefits.

Financial Analyses

At this point you are presented with several branching options for looking at the financial (i.e., "real money" outlay) implications of the project. This information may be useful for agency financial planning. First, you may examine the implied cash flows associated with the project, that is, the year-by-year outlays associated with the recurring costs (maintenance and operations). DSS simply projects forward, for each year of project life, the annual expenditures. The user may enter a constant INFLATION RATE to see how these recurring costs may be affected by inflation.

DSS applies the inflation rate savings in recurring costs (i.e., negative costs) by reducing the savings as a function of inflation for each year of the project life.

Whether or not you ask for the cash flow analysis, DSS will then offer you the opportunity to see the "local share" of the total cost stream associated with the project. This might be useful in differentiating between, for example, federal and local project costs. The user must specify a single percentage value, which is used to split ALL project costs between the "local" agency and other sources of funds. DSS then produces the time stream of "local" costs over the project life.

The user may cycle between inflation rate and local share calculations, following branching instructions shown on the screen.

> NOTE: If you do not elect to evaluate cash flows or local share, this information will NOT be available in the report that DSS prints at the end of the project analysis.

Cost per Vehicle Mile Analysis

One way to assess the worth of a proposed project is to estimate how much it costs for each vehicle mile (or entering vehicle) using the facility over the project life. Even if you cannot estimate the true benefits, getting an idea of the incremental provider costs per user-trip may help decision makers to evaluate whether or not the project is worthwhile.

This analysis is an option presented to the user as a branching opportunity. If it is selected, the cost allocation screen is presented and the user must then choose weights for allocating project costs to vehicle types. These weights represent "passenger car equivalents," reflecting the share of capacity used by each vehicle type and thus the share of project costs to be allocated to each.

Default vehicle type weights are offered to the user, which may be changed at run time to any values you wish to test. Assigning equal weights to all vehicle types will, of course, simply allocate total project life cycle costs equally across all vehicle miles (or entering vehicles for intersection improvements).

User Benefit Analysis

The next portion of DSS supports the evaluation of user benefits, in terms of travel time savings, fuel cost savings, and accident reduction savings. You may skip the benefit analysis entirely, or elect to analyze all or any of the three benefit types by following the branching instructions presented.

DSS has the capability to treat peak period and off-peak traffic operations differently. The first screen in the user benefits analysis asks you to enter the percentage of all-day traffic that travels in all peak periods. This is important if congestion and therefore traffic flow characteristics are significantly different during these two periods.

If you wish to separate peak/off-peak flow characteristics, you must estimate the percentage of traffic moving in the peak periods. For example, if 10% of the ADT moves in the morning peak, and 8 % in the evening peak, you should enter 18% in the peak percentage field. You may treat all time periods equally by setting the peak period percentage to zero (you may simply hit <return> in this field).

DSS provides limited treatment to cross-street flows which may be affected by the improvement. For roadway section improvements, flows on ALL cross streets over the section are aggregated. At this point in the program, you must enter the total cross-street ADT flow in response to the prompt on the screen. For intersections, the cross-street value should be the flow on the minor street, a <u>subset</u> of the total entering volume you supplied earlier. The cross-street volume may be zero.

Now DSS is ready to evaluate each benefit category. You are offered the opportunity to analyze travel time savings, fuel cost savings, and accident cost savings in sequence. You may select each, or skip any of these categories.

<u>Travel Time Savings</u>: To estimate travel time savings, you must estimate and enter the before- and after-improvement per vehicle travel times savings directly or, for roadway section improvements, you may provide the before and after average travel speeds, which DSS will convert to time and time savings.

> NOTE: For spot locations (intersections), be sure that you choose the option to enter travel time savings directly! If you elect to enter speeds, DSS will set all travel time changes to zero no matter what before and after speeds you enter!

For cross-street flows, you must provide an estimate of the before and after per vehicle travel times. These two ways of getting at user travel time savings are provided to give the DSS user more flexibility in application of the analysis system.

In either case, you will be presented with a new screen, on which you must enter the CURRENT or "before improvement" SPEED (or travel time) and the EXPECTED or "after improvement" SPEED (or travel time) on the MAINLINE, defined as the section of interest in your analysis. This must be done for both peak and off-peak flow conditions if you have elected to treat these periods separately.

DSS then computes the difference in speed (or travel time), and, for sections of non-zero length where you have entered speeds, it computes the current, expected, and saved per vehicle travel times. You are then presented with an opportunity to re-estimate before and after speeds or times.

This iterative opportunity is intended to allow users with limited objective information to make reasonable judgments in the benefit estimation process. As should be evident throughout this program, we have tried to make it difficult to ignore cost and benefit categories by making it easy to make good guesses about input data, and to test those guesses by seeing their implications (almost) immediately.

After you have completed data entry for the mainline flow travel time savings, DSS presents you with a similar input screen for data about cross-street flow. Here, you MUST provide before and after cross street per vehicle travel times, since there is no valid measure of the section length on the cross-streets which would support converting speed estimates to time saved. The data entry and iteration opportunity are identical to that used in the mainline analysis.

Valuing Travel Time: One of the more problematic tasks in performing an economic evaluation of travel time savings benefits is assigning monetary values to time saved. DSS supports this process by allowing the user to test different values of travel time, in terms of dollars per vehicle hour by each of the three vehicle types used in its accounting scheme. When you select an hourly rate, the aggregate present worth of travel time savings is immediately displayed on the screen, so you may see the implications, and judgmentally test the reasonableness, of your estimates.

> NOTE: You MUST select values for per vehicle travel time using this screen if benefits in this category are to be evaluated. That is, even if you know you want to accept the default per vehicle travel time values, you MUST select each vehicle type and explicitly accept the unit travel time value for each. Otherwise, these time savings will all be valued at zero.

The screen presented for this process is perhaps the most complicated element of DSS, yet the user should find it easy to work with after only a little experience with the program. The initial appearance of this screen is shown in Figure 5. You should focus your attention on the options menu in the lower right corner of the screen. To see, modify, and/or accept the value of auto travel time, press "1"; this will implement the default hourly value of auto travel time (set in the installation process, or at $2.50/hour otherwise). This value will appear as the first entry in the AUTO column. Below it DSS will write the present worth of the total time stream of savings over the project life. To the right, under TOTAL, you will see the

```
      VALUE OF TRAVEL TIME AND TIME SAVING BENEFIT ESTIMATE

                        AUTOS  LIGHT TRUCKS  HEAVY TRUCKS   TOTAL

VALUE OF TIME
$ PER VEHICLE HOUR

TIME SAVING BENEFIT
PRESENT WORTH TOTAL $

    INSTRUCTIONS                           OPTIONS MENU

Push F9 to decrease value       1> change auto time value
Push F10 to increase value      2> change light truck time value
Push <return> to go to MENU =>  3> change heavy truck time value
                                4> accept all values
                                          SELECT =>
```

FIGURE 5
TRAVEL TIME VALUATION SCREEN

aggregate value of time savings benefits for all vehicle types.

If you wish to change the value of auto travel time, you may:

- Use function keys 9 or 10 to decrease (F9) or increase (F10) the value of auto travel time in decrements or increments of $0.25/hour. The effect of such changes on aggregate time savings benefits will be computed and shown immediately on the screen.
- You may also use the number pad on you keyboard to enter travel time value increments directly, followed by <return> (no dollar signs, please).

When you are ready to look at the value of travel time for light trucks, press <return> to go back to the menu in the lower right corner, and the press "2". The procedure will be identical to that followed for auto travel time value at this point.

Having treated light truck travel time value, press <return> to go back to the menu, and then press "3" to see and explore time savings values for heavy trucks.

If you wish to accept the default values of travel time for each vehicle type, simply cycle through the process by pressing:

1 to select auto

<return> to accept value

2 to select light truck

<return> to accept value

3 to select heavy truck

<return> to accept value

When you have completed the analysis of each vehicle type, (be sure you return to the menu and) press 4 to accept all travel time values. You will be able to get back to this panel at the end of the analysis session if you wish.

The "final" form of the time valuation screen will look something like Figure 6.

Fuel Cost Saving Benefits: The next analysis option allows the user to estimate the savings in fuel costs produced by the proposed improvement. FUEL COST SAVINGS ONLY OCCUR IF THERE IS A CHANGE IN OPERATING SPEED (or travel time). If you showed no speed changes in the previous benefit module, it is not possible to estimate fuel cost savings, and DSS will skip over this component.

The display panel presented if you elect to estimate fuel cost savings asks you to confirm or enter new values for fuel consumption rates. For each of the three vehicle types, DSS requires a fleet average fuel economy figure, in miles traveled for each gallon of fuel. It also requires an estimate of fleet average idling fuel consumption by vehicle type to compute fuel consumption of vehicles waiting to proceed in traffic.

DSS also will need an estimate of the average cost per gallon of fuel. This number should be selected to reflect current and expected prices, and to be a judgmental composite of diesel fuel and gasoline costs.

The underlying fuel consumption model (see appendix A) requires that you decide whether the flow conditions on the

```
      VALUE OF TRAVEL TIME AND TIME SAVING BENEFIT ESTIMATE

                        AUTOS  LIGHT TRUCKS  HEAVY TRUCKS   TOTAL

VALUE OF TIME           $2.500    $5.000      $7.500
$ PER VEHICLE HOUR

TIME SAVING BENEFIT
PRESENT WORTH TOTAL $4,689,408  $1,213,729  $662,034   $6,565,171

    INSTRUCTIONS                            OPTIONS MENU

Push F9 to decrease value        1> change auto time value
Push F10 to increase value       2> change light truck time value
Push <return> to go to MENU =>   3> change heavy truck time value
                                 4> accept all values
                                            SELECT =>
```

FIGURE 6
TRAVEL TIME VALUATION SCREEN WITH EXAMPLE RESULTS

mainline are classified as "interrupted" or "uninterrupted." Interrupted traffic flow is what we experience when there are relatively frequent stops or acceleration/deceleration cycles. CROSS STREET FLOW IS ALWAYS ASSUMED TO BE INTERRUPTED. You should generally assume that when running speeds are less than 30 miles per hour, the flow condition is interrupted.

Default values for these parameters are built into DSS based on fleet average figures for the early 1980s. Using the INSTALL program, the user may define new default values which are entered each time DSS is started. Alternatively, the user may modify each of these values at run time using the fuel consumption cost screen.

> NOTE: In selecting values for fuel consumption and price, as well as for user travel time in the previous DSS element, you must select single parameter values to reflect these factors over the life of the project. Therefore, you should think ahead to make a reasonable, judgmental estimate of what these values may be over the project life.

You may accept all fuel consumption parameter values DSS presents by pressing <return>; pressing any other key will allow you to enter new values FOR EACH parameter. These new values will be written to the right of the old ones. Actual fuel cost savings will not be calculated until you respond to the CHANGE VALUES? prompt with either "n" or <return>.

Accident Cost Reduction Benefits: The last optional benefit category treated within the DSS system is the savings due to reductions in accident rates. If you elect not to evaluate accident reduction benefits, DSS skips directly to a benefit-cost summary.

DSS disaggregates accidents in two dimensions. First, it AUTOMATICALLY treats fatal, injury, and property damage only (PDO) accidents separately, since the economic costs of these accident types are radically different. Second, DSS gives you the OPTION to separate all accidents into one or two MUTUALLY EXCLUSIVE subcategories defined by the user. This will give you some added control over the benefit estimation process, allowing you to represent the effect of an improvement which has a differential impact on two classes of accidents. By MUTUALLY EXCLUSIVE we mean that the sum of the accidents in these one or two categories must be less than or equal to the total number of accidents.

For example, the two categories might be day and night accidents, which would be useful where the improvement is expected to affect daytime and nighttime accidents differently (e.g., reflectorized markings, lighting enhancements, etc.). If the improvement included larger clear zones and/or break-away sign posts, it would be important to separate run-off-the-road accidents from all other categories.

DSS begins the accident reduction analysis by asking you to specify the number of years of accident history you wish to enter; this information is used to establish a before-improvement accident rate by severity class (fatal, injury, and PDO accidents). You may enter no more than three year´s worth of accident histories. It is advisable to gather a full three year history to be confident that the computed accident rates are representative of conditions on the roadway.

Next you will be asked to define the number of MUTUALLY EXCLUSIVE classes into which you want to further disaggregate the accident experience at the project location. If you do not wish to implement this disaggregation, you must respond to the prompt with <return>. If you wish to disaggregate to two classes, enter "1" or "2" as desired. Working with 1 subcategory will allow you to name the single accident class to which unique reduction factors are to be applied. Of course you must have accident history data for this subcategory.

The next screen presented by DSS asks you to enter the accident history one year at a time, for the number of years you selected. These should be entered starting with the earliest year. DSS sends the cursor in sequence to the fatal, injury, and PDO accident categories. In each, enter the number of accidents in that severity class for that year, followed by <return>. Then enter the historical average daily traffic (ADT), which is necessary for computing the accident rates. For spot (intersection) locations, enter the ENTERING VOLUME.

DSS will accept data for one year at a time and allow you to modify the inputs for each year before proceeding to the next year. If you disaggregated the accident history further, DSS will ask for the accident history breakdown in these categories, using the names you just supplied.

> NOTE: Since the special accident categories are defined to be mutually exclusive and a subset of total accidents, the sum of the accidents across all special categories must be less than or equal to the total accidents you entered for each severity class. If not, DSS will recycle this input step.

DSS then computes the accident rates from the entered data history by finding the average number of accidents per year for each class and dividing that number by the average ADT (times the section length to get vehicle miles for non-intersection cases). Rates for fatal accidents are in terms of accidents per hundred million vehicle miles of travel (VMT) or (for intersections) per hundred million entering vehicles. Rates for injury and PDO accidents are per million VMT (or entering vehicles).

These rates are presented to you at this point for your review; you may proceed to the next screen by pressing <return>. DSS assumes these rates will remain constant into the future in computing the no-project (null alternative) accident costs; these rates will be reduced by application of the user-supplied constant reduction factors to project with-project accident costs. If there were no accidents of a particular class in the accident history provided, the rate for that class will be assigned a zero value for projecting future accidents.

At this point you are asked to supply accident reduction factors, the percent of accidents by severity classes and types which you expect to eliminate through the proposed project. These values may be available from your agency, or you may extract them from the published literature.

If in doubt, DSS will support intelligent guessing by showing you the "after" accident rates which result from your assumed reduction factors. It is wise to check these "after" rates against typical or average rates for similar locations in your area. If the resulting "after" rates are below the typical or average rates for your area, they are probably too low.

If you wish to assign the same percentage accident reduction for all severity classes, this single value should be entered under the TOTAL column. If you enter zero or <return> here, the cursor will bounce to FATALS, INJURIES, and PDOs in sequence to allow you to enter different reduction factors for each severity class. DSS asks you to supply reduction factors for the 1 or 2 special categories of accidents, using the same procedure as described for total accidents.

When you have entered the required reduction factors, DSS will compute projected after (with project built) accident rates and show them to you below the measured before rates from the previous screen. Then you are presented with the opportunity to revise the accident rates.

NOTE: As the prompt indicates, you must hit <return>

after modifying reduction factors to signal DSS to recompute the new projected accident rates.

If you elected to disaggregate accidents by one or two additional classes, DSS will separate the accidents in these classes from the total accidents in this historical data, compute class-specific accident rates, and apply the special reduction factors for these classes FIRST; the reduction factors you entered in the row labeled TOTAL (applicable to all accident types) will then be applied to the accident rates based on accidents REMAINING after the accidents from the special classes have been eliminated. This precludes double-counting accident reduction benefits.

Valuing Accident Costs: The next DSS screen asks you to assign unit monetary costs to each accident severity class, fatals, injuries and PDOs. This is necessary for the economic evaluation procedure. While some users may not be comfortable in assigning money costs to injury and fatal accidents, it is clear that there are real, economic costs associated with these accident types. Assigning money values to these accidents is a way to account for the safety benefits associated with the proposed project.

Technical professionals and policy makers who chose to avoid placing money values on accident outcomes set implicit values on such accidents when they make decisions about safety project implementation. For example, if a project costing $500,000 is expected to eliminate 10 fatal accidents over its life, and a decision is made against its implementation, an implicit value of <u>less than</u> or equal to \$50,000 per fatal accident is being set; a decision NOT to implement the project implies that the resulting benefits are less than or equal to the costs:

$$\text{BENEFITS} \leq \text{COSTS}$$

$$10 \cdot V \leq \$500,000$$

where V = economic cost of a fatal accident.

Therefore:

$$V \leq \$50,000.$$

On the other hand, if a decision is made to implement the project, then the implicit value of a fatal accident is <u>greater than</u> or equal to \$50,000:

$$\text{BENEFITS} \geq \text{COSTS}$$

$$10 \cdot V \geq \$500,000$$

$$V \geq \$50,000.$$

We argue that it is more sensible to make the values behind these decision explicit. DSS allows the user to see the implications of choices for accident costs immediately in terms of the resulting aggregate monetary benefits. This supports rapid testing of different accident cost values.

Both the National Safety Council and the National Highway Traffic Safety Administration periodically publish updated estimates of accident costs be severity class (Ref).

NOTE: DSS assumes that the entered accident cost values are on a per accident basis. Thus, for example, DSS expects to get the cost PER FATAL ACCIDENT, not the cost per fatality.

The accident cost valuation screen looks and functions much like the travel time value screen. The user should select options from the menu in the lower right part of the screen to review and modify costs for fatal (option"1"), injury (option "2") and PDO (option "3") accidents.

Initially assumed accident costs are built into the default parameter file which can be modified by the user with the INSTALL program. At run time, accident costs can be changed by selecting the severity class, then either typing in new costs from the keyboard or decrementing (F9) or incrementing (F10) the values in $500 steps with the function keys.

NOTE: As in the case of travel time savings, the user must select each of the severity classes on this screen to assign a non-zero value to the cost of accidents in each.

When you are satisfied with the accident cost values, return to the options menu and press "4" to accept all values and proceed to the next screen.

Economic Evaluation Summary

Now DSS has all the information it needs to develop a summary economic evaluation, which is presented on the next screen. All of the input data you have provided has been used to:

- Compute the annualized and present worth of the time streams of provider costs;

- Project the future traffic using the project section based on the initial average daily traffic and the constant annual growth rate of demand;
- Compute the annualized and present worths of the time streams of user travel time savings, based on future traffic and the travel time or speed changes provided, valued at the rates specified for each vehicle type;
- Compute the annualized and present worths of the time streams of fuel cost savings for each vehicle type, based on future traffic and the travel speed changes provided, using the fuel consumption and price data supplied for each vehicle type;
- Compute the annualized and present worths of accident cost savings, based on traffic growth, historical accident rates, and expected accident reduction factors, valued at the accident cost rates supplied for each severity class;
- Compute the total benefits, the sum of the savings of user travel time, fuel costs, and accident costs, as well as the percentage of benefits in each of these categories (useful for judging if the order of magnitude of benefits from each source is reasonable);
- Compute the net annualized and present worth of the proposed project, the benefits less the costs;
- compute the benefit cost ratio for the proposed project, the benefits divided by the costs.

DSS also performs and reports the results of a simple sensitivity analysis for the project. It shows:

- By what percent the benefits must change, assuming that the costs are accurate as indicated, to bring the project to the breakeven point (benefits = costs);
- By what percent the costs must change, assuming that the benefits are accurate as indicated, to bring the project to the breakeven point (benefits = costs).

This information gives a measure of how "solid" the project is, given the inherent uncertainties in estimates of both benefits and costs. It tells us how close the project is to the breakeven point, net worth = 0 or B/C = 1.0.

For example, if the project has a positive net worth (B/C > 1.0), if the benefits need fall by only a small percentage (e.g., 10-15%) to bring the project back to the breakeven point, the margin of error in benefit estimation is such that the project may not be very attractive. On the other hand, if the benefits must drop, for example, by 40% to bring the benefit/cost ratio back to 1.0, the confidence we have in the worthiness of the project should be greater. This interpretation presumes that the costs are estimated correctly.

Similarly, if the costs must decrease by only a small percent to bring an apparently worthy project down to the breakeven point, uncertainties in estimation might suggest that the project is "shaky." If costs must go up by a large amount for the project to become unworthy (B/C < 1.0), our confidence in the worthiness of the proposal should be greater. This interpretation presumes that the benefit estimates are accurate.

DSS does the same analysis, with analogous interpretations, if the initial analysis shows the project to be unworthy (negative net worth, B/C < 1.0).

The net worth measure of project value is generally more useful and meaningful than the benefit/cost ratio because it shows in absolute terms what the societal "profit" or "loss" is expected to be as a result of the proposed project. While the magnitude of the net worth has meaning, the only useful piece of information in the benefit/cost ratio is whether it is greater than or less than 1.0.

The net worth criterion can also be used in investment programming analysis, where decisions are to be made about selecting groups of worthy projects to be implemented within a given program budget. A simple program for performing this kind of analysis is described in the next chapter of this manual.

Recycling the Analysis, Printing Reports, and Quitting

After the economic analysis summary is presented, DSS offers you an opportunity to modify the values assigned to any of the benefits. This might be useful if, upon review of the results, you felt that the unit values for travel time, fuel consumption rates or costs, and/or accident costs were inappropriately specified. If you choose to re-value any benefits, you will be presented with a menu from which to select which benefits you want to re-value. Selecting one of these recycles the program back to the appropriate benefit valuation screen.

When you have completed this phase, another menu appears, giving you the opportunity to print a hard copy report, to review and revise the project analysis you are now working on, to analyze a new project (which takes you back to the "top" of the program), or to quit.

If you elect to review the current project, DSS recycles to its starting input screen, this time showing you all of the values which you entered, and giving you an opportunity to revise some or all data input items. This offers a chance to correct any input errors which were not caught on the last pass through the program at the "change inputs?" branches.

If you decide to print a hard copy report at this point, selecting the appropriate command will write your project data into a temporary file in a ramdisk (for floppy operations) or onto the active directory of your hard disk. Then the program PRINTDSS will be called to replace MAINDSS in your computer memory.

A prompt will ask you to prepare the printer. Position the paper so that the print is at the very top (don´t leave a top margin), and press any key when you are ready to print. The DSS system will ask you to signal it to return to MAINDSS after printing by entering the letter "r".

NOTE: DON´T RETURN TO MAINDSS UNTIL THE PRINTING PROCESS

HAS BEEN COMPLETED, EVEN IF YOUR COMPUTER SYSTEM HAS A PRINT BUFFER.

When you return to MAINDSS, you will see the initial menu, which will permit you to define a new project or retrieve an old one, as before. If you filed the current project to disk, you may review and revise it at this time. This supports analyzing different versions of the same project, printing hard copy reports on each before proceeding to the revision.

DSS prints a four-page report, after which control is returned to MAINDSS. An example of the DSS report is shown in Figure 7.

```
                  PROJECT  EVALUATION  REPORT

DATE: 07-29-1986                      TIME: 12:02:54

PROJECT LOCATION: Wilmette Avenue, Greenbay to Ridge

PROPOSED ACTION: resurface and modernize lighting

==============================================================

                 PROJECT CHARACTERISTICS AND COSTS

LENGTH, MILES:   2.00    PROJECT LIFE, YRS: 15       1985 AADT:  18,500

OPENS IN: 1987   ANNUAL TRAFFIC GROWTH:   1.50%   END YEAR VOLUME:  23,828

                                               DISCOUNT RATE: 10.0%

COST ITEM       FIRST    ITEM     ACTUAL       ANNUALIZED      PRESENT
              YEAR PAID  LIFE     AMOUNT          WORTH         WORTH

DESIGN          1985      15     $85,000        $11,175       $85,000

R-O-W              0       0          $0             $0            $0

CONSTRUCTION    1986      15    $650,000        $85,458      $650,000

MAINTENANCE     1987       1     -$3,000        -$3,300      -$25,100

OPERATION       1987       1     -$4,000        -$4,400      -$33,467

OTHER              0       0          $0             $0            $0

OTHER              0       0          $0             $0            $0

TOTAL                                           $88,933      $676,433


                         CASH FLOW ANALYSIS
           INFLATION RATE:   3.5%                   'LOCAL' SHARE:   10.0%

   YEAR         INFLATED COST        'LOCAL' SHARE        OTHER SHARE

   1985              $85,000               $8,500            $76,500
   1986             $650,000              $65,000           $585,000
   1987              -$7,000                -$700            -$6,300
   1988              -$7,000                -$700            -$6,300
   1989              -$6,755                -$676            -$6,080
   1990              -$6,501                -$650            -$5,851
   1991              -$6,239                -$624            -$5,615
   1992              -$5,967                -$597            -$5,371
   1993              -$5,686                -$569            -$5,118
   1994              -$5,395                -$540            -$4,856
   1995              -$5,094                -$509            -$4,585
   1996              -$4,782                -$478            -$4,304
   1997              -$4,460                -$446            -$4,014
   1998              -$4,126                -$413            -$3,713
   1999              -$3,780                -$378            -$3,402
   2000              -$3,423                -$342            -$3,080
   2001              -$3,052                -$305            -$2,747
   2002              -$2,669                -$267            -$2,402

  TOTAL             $653,070              $65,307           $587,763
```

FIGURE 7: EXAMPLE DSS PRINTED REPORT

Wilmette Avenue, Greenbay to Ridge
resurface and modernize lighting

ECONOMIC COST ALLOCATION

VEHICLE TYPE	PERCENT OF TRAFFIC	COST WEIGHT (E.G., PCE'S)	WTD. COST PER VEHICLE-MILE
AUTOS	90.0%	1.0	$0.0026
LIGHT TRUCKS	8.0%	2.0	$0.0052
HEAVY TRUCKS	2.0%	3.5	$0.0090

BENEFIT ANALYSIS

PEAK PERIOD PERCENTAGE: 25.0%

CROSS STREET TRAFFIC: 4,000

TIME SAVINGS BENEFITS
MAINLINE TRAFFIC

	CURRENT TRAVEL TIMES	EXPECTED TRAVEL TIMES	TRAVEL TIME SAVINGS
PEAK	6.0	5.5	0.5
SPEED	20.0	22.0	
OFF PEAK	5.0	4.8	0.2
SPEED	24.0	25.0	
OVERALL	5.3	5.0	0.3

TIME SAVINGS BENEFITS
CROSS STREET TRAFFIC

	CURRENT TRAVEL TIMES	EXPECTED TRAVEL TIMES	TRAVEL TIME SAVINGS
PEAK	0.5	0.5	0.0
OFF PEAK	0.5	0.5	0.0
OVERALL	0.5	0.5	0.0

	AUTOS	LIGHT TRUCKS	HEAVY TRUCKS
VALUE OF TIME	$2.50	$5.00	$7.50
PRESENT WORTH OF TIME SAVING	$636,157	$113,095	$42,410

Wilmette Avenue, Greenbay to Ridge
resurface and modernize lighting

FUEL COST SAVINGS

	AUTOS	LIGHT TRUCKS	HEAVY TRUCKS
FLEET AVERAGE MILES/GALLON:	16.30	12.90	5.70
FLEET AVG IDLE CONSUMPTION,GAL/HR	0.50	0.70	2.00

FUEL PRICE: $1.30 FLOW CONDITIONS: UNINTERRUPTED, >30 MPH

PRESENT WORTH FUEL COST SAVINGS

AUTOS	LIGHT TRUCKS	HEAVY TRUCKS	TOTAL
$53,133	$13,085	$11,439	$77,658

ACCIDENT COST SAVINGS
ACCIDENT HISTORY

YEAR		FATAL ACCIDENTS	INJURY ACCIDENTS	PDO ACCIDENTS	AADT
1	TOTAL	0	4	8	18,000
	day	0	1	3	
	night	0	3	5	
2	TOTAL	0	6	7	18,500
	day	0	2	4	
	night	0	4	3	
3	TOTAL	0	9	11	18,500
	day	0	3	6	
	night	0	6	5	

PERCENT OF ACCIDENTS ELIMINATED

ACCIDENT TYPE	TOTAL	FATALS	INJURIES	PDOs
TOTAL	10.0	0.0	0.0	0.0
day	0.0	0.0	0.0	0.0
night	40.0	0.0	0.0	0.0

ACCIDENT RATES

	FATALS	INJURIES	PDOs
PRESENT RATE	0.000	1.420	1.943
PROJECTED RATE	0.000	0.928	1.399

[PER MILLION VEHICLE MILES. FATALS, PER 100 MILLION]

COST/ACCIDENT	$350,000	$5,000	$1,000
PRESENT WORTH OF ACCIDENT SAVING	$0	$291,257	$64,448

Wilmette Avenue, Greenbay to Ridge
resurface and modernize lighting

ECONOMIC EVALUATION SUMMARY

ITEM	ANNUALIZED WORTH	PRESENT WORTH	% TOTAL BENEFIT
COST	$88,933	$676,433	
TIME SAVINGS	$104,083	$791,662	64.6%
FUEL COST SAVINGS	$10,210	$77,658	6.3%
ACC. COST SAVINGS	$46,766	$355,706	29.0%
TOTAL BENEFIT	$161,059	$1,225,025	100.0%
NET WORTH	$72,125	$548,592	
BENEFIT/COST RATIO	1.81		

With costs as shown, benefits MAY DECREASE -44.78% to bring B/C down to 1.0.

With benefits as shown, costs MAY INCREASE 81.10% to bring B/C down to 1.0.

CHAPTER FIVE
SUPPORTING PROGRAMS

Introduction

Two supporting programs are supplied as a part of the DSS system. VIEW allows the user to examine computer files of projects already analyzed to review the results and print a short report describing each. PROG supports investment programming decisions by finding the optimal package of projects for a given budget. These programs are described in this chapter.

Suporting Program "View"

VIEW is intended to support search and review of project data files. It may useful for finding a particular project file and/or for examining the benefit-cost summary quickly, without entering and moving through the MAINDSS program. VIEW will also print a short report containing (only) the benefit-cost summary for a project.

To activate VIEW, with that program in the active directory, simply type VIEW <return>. A screen will be presented allowing you to see a shortened directory of drive A:, B:, or the active directory in Drive C:. You select which directory you wish by typing A, B, or C. If you do not wish to see a directory, type <return>.

To retrieve a file for viewing, type the correct and complete file name, including the drive name, e.g.:

```
A:KNOXCO.DOT <return>
```

If the file you want to view is in the active directory, you should not include the drive identifier, e.g.:

KNOXCO.DOT <return>

If the file you name is not found, a message will be printed on the screen and VIEW will recycle. When VIEW finds the file, a listing of all projects in it will be displayed, and VIEW will ask you to verify that this is the correct file. If it is not, enter "n" <return> and VIEW will again recycle.

When the desired file is found, view will ask you to select which project on that file you wish to review. Select that project by entering its sequence number, followed by <return>.

VIEW will then show you a one screen project summary, with its location, action, project opening year, project life, the average daily traffic "now" and at the end of the project life, the discount rate, and the present worth and annualized costs, benefits by source, benefit-cost ratio and net worths. NOTE: VIEW has no computational capability; it only permits you to examine and print out results of prior DSS analyses.

At the bottom of the project summary screen, you will also be presented with a menu offering the following options, to be chosen by their single number identifiers shown to the left of each option:

```
1 see next project (in this file)  2 see previous project
                                     (in this file)
3 print short project report       4 return  to  list (of
                                     all projects in this
                                     file)
5 quit VIEW
```

If you want to examine <u>other</u> files, chose option 4 and, when the list of projects in the current file reappears on the screen, respond to the prompt "is this the correct file" with "n", which will recycle VIEW to its starting point.

Figure 8 shows an example of the printed report from VIEW.

Supporting Program "PROG"

Background and Concept: PROG is the final component of the DSS system. It is a simple package which supports investment programming, the selection of a subset of proposed projects which produce the maximum net worth (benefits - costs) and are within a user-specified budget.

PROG uses dynamic programming, an operations research method which can be used to formulate and solve investment programming problems. The general problem structure used in PROG is as follows:

> MAXIMIZE: {total net worth of the package of chosen projects}
>
> SUBJECT TO:
>
> 1) Requirement that the total cost of the package of chosen projects cannot exceed a specified budget level; and
>
> 2) Requirements (if any) that certain projects on the candidate list must be included in the final program.

Maximizing total net worth of the chosen package is the objective function, a mathematical statement of what we should be trying to achieve in the investment programming process. The next two statements in the formulation are constraints or requirements that must be met by the best or optimal solution to this problem.

PROJECT EVALUATION SUMMARY REPORT

DATE : 06-24-1988

PROJECT LOCATION : Wilmette Avenue at Greenbay Road

PROPOSED ACTION : Channelization, signals, improved lighting

LENGTH, MILES : 1.00

OPENING YEAR : 1989 **PROJECT LIFE, YRS :** 25

AADT : 27500 **FUTURE AADT :** 35976

DISCOUNT RATE : 8.0 %

	PRESENT VALUE	**ANNUALIZED VALUE**
COSTS		
DESIGN	$45,000	$4,216
R-O-W	$196,332	$18,392
CONSTRUCTION	$125,000	$11,710
MAINTENANCE	-$8,979	-$841
OPERATION	$13,835	$1,296
OTHER	$375,000	$35,130
OTHER	$0	$0
TOTAL	$746,188	$69,902
BENEFITS		
TIME SAVINGS	$865,235	$81,054
FUEL SAVINGS	$212,952	$19,949
ACCIDENT REDUCTION	$1,057,528	$99,068
TOTAL	$2,135,715	$200,071
NET WORTH	$1,389,528	$130,169

BENEFIT/COST RATIO 2.86

FIGURE 8: Example VIEW Printed Report

The first constraint is called the budget constraint, and it demands that the chosen package of projects be affordable, i.e., that in total their cost cannot exceed the available funds. The second constraint demands that certain, user-specified projects be included in the optimal collection of projects.

This might be used, for example, where a firm commitment has already been made to include a particular project. It might also be helpful in testing the impact on the overall program of including a specific project. This constraint is optional, that is, the user need not require that any particular project(s) be in the final solution.

In general, one should expect that requiring the inclusion of a particular project will reduce the value of the objective function, that is, it will give a poorer solution in terms of net worth. This follows the principle that the more one constrains an optimization problem, the more one restricts the value of the objective function at optimality.

Of course if a required project would have been in the optimal package found for the unconstrained problem anyway, requiring that project will not result in a lower (inferior) objective function.

Dynamic programming is a systematic, iterative approach to finding the optimal solution to problems of this type. Over a (potentially long) series of steps, the procedure adds candidate projects to the chosen set until it is no longer possible to add or substitute projects and improve the level of the objective function while meeting (both of) the constraints.

This is a many-step process which can take even the computer a long time. For a large set of candidate projects (i.e., more than 20) the solution times can be very long, measured in 10s of minutes. PROG is intended to support the development of relatively small investment programs.

NOTE: PROG IS LIMITED TO ANALYZING 50 OR FEWER PROJECTS! It should not be used for developing a statewide program that may have hundreds of candidate projects. This is a task which should be done on a mainframe computer, or with more specialized microcomputer software that takes advantage of math co-processor chips and uses a more efficient solution algorithm.

Using Prog: PROG requires that you enter project data from the keyboard; it does not extract data directly from files created in DSS. It does create and read its on files.

The program is run [from floppy diskette by inserting the diskette with PROG on it - the second bootable diskette you made -], making the drive or directory in which PROG resides the active directory, and typing PROG <return>. You will quickly see a menu of options: retrieve from disk file (created by PROG), define a new set of candidate projects, and quit. Choose your desired action by typing its number.

File retrieval works as it does in VIEW. You may call for a file directory of one of the floppy drives or the active fixed disk directory by typing the drive letter. Hitting <return>

skips the directory and asks you to enter the complete file name (including drive identifier if it is not on the active drive/directory and extension), followed by <return>. When the file is found, PROG shows you its contents and asks if you want to change them. It then proceeds through its routine analysis process.

PROG will not store multiple project lists under the same file name; you must pick a new name for each file you want to save.

When you retrieve data from a file, PROG will step you through the analysis, showing you the filed data and presenting you with opportunities to revise the inputs at each step.

When entering a new programming problem, PROG will first ask how many candidate projects are to be considered. This is the total number of projects on the list, from which a subset will be selected for the optimal program.

> NOTE: Once entered, you cannot change the number of candidate projects unless you define a new set of projects. If you think you might want to add a few projects to your initial list, enter them with zero costs and benefits. You can change these later.

Then, PROG will ask for the following data on each project in sequence: project name, costs, and benefits.

Costs and benefits may either be in terms of equivalent, uniform, annualized amounts or present worths, BUT BOTH MUST BE IN THE SAME TERMS. Furthermore, the total budget, which PROG asks for after all of the project data have been entered, MUST BE IN THE SAME TERMS AS PROJECT COSTS. Thus, if project costs (and benefits) are in terms of present worth, then the present worth (actual current dollar amount) of the budget must be used. If project costs (and benefits) are annualized values, then you must use an equivalent uniform annualized budget.

You can annualized a current dollar budget with the following formula:

$$\text{Annualized Budget} = \text{Present Worth Budget} \bullet \left[\frac{i\,(1+i)^n}{(1+i)^n - 1} \right]$$

After you enter the budget, PROG asks how many of the projects are required (<u>must</u> be included in the final program). You may select zero (or hit <return>) if none are required. If some must be included, you must first type how many, and then enter their numbers, followed by <return>.

You can cycle through PROG repeatedly testing the affect of requiring different projects. However, each type you revise the current set of inputs, you must tell PROG how many and which projects are required. It does not store this information, nor does it record required projects to disk files.

When data for all projects and the budget have been entered, PROG will display all of these inputs and give you an opportunity to revise any or all of them. It will ask if you want to change project data or the budget. If you choose to modify projects, it will ask which number, and then permit you to re-enter all data for that project. Then, after the modified input data have been echoed to the screen, you will be presented with another opportunity for revisions.

When you choose not to make any (more) revisions (by responding with "n", "N", or <return>, the solution process will begin. PROG will print to the screen some aspects of the intermediate solutions it considers while proceeding toward the optimal package. This will be done "on the fly," and with a small number of candidate projects, you will not be able to read the intermediate results because they will scroll by too quickly. As the number of candidate projects gets larger, the solution process gets more cumbersome and slows down. You will be able to read intermediate results, and for larger problems they will stay on the screen for quite a while. This display serves to confirm that the program is working toward the solution.

When it has found the optimal solution, PROG "beeps" and displays a table containing the chosen candidate projects (alternatives), the remaining alternatives, the costs, benefits, net worths, for each project, for the chosen package, and the remaining package.

At this point you can print a hard copy report using the current data. Whether or not you print this report, PROG will offer you the opportunity to revise the inputs. When you decline this opportunity, PROG asks if you want to file the CURRENT INPUT DATA to disk. If you say no, the program cycles back to its beginning point, at which you may quit, retrieve data from a file, or define a new program.

If you elect to file your data, you will be asked for a complete file name specification, e.g., A:KNOXCO.DOT <return>. If the data are to be filed in the active directory, you need not enter the drive identifier. As stated, each data set must be stored in a file with a unique name, one data set to each file.

While PROG is not fast enough to solve large problems, it should be very handy for finding optimal investment packages from smaller lists of candidates. Such cases would be quite burdensome to analyze by hand.

Its major advantage lies in its ability to cycle through optimization, revision, and testing the effects of "requiring" inclusion of different groups of projects.

Example Investment Programming Problem: Suppose we have the following set of projects, from which we wish to select the optimal investment program:

Project #	Project Name	PW Cost	PW Benefit
1	First Ave.	$125,000	$152,000
2	55th St.	$132,500	$195,000
3	Jones Bridge	$495,000	$758,000
4	Dann Ramp	$335,000	$405,000
5	Carr Road	$129,500	$151,000
6	Wesley St.	$225,000	$325,000

Suppose we have a budget of $950,000. Using PROG to solve this problem in its current form, we get the results shown in Figure 9. On a 4.77 mHz IBM XT, this solution required about 6 seconds.

```
                    INVESTMENT   PROGRAM
                       BUDGET =      $950,000

I.ALTERNATIVES CHOSEN

Project Name         Cost              Benefit             Net Worth

    55th Stree    $132,500            $195,000             $62,500
    Jones Brid    $495,000            $758,000            $263,000
    Wesley St.    $225,000            $325,000            $100,000
------------------------------------------------------------------------
    Total         $852,500          $1,278,000            $425,500

II.OTHER ALTERNATIVES

Project Name         Cost              Benefit             Net Worth

    First Ave.    $125,000            $152,000             $27,000
    Dann Ramp     $335,000            $405,000             $70,000
    Carr Road     $129,500            $151,000             $21,500
------------------------------------------------------------------------
    Total         $589,500            $708,000            $118,500
```

Figure 9: Example PROG Printed Report

```
                    INVESTMENT   PROGRAM
                       BUDGET =      $950,000

I.ALTERNATIVES CHOSEN

Project Name         Cost              Benefit             Net Worth

    Jones Brid    $495,000            $758,000            $263,000
    Wesley St.    $225,000            $325,000            $100,000
    First Ave.    $125,000            $152,000             $27,000
------------------------------------------------------------------------
    Total         $845,000          $1,235,000            $390,000

II.OTHER ALTERNATIVES

Project Name         Cost              Benefit             Net Worth

    55th Stree    $132,500            $195,000             $62,500
    Dann Ramp     $335,000            $405,000             $70,000
    Carr Road     $129,500            $151,000             $21,500
------------------------------------------------------------------------
    Total         $597,000            $751,000            $154,000
```

FIGURE 10: Example PROG Printed Report, Modified Problem

If we <ins>require</ins> that project 1 (First Avenue) be in the solution, we get the results shown in Figure 10. The net worth of the optimal package has declined by $35,500 because project 1 was required. This solution required approximately 3 seconds.

APPENDIX A
COMPUTATION OF FUEL CONSUMPTION SAVINGS

Fuel consumption savings are computed separately for (1) spot locations (intersections) and cross street traffic and (2) sections of non-zero length. Computations are further separated by each of three vehicle types and by peak and off-peak flows.

In all cases, the difference in the rate of fuel consumption (without project - with project) is multiplied by the total traffic (by time period and vehicle type) expected over the project during its life, and then by the fuel price. In actual computations, the discounted present worth of this money value is found by projecting year-by-year traffic, and discounting the traffic volume back by multiplying the traffic in each year by:

$$\frac{1}{(1+i)^n}$$

where

i = discount rate

n = project life in years.

This has the same effect as computing the savings in each year and discounting it back to present worth.

The effect of the peak/off-peak splits and percentage of traffic by vehicle type are handled by separating traffic flows into time periods and vehicle classes using simple percentage multipliers.

Spot locations and cross traffic are treated by assuming that changes in fuel consumption come from changes in the amount of engine idling time. If a vehicle spends 0.5 minutes less going through an intersection, it is assumed to consume fuel at the idling rate for 0.5 minutes less.

For non-zero length sections on the mainline portion of a project, fuel consumption models by vehicle type were derived from work done at Texas A & M University and General Motors Corporation. Models were estimated separately for interrupted (<30 m.p.h.) and uninterrupted flows. Uninterrupted flow fuel savings models are shown in Table 1; interrupted flow fuel savings models are shown in Table 2.

The models shown in these tables are the results of regressing data points reported in the sources cited above in order to develop the simplest, combined models possible.

TABLE 1
UNINTERRUPTED FUEL COST SAVINGS MODELS

Automobile Model

$$\delta \text{ Fuel Cost} = \frac{16.3}{MPG(A)} \left[497 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] + 0.0139 \cdot \left[VO(J)^{2} - VN(J)^{2} \right] \right] \cdot$$

$$PCT(Auto) \cdot PPCT(J) \cdot PWTFC \cdot GASPRICE \cdot LENGTH$$

Light Truck Model

$$\delta \text{ Fuel Cost} = \frac{12.9}{MPG(LT)} \left[832.19 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] + 0.016 \cdot \left[VO(J)^{2} - VN(J)^{2} \right] \right] \cdot$$

$$PCT(LT) \cdot PPCT(J) \cdot PWTFC(LT) \cdot GASPRICE \cdot LENGTH$$

Heavy Truck Model

$$\delta \text{ Fuel Cost} = \frac{5.7}{MPG(HT)} \left[1986.88 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] + 0.0178 \cdot \left[VO(J)^{2} - VN(J)^{2} \right] \right] \cdot$$

$$PCT(HT) \cdot PPCT(J) \cdot PWTFC \cdot GASPRICE \cdot LENGTH$$

Where:

MPG (A,LT,HT)	= projected fuel consumption by vehicle type
VO(J), VN(J)	= old and new vehicle speed for time period J (1= peak, 2 = off-peak)
PCT(A,LT,HT)	= fraction of given vehicle type in traffic volume
PPCT(J)	= fraction of flow in peak (J=1) and off-peak (J=2)
PWTFC	= discounted present worth of traffic over project life
GASPRICE	= price of fuel, $ per gallon
LENGTH	= section length in miles

TABLE 2
INTERRUPTED FUEL COST SAVING MODELS

<u>Automobile Model</u>

$$\delta \text{ Fuel Cost} = \frac{16.3}{MPG(A)} \cdot 601.69 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] \cdot PCT(A) \cdot PPCT(J) \cdot PWTFC \cdot GASPRICE \cdot LENGTH$$

<u>Light Truck Model</u>

$$\delta \text{ Fuel Cost} = \frac{12.9}{MPG(LT)} \cdot 1006.95 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] \cdot PCT(LT) \cdot PPCT(J) \cdot PWTFC \cdot GASPRICE \cdot LENGTH$$

<u>Heavy Truck Model</u>

$$\delta \text{ Fuel Cost} = \frac{5.7}{MPG(HT)} \left[2000689 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right]^2 + 14 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] \cdot \left[VO(J)^2 - VN(J)^2 \right] \right] \cdot$$

$$PCT(HT) \cdot PPCT(J) \cdot PWTFC \cdot GASPRICE \cdot LENGTH \cdot \left[\left[832.9 \cdot \left[\frac{1}{VO(J)} - \frac{1}{VN(J)} \right] + 0.16 \cdot \left[VO(J)^2 - VN(J)^2 \right] \right]^{-1}$$

Where all terms are as defined in Table 1.

APPENDIX B
PROGRAM LISTING

MAINDSS

```
10 REM JLS D S S June 17, 1988   <MAIN DSS>
30 CLS:KEY OFF:CLEAR:VNO%=1 'NEW PROJ RETURN
35 DIM CLASFAT(4,2),CLASINJ(4,2),CLASPDO(4,2),REDTOTC(2),REDFATC(2),REDINJC(2)
37 DIM REDPDOC(2),CYFAT(2),CYINJ(2),CYPDO(2),OLDPKTT(2),NEWPKTT(2),PKTTSAV(2),OL
DOPTT(2),NEWOPTT(2),OPTTSAV(2),NEWOATT(2),OATTSAV(2),OLDOATT(2),UNITBEN(2)
40 DIM CASHF(60),YR(7),COST(7),LIFE(7),SYR(7),TAG$(7),ANCOST(7),PWCOST(7),PPCT(2
),VO(2),VN(2),IDLE(3):IDLE(1)=.5:IDLE(2)=.7:IDLE(3)=2
45 DIM BENEFIT(3,3),BENVAL(3,3),CLAS$(2),FAT(4),INJ(4),PDO(4),TRAF(4),PWBEN(3),A
WBEN(3),D(20),B(20),PF%(6),MPG(3),FA(3),FB(3),FC(3),FADJ(3),PCT(3)
55 R$="N":TWO=0:FOR I=1 TO 6:PF%(I)=0:NEXT I:MPG(1)=16.3:MPG(2)=12.9:MPG(3)=5.7:
GASPRI=1.25:FLOWCON%=2:FA(1)=497:FA(2)=832.19:FA(3)=1986.88:FB(1)=.0139:FB(2)=.0
16:FB(3)=.0178:FC(1)=601.69:FC(2)=1006.95:FC(3)=1200
60 CHA$="CHANGE INPUTS (Y,N)?":LOCATE 1,10:PRINT "DECISION SUPPORT SYSTEM FOR TR
ANSPORTATION PROJECT ANALYSIS"
65 GOSUB 40000
70 LOCATE 2,25:PRINT "NORTHWESTERN UNIVERSITY  1988":LOCATE 5,35:PRINT"ACTIONS"
CHR$(13)
90 PRINT TAB(28)"1  RETRIEVE FROM FILE":PRINT TAB(28) "2  DEFINE NEW PROJECT":PR
INT TAB(28) "3  QUIT"
110 LOCATE 12,25,1:COLOR 0,7:PRINT "ENTER NUMBER OF DESIRED ACTION";:COLOR 7,0
120 C$=INKEY$:IF C$="" GOTO 120
130 IF C$="1" THEN GOSUB 34000
140 IF C$="2" GOTO 195
145 IF C$<>"3" GOTO 160 ELSE LOCATE 14,26,1:COLOR 0,7:PRINT "VERIFY QUIT BY ENTE
RING `Q`";:COLOR 7,0:BEEP
147 C$=INKEY$:IF C$="" GOTO 147
148 IF C$="q" OR C$="Q" THEN STOP ELSE 30
160 LOCATE 22,23:COLOR 0,7:PRINT"PLEASE ENTER CORRECT INSTRUCTION";:COLOR 7,0:GO
SUB 12500:GOTO 30 '***INPUT ROUTINE NEXT***
195 CLS:LOCATE 1,32:PRINT "AUTO FILER SETUP":LOCATE 3,9,1:COLOR 0,7:PRINT "DO YO
U WANT TO FILE PROJECT CHARACTERISTICS ON DISKETTE (Y,N)?";:COLOR 7,0
200 C$=INKEY$:IF C$="" GOTO 200
205 IF C$ = "Y" OR C$ = "y" THEN GOSUB 27500
210 CLS:PRINT TAB(23)"PROJECT DESCRIPTION INPUT ROUTINE"
220 LOCATE 3,1:PRINT "1=PROJECT LOCATION: " 'PLACE$=LOCATION
230 LOCATE 5,1:PRINT "2=PROPOSED ACTION: " 'ACT$=ACTION
240 LOCATE 7,5 :PRINT "3=SECTION LENGTH [Mi.]: " 'LENG=LENGTH
250 LOCATE 7,40: PRINT "4=PROJECT LIFE, Yrs: " 'NYEAR=PROJ LIFE
260 LOCATE 9,5:PRINT "5=AADT: " TAB(25) "6=Yr of AADT: " SPC(13) "7=Yr PROJECT O
PENS: " 'YAADT=YEAR AADT;YOP=YEAR OPENS
270 LOCATE 11,18:PRINT "8=PERCENT AUTOS: " 'PCAR=% AUTO
280 LOCATE 13,18:PRINT "9=PERCENT LIGHT TRUCKS: " 'PLTRUK=% LT TRUCK
290 LOCATE 15,17:PRINT "10=PERCENT HEAVY TRUCKS: " 'PHTRUK=% HEAV TRK
300 LOCATE 17,20:PRINT "TOTAL: " 'PTOT=TOTAL %
310 LOCATE 19,2:PRINT "11=ANNUAL TRAFFIC GROWTH, % per YEAR: _____" 'ANGROW
320 LOCATE 20,5:PRINT "[enter zero to use end-year AADT forecast]" 'INP.SCREEN*
335 IF R$="Y" THEN GOSUB 13005
340 LOCATE 3,21:FOR M%=1 TO 58:PRINT "_";:NEXT M%:LOCATE 3,21,1:LINE INPUT "",PL
ACE$:IF CH$="Y" GOTO 360
350 LOCATE 5,20:FOR M%=1 TO 59:PRINT "_";:NEXT M%:LOCATE 5,20,1:LINE INPUT "",AC
T$
360 LOCATE 6,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
370 C$=INKEY$:IF C$="" GOTO 370
380 LOCATE 6,30:PRINT SPC(20):IF C$="Y" OR C$="y" GOTO 390 ELSE 405
390 LOCATE 6,25,1:COLOR 0,7:PRINT "ENTER ITEM NUMBER TO CHANGE: ";:COLOR 7,0
400 N$=INKEY$:IF N$=""GOTO 400
402 NS=VAL(N$):LOCATE 6,25:PRINT SPC(31):CH$="Y":IF NS<1 OR NS>2 GOTO 360
404 ON NS GOTO 340,350
405 CH$="N":IF R$="Y" THEN GOSUB 13020
```

```
410 LOCATE 7,29:PRINT "_____ ":LOCATE 7,29,1:GOSUB 30000:LENG=NEWNUM:IF CH$="Y"
GOTO 500
420 LOCATE 7,61:PRINT "_____ ":LOCATE 7,61,1:GOSUB 30000:NYEAR=NEWNUM
430 IF NYEAR<41 GOTO 465 ELSE LOCATE 8,40:COLOR 0,7:PRINT "LIFE MUST BE =< 40 YE
ARS!";:COLOR 7,0:GOSUB 12500
440 LOCATE 8,40:PRINT SPC(38):LOCATE 7,61:PRINT "_____":GOTO 420
450 GOSUB 12500:LOCATE 8,40:PRINT SPC(38)
465 IF CH$="Y" GOTO 500
470 LOCATE 9,13:PRINT "__________ ":LOCATE 9,13,1:GOSUB 30000:AADT=NEWNUM:IF CH$=
"Y" GOTO 500
480 LOCATE 9,39:PRINT "_____ ":LOCATE 9,39,1:GOSUB 30000:YAADT=NEWNUM:IF CH$="Y"
 GOTO 500
490 LOCATE 9,72:PRINT "_____ ":LOCATE 9,72,1:GOSUB 30000:YOP=NEWNUM
500 LOCATE 10,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
510 C$=INKEY$:IF C$="" GOTO 510
520 LOCATE 10,30:PRINT SPC(22):IF C$="Y" OR C$="y" GOTO 530 ELSE 555
530 LOCATE 10,25,1:COLOR 0,7:PRINT "ENTER ITEM NUMBER TO CHANGE: ";:COLOR 7,0
540 N$=INKEY$:IF N$=""GOTO 540
550 NS=VAL(N$):LOCATE 10,25:PRINT SPC(31):CH$="Y":IF NS<3 OR NS>7 GOTO 500
553 ON NS-2 GOTO 410,420,470,480,490
555 CH$="N":IF R$="Y" THEN GOSUB 13040
560 LOCATE 11,41:PRINT "_____":LOCATE 11,42,1:GOSUB 30000:PCAR=NEWNUM:IF PCAR >=
100 THEN LOCATE 13,41:PRINT "_____":LOCATE 15,41:PRINT "_____":GOTO 590
570 LOCATE 13,41:PRINT "_____":LOCATE 13,42,1:GOSUB 30000:PLTRUK=NEWNUM:IF PCAR+
PLTRUK>=100 THEN LOCATE 15,41:PRINT "_____":GOTO 590
580 LOCATE 15,41:PRINT "_____":LOCATE 15,42,1:GOSUB 30000:PHTRUK=NEWNUM
590 PTOT=PCAR+PLTRUK+PHTRUK:LOCATE 17,41:PRINT USING "###.#";PTOT:IF PTOT<>100 T
HEN BEEP:PCAR=0:PLTRUK=0:PHTRUK=0:GOTO 560
595 LOCATE 13,55:COLOR 0,7:PRINT CHA$;:COLOR 7,0
597 C$=INKEY$:IF C$="" GOTO 597
598 IF C$="Y" OR C$="y" THEN PCAR=0:PLTRUK=0:PHTRUK=0:GOTO 560
605 LOCATE 13,55:PRINT SPC(25)
607 IF R$="Y" THEN GOSUB 13060
610 LOCATE 19,40,1:GOSUB 30000:ANGROW=NEWNUM:IF ANGROW=0 GOTO 620 ELSE 670
620 LOCATE 22,5:PRINT "ESTIMATED AADT, YEAR " YOP+NYEAR ": _____"
630 LOCATE 22,34,1:GOSUB 30000:FUTAADT=NEWNUM
640 VRATIO=FUTAADT/AADT 'RATIO FUTURE TO PRESENT AADT
650 EX=1/(NYEAR+YOP-YAADT):ANGROW=((VRATIO^EX)-1)*100
660 LOCATE 19,40:PRINT USING "###.#";ANGROW:GOTO 700
670 FUTAADT=AADT*((1+ANGROW/100)^(NYEAR+YOP-YAADT))
680 LOCATE 22,5:PRINT "ESTIMATED AADT, YEAR " YOP+NYEAR ":";
690 PRINT USING "#########,";FUTAADT
700 LOCATE 19,50:COLOR 0,7:PRINT CHA$;:COLOR 7,0
710 LOCATE 19,78:C$=INKEY$:IF C$="" GOTO 710
720 IF C$="Y" OR C$="y" GOTO 730 ELSE 750
730 LOCATE 19,40:PRINT "_____":LOCATE 22,32:PRINT "____________":GOTO 610
750 CLS:GOSUB 32100
760 LOCATE 1,1:PRINT SPC(20) "PROVIDER COST DATA INPUT ROUTINE"
770 LOCATE 3,1:PRINT "    ITEM         (FIRST)     LUMP  or  PER      LIFE"
780 LOCATE 4,1:PRINT "    NAME        YEAR PAID    SUM $ ==> MILE $  (YEARS)
790 LOCATE 5,1:PRINT "¦==============¦==========¦=========¦=========¦=======¦"
800 LOCATE 7,2:PRINT "1>DESIGN":LOCATE 9,2:PRINT "2>R-O-W":LOCATE 11,2:PRINT "3>
CONSTRUCTION":LOCATE 13,2:PRINT "4>MAINTENANCE "CHR$(4):LOCATE 15,2:PRINT "5>OPE
RATIONS "CHR$(4)
810 LOCATE 17,2:PRINT "6>OTHER":LOCATE 19,2:PRINT "7>OTHER"
812 LOCATE 20,2:PRINT "("CHR$(4) " = ONLY recurring cost categories)"
815 IF R$="Y" THEN GOSUB 13080
820 LOCATE 7,19,1:GOSUB 30000:YR(1)=NEWNUM 'YR(1)=YR DESIGN COST PAID
830 IF YR(1)= 0 GOTO 880
840 LOCATE 7,28,1:GOSUB 30000:COST(1)=NEWNUM 'COST(1)=DESIGN COST
```

```
850 IF COST(1)<>0 GOTO 870
860 LOCATE 7,38,1:GOSUB 30000:COST(1)=NEWNUM*LENG:LOCATE 7,28:PRINT USING "#####
###";COST(1)
870 LOCATE 7,48,1:GOSUB 30000:LIFE(1)=NEWNUM 'LIFE(1)=LIFE DESCOS
880 IF F=1 GOTO 1310    'F=CORRECTION FLAG
890 LOCATE 9,19,1:GOSUB 30000:YR(2)=NEWNUM 'YR(2)=YEAR ROW PD
900 IF YR(2)=0 GOTO 950
910 LOCATE 9,28,1:GOSUB 30000:COST(2)=NEWNUM 'COST(2)=R-O-W COST
920 IF COST(2)<>0 GOTO 940
930 LOCATE 9,38,1:GOSUB 30000:COST(2)=NEWNUM*LENG:LOCATE 9,28:PRINT USING "#####
###";COST(2)
940 LOCATE 9,48,1:GOSUB 30000:LIFE(2)=NEWNUM 'LIFE(2)=LIFE OF R-O-W
950 IF F=1 GOTO 1310
960 LOCATE 11,19,1:GOSUB 30000:YR(3)=NEWNUM 'YR(3)=YEAR CON PAID
970 IF YR(3)=0 GOTO 1020
980 LOCATE 11,28,1:GOSUB 30000:COST(3)=NEWNUM 'COST(3)=CONST COST
990 IF COST(3)<>0 GOTO 1010
1000 LOCATE 11,38,1:GOSUB 30000:COST(3)=NEWNUM*LENG:LOCATE 11,28:PRINT USING "##
######";COST(3)
1010 LOCATE 11,48,1:GOSUB 30000:LIFE(3)=NEWNUM 'LIFE(3)=LIFE OF CONST
1020 IF F=1 GOTO 1310
1030 LOCATE 13,19,1:GOSUB 30000:YR(4)=NEWNUM 'YR(4)= 1st MAINT YR
1040 IF YR(4)=0 GOTO 1090
1050 LOCATE 13,28,1:GOSUB 30000:COST(4)=NEWNUM 'COST(4)=MAINT COST
1060 IF COST(4)<>0 GOTO 1080
1070 LOCATE 13,38,1:GOSUB 30000:COST(4)=NEWNUM*LENG:LOCATE 13,28:PRINT USING "##
######";COST(4)
1080 LOCATE 13,48,1:GOSUB 30000:LIFE(4)=NEWNUM 'LIFE(4)=MAINT INTERVAL
1090 IF F=1 GOTO 1310
1100 LOCATE 15,19,1:GOSUB 30000:YR(5)=NEWNUM 'YR(5)=1st OP COST YR
1110 IF YR(5)=0 GOTO 1160
1120 LOCATE 15,28,1:GOSUB 30000:COST(5)=NEWNUM 'COST(5)=OP COST
1130 IF COST(5)<>0 GOTO 1150
1140 LOCATE 15,38,1:GOSUB 30000:COST(5)=NEWNUM*LENG:LOCATE 15,28:PRINT USING "##
######";COST(5)
1150 LOCATE 15,48,1:GOSUB 30000:LIFE(5)=NEWNUM 'LIFE(5)=OP COST INTERVAL
1160 IF F=1 GOTO 1310
1170 LOCATE 17,19,1:GOSUB 30000:YR(6)=NEWNUM 'YR(6)=1st YR OTHER COST1
1180 IF YR(6)=0 GOTO 1310
1190 LOCATE 17,28,1:GOSUB 30000:COST(6)=NEWNUM 'COST(6)=1st OTHER COST
1200 IF COST(6)<>0 GOTO 1220
1210 LOCATE 17,38,1:GOSUB 30000:COST(6)=NEWNUM*LENG:LOCATE 17,28:PRINT USING "##
######";COST(6)
1220 LOCATE 17,48,1:GOSUB 30000:LIFE(6)=NEWNUM 'LIFE(6)=INTERVAL O1 COST
1230 IF F=1 GOTO 1310
1240 LOCATE 19,19,1:GOSUB 30000:YR(7)=NEWNUM 'YR(7)=1st YR OTHER COST 2
1250 IF YR(7)=0 GOTO 1310
1260 LOCATE 19,28,1:GOSUB 30000:COST(7)=NEWNUM 'COST(7)=2nd OTHER COST
1270 IF COST(7)<>0 GOTO 1290
1280 LOCATE 19,38,1:GOSUB 30000:COST(7)=NEWNUM*LENG:LOCATE 19,28:PRINT USING "##
######";COST(7)
1290 LOCATE 19,48,1:GOSUB 30000:LIFE(7)=NEWNUM 'LIFE(7)=INTERVAL O2 COST
1310 LOCATE 21,25:COLOR 0,7:PRINT CHA$;:COLOR 7,0
1320 C$=INKEY$:IF C$="" GOTO 1320
1330 IF C$="Y" OR C$="y" GOTO 1340 ELSE 1460
1340 LOCATE 23,20:COLOR 0,7:PRINT "ENTER LINE NUMBER TO BE CHANGED:";:COLOR 7,0
1350 C$=INKEY$:IF C$="" GOTO 1350
1360 LOCATE 21,25:PRINT SPC(25):LOCATE 23,20:PRINT SPC(35)
1370 F=0 '***F=FLAG INDICATES REVISION PASS THRU INPUT SCRN LN
1380 IF C$>"7" OR C$<"1" THEN BEEP:GOTO 1340
```

```
1390 IF C$="1" THEN LOCATE 7,19:PRINT SPC(60):F=1:GOTO 820
1400 IF C$="2" THEN LOCATE 9,19:PRINT SPC(60):F=1:GOTO 890
1410 IF C$="3" THEN LOCATE 11,19:PRINT SPC(60):F=1:GOTO 960
1420 IF C$="4" THEN LOCATE 13,19:PRINT SPC(60):F=1:GOTO 1030
1430 IF C$="5" THEN LOCATE 15,19:PRINT SPC(60):F=1:GOTO 1100
1440 IF C$="6" THEN LOCATE 17,19:PRINT SPC(60):F=1:GOTO 1170
1450 IF C$="7" THEN LOCATE 19,19:PRINT SPC(60):F=1:GOTO 1240
1460 LOCATE 21,1:PRINT SPC(79):LOCATE 23,20:PRINT SPC(50)
1464 IF R$="Y" GOTO 1466 ELSE 1470
1466 LOCATE 21,1:PRINT "DISCOUNT RATE: ";:PRINT USING "##.#";DISCO*100:GOTO 1580

1470 LOCATE 21,1:PRINT "DISCOUNT RATE= 10% ";:COLOR 0,7:PRINT "(<RETURN> TO ACCE
PT; ANY OTHER KEY TO REJECT)";:COLOR 7,0
1480 C$=INKEY$:IF C$="" GOTO 1480
1490 IF C$<>CHR$(13) GOTO 1510 ELSE DISCO=.1:GOTO 1500 'DEFAULT DISCO=10%
1500 LOCATE 21,19:PRINT SPC(60):GOTO 1580
1510 LOCATE 21,16:PRINT SPC(65):LOCATE 23,20:PRINT SPC(50)
1520 LOCATE 21,1:PRINT "DISCOUNT RATE: ";
1530 LOCATE 21,16,1:GOSUB 30000:DISCO=NEWNUM/100 'DISCO=DISCOUNT RATE
1540 IF DISCO>0 GOTO 1580 ELSE LOCATE 21,1:COLOR 0,7:PRINT "DISCOUNT RATE MUST B
E POSITIVE!";:COLOR 7,0:GOSUB 12500:GOTO 1460
1580 LOCATE 3,56:PRINT " ANNUALIZED    PRESENT":'COMPUTE/PRINT AW&PW COST
1590 LOCATE 4,56:PRINT "    COST         COST "
1600 LOCATE 5,56:PRINT "===========|===========|
1610 I=DISCO:TOTPW=0:TOTAW=0:FOR L=1 TO 7:ANCOST(L)=0:PWCOST(L)=0:NEXT L
1620 FOR L=1 TO 7
1630 IF YR(L)=0 OR COST(L)=0 OR LIFE(L)=0 GOTO 1680
1640 PRINC=COST(L):N=LIFE(L):FYEAR=YR(L)
1650 GOSUB 28000
1660 LOCATE 5+2*L,55:PRINT USING "$$#########,";AW;:PRINT USING "_ _ $$#########
,";PW
1670 TOTAW=TOTAW+AW:TOTPW=TOTPW+PW
1675 ANCOST(L)=AW:PWCOST(L)=PW
1680 NEXT L
1690 LOCATE 21,35:PRINT "COST TOTALS ==>"
1700 LOCATE 21,55:PRINT USING "$$#########,";TOTAW;:PRINT USING "_ _ $$#########
,";TOTPW '***TOTAW= ANN WORTH; TOTPW=PW***RECYCLE DISCO NEXT
1720 LOCATE 22,25:COLOR 0,7:PRINT "TRY ANOTHER DISCOUNT RATE (Y,N)?";:COLOR 7,0
1730 C$=INKEY$:IF C$="" GOTO 1730
1740 IF C$="Y" OR C$="y" GOTO 1750 ELSE 1760
1750 LOCATE 22,1:PRINT SPC(79):GOTO 1510
1760 LOCATE 22,25:PRINT SPC(35)
1790 LOCATE 22,33:COLOR 0,7:PRINT "DO CASH FLOW ANALYSIS (Y,N)?";:COLOR 7,0
1800 C$=INKEY$:IF C$="" GOTO 1800
1810 IF C$="Y" OR C$="y" GOTO 1840 ELSE 2610 'CASHFLO ANAL
1840 PF%(6)=1:CLS:GOSUB 32100
1850 LOCATE 1,31:PRINT  "CASH FLOW ANALYSIS":TWO=1 'FIND 1st DATA YR
1880 FOR L=1 TO 7:SYR(L)=YR(L):NEXT L
1890 FOR I=1 TO 7:FOR J=1 TO 7
1900 IF SYR(I)=<SYR(J) GOTO 1920 ELSE 1910
1910 SWAP SYR(I),SYR(J)
1920 NEXT J
1930 NEXT I
1940 FOR I = 0 TO 6
1950 IF SYR(7-I)=0 GOTO 1980 '1sT YR = yx
1970 YX=SYR(7-I):GOTO 1985
1980 NEXT I
1985 IF R$="Y" GOTO 2010
1990 LOCATE 3,29,1:COLOR 0,7:PRINT "ENTER INFLATION RATE: ";:COLOR 7,0
2000 GOSUB 30000:INFLATE=NEWNUM/100 'INFLATE=INFLATION
```

```
2002 IF INFLATE=<.25 GOTO 2010 ELSE LOCATE 5,32:COLOR 0,7:PRINT "MUST BE <= 25%!
";:COLOR 7,0:GOSUB 12500
2004 LOCATE 5,32:PRINT SPC(15):LOCATE 3,51:PRINT SPC(5):GOTO 1990
2010 LOCATE 3,29:PRINT SPC(30):LOCATE 1,52:COLOR 0,7:PRINT "(INFLATION = ";
2020 PRINT USING "##.#_%_)";INFLATE*100:COLOR 7,0
2030 LOCATE 3,1:PRINT "YEAR        COST              YEAR        COST
 YEAR        COST" 'ACCUM CASHFLOW/YR
2110 LOCATE 4,1,0:FOR J=1 TO 60:CASHF(J)=0:NEXT J:TOTCASHF=0
2120 NOWYEAR=VAL(RIGHT$(DATE$,4))-YX+1 'INDEX CURRENT YR
2130 OINDEX=YOP-YX+1 'INDEX OPEN YR
2140 NINDEX=OINDEX+NYEAR 'HORIZON YR INDEX
2150 FOR L=1 TO 7:MINUS=0  'MINUS=COST<0
2155 IF YR(L)=0 GOTO 2240
2160 YINDEX=YR(L)-YX+1 'YR COST L 1st PD
2170 IF YINDEX>NINDEX GOTO 2240 'NEXT ITEM
2180 IF YINDEX<=NOWYEAR THEN ADD=COST(L):GOTO 2210
2185 R=ABS(COST(L)):IF COST(L)<0 GOTO 2187 ELSE 2190
2187 MINUS=1  'account for cost save
2190 ADD=(R*(1+INFLATE)^(YINDEX-NOWYEAR))-2*R*MINUS
2210 CASHF(YINDEX)=CASHF(YINDEX) +ADD
2214 TOTCASHF=TOTCASHF +ADD
2216 IF L<4 OR L>5 GOTO 2240 'PREVENT REPAY NONRECURRING COSTS
2220 YINDEX=YINDEX+LIFE(L):GOTO 2170
2240 NEXT L 'PRN CASHFLOWS
2250 FOR I=1 TO (YOP-YX)+1+NYEAR  '=FROM 1 TO NINDEX
2260 IF I<19 GOTO 2270 ELSE 2280
2270 LOCATE 4+I,1:PRINT USING "####";YX-1+I;:PRINT USING "_ _ $$#########,";CASH
F(I);:GOTO 2310
2280 IF I<37 GOTO 2290 ELSE 2300
2290 LOCATE I-14,27:PRINT USING "####";YX-1+I;:PRINT USING "_ _ $$#########,";CA
SHF(I);:GOTO 2310
2300 LOCATE I-32,59:PRINT USING "####";YX-1+I;:PRINT USING "_ _ $$#########,";CA
SHF(I);
2310 NEXT I
2320 ALIN=CSRLIN:ACOL=POS(0):LOCATE ALIN+1,ACOL-18:PRINT "TOTAL";:PRINT USING "_
 _ $$#########,";TOTCASHF
2330 LOCATE 25,50,1:COLOR 0,7:PRINT "TRY NEW INFLATION RATE (Y,N)?";:COLOR 7,0
2340 C$=INKEY$:IF C$="" GOTO 2340
2350 IF C$="y" OR C$="Y" GOTO 2360 ELSE 2365
2360 LOCATE 1,1:PRINT SPC(25):LOCATE 25,50:PRINT SPC(29):LOCATE 3,1:PRINT SPC(79
):GOTO 1990
2365 LOCATE 25,50:PRINT SPC(29)
2370 LOCATE 25,50,1:COLOR 0,7:PRINT "SEE `LOCAL` SHARE (Y,N)? ";:COLOR 7,0
2380 C$=INKEY$:IF C$="" GOTO 2380
2390 IF C$="Y" OR C$="y" GOTO 2400 ELSE 2610
2400 LOCATE 25,50:PRINT SPC(29)
2405 IF R$="Y" AND TWO>0 GOTO 2415
2410 LOCATE 25,50,1:COLOR 0,7:PRINT "ENTER LOCAL SHARE, %: ";:COLOR 7,0:GOSUB 30
000:SHARE=NEWNUM/100
2415 CLS:LOCATE 1,1:COLOR 0,7:PRINT USING "##.#";SHARE*100;:PRINT  "% `LOCAL` SH
ARE";:LOCATE 1,52:PRINT USING "_I_N_F_L_A_T_I_O_N_=_ ##.#_%";INFLATE*100;:COLOR
7,0
2420 LOCATE 1,31:PRINT "CASH FLOW ANALYSIS";
2422 LOCATE 3,1:PRINT "YEAR        COST              YEAR        COST
 YEAR        COST";
2425 LOCATE 25,1:PRINT SPC(79):GOSUB 32100
2430 REM ***PRINT LOC SHR CASH FLOWS***
2440 FOR I=1 TO (YOP-YX)+1+NYEAR  '=FROM 1 TO NINDEX
2450 IF I<19 GOTO 2460 ELSE 2470
2460 LOCATE 4+I,1:PRINT USING "####";YX-1+I;:PRINT USING "_ _ $$#########,";CASH
```

```
F(I)*SHARE;:GOTO 2500
2470 IF I<37 GOTO 2480 ELSE 2490
2480 LOCATE I-14,27:PRINT USING "####";YX-1+I;:PRINT USING "_ _ $$#########,";CA
SHF(I)*SHARE;:GOTO 2500
2490 LOCATE I-32,59:PRINT USING "####";YX-1+I;:PRINT USING "_ _ $$#########,";CA
SHF(I)*SHARE;
2500 NEXT I
2510 ALIN=CSRLIN:ACOL=POS(0):LOCATE ALIN+1,ACOL-18:PRINT "TOTAL";:PRINT USING "_
 _ $$#########,";TOTCASHF*SHARE
2515 TWO=TWO-1
2520 GOTO 2330
2610 TWO=1
2620 LOCATE 25,34,1:COLOR 0,7:PRINT "DO COST PER VEHICLE [MILE] ANALYSIS (Y,N)?
 ";:COLOR 7,0
2630 C$=INKEY$:IF C$="" GOTO 2630
2640 IF C$="Y" OR C$="y" GOTO 2650 ELSE 3080
2650 PF%(1)=1:CLS:GOSUB 32100
2660 LOCATE 1,28:PRINT "ECONOMIC COST ALLOCATION" CHR$(13)
2670 PRINT "VEHICLE TYPE" SPC(12) "PERCENT" SPC(8) "COST WEIGHT"
2680 PRINT SPC(23) "OF TRAFFIC" SPC(5) "(E.G., PCE`S)"
2690 LOCATE 6,1:PRINT "AUTOS":LOCATE 8,1:PRINT "LIGHT TRUCKS":LOCATE 10,1:PRINT
"HEAVY TRUCKS"
2700 LOCATE 6,25:PRINT USING "###.#_%";PCAR
2710 LOCATE 8,25:PRINT USING "###.#_%";PLTRUK
2720 LOCATE 10,25:PRINT USING "###.#_%";PHTRUK
2722 IF R$="Y" THEN GOSUB 13120
2725 WCAR=1:WLTRUK=2:WHTRUK=3.5  'COST ALLOC WTS
2730 LOCATE 6,45:PRINT "1.0":LOCATE 8,45:PRINT "2.0":LOCATE 10,45:PRINT "3.5"
2740 LOCATE 12,1,1:COLOR 0,7:PRINT "DEFAULT WEIGHTS SHOWN; <RETURN> TO ACCEPT; A
NY OTHER KEY TO CHANGE";:COLOR 7,0
2750 C$=INKEY$:IF C$="" GOTO 2750
2760 IF C$=CHR$(13) GOTO 2850  'USE DEFAULT VALS
2780 LOCATE 6,45:PRINT SPC(5):LOCATE 6,45,1:GOSUB 30000:WCAR=NEWNUM  'REVISE WT
2790 LOCATE 8,45:PRINT SPC(5):LOCATE 8,45,1:GOSUB 30000:WLTRUK=NEWNUM
2800 LOCATE 10,45:PRINT SPC(5):LOCATE 10,45,1:GOSUB 30000:WHTRUK=NEWNUM
2810 LOCATE 12,1:PRINT SPC(79)
2820 LOCATE 12,30,1:COLOR 0,7:PRINT CHA$;:COLOR 7,0
2830 C$=INKEY$:IF C$="" GOTO 2830
2840 IF C$="Y" OR C$="y" GOTO 2780
2850 TRAFY=AADT*(1+ANGROW/100)^(YOP-YAADT) 'COMPUTE OPEN YR TFC
2860 TOTRAF=TRAFY 'TRAFY=TOT TFC VOL OVER LIFE
2870 GFACTOR = (1+ANGROW/100)
2880 FOR J = 1 TO NYEAR-1
2890 TRAFY = TRAFY*(GFACTOR)
2900 TOTRAF=TOTRAF + TRAFY
2910 NEXT J
2915 LFLAG=0  'LFLAG=0=NOT SPOT
2920 IF LENG=0 GOTO 2930 ELSE 2940
2930 LFLAG=1:XLENG=1:GOTO 2950  'LFLAG=1=SPOT
2940 XLENG=LENG
2950 VMT = 365*XLENG*TOTRAF 'VMT=VMT OR VEH IF SPOT;TUNITS=PCUs
2960 TUNITS=VMT*((WCAR*PCAR/100)+(WLTRUK*PLTRUK/100)+(WHTRUK*PHTRUK/100))
2970 CPU=TOTPW/TUNITS 'COST PER TFC UNIT
2980 CARSHARE=CPU*WCAR:LTSHARE=CPU*WLTRUK:HTSHARE=CPU*WHTRUK
2985 IF LFLAG=0 GOTO 2995
2990 LOCATE 3,57:PRINT "WTD. COST/VEHICLE":GOTO 2997
2995 LOCATE 3,57:PRINT "WTD. COST PER":LOCATE 4,57:PRINT "VEHICLE-MILE"
2997 IF PCAR=0 GOTO 3005
3000 LOCATE 6,55:PRINT USING "$$####.####";CARSHARE
3005 IF PLTRUK=0 GOTO 3015
```

```
3010 LOCATE 8,55:PRINT USING "$$####.####";LTSHARE
3015 IF PHTRUK=0 GOTO 3030
3020 LOCATE 10,55:PRINT USING "$$####.####";HTSHARE
3030 LOCATE 12,1:PRINT SPC(79)
3040 LOCATE 12,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
3050 C$=INKEY$:IF C$="" GOTO 3050
3060 IF C$="Y" OR C$ = "y" GOTO 3070 ELSE 3080
3070 LOCATE 12,21:PRINT SPC(58):GOTO 2780
3080 CLS:REM BEN ANAL ROUT
3090 LOCATE 14,24:COLOR 0,7:PRINT "PERFORM BENEFITS ANALYSIS (Y,N)?";:COLOR 7,0
3100 C$=INKEY$:IF C$="" GOTO 3100
3110 IF C$="Y" OR C$="y" GOTO 3120 ELSE 3115
3115 IF R$="Y" AND PF%(2)>0 GOTO 9200 ELSE 12000
3120 PF%(2)=1:CLS:PRINT TAB(27) "BENEFITS ANALYSIS ROUTINE"
3125 GOSUB 32100
3130 LOCATE 3,15:PRINT "ENTER PERCENTAGE OF TRAFFIC IN ALL PEAK PERIODS:  "
3140 LOCATE 4,7:PRINT "(Enter zero if you do not wish to treat peak & off peak s
eparately)"
3145 IF R$="Y" THEN GOSUB 13140
3150 LOCATE 3,64:PRINT SPC(5):LOCATE 3,64,1:GOSUB 30000:PEAKPCT=NEWNUM
3160 IF PEAKPCT<0 OR PEAKPCT>100 THEN BEEP:GOTO 3120
3170 OFFPEAK=100-PEAKPCT:LOCATE 6,26:PRINT "PEAK HOUR TRAFFIC PERCENT", PEAKPCT:
LOCATE 8,26:PRINT "OFF-PEAK TRAFFIC PERCENT  ",OFFPEAK
3180 LOCATE 10,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
3190 C$=INKEY$:IF C$="" GOTO 3190
3200 LOCATE 10,26:PRINT SPC(30):IF C$="Y" OR C$="y" GOTO 3150
3202 IF LENG=0 GOTO 3204 ELSE 3206
3204 LOCATE 15,16,1:PRINT "TOTAL AADT IS: ";:PRINT USING "######,";AADT;:PRINT "
. ENTER CROSS STREET AADT: ";:GOTO 3207
3206 LOCATE 15,7,1:PRINT "AADT IS: ";:PRINT USING "######,";AADT;:PRINT ". ENTER
 TOTAL AADT, ALL CROSS STREETS IN SECTION: ";
3207 IF CH$="Y" GOTO 3209
3208 IF R$="Y" THEN LOCATE 15,72:PRINT USING "######,";XAADT:GOTO 3212
3209 LOCATE 15,72:PRINT SPC(7):LOCATE 15,72:GOSUB 30000:XAADT=NEWNUM:IF LENG<>0
GOTO 3212
3210 IF XAADT=<AADT GOTO 3212 ELSE LOCATE 16,16:COLOR 0,7:PRINT "CROSS STREET VO
LUME CANNOT EXCEED TOTAL VOLUME!";:COLOR 7,0:GOSUB 12500
3211 LOCATE 16,16:PRINT SPC(50):GOTO 3202
3212 LOCATE 16,30,1:COLOR 0,7:PRINT CHA$;:COLOR 7,0
3214 C$=INKEY$:IF C$="" GOTO 3214
3216 LOCATE 16,30:PRINT SPC(21):IF C$="Y" OR C$="y" THEN LOCATE 16,30:PRINT SPC(
25):CH$="Y":GOTO 3202
3218 CH$="N":LOCATE 18,24:PRINT "COMPUTING FUTURE TRAFFIC VOLUMES":PWTFC=0:PWXTF
C=0:XFRACT=0:IF LENG=0 THEN XFRACT=XAADT/AADT
3220 FOR I=1 TO NYEAR+1  'TTFC(I)=TOT TFC, YR I
3230 TTFC=AADT*(1+(ANGROW/100))^(YOP-YAADT+I-1):XTTFC=XAADT*(1+(ANGROW/100))^(YO
P-YAADT+I-1)
3235 DISTFC=TTFC*365/((1+DISCO)^I):PWTFC=PWTFC+DISTFC:DISXTFC=XTTFC*365/((1+DISC
O)^I):PWXTFC=PWXTFC+DISXTFC:NEXT I
3240 LOCATE 18,23:COLOR 0,7:PRINT "COMPUTE TRAVEL TIME SAVINGS (Y,N)?";:COLOR 7,
0
3250 C$=INKEY$:IF C$="" GOTO 3250
3260 IF C$="Y" OR C$="y" GOTO 3270 ELSE 3910  'OP COST SAVING
3270 LOCATE 18,19:COLOR 0,7:PRINT "ENTER TRAVEL TIME SAVINGS DIRECTLY (Y,N)?";:C
OLOR 7,0:LOCATE 19,3:PRINT "If NO, must enter speed change estimates; Y required
 for spot locations!":LOCATE 18,61:PF%(3)=1
3280  C$=INKEY$:IF C$="" GOTO 3280
3290 IF C$="n" OR C$="N" AND LENG<>0 GOTO 3500
3295 X%=1
3300 CLS:PRINT TAB(14) "ESTIMATED PER VEHICLE TRAVEL TIME SAVINGS IN MINUTES":LO
```

```
CATE 20,33:COLOR 0,7:IF X%=1 THEN PRINT "MAIN LINE FLOW":COLOR 7,0:GOTO 3305
3302 LOCATE 20,32:COLOR 0,7:PRINT "CROSS STREET FLOW":COLOR 7,0
3305 GOSUB 32100:LOCATE 2,23: PRINT "SECTION LENGTH (MILES) = ", LENG
3310 LOCATE 3,1:PRINT "<= CURRENT TRAVEL TIMES => <= EXPECTED TRAVEL TIMES => <=
 TRAVEL TIME SAVINGS =>"
3320 LOCATE 5,1:PRINT "PEAK ":LOCATE 7,1:PRINT "OFF PEAK":LOCATE 9,1:PRINT "OVER
ALL"
3330 IF PEAKPCT>0 GOTO 3335 ELSE 3425
3335 IF R$="Y" AND TWO>0 THEN GOSUB 13160
3340 LOCATE 5,14:PRINT SPC(10):LOCATE 5,14,1:GOSUB 30000:OLDPKTT(X%)=NEWNUM
3345 IF NEWNUM=0 THEN LOCATE 18,27:COLOR 0,7:PRINT "TRAVEL TIME CANNOT BE ZERO!"
;:COLOR 7,0:BEEP:GOSUB 12500:LOCATE 18,27:PRINT SPC(30):GOTO 3340
3350 LOCATE 5,35:PRINT SPC(10):LOCATE 5,35,1:GOSUB 30000:NEWPKTT(X%)=NEWNUM
3355 IF NEWNUM=0 THEN LOCATE 18,27:COLOR 0,7:PRINT "TRAVEL TIME CANNOT BE ZERO!"
;:COLOR 7,0:BEEP:GOSUB 12500:LOCATE 18,27:PRINT SPC(30):GOTO 3350
3360 PKTTSAV(X%)=OLDPKTT(X%)-NEWPKTT(X%):LOCATE 5,65:PRINT USING "###.#";PKTTSAV
(X%)
3370 LOCATE 7,14:PRINT SPC(10):LOCATE 7,14,1:GOSUB 30000:OLDOPTT(X%)=NEWNUM
3375 IF NEWNUM=0 THEN LOCATE 18,27:COLOR 0,7:PRINT "TRAVEL TIME CANNOT BE ZERO!"
;:COLOR 7,0:BEEP:GOSUB 12500:LOCATE 18,27:PRINT SPC(30):GOTO 3370
3380 LOCATE 7,35:PRINT SPC(10):LOCATE 7,35,1:GOSUB 30000:NEWOPTT(X%)=NEWNUM
3385 IF NEWNUM=0 THEN LOCATE 18,27:COLOR 0,7:PRINT "TRAVEL TIME CANNOT BE ZERO!"
;:COLOR 7,0:BEEP:GOSUB 12500:LOCATE 18,27:PRINT SPC(30):GOTO 3380
3390 OPTTSAV(X%)=OLDOPTT(X%)-NEWOPTT(X%):LOCATE 7,65:PRINT USING "###.#";OPTTSAV
(X%)
3400 OLDOATT(X%)=(PEAKPCT*OLDPKTT(X%)/100) +(100-PEAKPCT)*OLDOPTT(X%)/100:LOCATE
 9,14:PRINT USING "###.#";OLDOATT(X%)
3410 NEWOATT(X%)=(PEAKPCT*NEWPKTT(X%)/100) +(100-PEAKPCT)*NEWOPTT(X%)/100:LOCATE
 9,35:PRINT USING "###.#";NEWOATT(X%)
3420 OATTSAV(X%)=(PEAKPCT*PKTTSAV(X%)/100) +(100-PEAKPCT)*OPTTSAV(X%)/100:LOCATE
 9,65:PRINT USING "###.#";OATTSAV(X%):GOTO 3452
3425 IF R$="Y" AND TWO>0 THEN GOSUB 13200
3430 LOCATE 9,14:PRINT SPC(10):LOCATE 9,14,1:GOSUB 30000:OLDOATT(X%)=NEWNUM
3435 IF NEWNUM=0 THEN LOCATE 18,27:COLOR 0,7:PRINT "TRAVEL TIME CANNOT BE ZERO!"
;:COLOR 7,0:BEEP:GOSUB 12500:LOCATE 18,27:PRINT SPC(30):GOTO 3430
3440 LOCATE 9,35:PRINT SPC(10):LOCATE 9,35,1:GOSUB 30000:NEWOATT(X%)=NEWNUM
3445 IF NEWNUM=0 THEN LOCATE 18,27:COLOR 0,7:PRINT "TRAVEL TIME CANNOT BE ZERO!"
;:COLOR 7,0:BEEP:GOSUB 12500:LOCATE 18,27:PRINT SPC(30):GOTO 3440
3450 OATTSAV(X%)=OLDOATT(X%)-NEWOATT(X%):LOCATE 9,65:PRINT USING "###.#";OATTSAV
(X%)
3452 IF X%>1 GOTO 3474
3455 IF LENG>0 GOTO 3456 ELSE 3474
3456 NUM=60*LENG:OLDPKSP=NUM/OLDPKTT(X%):NEWPKSP=NUM/NEWPKTT(X%):OLDOPSP=NUM/OLD
OPTT(X%):NEWOPSP=NUM/NEWOPTT(X%):OLDOASP=NUM/OLDOATT(X%):NEWOASP=NUM/NEWOATT(X%)

3457 IF PEAKPCT>0 GOTO 3465 ELSE 3471
3465 LOCATE 6,4:PRINT "SPEED =>":LOCATE 8,4:PRINT "SPEED =>":LOCATE 6,17:COLOR 0
,7:PRINT USING "##.#";OLDPKSP:LOCATE 6,38:PRINT USING "##.#";NEWPKSP
3468 LOCATE 8,17:PRINT USING "##.#";OLDOPSP:LOCATE 8,38:PRINT USING "##.#";NEWOP
SP:COLOR 7,0
3471 LOCATE 10,4:PRINT "SPEED =>":LOCATE 10,17:COLOR 0,7:PRINT USING "##.#";OLDO
ASP:LOCATE 10,38:PRINT USING "###.#";NEWOASP:COLOR 7,0
3474 LOCATE 12,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
3477 C$=INKEY$:IF C$="" GOTO 3477
3480 IF C$="Y" OR C$="y" GOTO 3485 ELSE 3487
3485 TWO=TWO-1:GOTO 3330
3487 X%=X%+1:TWO=1:IF X%>2 GOTO 3770 ELSE 3300 '***TAKE SPEED CHANGE NEXT***
3500 CLS: PRINT TAB(20) "ESTIMATED PER VEHICLE SPEED CHANGES, MPH"
3510 GOSUB 32100:LOCATE 2,23: PRINT "SECTION LENGTH (MILES) = ", LENG
3515 LOCATE 20,33:COLOR 0,7:PRINT "MAIN LINE FLOW":COLOR 7,0
```

```
3520 LOCATE 3,1:PRINT "<= CURRENT TRAVEL SPEED => <= EXPECTED TRAVEL SPEED => <=
 TRAVEL SPEED CHANGE =>"
3530 LOCATE 5,1:PRINT "PEAK ":LOCATE 8,1:PRINT "OFF PEAK":LOCATE 11,1:PRINT "OVE
RALL"
3540 IF PEAKPCT>0 GOTO 3545 ELSE 3635
3545 IF R$="Y" AND TWO>0 THEN GOSUB 13210
3550 LOCATE 5,14:PRINT SPC(10):LOCATE 5,14,1:GOSUB 30000:OLDPKSP=NEWNUM
3555 IF NEWNUM=0 THEN LOCATE 18,30:COLOR 0,7:PRINT "SPEED CANNOT BE ZERO!";:COLO
R 7,0:BEEP:GOSUB 12500:LOCATE 18,30:PRINT SPC(25):GOTO 3550
3560 LOCATE 5,35:PRINT SPC(10):LOCATE 5,35,1:GOSUB 30000:NEWPKSP=NEWNUM
3565 IF NEWNUM=0 THEN LOCATE 18,30:COLOR 0,7:PRINT "SPEED CANNOT BE ZERO!";:COLO
R 7,0:BEEP:GOSUB 12500:LOCATE 18,30:PRINT SPC(25):GOTO 3560
3570 PKSPSAV=OLDPKSP-NEWPKSP:LOCATE 5,65:PRINT USING "###.#";-PKSPSAV
3580 LOCATE 8,14:PRINT SPC(10):LOCATE 8,14,1:GOSUB 30000:OLDOPSP=NEWNUM
3585 IF NEWNUM=0 THEN LOCATE 18,30:COLOR 0,7:PRINT "SPEED CANNOT BE ZERO!";:COLO
R 7,0:BEEP:GOSUB 12500:LOCATE 18,30:PRINT SPC(25):GOTO 3580
3590 LOCATE 8,35:PRINT SPC(10):LOCATE 8,35,1:GOSUB 30000:NEWOPSP=NEWNUM
3595 IF NEWNUM=0 THEN LOCATE 18,30:COLOR 0,7:PRINT "SPEED CANNOT BE ZERO!";:COLO
R 7,0:BEEP:GOSUB 12500:LOCATE 18,30:PRINT SPC(25):GOTO 3590
3600 OPSPSAV=OLDOPSP-NEWOPSP:LOCATE 8,65:PRINT USING "###.#";-OPSPSAV
3610 OLDOASP=(PEAKPCT*OLDPKSP/100) +(100-PEAKPCT)*OLDOPSP/100:LOCATE 11,13:PRINT
 USING "###.#";OLDOASP
3620 NEWOASP=(PEAKPCT*NEWPKSP/100) +(100-PEAKPCT)*NEWOPSP/100:LOCATE 11,34:PRINT
 USING "###.#";NEWOASP
3630 OASPSAV=(PEAKPCT*PKSPSAV/100) +(100-PEAKPCT)*OPSPSAV/100:LOCATE 11,65:PRINT
 USING "###.#";-OASPSAV :GOTO 3670
3635 IF R$="Y" AND TWO>0 THEN GOSUB 13240
3640 LOCATE 11,14:PRINT SPC(10):LOCATE 11,14,1:GOSUB 30000:OLDOASP=NEWNUM
3645 IF NEWNUM=0 THEN LOCATE 18,30:COLOR 0,7:PRINT "SPEED CANNOT BE ZERO!";:COLO
R 7,0:BEEP:GOSUB 12500:LOCATE 18,30:PRINT SPC(25):GOTO 3640
3650 LOCATE 11,35:PRINT SPC(10):LOCATE 11,35,1:GOSUB 30000:NEWOASP=NEWNUM
3655 IF NEWNUM=0 THEN LOCATE 18,30:COLOR 0,7:PRINT "SPEED CANNOT BE ZERO!";:COLO
R 7,0:BEEP:GOSUB 12500:LOCATE 18,30:PRINT SPC(25):GOTO 3650
3660 OASPSAV=OLDOASP-NEWOASP:LOCATE 11,65:PRINT USING "###.#";OASPSAV
3661 OLDPKSP=OLDOASP:NEWPKSP=NEWOASP:OLDOPSP=OLDOASP:NEWOPSP=NEWOASP
3670 OLDPKTT(1)=LENG*60/OLDPKSP:NEWPKTT(1)=LENG*60/NEWPKSP:PKTTSAV(1)=OLDPKTT(1)
-NEWPKTT(1)
3680 OLDOPTT(1)=LENG*60/OLDOPSP:NEWOPTT(1)=LENG*60/NEWOPSP:OPTTSAV(1)=OLDOPTT(1)
-NEWOPTT(1)
3690 OLDOATT(1)=(PEAKPCT*OLDPKTT(1)/100)+(100-PEAKPCT)*OLDOPTT(1)/100:NEWOATT(1)
=(PEAKPCT*NEWPKTT(1)/100)+(100-PEAKPCT)*NEWOPTT(1)/100:OATTSAV(1)=OLDOATT(1)-NEW
OATT(1)
3700 LOCATE 6,4:PRINT "TRAVEL TIME =>":LOCATE 9,4:PRINT "TRAVEL TIME =>":LOCATE
12,4:PRINT "TRAVEL TIME =>"
3705 COLOR 0,7
3710 LOCATE 6,20:PRINT USING "###.#";OLDPKTT(1):LOCATE 6,41:PRINT USING "###.#";
NEWPKTT(1):LOCATE 6,71:PRINT USING "###.#";PKTTSAV(1)
3720 LOCATE 9,20:PRINT USING "###.#";OLDOPTT(1):LOCATE 9,41:PRINT USING "###.#";
NEWOPTT(1):LOCATE 9,71:PRINT USING "###.#";OPTTSAV(1)
3730 LOCATE 12,20:PRINT USING "###.#";OLDOATT(1):LOCATE 12,41:PRINT USING "###.#
";NEWOATT(1):LOCATE 12,71:PRINT USING "###.#";OATTSAV(1)
3735 COLOR 7,0
3740 LOCATE 14,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
3750 C$=INKEY$:IF C$="" GOTO 3750
3760 IF C$="Y" OR C$="y" GOTO 3765 ELSE 3767
3765 TWO=TWO-1:GOTO 3540
3767 X%=2:TWO=1:GOTO 3300
3770 CLS:GOSUB 32100
3780 LOCATE 1,13:PRINT "VALUE OF TRAVEL TIME AND TIME SAVING BENEFIT ESTIMATE"
3790 LOCATE 4,25:PRINT "AUTOS" SPC(7) "LIGHT TRUCKS" SPC(3) "HEAVY TRUCKS" SPC(6
) "TOTAL"
3800 LOCATE 6,1:PRINT "VALUE OF TIME":LOCATE 7,1:PRINT "$ PER VEHICLE HOUR"
3810 LOCATE 11,1:PRINT "TIME SAVING BENEFIT":LOCATE 12,1:PRINT "PRESENT WORTH TO
TAL $"
3820 LOCATE 16,5:PRINT "INSTRUCTIONS":LOCATE 18,1:PRINT "Push F9 to decrease val
ue":LOCATE 19,1:PRINT "Push F10 to increase value":LOCATE 20,1:PRINT "Push <retu
rn> to go to MENU =>"
3830 LOCATE 16,55: PRINT "OPTIONS MENU":LOCATE 18,45:PRINT "1> change auto time
value":LOCATE 19,45:PRINT "2> change light truck time value":LOCATE 20,45:PRINT
"3> change heavy truck time value":LOCATE 21,45:PRINT "4> accept all values"
3835 UNITBEN(1)=OATTSAV(1)/60:UNITBEN(2)=OATTSAV(2)/60
3840 LOCATE 17+BENFLAG,40:PRINT SPC(3):REM RETURN HERE
3850 TYPE = 1:LOCATE 22,55,1:COLOR 0,7:PRINT "SELECT =>";:COLOR 7,0
3860 C$=INKEY$: IF C$="" GOTO 3860
3870 C=VAL(C$):ON C GOTO 3880,3890,3900,3910
3880 BENFLAG=1:LOCATE 18,40:COLOR 0,7:PRINT "==>":COLOR 7,0:GOSUB 35000 'ADJUST
AUTO TT BEN
3890 BENFLAG=2:LOCATE 19,40:COLOR 0,7:PRINT "==>":COLOR 7,0:GOSUB 35000 'ADJUST
LT TRK TT BEN
3900 BENFLAG=3:LOCATE 20,40:COLOR 0,7:PRINT "==>":COLOR 7,0:GOSUB 35000 'ADJUST
HV TRK TT BEN
3910 IF PONTFLAG = 1 GOTO 9200
3915 REM COMPUTE OP COST SAV BEN
3920 CLS:LOCATE 14,24:COLOR 0,7:PRINT "COMPUTE FUEL COST SAVINGS (Y,N)?":COLOR 7
,0
3930 C$=INKEY$:IF C$="" GOTO 3930
3940 IF C$="Y" OR C$="y" GOTO 3950 ELSE 8000 'TO ACC RED
3950 PF%(4)=1:CLS:GOSUB 32100
3960 LOCATE 1,15:PRINT "ESTIMATED PER VEHICLE FUEL COST SAVINGS, DOLLARS"
3970 LOCATE 2,3:PRINT "(These are savings IN ADDITION TO those due to speed (tra
vel time savings)"
3975 IF PF%(3)=1 GOTO 4020
3980 BEEP:LOCATE 4,5:COLOR 0,7:PRINT "TIME SAVINGS ESTIMATES ARE REQUIRED TO EST
IMATE FUEL COST SAVINGS!":COLOR 7,0:LOCATE 6,5:PRINT "HIT <RETURN> TO CONTINUE;
ANY OTHER KEY TO GO BACK TO TIME SAVINGS.";
3990 C$=INKEY$:IF C$=""GOTO 3990
4000 IF C$<>CHR$(13) THEN CLS:GOTO 3270
4010 LOCATE 4,1:PRINT SPC(79):LOCATE 6,1:PRINT SPC(79)
4020 LOCATE 4,5:COLOR 0,7:PRINT "TO ACCEPT DEFAULT VALUES, HIT <RETURN>; TO CHAN
GE, ENTER NEW VALUE.":COLOR 7,0:TWO=1
4030 LOCATE 6,1:PRINT "FLEET AVG.==>      FUEL ECONOMY (MPG)":LOCATE 8,1:PRINT "A
UTOS":LOCATE 8,20:PRINT USING "##.##";MPG(1):LOCATE 9,1:PRINT "LIGHT TRUCKS":LOC
ATE 9,20:PRINT USING "##.##";MPG(2):LOCATE 10,1:PRINT "HEAVY TRUCKS"
4035 LOCATE 6,45:PRINT "IDLING FUEL CONSUMPTION (GPH)":LOCATE 8,55:PRINT USING "
##.##";IDLE(1):LOCATE 9,55:PRINT USING "##.##";IDLE(2):LOCATE 10,55:PRINT USING
"##.##";IDLE(3)
4040 LOCATE 10,20:PRINT USING "##.##";MPG(3):LOCATE 12,1:PRINT "AVERAGE FUEL PRI
CE, DOLLARS PER GALLON":LOCATE 12,50:PRINT USING "##.##";GASPRI
4045 LOCATE 8,30:PRINT SPC(5):LOCATE 9,30:PRINT SPC(5):LOCATE 10,30:PRINT SPC(5)
:LOCATE 12,60:PRINT SPC(5):LOCATE 14,77:PRINT SPC(2):LOCATE 8,65:PRINT SPC(5):LO
CATE 9,65:PRINT SPC(5):LOCATE 10,65:PRINT SPC(5)
4050 LOCATE 14,1:PRINT "MAINLINE FLOW TYPE: [1=INTERRUPTED, <30 mph; 2=UNINTERRU
PTED, >30 mph] " FLOWCON%:FOR T=1 TO 3:BENEFIT(2,T)=0:NEXT T
4055 IF TWO>0 GOTO 4087
4060 LOCATE 8,30,1:GOSUB 30000:IF NEWNUM>0 THEN MPG(1)=NEWNUM
4065 LOCATE 9,30,1:GOSUB 30000:IF NEWNUM>0 THEN MPG(2)=NEWNUM
4070 LOCATE 10,30,1:GOSUB 30000:IF NEWNUM>0 THEN MPG(3)=NEWNUM
4072 LOCATE 8,65,1:GOSUB 30000:IF NEWNUM>0 THEN IDLE(1)=NEWNUM
4073 LOCATE 9,65,1:GOSUB 30000:IF NEWNUM>0 THEN IDLE(2)=NEWNUM
4074 LOCATE 10,65,1:GOSUB 30000:IF NEWNUM>0 THEN IDLE(3)=NEWNUM
```

```
4075 LOCATE 12,60,1:GOSUB 30000:IF NEWNUM>0 THEN GASPRI=NEWNUM
4080 LOCATE 14,77,1:GOSUB 30000:IF NEWNUM>0 THEN FLOWCON%=NEWNUM
4087 LOCATE 22,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0:TWO=0
4088 C$=INKEY$:IF C$="" GOTO 4088
4089 LOCATE 22,30:PRINT SPC(30)
4090 IF C$="Y" OR C$="y" GOTO 4030
4092 FADJ(1)=16.3/MPG(1):FADJ(2)=12.9/MPG(2):FADJ(3)=5.7/MPG(3)
4095 PCT(1)=PCAR/100:PCT(2)=PLTRUK/100:PCT(3)=PHTRUK/100
4097 IF LENG<=0 THEN X%=1:GOTO 4202
4100 IF PEAKPCT>0 GOTO 4110 ELSE VO(1)=OLDOASP:VO(2)=OLDOASP:VN(1)=NEWOASP:VN(2)
=NEWOASP:PPCT(1)=1:GOTO 4120
4110 VO(1)=OLDPKSP:VO(2)=OLDOPSP:VN(1)=NEWPKSP:VN(2)=NEWOPSP:PPCT(1)=PEAKPCT/100
:PPCT(2)=1-PPCT(1)
4120 FOR J=1 TO 2:NUM=(1/VO(J))-(1/VN(J)):DENOM=(VO(J)^2)-VN(J)^2
4125 FOR T=1 TO 3
4130 IF J=2 AND PEAKPCT=0 GOTO 4200
4160 IF FLOWCON%=1 GOTO 4180
4170 NUCOST=FADJ(T)*((FA(T)*NUM)+FB(T)*DENOM)*PCT(T)*PPCT(J)*(PWTFC/1000)*GASPRI
*LENG:GOTO 4190
4180 IF T=3 GOTO 4182 ELSE 4185
4182 NUCOST=FADJ(T)*PCT(T)*PPCT(J)*(PWTFC/1000)*GASPRI*LENG*((2000689!*NUM^2)+(1
4*NUM*DENOM))/((832.9*NUM)+(.016*DENOM)):GOTO 4190
4185 NUCOST=FADJ(T)*FC(T)*NUM*PCT(T)*PPCT(J)*(PWTFC/1000)*GASPRI*LENG
4190 BENEFIT(2,T)=BENEFIT(2,T)+NUCOST
4200 NEXT T:NEXT J:GOTO 4210
4202 REM SPOT LOCATIONS MAINLINE
4204 FOR T=1 TO 3:NUCOST=OATTSAV(1)*IDLE(T)*PCT(T)*GASPRI*PWTFC/60
4206 BENEFIT(2,T)=BENEFIT(2,T)+NUCOST:NEXT T
4210 REM HERE IS CROSS STREET FUEL SAVINGS
4213 FOR T=1 TO 3:NUCOST=OATTSAV(2)*IDLE(T)*PCT(T)*GASPRI*PWXTFC/60
4216 BENEFIT(2,T)=BENEFIT(2,T)+NUCOST:NEXT T
4220 LOCATE 19,27:PRINT"PRESENT WORTH FUEL SAVINGS":LOCATE 20,10:PRINT "AUTOS" S
PC(10) "LIGHT TRUCKS"SPC(6)"HEAVY TRUCKS"SPC(9) "TOTAL":FOR T=1 TO 3:LOCATE 21,6
+18*(T-1):PRINT USING "$$#########,";BENEFIT(2,T);:NEXT T
4230 LOCATE 21,60:PRINT USING "$$#########,";BENEFIT(2,1)+BENEFIT(2,2)+BENEFIT(2
,3):LOCATE 22,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
4240 C$=INKEY$:IF C$="" GOTO 4240
4250 LOCATE 22,30:PRINT SPC(30)
4260 IF C$="Y" OR C$="y" GOTO 4030 ELSE 4990
4990 IF PONTFLAG = 1 GOTO 9200
5000 REM ACC RED BEN
8000 CLS:LOCATE 14,24:COLOR 0,7:PRINT "COMPUTE ACCIDENT COST SAVINGS (Y,N)?";:CO
LOR 7,0
8010 C$=INKEY$:IF C$="" GOTO 8010
8020 IF C$="Y" OR C$="y" GOTO 8030 ELSE 9400
8030 REM ACC RED BEN ANAL
8040 PF%(5)=1:CLS:GOSUB 32100 'INPUT ACC YRS + TYPES
8047 IF R$="Y" THEN GOSUB 13250
8050 LOCATE 2,15,1:COLOR 0,7:PRINT "HOW MANY YEARS OF DATA WILL YOU ENTER (LESS
THAN 4)?";:COLOR 7,0
8060 LOCATE 2,66:PRINT SPC(8):LOCATE 2,68:GOSUB 30000: ACCYEARS=NEWNUM
8070 IF ACCYEARS<4 GOTO 8090 ELSE LOCATE 3,18:COLOR 0,7:PRINT "ACCIDENT HISTORY
MUST BE LESS THAN 4 YEARS!";:COLOR 7,0:GOSUB 12500:LOCATE 3,18:PRINT SPC(50):GOT
O 8050
8090 LOCATE 4,15:PRINT "ACCIDENTS WILL BE EXAMINED BY SEVERITY CLASS:"CHR$(13) "
   Fatals Accidents, Injury Accidents, and Property Damage [only] Accidents."
8100 LOCATE 7,1:PRINT "YOU MAY IDENTIFY UP TO 2 MUTUALLY EXCLUSIVE ACCIDENT TYPE
S FOR SEPARATE ANALYSIS":ACCLASS=0
8110 LOCATE 9,16,1:COLOR 0,7:PRINT "HOW MANY ADDITIONAL CLASSES DO YOU WANT TO U
SE?";:COLOR 7,0:PRINT SPC(5)
```

```
8120 LOCATE 9,64,1:GOSUB 30000:ACCLASS=NEWNUM
8130 IF ACCLASS=<0 GOTO 8132 ELSE 8135
8132 LOCATE 13,1:PRINT SPC(75):LOCATE 15,1:PRINT SPC(75):GOTO 8160
8135 IF ACCLASS=<2 GOTO 8140 ELSE LOCATE 19,24:COLOR 0,7:PRINT "CLASSES MUST BE
LESS THAN 3!";:COLOR 7,0:GOSUB 12500:LOCATE 19,20:PRINT SPC(50):GOTO 8110
8140 LOCATE 11,20:PRINT "CLASS NUMBER" SPC(20) "CLASS NAME"
8150 FOR I=1 TO ACCLASS:LOCATE 11+2*I,25:PRINT I, SPC(40):LOCATE 11+2*I,55:INPUT
 "",CLASS$(I):NEXT I
8160 LOCATE 19,30,1:COLOR 0,7:PRINT CHA$;:COLOR 7,0
8170 C$=INKEY$:IF C$="" GOTO 8170
8180 IF C$="Y" OR C$="y" GOTO 8050
8190 REM ACC HIST INPUT
8200 CLS:GOSUB 32100
8210 LOCATE 1,20:PRINT "FATAL" SPC(10) "INJURY" SPC(10) "PDO" SPC(10) "AADT"
8220 LOCATE 2,1:PRINT "YEAR" SPC(13) "ACCIDENTS" SPC(7) "ACCIDENTS" SPC(5) "ACCI
DENTS"
8230 FOR J=1 TO ACCYEARS
8240 TWO=1:LOCATE J*5,1:PRINT USING "##";J;:PRINT "    TOTAL"
8241 IF R$<>"Y" GOTO 8250
8242 LOCATE (J)*5,20:LOCATE 5*J,20:PRINT USING "####";FAT(J)
8244 LOCATE (J)*5,35:LOCATE 5*J,35:PRINT USING "####";INJ(J)
8246 LOCATE (J)*5,51:LOCATE 5*J,51:PRINT USING "####";PDO(J)
8248 LOCATE (J)*5,64:LOCATE 5*J,64:PRINT USING "######";TRAF(J):GOTO 8285
8250 LOCATE (J)*5,20:PRINT SPC(10):LOCATE 5*J,20,1:GOSUB 30000:FAT(J)=NEWNUM
8260 LOCATE (J)*5,35:PRINT SPC(10):LOCATE 5*J,35,1:GOSUB 30000:INJ(J)=NEWNUM
8270 LOCATE (J)*5,51:PRINT SPC(10):LOCATE 5*J,51,1:GOSUB 30000:PDO(J)=NEWNUM
8280 LOCATE (J)*5,64:PRINT SPC(10):LOCATE 5*J,64,1:GOSUB 30000:TRAF(J)=NEWNUM
8285 IF ACCLASS>0 GOTO 8290 ELSE 8287
8287 P=5*J:GOTO 8390
8290 FOR JJ=1 TO ACCLASS:PRINT
8295 REM INPUT YRLY ACCS BY SEV FOR CLASSES
8300 P=JJ+5*(J):LOCATE P,7:PRINT LEFT$(CLASS$(JJ),8)
8301 IF R$<>"Y" OR TWO=<0 GOTO 8310
8302 LOCATE P,20:LOCATE P,20:PRINT USING "####";CLASFAT(J,JJ)
8304 LOCATE P,35:LOCATE P,35:PRINT USING "####";CLASINJ(J,JJ)
8306 LOCATE P,51:LOCATE P,51:PRINT USING "####";CLASPDO(J,JJ):GOTO 8335
8310 LOCATE P,20:PRINT SPC(10):LOCATE P,20,1:GOSUB 30000:CLASFAT(J,JJ)=NEWNUM
8320 LOCATE P,35:PRINT SPC(10):LOCATE P,35,1:GOSUB 30000:CLASINJ(J,JJ)=NEWNUM
8330 LOCATE P,51:PRINT SPC(10):LOCATE P,51,1:GOSUB 30000:CLASPDO(J,JJ)=NEWNUM
8335 NEXT JJ
8340 YFAT=0:YINJ=0:YPDO=0
8350 FOR I=1 TO ACCLASS:YFAT=YFAT+CLASFAT(J,I):YINJ=YINJ+CLASINJ(J,I):YPDO=YPDO+
CLASPDO(J,I):NEXT I
8360 IF YFAT>FAT(J) OR YINJ>INJ(J) OR YPDO>PDO(J) GOTO 8370 ELSE 8390
8370 LOCATE P+2,25:COLOR 0,7:PRINT "CLASS TOTALS MUST NOT EXCEED TOTAL ACCIDENTS
!";:COLOR 7,0:GOSUB 12500:LOCATE P+2,25:PRINT SPC(50):GOTO 8250
8390 LOCATE P+2,30:COLOR 0,7:PRINT CHA$;:COLOR 7,0
8400 C$=INKEY$:IF C$="" GOTO 8400
8410 LOCATE P+2,30:PRINT SPC(21):IF C$="Y" OR C$="y" THEN TWO=TWO-1:GOTO 8250
8420 LOCATE P+2,30:PRINT SPC(20):NEXT J '***COMPUTE ACC RATES NOW***
8430 TVOL=0:TFAT=0:TINJ=0:TPDO=0:FOR J=1 TO ACCYEARS:TVOL=TVOL+TRAF(J):TFAT=TFAT
+FAT(J):TINJ=TINJ+INJ(J):TPDO=TPDO+PDO(J):NEXT J
8440 IF LENG<=0 THEN ALENG=1 ELSE ALENG=LENG
8450 EXPO=(TVOL/ACCYEARS)*365*ALENG
8460 FRAT=1E+08*TFAT/EXPO:IRAT=1000000!*TINJ/EXPO:PDORAT=1000000!*TPDO/EXPO
8470 TOTRAT=1000000!*(TFAT+TINJ+TPDO)/EXPO
8480 LOCATE P+2,1:PRINT "ACCIDENT RATE":IF LENG<=0 GOTO 8482 ELSE 8484
8482 LOCATE P+3,1:PRINT "PER MILLION VEHS":GOTO 8490
8484 LOCATE P+3,1:PRINT "PER MILLION VMT"
8490 LOCATE P+2,20:PRINT USING "##.###";FRAT:LOCATE P+2,35:PRINT USING "##.###";
```

```
IRAT:LOCATE P+2,51:PRINT USING "##.###";PDORAT:LOCATE P+2,60:PRINT USING "_T_O_T
_A_L_ _=_ ##.###";TOTRAT
8500 LOCATE P+3,19:PRINT "(100 MIL)"
8510 LOCATE P+4,25:COLOR 0,7:PRINT "PAUSING...PRESS ANY KEY TO CONTINUE";:COLOR
7,0
8515 C$=INKEY$:IF C$="" GOTO 8515
8520 CLS:GOSUB 32100
8525 REM INPUT RED FACT
8530 LOCATE 1,24:PRINT  "ENTER ACCIDENT REDUCTION FACTORS"
8540 PRINT "   NOTE: Reduction factors supplied for TOTAL accidents will be appl
ied to all  accident types UNLESS additional classes were defined for separate t
reatment."
8550 PRINT "   If additional classes are specified, reduction factors FOR THESE
CLASSES willbe applied to those accidents FIRST;TOTAL reduction factors will the
n be appliedto remaining accidents ONLY."
8560 PRINT "   A zero in the TOTAL COLUMN allows entry of separate reduction fac
tors by sev-erity class."
8570 LOCATE 10,36:PRINT "PERCENT OF ACCIDENTS ELIMINATED"
8580 LOCATE 12,1:PRINT "ACCIDENT TYPE" SPC(10) "TOTAL" SPC(10) "FATALS" SPC(10)
"INJURIES" SPC(10) "PDOs"
8590 LOCATE 14,5:PRINT "TOTAL":IF ACCLASS>0 GOTO 8600 ELSE 8605
8595 REM INPUT RED FACT CLASSES
8600 FOR JJ=1 TO ACCLASS:LOCATE 14+2*JJ,5:PRINT LEFT$(CLAS$(JJ),8):NEXT JJ
8605 IF R$="Y" THEN GOSUB 13630
8610 LOCATE 14,24:PRINT SPC(5):LOCATE 14,24,1:GOSUB 30000:REDTOT=NEWNUM
8614 IF (REDTOT>=0) AND (REDTOT<=100) GOTO 8620
8616 LOCATE 15,30:COLOR 0,7:PRINT "UNREASONABLE REDUCTION FACTOR!";:COLOR 7,0:GO
SUB 12500:LOCATE 15,30:PRINT SPC(35):GOTO 8610
8620 IF REDTOT>0 THEN LOCATE 14,39:PRINT SPC(40):GOTO 8660
8630 LOCATE 14,39:PRINT SPC(5):LOCATE 14,39,1:GOSUB 30000:REDFAT=NEWNUM
8640 LOCATE 14,55:PRINT SPC(5):LOCATE 14,55,1:GOSUB 30000:REDINJ=NEWNUM
8650 LOCATE 14,73:PRINT SPC(5):LOCATE 14,73,1:GOSUB 30000:REDPDO=NEWNUM
8652 IF (REDFAT>=0) AND (REDFAT<=100) GOTO 8654 ELSE 8658
8654 IF (REDINJ>=0) AND (REDINJ<=100) GOTO 8656 ELSE 8658
8656 IF (REDPDO>=0) AND (REDPDO<=100) GOTO 8660 ELSE 8658
8658 LOCATE 15,30:COLOR 0,7:PRINT "UNREASONABLE REDUCTION FACTOR!";:COLOR 7,0:GO
SUB 12500:LOCATE 15,30:PRINT SPC(35):GOTO 8610
8659 LOCATE 15,30:PRINT SPC(35):GOTO 8610
8660 IF ACCLASS>0 GOTO 8670 ELSE 8730
8670 FOR JJ=1 TO ACCLASS
8672 LOCATE 14+2*JJ,24:PRINT SPC(5):LOCATE 14+2*JJ,24,1:GOSUB 30000:REDTOTC(JJ)=
NEWNUM
8674 IF (REDTOTC(JJ)>=0) AND (REDTOTC(JJ)<=100) GOTO 8680
8676 LOCATE 15+2*JJ,30:COLOR 0,7:PRINT "UNREASONABLE REDUCTION FACTOR!";:COLOR 7
,0:GOSUB 12500:LOCATE 15+2*JJ,30:PRINT SPC(40):GOTO 8672
8680 IF REDTOTC(JJ)>0 THEN LOCATE 14+2*JJ,39:PRINT SPC(40):GOTO 8720
8690 LOCATE 14+2*JJ,39:PRINT SPC(5):LOCATE 14+2*JJ,39,1:GOSUB 30000:REDFATC(JJ)=
NEWNUM
8700 LOCATE 14+2*JJ,55:PRINT SPC(5):LOCATE 14+2*JJ,55,1:GOSUB 30000:REDINJC(JJ)=
NEWNUM
8710 LOCATE 14+2*JJ,73:PRINT SPC(5):LOCATE 14+2*JJ,73,1:GOSUB 30000:REDPDOC(JJ)=
NEWNUM
8712 IF (REDFATC(JJ)>=0) AND (REDFATC(JJ)<=100) GOTO 8714 ELSE 8718
8714 IF (REDINJC(JJ)>=0) AND (REDINJC(JJ)<=100) GOTO 8716 ELSE 8718
8716 IF (REDPDOC(JJ)>=0) AND (REDPDOC(JJ)<=100) GOTO 8720
8718 BEEP:LOCATE 15+2*JJ,30:COLOR 0,7:PRINT "UNREASONABLE REDUCTION FACTOR!";:CO
LOR 7,0:GOSUB 12500:LOCATE 15+2*JJ,30:PRINT SPC(40):GOTO 8672
8720 NEXT JJ:JJ=ACCLASS
8730 LOCATE 19+2*JJ,30,1:COLOR 0,7:PRINT CHA$;:COLOR 7,0
8740 C$=INKEY$:IF C$="" GOTO 8740
```

```
8750 IF C$="Y" OR C$="y" THEN LOCATE 19+2*JJ,25:PRINT SPC(35):GOTO 8610
8755 LOCATE 19+2*JJ,25:PRINT SPC(35) 'COMPUTE ACC REDUCT
8770 FOR JJ=1 TO ACCLASS:CYFAT(JJ)=0:CYINJ(JJ)=0:CYPDO(JJ)=0:NEXT JJ:TFAT=0:TINJ
=0:TPDO=0 'T=TOTALS, C=CLASSES
8780 FOR J=1 TO ACCYEARS 'SUM DATA YRS
8790 TFAT=TFAT+FAT(J):TINJ=TINJ+INJ(J):TPDO=TPDO+PDO(J)
8800 FOR JJ=1 TO ACCLASS:CYFAT(JJ)=CYFAT(JJ)+CLASFAT(J,JJ): CYINJ(JJ)=CYINJ(JJ)+
CLASINJ(J,JJ):CYPDO(JJ)=CYPDO(JJ)+CLASPDO(J,JJ)
8810 NEXT JJ:NEXT J
8820 ELFATC=0:ELINJC=0:ELPDOC=0
8830 FOR JJ=1 TO ACCLASS
8840 IF REDTOTC(JJ)<=0 GOTO 8850 ELSE 8880
8850 ELFATC=ELFATC + CYFAT(JJ)*(REDFATC(JJ)/100)
8860 ELINJC=ELINJC + CYINJ(JJ)*(REDINJC(JJ)/100)
8870 ELPDOC=ELPDOC + CYPDO(JJ)*(REDPDOC(JJ)/100):GOTO 8890
8880 ELFATC=ELFATC + CYFAT(JJ)*(REDTOTC(JJ)/100):ELINJC=ELINJC + CYINJ(JJ)*(REDT
OTC(JJ)/100):ELPDOC=ELPDOC +CYPDO(JJ)*(REDTOTC(JJ)/100)
8890 NEXT JJ
8900 REMFAT=TFAT-ELFATC:REMINJ=TINJ-ELINJC:REMPDO=TPDO-ELPDOC
8910 IF REDTOT<=0 GOTO 8920 ELSE 8930
8920 RREMFAT=REMFAT*(1-REDFAT/100):RREMINJ=REMINJ*(1-REDINJ/100):RREMPDO=REMPDO*
(1-REDPDO/100):GOTO 8940
8930 RREMFAT=REMFAT*(1-REDTOT/100):RREMINJ=REMINJ*(1-REDTOT/100):RREMPDO=REMPDO*
(1-REDTOT/100)
8940 REM COMPUTE AFTER ACC RATE
8950 NEWFRAT=1E+08*RREMFAT/EXPO:NEWIRAT=1000000!*RREMINJ/EXPO:NEWPRAT=1000000!*R
REMPDO/EXPO:JJ=ACCLASS
8960 LOCATE 16+2*JJ,39:PRINT USING "##.###";FRAT:LOCATE 16+2*JJ,55:PRINT USING "
##.###";IRAT:LOCATE 16+2*JJ,73:PRINT USING "##.###";PDORAT
8970 LOCATE 18+2*JJ,39:PRINT USING "##.###";NEWFRAT:LOCATE 18+2*JJ,55:PRINT USIN
G "##.###";NEWIRAT:LOCATE 18+2*JJ,73:PRINT USING "##.###";NEWPRAT
8980 LOCATE 15+2*JJ,40:PRINT "============ACCIDENT RATES============="
8990 LOCATE 16+2*JJ,1:PRINT "PRESENT RATE ==>":LOCATE 18+2*JJ,1:PRINT "PROJECTED
 RATE ==>                    "
9000 LOCATE 19+2*JJ,30,1:COLOR 0,7:PRINT CHA$;:COLOR 7,0
9010 C$=INKEY$:IF C$="" GOTO 9010
9020 IF C$="Y" OR C$="y" GOTO 9025 ELSE 9030
9025 LOCATE 19+2*JJ,25:PRINT SPC(35):LOCATE 18+2*JJ,20:COLOR 0,7:PRINT "MUST UPD
ATE ==>":COLOR 7,0:GOTO 8610
9030 SAVACF=ALENG*(FRAT-NEWFRAT)/1E+08 'SAVED FAT RATE*LENG
9040 SAVACI=ALENG*(IRAT-NEWIRAT)/1000000! 'SAVED INJ RATE*LENG
9050 SAVACP=ALENG*(PDORAT-NEWPRAT)/1000000! 'SAVED PDO RATE*LENG
9060 CLS:GOSUB 32100
9070 LOCATE 1,15:PRINT "COST OF ACCIDENTS AND VALUE OF ACCIDENTS AVOIDED"
9080 LOCATE 4,25:PRINT "FATALS" SPC(6) "INJURIES" SPC(7) "PROP. DAMAGE" SPC(6) "
TOTAL"
9090 LOCATE 6,1:PRINT "ACCIDENT COST":LOCATE 7,1:PRINT "$ PER ACCIDENT"
9100 LOCATE 11,1:PRINT "ACC. SAVING BENEFIT":LOCATE 12,1:PRINT "PRESENT WORTH TO
TAL $"
9110 LOCATE 16,5:PRINT "INSTRUCTIONS":LOCATE 18,1:PRINT "Push F9 to decrease val
ue":LOCATE 19,1:PRINT "Push F10 to increase value":LOCATE 20,1:PRINT "Push <retu
rn> to go to MENU =>"
9120 LOCATE 16,55: PRINT "OPTIONS MENU":LOCATE 18,45:PRINT "1> change fatal acc.
 cost":LOCATE 19,45:PRINT "2> change injury acc. cost":LOCATE 20,45:PRINT "3> ch
ange prop. damage acc. cost":LOCATE 21,45:PRINT "4> accept all values"
9130 LOCATE 17+BENFLAG,40:PRINT SPC(3):REM RETURN HERE
9140 TYPE = 3:LOCATE 22,55,1:COLOR 0,7:PRINT "SELECT =>";:COLOR 7,0
9150 C$=INKEY$: IF C$="" GOTO 9150
9160 C=VAL(C$):ON C GOTO 9170,9180,9190,9200
9170 BENFLAG=1:UNITBEN(1)=SAVACF: LOCATE 18,40:COLOR 0,7:PRINT "==>":COLOR 7,0:GO
```

```
SUB 35000 'ADJ FAT ACC COST
9180 BENFLAG=2:UNITBEN(1)=SAVACI:LOCATE 19,40:COLOR 0,7:PRINT "==>":COLOR 7,0:GO
SUB 35000 'ADJ INJ ACC COST
9190 BENFLAG=3:UNITBEN(1)=SAVACP:LOCATE 20,40:COLOR 0,7:PRINT "==>":COLOR 7,0:GO
SUB 35000 'ADJ PDO ACC COST
9200 'AGGR BEN, DO ECON EVAL
9400 TOP=DISCO*((1+DISCO)^NYEAR):BOT=((1+DISCO)^NYEAR)-1:CRF=TOP/BOT
9405 FOR TYPE=1 TO 3:PWBEN(TYPE)=0:NEXT TYPE:TOTPWBEN=0:TOTAWBEN=0
9410 FOR TYPE=1 TO 3:FOR BENFLAG=1 TO 3
9420 PWBEN(TYPE)=PWBEN(TYPE)+BENEFIT(TYPE,BENFLAG)
9430 NEXT BENFLAG
9440 AWBEN(TYPE)=CRF*PWBEN(TYPE)
9450 TOTPWBEN=TOTPWBEN+PWBEN(TYPE)
9460 TOTAWBEN=TOTAWBEN+AWBEN(TYPE)
9465 NEXT TYPE
9470 CLS:GOSUB 32100
9480 LOCATE 1,30:PRINT "ECONOMIC EVALUATION"
9490 LOCATE 3,1:PRINT "ITEM" SPC(18) "ANNUALIZED WORTH" SPC(7) "PRESENT WORTH" S
PC(4) "% TOTAL BENEFIT"
9500 LOCATE 5,1:PRINT "COST":LOCATE 5,24:PRINT USING "$$#########,";TOTAW
9510 LOCATE 5,47:PRINT USING "$$#########,";TOTPW
9520 LOCATE 7,1:PRINT "TIME SAVINGS":LOCATE 7,24:PRINT USING "$$#########,";AWBE
N(1)
9530 LOCATE 7,47:PRINT USING "$$#########,";PWBEN(1)
9535 LOCATE 7,67:PRINT USING "###.#_%";100*PWBEN(1)/TOTPWBEN
9540 LOCATE 9,1:PRINT "FUEL COST SAVINGS":LOCATE 9,24:PRINT USING "$$#########,"
;AWBEN(2)
9550 LOCATE 9,47:PRINT USING "$$#########,";PWBEN(2)
9555 LOCATE 9,67:PRINT USING "###.#_%";100*PWBEN(2)/TOTPWBEN
9560 LOCATE 11,1:PRINT "ACC. COST SAVINGS":LOCATE 11,24:PRINT USING "$$#########
,";AWBEN(3)
9570 LOCATE 11,47:PRINT USING "$$#########,";PWBEN(3)
9575 LOCATE 11,67:PRINT USING "###.#_%";100*PWBEN(3)/TOTPWBEN
9580 LOCATE 13,1:PRINT "TOTAL BENEFIT":LOCATE 13,24:PRINT USING "$$#########,";T
OTAWBEN
9590 LOCATE 13,47:PRINT USING "$$#########,";TOTPWBEN
9595 LOCATE 13,67:PRINT USING "###.#_%";100*(PWBEN(1)+PWBEN(2)+PWBEN(3))/TOTPWBE
N
9600 LOCATE 15,1:PRINT "NET WORTH":LOCATE 15,24:PRINT USING "$$#########,";TOTAW
BEN-TOTAW
9610 LOCATE 15,47:PRINT USING "$$#########,";TOTPWBEN-TOTPW
9620 LOCATE 17,1:PRINT "BENEFIT/COST RATIO":LOCATE 17,24:PRINT USING "####.##";T
OTAWBEN/TOTAW
9630 LOCATE 17,47:PRINT USING "####.##";TOTPWBEN/TOTPW 'BREAKEVEN ANAL
9645 IF (TOTPWBEN<>0) AND (TOTPW<>0) GOTO 9647 ELSE 9710 '9710=REVALUE
9647 LOCATE 18,27:PRINT "** SENSITIVITY ANALYSIS **"
9650 BENCH=100*(TOTPW-TOTPWBEN)/TOTPWBEN:COSTCH=100*(TOTPWBEN-TOTPW)/TOTPW
9655 IF BENCH>=0 GOTO 9660 ELSE 9670
9660 LOCATE 19,1:PRINT "With costs as shown, BENEFITS MUST INCREASE";:PRINT USIN
G " ####.##_%";BENCH;:PRINT " to bring B/C up to 1.0.":GOTO 9680
9670 LOCATE 19,1:PRINT "With costs as shown, benefits MAY DECREASE";:PRINT USING
 " ####.##_%";BENCH;:PRINT " to bring B/C down to 1.0."
9680 IF COSTCH>=0 GOTO 9690 ELSE 9700
9690 LOCATE 20,1:PRINT "With benefits as shown, costs MAY INCREASE";:PRINT USING
 " ####.##_%";COSTCH;:PRINT " to bring B/C down to 1.0.":GOTO 9710
9700 LOCATE 20,1:PRINT "With benefits as shown, COSTS MUST DECREASE";:PRINT USIN
G " ####.##_%";COSTCH;:PRINT " to bring B/C up to 1.0."
9710 LOCATE 21,26:COLOR 0,7:PRINT "RE-VALUE ANY BENEFITS (Y,N)?";:COLOR 7,0
9720 C$=INKEY$:IF C$="" GOTO 9720
9730 IF C$="Y" OR C$="y" GOTO 9740 ELSE 9840
```

```
9740 LOCATE 21,26:PRINT SPC(30)
9750 LOCATE 21,25:COLOR 0,7:PRINT "ENTER NUMBER TO SELECT CHANGE"
9760 LOCATE 22,25:PRINT "1 ==> TRAVEL TIME"
9770 LOCATE 23,25:PRINT "2 ==> OPERATING COST"
9780 LOCATE 24,25:PRINT "3 ==> ACCIDENT COST";:COLOR 7,0
9790 C$=INKEY$:IF C$=""GOTO 9790
9800 PONT=VAL(C$)
9805 PONTFLAG=1
9810 IF (PONT>0) AND (PONT<4) GOTO 9830 ELSE 9820
9820 FOR I=1 TO 3:LOCATE 20+I,25:PRINT SPC(35):NEXT I:GOTO 9710
9830 ON PONT GOTO 3770,3950,9060
9840 PONTFLAG=0
12000 REM TO FILER
12010 IF DOFILE$="Y" OR ADFILE$="Y" THEN GOSUB 27700
12020 CLS:GOSUB 32100:LOCATE 1,26:COLOR 0,7:PRINT "ENTER NUMBER OF NEXT ACTION":
COLOR 7,0
12030 LOCATE 3,15:PRINT "1> PRINT REPORT FOR CURRENT PROJECT."
12035 LOCATE 5,15:PRINT "2> REVIEW/REVISE THIS PROJECT."
12040 LOCATE 7,15:PRINT "3> EVALUATE ANOTHER PROJECT (WITHOUT PRINTING!)."
12050 LOCATE 9,15:PRINT "4> QUIT (WITHOUT PRINTING)."
12060 LOCATE 11,36,1:COLOR 0,7:PRINT "CHOICE =>";:COLOR 7,0
12070 C$=INKEY$:IF C$="" GOTO 12070
12075 IF C$ <>"1" AND C$<>"2" AND C$<>"3" AND C$<>"4" THEN BEEP:GOTO 12020
12080 PIK=VAL(C$)
12085 IF PIK = 2 THEN R$="Y" ELSE R$="N"
12090 ON PIK GOTO 37330,210,30,12100
12100 BEEP:LOCATE 12,26,1:COLOR 0,7:PRINT "VERIFY QUIT BY ENTERING `Q`";:COLOR 7
,0
12110 C$=INKEY$:IF C$="" GOTO 12110
12120 IF C$="Q" OR C$="q" THEN STOP ELSE 12020
12500 BEEP:FOR DELAY=1 TO 1000:WAT=DELAY*2:NEXT DELAY:RETURN:´>>DELAY<<
13000 REM DISPLAY FILED DATA
13005 REM PROJ DESCRIP
13010 LOCATE 3,21:PRINT LEFT$(PLACE$,58):LOCATE 5,20:PRINT LEFT$(ACT$,59):RETURN
 360
13020 REM LIFE & TFC
13030 LOCATE 7,30:PRINT USING "##.##";LENG:LOCATE 7,61:PRINT USING "####";NYEAR:
LOCATE 9,15:PRINT USING "######,";AADT:LOCATE 9,39:PRINT USING "####";YAADT:LOCA
TE 9,72:PRINT USING "####";YOP:RETURN 500
13040 REM TFC MIX
13050 LOCATE 11,41:PRINT USING "###.#";PCAR:LOCATE 13,41:PRINT USING "###.#";PLT
RUK:LOCATE 15,41:PRINT USING "###.#";PHTRUK:RETURN 590
13060 REM TFC GROW
13070 LOCATE 19,40:PRINT USING "###.#";ANGROW:LOCATE 22,34:PRINT USING "########
#";FUTAADT
13075 LOCATE 22,5:PRINT "ESTIMATED AADT, YEAR " YOP+NYEAR ": ":RETURN 700
13080 REM COST
13090 FOR L=1 TO 7:LOCATE 5+2*L,19:PRINT USING "####";YR(L):NEXT L
13100 FOR L=1 TO 7:LOCATE 5+2*L,28:PRINT USING "$$#########,";COST(L):NEXT L
13110 FOR L=1 TO 7:LOCATE 5+2*L,48:PRINT USING "###";LIFE(L):NEXT L:RETURN 1310
13120 REM VEH WTS
13130 LOCATE 6,45:PRINT USING "##.#";WCAR:LOCATE 8,45:PRINT USING "##.#";WLTRUK:
LOCATE 10,45:PRINT USING "##.#";WHTRUK:RETURN 2740
13140 REM PEAK %
13150 LOCATE 3,64:PRINT USING "##.#";PEAKPCT:RETURN 3170
13160 REM TIME SAV
13170 LOCATE 5,14:PRINT USING "###.#";OLDPKTT(X%):LOCATE 5,35:PRINT USING "###.#
";NEWPKTT(X%)
13180 PKTTSAV(X%)=OLDPKTT(X%)-NEWPKTT(X%):LOCATE 5,65:PRINT USING "###.#";PKTTSA
V(X%)
```

```
13190 LOCATE 7,14:PRINT USING "###.#";OLDOPTT(X%):LOCATE 7,35:PRINT USING "###.#
";NEWOPTT(X%):RETURN 3390
13200 LOCATE 9,14:PRINT USING "###.#";OLDOATT(X%):LOCATE 9,35:PRINT USING "###.#
";NEWOATT(X%):RETURN 3450
13210 LOCATE 5,14:PRINT USING "###.#";OLDPKSP:LOCATE 5,35:PRINT USING "###.#";NE
WPKSP
13220 PKSPSAV=OLDPKSP-NEWPKSP:LOCATE 5,65:PRINT USING "###.#";-PKSPSAV
13230 LOCATE 8,14:PRINT USING "###.#";OLDOPSP:LOCATE 8,35:PRINT USING "###.#";NE
WOPSP:RETURN 3600
13240 LOCATE 11,14:PRINT USING "###.#";OLDOASP:LOCATE 11,35:PRINT USING "###.#";
NEWOASP:RETURN 3660
13250 REM ACC DATA SPEC
13260 LOCATE 2,15,1:COLOR 0,7:PRINT "HOW MANY YEARS OF DATA WILL YOU ENTER (LESS
 THAN 4)?";:COLOR 7,0
13270 LOCATE 2,70:PRINT USING "#";ACCYEARS
13280 LOCATE 4,15:PRINT "ACCIDENTS WILL BE EXAMINED BY SEVERITY CLASS:"CHR$(13)
"Fatals Accidents, Injury Accidents, and Property Damage [only] Accidents."
13290 LOCATE 7,1:PRINT "YOU MAY IDENTIFY UP TO 2 MUTUALLY EXCLUSIVE ACCIDENT TYP
ES FOR SEPARATE ANALYSIS"
13300 LOCATE 9,16,1:COLOR 0,7:PRINT "HOW MANY ADDITIONAL CLASSES DO YOU WANT TO
USE?";:COLOR 7,0:PRINT SPC(5)
13310 LOCATE 9,64:PRINT USING "#";ACCLASS
13315 LOCATE 11,20:PRINT "CLASS NUMBER" SPC(20) "CLASS NAME"
13320 FOR I=1 TO ACCLASS:LOCATE 11+2*I,25:PRINT I, SPC(26);:PRINT CLASS$(I):NEXT
I
13330 RETURN 8160
13630 REM DISPLAY RED FACT
13650 LOCATE 14,24:PRINT USING "##.#";REDTOT
13660 LOCATE 14,39:PRINT USING "##.#";REDFAT
13670 LOCATE 14,55:PRINT USING "##.#";REDINJ
13680 LOCATE 14,73:PRINT USING "##.#";REDPDO
13690 IF ACCLASS>0 GOTO 13700 ELSE RETURN 8730
13700 FOR JJ=1 TO ACCLASS
13710 LOCATE 14+2*JJ,24:PRINT USING "##.#";REDTOTC(JJ)
13720 LOCATE 14+2*JJ,39:PRINT USING "##.#";REDFATC(JJ)
13730 LOCATE 14+2*JJ,55:PRINT USING "##.#";REDINJC(JJ)
13740 LOCATE 14+2*JJ,73:PRINT USING "##.#";REDPDOC(JJ)
13750 NEXT JJ:JJ=ACCLASS
13760 RETURN 8730
27499 STOP
27500 REM AUTOFILE SETUP DOFILES=NEW FILE;ADFILES=APPEND
27510 LOCATE 5,12,1:COLOR 0,7:PRINT "TYPE DRIVE LETTER (A,B,C...):FILE NAME, THE
N HIT <RETURN>";:COLOR 7,0:LOCATE 7,35,1:INPUT "",NAM$
27520 ON ERROR GOTO 27580 'NO FILE THIS NAME
27530 OPEN NAM$ FOR INPUT AS #1  'TEST EXISTENCE
27535 CLOSE #1
27540 BEEP:LOCATE 9,25:COLOR 0,7:PRINT "THIS FILE NAME ALREADY IN USE!";:COLOR 7
,0:LOCATE 11,25:COLOR 0,7:PRINT "C> CHANGE FILE NAME";:COLOR 7,0
27545 LOCATE 13,25:COLOR 0,7:PRINT "A> ADD NEW RECORD TO THIS FILE";:COLOR 7,0:L
OCATE 15,25:COLOR 0,7:PRINT "W> WRITE OVER THIS FILE (!)";:COLOR 7,0
27550 LOCATE 17,25,1:COLOR 0,7:PRINT "ENTER SELECTION (C,A,W)?";:COLOR 7,0
27560 C$=INKEY$:IF C$="" GOTO 27560
27570 IF C$="W" OR C$="w" GOTO 27590 ELSE GOTO 27575
27575 IF C$="C" OR C$="c" THEN RETURN 195 ELSE 27577
27577 ADFILE$="Y":ON ERROR GOTO 0:RETURN
27580 IF ERR=53 THEN DOFILE$="Y":RESUME 27590
27582 IF ERR=64 THEN PRINT SPC(27) "BAD FILE NAME: TRY AGAIN.":RESUME 27588
27584 IF ERR=68 OR ERR=71 THEN PRINT SPC(23) "DEVICE UNAVAILABLE; CHECK IT OUT!"
:RESUME 27588
27586 IF ERR=72 THEN PRINT SPC(26) "DISK MEDIA ERROR; TRY AGAIN.":RESUME 27588
```

```
27588 GOSUB 12500:GOTO 30
27590 ON ERROR GOTO 0  'CHANGE TO ERROR TRAP
27600 DOFILE$="Y":RETURN
27610 REM END AUTOFILER SETUP
27699 STOP
27700 REM AUTOFILER WRITE-TO-FILE
27702 IF DOFILE$="Y" GOTO 27705 ELSE 27707
27705 OPEN NAM$ FOR OUTPUT AS #1:GOTO 27708
27707 OPEN NAM$ FOR APPEND AS #1
27708 WRITE #1,LENG,PLACE$
27709 WRITE #1,ACT$
27710 WRITE #1,NYEAR,AADT,YAADT,YOP,PCAR,PLTRUK,PHTRUK,ANGROW,FUTAADT,DATE$,TIME
$,PWTFC,PWXTFC,XFRACT,YX,TOTCASHF,MPG(1),MPG(2),MPG(3),GASPRI,FLOWCON%,VNO%
27720 FOR L=1 TO 7:WRITE #1,YR(L),COST(L),LIFE(L),ANCOST(L),PWCOST(L):NEXT L
27730 WRITE #1, INFLATE,SHARE,WCAR,WLTRUK,WHTRUK,PEAKPCT,OLDPKSP,NEWPKSP,OLDOPSP
,NEWOPSP,OLDOASP,NEWOASP,ACCYEARS,ACCLASS,XAADT,SAVACF,SAVACI,SAVACP,CARSHARE,LT
SHARE,HTSHARE
27732 FOR X%=1 TO 2:WRITE #1,OLDPKTT(X%),NEWPKTT(X%),OLDOPTT(X%),NEWOPTT(X%),OLD
OATT(X%),NEWOATT(X%):NEXT X%
27735 FOR TYPE=1 TO 3:WRITE #1,PWBEN(TYPE),AWBEN(TYPE):FOR BENFLAG=1 TO 3:WRITE
#1,BENEFIT(TYPE,BENFLAG),BENVAL(TYPE,BENFLAG):NEXT BENFLAG:NEXT TYPE
27737 FOR I=1 TO (YOP-YX)+1+NYEAR:WRITE #1,CASHF(I):NEXT I
27740 FOR I = 1 TO ACCYEARS:WRITE #1,FAT(I),INJ(I),PDO(I),TRAF(I)
27750 FOR J=1 TO ACCLASS:WRITE #1,CLASFAT(I,J),CLASINJ(I,J),CLASPDO(I,J)
27760 NEXT J: NEXT I
27770 WRITE #1, DISCO,PF%(6),PF%(1),PF%(2),PF%(3),PF%(4),PF%(5),REDTOT,REDFAT,RE
DINJ,REDPDO,FRAT,IRAT,PDORAT,NEWFRAT,NEWIRAT,NEWPRAT
27775 WRITE #1,TOTPWBEN,TOTAWBEN
27780 FOR J=1 TO ACCLASS:WRITE #1,REDTOTC(J),REDFATC(J),REDINJC(J),REDPDOC(J),CL
AS$(J):NEXT J
27785 WRITE #1,TOTAW,TOTPW,IDLE(1),IDLE(2),IDLE(3)
27790 CLOSE #1:RETURN 12020
27999 STOP
28000 'COMPUTE PW & AWS
28010 AW=0:PW=0:'ANNUALIZE OVER OWN LIFE
28020 EXPO=(1+I)^N:TOP=I*EXPO:BOT=EXPO-1:AWO=PRINC*TOP/BOT
28040 'DISC ANN WORTH TO PW 1st YR
28050 IF FYEAR<YOP THEN FYEAR=YOP 'IF PAID BEFORE OPEN, TREAT AS AT OPEN
28060 N2=YOP+NYEAR-FYEAR:EXPO=(1+I)^N2:TOP=EXPO-1:BOT=I*EXPO
28080 PWPAID=AWO*TOP/BOT 'PRESENT WORTH WHEN PAID OR OPEN
28100 IF FYEAR<=YOP GOTO 28140
28110 'DISCOUNT PWs ITEMS PD POST-OPEN TO OPEN
28120 PWOPEN=PWPAID/(1+I)^(FYEAR-YOP):PWPAID=PWOPEN
28130 'ANNUALIZE OPEN PWs OVER LIFE
28140 EXPO=(1+I)^NYEAR:TOP=I*EXPO:BOT=EXPO-1:AW=PWPAID*TOP/BOT
28170 PW=PWPAID 'RETURN AW & PW
28180 RETURN
29999 STOP
30000 ' NON-NUMERIC INPUT TRAP
30010 C$="":Z%=0:DEC=0:W%=0:M%=0:NEG=1:R=0
30020 V$=INKEY$:IF V$="" GOTO 30020
30030 IF V$="\" GOTO 30040 ELSE 30070
30040 FOR BS=1 TO Z%+W%+DEC+R
30050 PRINT CHR$(29) CHR$(32) CHR$(29);
30060 NEXT BS: GOTO 30010
30070 IF V$=CHR$(13) GOTO 30450
30080 IF V$="-" THEN NEG=-1!:R=1:GOTO 30430
30090 IF DEC=1 THEN W%=W%+1:GOTO 30220
30100 IF V$="." THEN DEC=1 : GOTO 30430
30110 IF V$="0" GOTO 30330 ELSE 30120
```

```
30120 IF V$="1" GOTO 30340 ELSE 30130
30130 IF V$="2" GOTO 30350 ELSE 30140
30140 IF V$="3" GOTO 30360 ELSE 30150
30150 IF V$="4" GOTO 30370 ELSE 30160
30160 IF V$="5" GOTO 30380 ELSE 30170
30170 IF V$="6" GOTO 30390 ELSE 30180
30180 IF V$="7" GOTO 30400 ELSE 30190
30190 IF V$="8" GOTO 30410 ELSE 30200
30200 IF V$="9" GOTO 30420
30210 BEEP:BEEP:GOTO 30020
30220 IF V$="0" THEN D(W%)=0:GOTO 30430
30230 IF V$="1" THEN D(W%)=1:GOTO 30430
30240 IF V$="2" THEN D(W%)=2:GOTO 30430
30250 IF V$="3" THEN D(W%)=3:GOTO 30430
30260 IF V$="4" THEN D(W%)=4:GOTO 30430
30270 IF V$="5" THEN D(W%)=5:GOTO 30430
30280 IF V$="6" THEN D(W%)=6:GOTO 30430
30290 IF V$="7" THEN D(W%)=7:GOTO 30430
30300 IF V$="8" THEN D(W%)=8:GOTO 30430
30310 IF V$="9" THEN D(W%)=9:GOTO 30430
30320 W%=W%-1:GOTO 30210
30330 Z%=Z%+1:B(Z%)=0:GOTO 30430
30340 Z%=Z%+1:B(Z%)=1:GOTO 30430
30350 Z%=Z%+1:B(Z%)=2:GOTO 30430
30360 Z%=Z%+1:B(Z%)=3:GOTO 30430
30370 Z%=Z%+1:B(Z%)=4:GOTO 30430
30380 Z%=Z%+1:B(Z%)=5:GOTO 30430
30390 Z%=Z%+1:B(Z%)=6:GOTO 30430
30400 Z%=Z%+1:B(Z%)=7:GOTO 30430
30410 Z%=Z%+1:B(Z%)=8:GOTO 30430
30420 Z%=Z%+1:B(Z%)=9:GOTO 30430
30430 PRINT V$;
30440 GOTO 30020
30450 XC=0:YC=0
30460 FOR Y%=1 TO Z%
30470 XC=XC+(B(Y%)*10^(Z%-Y%))
30480 NEXT Y%
30490 FOR M%=1 TO W%
30500 YC=YC+D(M%)*10^(-M%)
30510 NEXT M%
30520 NEWNUM=(XC+YC)*NEG:NEG=1
30530 RETURN
30540 'RETURN MAIN
32099 STOP
32100 REM SCREEN FOOTER
32110 LOCATE 25,1:PRINT LEFT$(ACT$,20) " " LEFT$(PLACE$,30) " OPENS: ";:PRINT US
ING "####";YOP;:PRINT " LIFE: ";:PRINT USING "##_ _Y_R_S_.";NYEAR;
32120 RETURN 'END FOOTER
33999 STOP
34000 REM RETRIEVE FILE
34010 CLS:LOCATE 1,21,1:COLOR 0,7:PRINT "DO YOU WANT TO SEE DISKETTE DIRECTORY?"
;:COLOR 7,0
34020 LOCATE 3,21:PRINT "ENTER DRIVE NAME ==> A,B,C... <==FOR DIRECTORY;"
34030 LOCATE 5,21:PRINT "ENTER <RETURN> FOR NO DIRECTORY.":LOCATE 5,1,1
34050 C$=INKEY$:IF C$="" GOTO 34050
34080 IF C$=CHR$(13) GOTO 34110 ELSE FILES C$+":*.*"
34110 LOCATE 18,14,1:COLOR 0,7:PRINT "TYPE DRIVE NAME (A,B,C...):FILENAME and HI
T <RETURN>";:COLOR 7,0
34120 LOCATE 19,35,1:INPUT "",NAM$
34130 ON ERROR GOTO 34160
```

```
34140 OPEN NAM$ FOR INPUT AS #1
34150 GOTO 34202
34160 IF ERR=53 THEN PRINT "FILE (or drive) DOES NOT EXIST; TRY AGAIN":RESUME 34
190
34170 IF ERR=64 THEN PRINT "BAD FILE NAME; TRY AGAIN":RESUME 34190
34175 IF ERR=68 OR ERR=71 THEN PRINT "DEVICE UNAVAILABLE; CHECK IT OUT!":RESUME
34190
34180 IF ERR=72 THEN PRINT "DISK MEDIA ERROR; TRY AGAIN":RESUME 34190
34185 PRINT "ENCOUNTERED ERROR # " ERR " LINE " ERL:RESUME 34190
34190 GOSUB 12500:CLOSE #1:GOTO 34000
34202 IF EOF(1) THEN PRINT "THIS FILE IS EMPTY!":CLOSE #1:GOTO 34190
34205 INPUT #1,LENG,PLACE$
34207 INPUT #1,ACT$
34210 INPUT #1,NYEAR,AADT,YAADT,YOP,PCAR,PLTRUK,PHTRUK,ANGROW,FUTAADT,XDATE$,XTI
ME$,PWTFC,PWXTFC,XFRACT,YX,TOTCASHF,MPG(1),MPG(2),MPG(3),GASPRI,FLOWCON%,VNO%
34214 FOR L=1 TO 7:INPUT #1,YR(L),COST(L),LIFE(L),ANCOST(L),PWCOST(L):NEXT L
34216 INPUT #1,INFLATE,SHARE,WCAR,WLTRUK,WHTRUK,PEAKPCT,OLDPKSP,NEWPKSP,OLDOPSP,
NEWOPSP,OLDOASP,NEWOASP,ACCYEARS,ACCLASS,XAADT,SAVACF,SAVACI,SAVACP,CARSHARE,LTS
HARE,HTSHARE
34218 FOR X%=1 TO 2:INPUT #1,OLDPKTT(X%),NEWPKTT(X%),OLDOPTT(X%),NEWOPTT(X%),OLD
OATT(X%),NEWOATT(X%):NEXT X%
34220 FOR TYPE=1 TO 3:INPUT #1,PWBEN(TYPE),AWBEN(TYPE):FOR BENFLAG=1 TO 3:INPUT
#1,BENEFIT(TYPE,BENFLAG),BENVAL(TYPE,BENFLAG):NEXT BENFLAG:NEXT TYPE
34221 FOR I=1 TO (YOP-YX)+1+NYEAR:INPUT #1,CASHF(I):NEXT I
34223 FOR I = 1 TO ACCYEARS:INPUT #1,FAT(I),INJ(I),PDO(I),TRAF(I)
34224 FOR J=1 TO ACCLASS:INPUT #1,CLASFAT(I,J),CLASINJ(I,J),CLASPDO(I,J)
34225 NEXT J:NEXT I
34226 INPUT #1,DISCO,PF%(6),PF%(1),PF%(2),PF%(3),PF%(4),PF%(5),REDTOT,REDFAT,RED
INJ,REDPDO,FRAT,IRAT,PDORAT,NEWFRAT,NEWIRAT,NEWPRAT
34227 INPUT #1,TOTPWBEN,TOTAWBEN
34228 FOR J=1 TO ACCLASS:INPUT #1,REDTOTC(J),REDFATC(J),REDINJC(J),REDPDOC(J),CL
AS$(J):NEXT J
34229 INPUT #1,TOTAW,TOTPW,IDLE(1),IDLE(2),IDLE(3)
34231 FOR X%=1 TO 2:OPTTSAV(X%)=OLDOPTT(X%)-NEWOPTT(X%):PKTTSAV(X%)=OLDPKTT(X%)-
NEWPKTT(X%):OATTSAV(X%)=OLDOATT(X%)-NEWOATT(X%):NEXT X%
34235 LOCATE 21,1:PRINT PLACE$:PRINT ACT$:PRINT XDATE$:LOCATE 24,24,1:COLOR 0,7:
PRINT "IS THIS THE CORRECT FILE (Y,N)?";:COLOR 7,0
34240 C$=INKEY$:IF C$="" GOTO 34240
34250 IF C$="N" OR C$="n" GOTO 34253 ELSE 34260
34253 PLACE$="":ACT$="":NYEAR=0:YOP=0:YX=0
34255 LOCATE 21,1:PRINT SPC(79):LOCATE 22,1:PRINT SPC(79):LOCATE 23,1:PRINT SPC(
79):LOCATE 24,24:PRINT SPC(35):LOCATE 21,1:GOTO 34202
34260 R$="Y":CLOSE #1 :ON ERROR GOTO 0
34270 LOCATE 24,1:PRINT SPC(79):LOCATE 24,24,1 :COLOR 0,7:PRINT "SAVE CHANGES IN
 THIS FILE (Y,N)?";:COLOR 7,0
34280 C$=INKEY$:IF C$="" GOTO 34280
34290 IF C$="Y" OR C$="y" THEN ADFILE$="Y" ELSE ADFILE$="N" 'ADFILE$=APPEND FILE

34292 IF ADFILE$="N" GOTO 34294 ELSE VNO%=VNO%+1:PLACE$=PLACE$+" V."+STR$(VNO%)
34294 ON ERROR GOTO 0:TWO=1:RETURN 210 'END RETRIEVE
34999 STOP
35000 REM COMPUTE BENEFITS
35010 LOCATE 22,55:PRINT SPC(10)
35020 ON TYPE GOTO 35030,35040,35050
35030 KEY 9, "-0.25" + CHR$(13):KEY 10, "0.25" + CHR$(13):GOTO 35060
35040 KEY 9, "-0.005" + CHR$(13):KEY 10, "0.005" + CHR$(13):GOTO 35060
35050 KEY 9, "-500" + CHR$(13):KEY 10, "500" + CHR$(13):VEHMIX=1:GOTO 35100
35060 ON BENFLAG GOTO 35070, 35080,35090
35070 VEHMIX=PCAR/100:GOTO 35100
35080 VEHMIX=PLTRUK/100:GOTO 35100
```

```
35090 VEHMIX=PHTRUK/100
35100 LOCATE 7,9+14*BENFLAG:COLOR 0,7:PRINT USING "$$######.###";BENVAL(TYPE.BEN
FLAG)
35110 BENVAL(TYPE,BENFLAG)=BENVAL(TYPE,BENFLAG)+VALAD:LOCATE 7,9+14*BENFLAG:PRIN
T USING "$$######.###";BENVAL(TYPE,BENFLAG)
35112 IF TYPE>1 THEN UNITBEN(2)=0:GOTO 35120
35114 IF LENG=0 GOTO 35118
35116 BENEFIT(TYPE,BENFLAG)=PWTFC*VEHMIX*UNITBEN(1)*BENVAL(TYPE,BENFLAG)+PWXTFC*
VEHMIX*UNITBEN(2)*BENVAL(TYPE,BENFLAG):COL=8+15*BENFLAG+(1-BENFLAG):LOCATE 12,CO
L:GOTO 35130
35118 BENEFIT(TYPE,BENFLAG)=(1-XFRACT)*PWTFC*VEHMIX*UNITBEN(1)*BENVAL(TYPE,BENFL
AG)+XFRACT*PWXTFC*VEHMIX*UNITBEN(2)*BENVAL(TYPE,BENFLAG):COL=8+15*BENFLAG+(1-BEN
FLAG):LOCATE 12,COL:GOTO 35130
35120 BENEFIT(TYPE,BENFLAG)=PWTFC*VEHMIX*UNITBEN(1)*BENVAL(TYPE,BENFLAG):COL=8+1
5*BENFLAG+(1-BENFLAG):LOCATE 12,COL
35130 PRINT USING "$$##########,";BENEFIT(TYPE,BENFLAG)
35140 LOCATE 12,66:PRINT USING "$$##########,";BENEFIT(TYPE,1)+BENEFIT(TYPE,2)+B
ENEFIT(TYPE,3)
35150 COLOR 7,0
35160 REM
35180 LOCATE 23,50:PRINT "INCREMENT:          ":LOCATE 23,61
35190 GOSUB 30000:VALAD=NEWNUM
35200 IF VALAD<>0 GOTO 35100 ELSE 35210
35210 COLOR 7,0:KEY 9,"":KEY 10,"":LOCATE 23,55:PRINT SPC(20)
35220 ON TYPE GOTO 35230,35230,35240 '*** 2nd VAL MUST BE CHANGED***
35230 LOCATE 23,50:PRINT SPC(20):RETURN 3840
35240 LOCATE 23,50:PRINT SPC(20):RETURN 9130
35250 STOP 'END BENEFIT COMP
37330 REM PRINT COST DATA
37340 CLS:GOSUB 32100
37345 ON ERROR GOTO 37365
37350 LOCATE 5,10:PRINT "TO PRINT, WE WRITE FILE TO RAMDISK AND EXIT TO PRINT RO
UTINE."
37360 FSTR$="D:SJS.DSS":GOTO 37385
37365 RESUME 37370
37370 ON ERROR GOTO 37377:GOTO 37375
37375 CLOSE #2:FSTR$="C:SJS.DSS":GOTO 37385
37377 LOCATE 10,19:BEEP:COLOR 0,7:PRINT "ERROR NO. " ERR " ENCOUNTERED; UNABLE T
O PRINT!":COLOR 7,0:RESUME 37378
37378 LOCATE 12,25:PRINT "1 SAVE TO DISKETTE (if not saved before)":LOCATE 13,25
:PRINT "2 RETURN TO MAIN MENU":LOCATE 15,33,1:COLOR 0,7:PRINT "ENTER CHOICE: ";:
COLOR 7,0
37379 C$=INKEY$:IF C$=""GOTO 37379
37380 IF C$="2" GOTO 12020 ELSE LOCATE 17,18:COLOR 0,7:BEEP:PRINT "ENTER DRIVE L
ETTER FOR SAVING AS <ERRORFIL>:";:COLOR 7,0
37382 C$=INKEY$:IF C$=""GOTO 37382
37384 OPEN C$+":"+"ERRORFIL" FOR OUTPUT AS #1:ON ERROR GOTO 0:GOSUB 27708
37385 OPEN FSTR$ FOR OUTPUT AS #2
37390 ON ERROR GOTO 0
37395 IF XDATE$="" GOTO 37397 ELSE 37398
37397 XDATE$=DATE$:XTIME$=TIME$
37398 WRITE #2,LENG,PLACE$
37399 WRITE #2,ACT$
37400 WRITE #2,NYEAR,AADT,YAADT,YOP,PCAR,PLTRUK,PHTRUK,ANGROW,FUTAADT,XDATE$,XTI
ME$,PWTFC,PWXTFC,XFRACT,YX,TOTCASHF,MPG(1),MPG(2),MPG(3),GASPRI,FLOWCON%,VNO%
37410 FOR L=1 TO 7:WRITE #2,YR(L),COST(L),LIFE(L),ANCOST(L),PWCOST(L):NEXT L
37420 WRITE #2, INFLATE,SHARE,WCAR,WLTRUK,WHTRUK,PEAKPCT,OLDPKSP,NEWPKSP,OLDOPSP
,NEWOPSP,OLDOASP,NEWOASP,ACCYEARS,ACCLASS,XAADT,SAVACF,SAVACI,SAVACP,CARSHARE,LT
SHARE,HTSHARE
37430 FOR X%=1 TO 2:WRITE #2,OLDPKTT(X%),NEWPKTT(X%),OLDOPTT(X%),NEWOPTT(X%),OLD
```

```
OATT(X%),NEWOATT(X%):NEXT X%
37440 FOR TYPE=1 TO 3:WRITE #2,PWBEN(TYPE),AWBEN(TYPE):FOR BENFLAG=1 TO 3:WRITE
#2,BENEFIT(TYPE,BENFLAG),BENVAL(TYPE,BENFLAG):NEXT BENFLAG:NEXT TYPE
37450 FOR I=1 TO (YOP-YX)+1+NYEAR:WRITE #2,CASHF(I):NEXT I
37460 FOR I = 1 TO ACCYEARS:WRITE #2,FAT(I),INJ(I),PDO(I),TRAF(I)
37470 FOR J=1 TO ACCLASS:WRITE #2,CLASFAT(I,J),CLASINJ(I,J),CLASPDO(I,J)
37480 NEXT J: NEXT I
37490 WRITE #2, DISCO,PF%(6),PF%(1),PF%(2),PF%(3),PF%(4),PF%(5),REDTOT,REDFAT,RE
DINJ,REDPDO,FRAT,IRAT,PDORAT,NEWFRAT,NEWIRAT,NEWPRAT
37495 WRITE #2,TOTPWBEN,TOTAWBEN
37500 FOR J=1 TO ACCLASS:WRITE #2,REDTOTC(J),REDFATC(J),REDINJC(J),REDPDOC(J),CL
AS$(J):NEXT J
37510 WRITE #2,TOTAW,TOTPW,IDLE(1),IDLE(2),IDLE(3)
37520 CLOSE #2
37530 CHAIN "PRINTDSS.EXE"
40000 REM READ DEFAULT VALUES
40010 ON ERROR GOTO 41000
40020 OPEN "STARTUP.DSS" FOR INPUT AS #3
40030 INPUT #3,BENVAL(1,1),BENVAL(1,2),BENVAL(1,3),BENVAL(3,1),BENVAL(3,2),BENVA
L(3,3),GASPRI,MPG(1),MPG(2),MPG(3),IDLE(1),IDLE(2),IDLE(3)
40040 CLOSE #3
40050 ON ERROR GOTO 0:RETURN
41000 BEEP:LOCATE 10,10:COLOR 0,7:PRINT "CANNOT ACCESS DEFAULT VALUE FILE. ERROR
 #" ERR ". CONTINUING...";:COLOR 7,0:GOSUB 12500:LOCATE 10,10:PRINT SPC(65):RESU
ME 40050
```

APPENDIX C
PROGRAM LISTING

PRINTDSS

```
10 REM JLS D S S JUNE 24,1988  PRINTDSS ROUTINE
30 CLS:KEY OFF:CLEAR:ON KEY(1) GOSUB 35000:VNO%=1 'NEW PROJ RETURN
35 DIM CLASFAT(4,2),CLASINJ(4,2),CLASPDO(4,2),REDTOTC(2),REDFATC(2),REDINJC(2)
37 DIM REDPDOC(2),CYFAT(2),CYINJ(2),CYPDO(2),OLDPKTT(2),NEWPKTT(2),PKTTSAV(2),OL
DOPTT(2),NEWOPTT(2),OPTTSAV(2),NEWOATT(2),OATTSAV(2),OLDOATT(2),UNITBEN(2)
40 DIM CASHF(60),YR(7),COST(7),LIFE(7),SYR(7),TAG$(7),ANCOST(7),PWCOST(7),PPCT(2
),VO(2),VN(2),IDLE(3):IDLE(1)=.5:IDLE(2)=.7:IDLE(3)=1
45 DIM BENEFIT(3,3),BENVAL(3,3),CLAS$(2),FAT(4),INJ(4),PDO(4),TRAF(4),PWBEN(3),A
WBEN(3),D(20),B(20),PF%(6),MPG(3),FLOWCON%(2),FA(3),FB(3),FC(3),FADJ(3),PCT(3)
55 R$="N":TWO=0:FOR I=1 TO 6:PF%(I)=0:NEXT I:MPG(1)=16.3:MPG(2)=12.9:MPG(3)=5.7:
GASPRI=1.25:FLOWCON%(1)=2:FLOWCON%(2)=2:FA(1)=497:FA(2)=832.19:FA(3)=1986.88:FB(
1)=.0139:FB(2)=.016:FB(3)=.0178:FC(1)=601.69:FC(2)=1006.95:FC(3)=1200
60 CHA$="CHANGE INPUTS (Y,N)?":LOCATE 1,10:PRINT "DECISION SUPPORT SYSTEM FOR TR
ANSPORTATION PROJECT ANALYSIS"
70 LOCATE 2,25:PRINT "NORTHWESTERN UNIVERSITY  1985":LOCATE 4,19:PRINT"THIS ROUT
INE PRINTS YOUR HARD COPY REPORT."
75 LOCATE 6,15:COLOR 0,7:BEEP:PRINT "TURN ON PRINTER AND POSITION PAPER TO TOP O
F PAGE!":LOCATE 8,28,1:PRINT "PRESS ANY KEY WHEN READY.";:COLOR 7,0
80 C$=INKEY$:IF C$="" GOTO 80
90 LOCATE 6,1:PRINT SPC(79):LOCATE 8,1:PRINT SPC(79):LOCATE 8,19:PRINT "RELAX UN
TIL WE RETURN TO THE MAIN ROUTINE."
100 PLAY "MB T250 O2 L8;ABO3CDEFGFEDCO2BA"
110 ON ERROR GOTO 120:GOTO 200
120 CLS:'IF ERR=68 OR ERR=57 OR ERR=25 OR ERR=24 GOTO 140
130 BEEP:LOCATE 2,24:COLOR 0,7:PRINT "ERROR NO. " ERR " HAS BEEN DETECTED!":COLO
R 7,0
140 BEEP:LOCATE 4,27:COLOR 0,7:PRINT "PLEASE CHECK YOUR PRINTER!":COLOR 7,0:RESU
ME 145
145 LOCATE 6,20:PRINT "1 RE-START PRINT (SET FORM TO TOP OF PAGE FIRST!)"
147 LOCATE 8,20:PRINT "2 RETURN TO MAIN PROGRAM (TERMINATE PRINT JOB)"
148 LOCATE 10,33,1:COLOR 0,7:PRINT "ENTER CHOICE: ";:COLOR 7,0
160 C$=INKEY$:IF C$="" GOTO 160
195 IF C$="2" THEN CHAIN "MAINDSS.EXE" ELSE 10
200 REM FIND THE RAMDISK FILE
205 REM
210 ON ERROR GOTO 230
220 FSTR$="D:SJS.DSS":GOTO 300
230 RESUME 235
235 ON ERROR GOTO 250:GOTO 240
240 CLOSE #1:FSTR$="C:SJS.DSS":GOTO 300
250 LOCATE 10,10:BEEP:COLOR 0,7:PRINT "ERROR NO. " ERR "ENCOUNTERED; UNABLE TO R
ETRIEVE THE PRINT FILE!":COLOR 7,0:GOSUB 12500:RESUME 260
260 CHAIN "MAINDSS.EXE"
300 PRINT FSTR$:OPEN FSTR$ FOR INPUT AS #1
304 INPUT #1,LENG,PLACE$
306 INPUT #1,ACT$
310 INPUT #1,NYEAR,AADT,YAADT,YOP,PCAR,PLTRUK,PHTRUK,ANGROW,FUTAADT,XDATE$,XTIME
$,PWTFC,PWXTFC,XFRACT,YX,TOTCASHF,MPG(1),MPG(2),MPG(3),GASPRI,FLOWCON%,VNO%
320 FOR L=1 TO 7:INPUT #1,YR(L),COST(L),LIFE(L),ANCOST(L),PWCOST(L):NEXT L
330 INPUT #1, INFLATE,SHARE,WCAR,WLTRUK,WHTRUK,PEAKPCT,OLDPKSP,NEWPKSP,OLDOPSP,N
EWOPSP,OLDOASP,NEWOASP,ACCYEARS,ACCLASS,XAADT,SAVACF,SAVACI,SAVACP,CARSHARE,LTSH
ARE,HTSHARE
340 FOR X%=1 TO 2:INPUT #1,OLDPKTT(X%),NEWPKTT(X%),OLDOPTT(X%),NEWOPTT(X%),OLDOA
TT(X%),NEWOATT(X%):NEXT X%
350 FOR TYPE=1 TO 3:INPUT #1,PWBEN(TYPE),AWBEN(TYPE):FOR BENFLAG=1 TO 3:INPUT #1
,BENEFIT(TYPE,BENFLAG),BENVAL(TYPE,BENFLAG):NEXT BENFLAG:NEXT TYPE
360 FOR I=1 TO (YOP-YX)+1+NYEAR:INPUT #1,CASHF(I):NEXT I
370 FOR I = 1 TO ACCYEARS:INPUT #1,FAT(I),INJ(I),PDO(I),TRAF(I)
380 FOR J=1 TO ACCLASS:INPUT #1,CLASFAT(I,J),CLASINJ(I,J),CLASPDO(I,J)
390 NEXT J: NEXT I
```

```
400 INPUT #1, DISCO,PF%(6),PF%(1),PF%(2),PF%(3),PF%(4),PF%(5),REDTOT,REDFAT,REDI
NJ,REDPDO,FRAT,IRAT,PDORAT,NEWFRAT,NEWIRAT,NEWPRAT,TOTPWBEN,TOTAWBEN
410 FOR J=1 TO ACCLASS: INPUT #1,REDTOTC(J),REDFATC(J),REDINJC(J),REDPDOC(J),CLAS
$(J):NEXT J
420 INPUT #1,TOTAW,TOTPW,IDLE(1),IDLE(2),IDLE(3)
430 FOR X%=1 TO 2: OPTTSAV(X%)=OLDOPTT(X%)-NEWOPTT(X%):PKTTSAV(X%)=OLDPKTT(X%)-N
EWPKTT(X%):OATTSAV(X%)=OLDOATT(X%)-NEWOATT(X%):NEXT X%
500 GOSUB 32100
510 ON ERROR GOTO 120:REM ****HERE COMES THE PRINT ROUTINE****
515 KEY(1) ON :LOCATE 6,25:PRINT "HIT F1 TO INTERRUPT PRINTING"
520 LPRINT TAB(15) CHR$(14) "PROJECT EVALUATION REPORT" CHR$(13)
530 LPRINT CHR$(27) "E" "DATE: " CHR$(27) "F" XDATE$ SPC(20) CHR$(27) "E" "TIME:
 " CHR$(27) "F" XTIME$ CHR$(13)
540 LPRINT CHR$(27) "E" "PROJECT LOCATION: " CHR$(27) "F" LEFT$(PLACE$,60) CHR$(
13)
550 LPRINT CHR$(27) "E" "PROPOSED ACTION: " CHR$(27) "F" LEFT$(ACT$,60) CHR$(13)

560 FOR P=1 TO 79:LPRINT CHR$(61);:NEXT P:LPRINT CHR$(13)
570 LPRINT CHR$(27) "E" TAB(23) "PROJECT CHARACTERISTICS AND COSTS" CHR$(27) "F"
 CHR$(13)
580 LPRINT CHR$(27) "E" "LENGTH, MILES: " CHR$(27) "F";:LPRINT USING "###.##";LE
NG;
590 LPRINT CHR$(27) "E" "     PROJECT LIFE, YRS: " CHR$(27) "F";:LPRINT USING "##
";NYEAR;:LPRINT SPC(8);
600 LPRINT USING "####";YAADT;:LPRINT CHR$(27) "E" " AADT: "CHR$(27) "F";:LPRINT
 USING "######,";AADT
610 LPRINT CHR$(13) CHR$(27) "E" "OPENS IN: " CHR$(27) "F";:LPRINT USING "####";
YOP;
620 LPRINT CHR$(27) "E" "    ANNUAL TRAFFIC GROWTH: " CHR$(27) "F";:LPRINT USING
"###.##_%";ANGROW;
630 LPRINT CHR$(27) "E" "    END YEAR VOLUME: " CHR$(27) "F";:LPRINT USING "#####
#,";FUTAADT
640 LPRINT CHR$(27) "E" TAB(50) "DISCOUNT RATE: " CHR$(27) "F";:LPRINT USING "##
.#_%";DISCO*100
650 LPRINT CHR$(13) CHR$(27) "E" "COST ITEM" SPC(6) "FIRST" SPC(5) "ITEM" SPC(5)
 "ACTUAL" SPC(7)"ANNUALIZED" SPC(7) "PRESENT"
660 LPRINT SPC(13) "YEAR PAID" SPC(3) "LIFE" SPC(5) "AMOUNT" SPC(10) "WORTH" SPC
(10) "WORTH" CHR$(27) "F"
670 TAG$(1)="DESIGN":TAG$(2)="R-O-W":TAG$(3)="CONSTRUCTION":TAG$(4)="MAINTENANCE
":TAG$(5)="OPERATION":TAG$(6)="OTHER":TAG$(7)="OTHER"
680 FOR J= 1 TO 7
690 LPRINT CHR$(13) CHR$(27) "E" TAG$(J) CHR$(27) "F";
700 LPRINT TAB(18);:LPRINT USING "####";YR(J);
710 LPRINT TAB(27);:LPRINT USING "####";LIFE(J);
720 LPRINT TAB(32);:LPRINT USING "$$#########,";COST(J);
730 LPRINT TAB(46);:LPRINT USING "$$#########,";ANCOST(J);
740 LPRINT TAB(62);:LPRINT USING "$$#########,";PWCOST(J)
750 NEXT J
760 LPRINT CHR$(13) CHR$(27) "E" "TOTAL" CHR$(27) "F" TAB(46);
770 LPRINT USING "$$#########,";TOTAW;
780 LPRINT TAB(62);:LPRINT USING "$$#########,";TOTPW
790 LNCT=19
800 REM INSERT PAPER SPACER HERE; AFTER PRINTING, RETURN TO 12020
810 IF PF%(6)>0 THEN GOTO 820 ELSE 1030
820 REM PRINT CASH FLOWS
830 LPRINT:LPRINT:LPRINT CHR$(27) "E" TAB(31) "CASH FLOW ANALYSIS"
840 LPRINT SPC(14) "INFLATION RATE: " CHR$(27) "F";
850 LPRINT USING "###.#_%";INFLATE*100;
860 LPRINT SPC(15) CHR$(27) "E" "`LOCAL` SHARE: " CHR$(27) "F";
870 LPRINT USING "###.#_%";SHARE*100;:LPRINT CHR$(13)
```

```
880 LPRINT CHR$(27) "E" SPC(4) "YEAR" SPC(7) "INFLATED COST" SPC(8) "`LOCAL` SHA
RE" SPC(8) "OTHER SHARE" CHR$(27) "F" CHR$(13)
890 FOR I=1 TO (YOP-YX)+1+NYEAR
900 LNCT=LNCT-1:IF LNCT<0 THEN LPRINT:LPRINT:LPRINT:LPRINT PLACE$:LPRINT ACT$:LP
RINT:LPRINT CHR$(27) "E" SPC(4) "YEAR" SPC(7) "INFLATED COST" SPC(8) "`LOCAL` SH
ARE" SPC(8) "OTHER SHARE" CHR$(27) "F" CHR$(13):LNCT=57
910 LPRINT SPC(4);:LPRINT USING "####";YX-1+I;
920 LPRINT SPC(7);:LPRINT USING "$$#########,";CASHF(I);
930 LPRINT SPC(8);:LPRINT USING "$$#########,";CASHF(I)*SHARE;
940 LPRINT SPC(8);:LPRINT USING "$$#########,";CASHF(I)*(1-SHARE)
950 NEXT I
960 LNCT=LNCT-1:IF LNCT<0 THEN LPRINT:LPRINT:LPRINT:LPRINT:LPRINT PLACE$:LPRINT
ACT$:LPRINT:LPRINT CHR$(27) "E" SPC(4) "YEAR" SPC(7) "INFLATED COST" SPC(8) "`LO
CAL` SHARE" SPC(8) "OTHER SHARE" CHR$(27) "F" CHR$(13):LNCT=57
970 LPRINT CHR$(13) CHR$(27) "E" SPC(3)"TOTAL" CHR$(27) "F";
980 LPRINT SPC(7);:LPRINT USING "$$#########,";TOTCASHF;
990 LPRINT SPC(8);:LPRINT USING "$$#########,";TOTCASHF*SHARE;
1000 LPRINT SPC(8);:LPRINT USING "$$#########,";TOTCASHF*(1-SHARE)
1010 LNCT=LNCT-1:IF LNCT<0 THEN LPRINT :LPRINT:LPRINT PLACE$:LPRINT ACT$:LPRINT
:LNCT=57
1020 'END CASHFLOW PRINT
1030 IF PF%(1)>0 THEN GOTO 1040 ELSE 1160
1040 'PRN COST ALLOC
1050 IF LNCT<11 THEN FOR I=1 TO LNCT+1:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:L
PRINT:LNCT=57
1060 LPRINT:LPRINT CHR$(27) "E" SPC(28) "ECONOMIC COST ALLOCATION"
1070 LPRINT :LPRINT "VEHICLE TYPE" SPC(12) "PERCENT" SPC(8) "COST WEIGHT" TAB(57
);"WTD. COST PER"
1080 LPRINT SPC(23) "OF TRAFFIC" SPC(5) "(E.G., PCE`S)";
1090 IF LFLAG=0 GOTO 1110
1100 LPRINT TAB(60) "VEHICLE":GOTO 1120
1110 LPRINT TAB(58) "VEHICLE-MILE"
1120 LPRINT:LPRINT:LPRINT "AUTOS" CHR$(27) "F" TAB(25);:LPRINT USING "###.#_%";P
CAR;:LPRINT TAB(45);:LPRINT USING "##.#";WCAR;:LPRINT TAB(55);:LPRINT USING "$$#
###.####";CARSHARE
1130 LPRINT:LPRINT CHR$(27) "E" "LIGHT TRUCKS" CHR$(27) "F" TAB(26);:LPRINT USIN
G "###.#_%";PLTRUK;:LPRINT TAB(46);:LPRINT USING "##.#";WLTRUK;:LPRINT TAB(56);:
LPRINT USING "$$####.####";LTSHARE
1140 LPRINT:LPRINT CHR$(27) "E" "HEAVY TRUCKS" CHR$(27) "F" TAB(26);:LPRINT USIN
G "###.#_%";PHTRUK;:LPRINT TAB(46);:LPRINT USING "##.#";WHTRUK;:LPRINT TAB(56);:
LPRINT USING "$$####.####";HTSHARE
1150 LPRINT :LPRINT:LNCT=LNCT-13
1160 IF PF%(2)>0 THEN GOTO 1170 ELSE 1210
1170 'PRN BEN INTRO
1180 IF LNCT<5 THEN FOR I=1 TO LNCT+1:LPRINT :NEXT I:LPRINT PLACE$:LPRINT ACT$:L
PRINT:LNCT=57
1190 LPRINT CHR$(27) "E" TAB(32) "BENEFIT ANALYSIS" CHR$(13)
1200 LPRINT TAB(25) "PEAK PERIOD PERCENTAGE: " CHR$(27) "F";:LPRINT USING "##.#_
%";PEAKPCT:LPRINT:LPRINT TAB(25) CHR$(27) "E" "CROSS STREET TRAFFIC: " CHR$(27)
"F";:LPRINT USING "######,";XAADT:LNCT=LNCT-5
1210 IF PF%(3)>0 THEN GOTO 1220 ELSE 1390
1220 X%=1:'TIME SAV PRN
1230 IF LNCT<13 THEN FOR I=1 TO LNCT+1:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:L
PRINT:LNCT=57
1240 LNCT=LNCT-13:LPRINT CHR$(13) CHR$(27) "E" TAB(29) "TIME SAVINGS BENEFITS":I
F X%=1 GOTO 1250 ELSE LPRINT TAB(29) "CROSS STREET TRAFFIC" CHR$(13):GOTO 1260
1250 LPRINT TAB(31) "MAINLINE TRAFFIC" CHR$(13)
1260 LPRINT TAB(10) "CURRENT TRAVEL TIMES" TAB(35) "EXPECTED TRAVEL TIMES" TAB(6
1) "TRAVEL TIME SAVINGS"
1270 LPRINT:LPRINT "PEAK" TAB(15) CHR$(27) "F";:LPRINT USING "###.#";OLDPKTT(X%)
```

```
;:LPRINT TAB(41);:LPRINT USING "###.#";NEWPKTT(X%);:LPRINT TAB(68);:LPRINT USING
 "###.#";PKTTSAV(X%)
1280 IF LFLAG=0 AND X%=1 GOTO 1290 ELSE 1300
1290 LPRINT CHR$(27) "E" "  SPEED" CHR$(27) "F" TAB(15);:LPRINT USING "##.#";LEN
G*60/OLDPKTT(X%);:LPRINT TAB(40);:LPRINT USING "##.#";LENG*60/NEWPKTT(X%)
1300 LPRINT:LPRINT CHR$(27) "E" "OFF PEAK" TAB(16) CHR$(27) "F";:LPRINT USING "#
##.#";OLDOPTT(X%);:LPRINT TAB(42);:LPRINT USING "###.#";NEWOPTT(X%);:LPRINT TAB(
69);:LPRINT USING "###.#";OPTTSAV(X%)
1310 IF LFLAG=0 AND X%=1 GOTO 1320 ELSE 1330
1320 LPRINT CHR$(27) "E" "  SPEED" CHR$(27) "F" TAB(15);:LPRINT USING "##.#";LEN
G*60/OLDOPTT(X%);:LPRINT TAB(40);:LPRINT USING "##.#";LENG*60/NEWOPTT(X%)
1330 LPRINT:LPRINT CHR$(27) "E" "OVERALL" CHR$(27) "F" TAB(17);:LPRINT USING "##
#.#";OLDOATT(X%);:LPRINT TAB(42);:LPRINT USING "###.#";NEWOATT(X%);:LPRINT TAB(6
9);:LPRINT USING "###.#";OATTSAV(X%)
1340 IF X%=2 GOTO 1350 ELSE X%=2:GOTO 1230
1350 IF LNCT<6 THEN FOR I=1 TO LNCT+1:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:LP
RINT:LNCT=57
1360 LNCT=LNCT-6:LPRINT:LPRINT CHR$(27) "E" TAB(40) "AUTOS" TAB(50) "LIGHT TRUCK
S" TAB(65) "HEAVY TRUCKS":LPRINT:LPRINT "VALUE OF TIME" CHR$(27) "F" TAB(37);:LP
RINT USING "$$###.##";BENVAL(1,1);:LPRINT TAB(50);
1370 LPRINT USING "$$###.##";BENVAL(1,2);:LPRINT TAB(65);:LPRINT USING "$$###.##
";BENVAL(1,3):LPRINT
1380 LPRINT CHR$(27) "E" "PRESENT WORTH OF TIME SAVING" CHR$(27) "F";:LPRINT TAB
(35);:LPRINT USING"$$#########,";BENEFIT(1,1);:LPRINT TAB(47);:LPRINT USING"$$###
#####,";BENEFIT(1,2);:LPRINT TAB(62);:LPRINT USING"$$#########,";BENEFIT(1,3):LPR
INT
1390 IF PF%(4)>0 THEN GOTO 1400 ELSE 1500 'FUEL COST SAVE
1400 IF LNCT<12 THEN FOR I=1 TO LNCT+6:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:L
NCT=57
1420 LNCT=LNCT-12:LPRINT:LPRINT
1430 LPRINT CHR$(27) "E" SPC(32) "FUEL COST SAVINGS" CHR$(13) CHR$(13)SPC(35) "A
UTOS" SPC(10) "LIGHT TRUCKS   HEAVY TRUCKS" CHR$(13) CHR$(13) "FLEET AVERAGE MIL
ES/GALLON:" CHR$(27) "F" TAB(37);
1440 LPRINT USING "###.##";MPG(1);:LPRINT TAB(52);:LPRINT USING "###.##";MPG(2);
:LPRINT TAB(67);:LPRINT USING "###.##";MPG(3):LPRINT
1445 LPRINT CHR$(27) "E" "FLEET AVG IDLE CONSUMPTION,GAL/HR" CHR$(27) "F" TAB(37
);
1447 LPRINT USING "###.##";IDLE(1);:LPRINT TAB(52);:LPRINT USING "###.##";IDLE(2
);:LPRINT TAB(67);:LPRINT USING "###.##";IDLE(3):LPRINT
1450 LPRINT CHR$(27) "E" "FUEL PRICE: " CHR$(27) "F";:LPRINT USING "$$##.##";GAS
PRI;:LPRINT CHR$(27) "E" "  FLOW CONDITIONS: " CHR$(27) "F";
1460 IF FLOWCON%=1 THEN LPRINT "INTERRUPTED, <30 MPH"; ELSE LPRINT "UNINTERRUPTE
D, >30 MPH"
1480 LPRINT :LPRINT SPC(27) CHR$(27) "E" "PRESENT WORTH FUEL COST SAVINGS":LPRIN
T SPC(10) "AUTOS" SPC(10) "LIGHT TRUCKS"SPC(6)"HEAVY TRUCKS"SPC(9)"TOTAL"CHR$(27
) "F";:LPRINT TAB(7);
1490 :LPRINT USING "$$#########,";BENEFIT(2,1);:LPRINT TAB(26);:LPRINT USING "$$
#########,";BENEFIT(2,2);:LPRINT TAB(44);:LPRINT USING "$$#########,";BENEFIT(2,
3);:LPRINT TAB(60);:LPRINT USING "$$#########,";BENEFIT(2,1)+BENEFIT(2,2)+BENEFI
T(2,3)
1500 IF PF%(5)>0 THEN GOTO 1510 ELSE 1780 '*** PRN ACC SAV ***
1510 IF LNCT<10 THEN FOR I =1 TO LNCT+3:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:
LPRINT:LNCT=57
1520 LNCT=LNCT-10
1530 LPRINT:LPRINT CHR$(27) "E" SPC(29) "ACCIDENT COST SAVINGS" CHR$(13) SPC(32)
 "ACCIDENT HISTORY" CHR$(13) CHR$(13) SPC(20) "FATAL" SPC(10) "INJURY" SPC(10) "
PDO" SPC(10) "AADT"
1540 LPRINT "YEAR" SPC(13) "ACCIDENTS" SPC(7) "ACCIDENTS" SPC(5) "ACCIDENTS" CHR
$(27) "F"
1550 FOR Y=1 TO ACCYEARS:LPRINT:LPRINT USING "##";Y;:LPRINT "     TOTAL" TAB(20);
```

```
:LPRINT USING "####";FAT(Y);:LPRINT TAB(35);:LPRINT USING "####";INJ(Y);:LPRINT
TAB(51);:LPRINT USING "####";PDO(Y);:LPRINT TAB(64);:LPRINT USING "######,";TRAF
(Y)
1560 IF ACCLASS<=0 GOTO 1600
1570 FOR JJ=1 TO ACCLASS
1580 LPRINT TAB(7) LEFT$(CLAS$(JJ),8) TAB(20);:LPRINT USING "####";CLASFAT(Y,JJ)
;:LPRINT TAB(35);:LPRINT USING "####";CLASINJ(Y,JJ);:LPRINT TAB(51);:LPRINT USIN
G "####";CLASPDO(Y,JJ)
1590 NEXT JJ
1600 IF LNCT<4 THEN FOR I=1 TO LNCT+1:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:LP
RINT:LNCT=57
1605 LNCT=LNCT-4
1610 NEXT Y
1620 IF LNCT<5 THEN FOR I=1 TO LNCT+1:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:LP
RINT:LNCT=57
1630 LNCT=LNCT-5:LPRINT:LPRINT CHR$(27) "E" TAB(24) "PERCENT OF ACCIDENTS ELIMIN
ATED" CHR$(13) "ACCIDENT TYPE" SPC(10) "TOTAL" SPC(10) "FATALS" SPC(10) "INJURIE
S" SPC(10) "PDOs" CHR$(13) CHR$(13) SPC(6) "TOTAL" CHR$(27) "F";
1640 LPRINT TAB(25);:LPRINT USING "##.#";REDTOT;:LPRINT TAB(40);:LPRINT USING "#
#.#";REDFAT;:LPRINT TAB(56);:LPRINT USING "##.#";REDINJ;:LPRINT TAB(74);:LPRINT
USING "##.#";REDPDO
1650 IF ACCLASS<=0 GOTO 1680
1660 FOR JJ=1 TO ACCLASS:LPRINT TAB(7) LEFT$(CLAS$(JJ),8) TAB(24);:LPRINT USING
"##.#";REDTOTC(JJ);:LPRINT TAB(39);:LPRINT USING "##.#";REDFATC(JJ);:LPRINT TAB(
55);:LPRINT USING "##.#";REDINJC(JJ);:LPRINT TAB(73);:LPRINT USING "##.#";REDPDO
C(JJ)
1670 NEXT JJ
1680 IF LNCT<11 THEN FOR I=1 TO LNCT+1:LPRINT :NEXT I:LPRINT PLACE$:LPRINT ACT$:
LPRINT:LNCT=57
1690 LNCT=LNCT-11:LPRINT:LPRINT CHR$(27) "E" TAB(33) "ACCIDENT RATES" CHR$(13) T
AB(38) "FATALS" TAB(54) "INJURIES" TAB(72) "PDOs" CHR$(13) CHR$(13) "PRESENT RAT
E" CHR$(27) "F";
1700 LPRINT TAB(38);:LPRINT USING "##.###";FRAT;:LPRINT TAB(54);:LPRINT USING "#
#.###";IRAT;:LPRINT TAB(72);:LPRINT USING "##.###";PDORAT
1710 LPRINT CHR$(27) "E" "PROJECTED RATE" CHR$(27) "F" TAB(39);:LPRINT USING "##
.###";NEWFRAT;:LPRINT TAB(55);:LPRINT USING "##.###";NEWIRAT;:LPRINT TAB(73);:LP
RINT USING "##.###";NEWPRAT
1720 LPRINT TAB(12) "[PER MILLION ";:IF LENG>0 THEN LPRINT "VEHICLE MILES.";:GOT
O 1740
1730 LPRINT "ENTERING VEHICLES.";
1740 LPRINT " FATALS, PER 100 MILLION]":LPRINT
1750 LPRINT CHR$(27) "E" "COST/ACCIDENT" CHR$(27) "F" TAB(39);:LPRINT USING "$$#
#####,";BENVAL(3,1);:LPRINT TAB(54);:LPRINT USING "$$######,";BENVAL(3,2);:LPRIN
T TAB(72);:LPRINT USING "$$######,";BENVAL(3,3): LPRINT
1760 LPRINT CHR$(27) "E" "PRESENT WORTH OF ACCIDENT SAVING" CHR$(27) "F" TAB(37)
;:LPRINT USING "$$########,";BENEFIT(3,1);:LPRINT TAB(53);:LPRINT USING "$$#####
###,";BENEFIT(3,2);:LPRINT TAB(70);:LPRINT USING "$$########,";BENEFIT(3,3)
1770 LPRINT
1780 IF LNCT<25 THEN FOR I=1 TO LNCT+6:LPRINT:NEXT I:LPRINT PLACE$:LPRINT ACT$:L
PRINT
1785 LPRINT :LPRINT :LPRINT
1790 LPRINT TAB(26) CHR$(27) "E" "ECONOMIC EVALUATION SUMMARY":LPRINT
1800 LPRINT CHR$(27) "E" "ITEM" SPC(18) "ANNUALIZED WORTH" SPC(7) "PRESENT WORTH
" SPC(4) "% TOTAL BENEFIT":LPRINT:LPRINT
1810 LPRINT "COST" TAB(24) CHR$(27) "F";:LPRINT USING "$$#########,";TOTAW;
1820 LPRINT TAB(47);:LPRINT USING "$$#########,";TOTPW:LPRINT
1830 LPRINT CHR$(27) "E" "TIME SAVINGS" CHR$(27) "F" TAB(26);:LPRINT USING "$$##
#######,";AWBEN(1);
1840 LPRINT TAB(48);:LPRINT USING "$$#########,";PWBEN(1);
1850 LPRINT TAB(67);:LPRINT USING "###.#_%";100*PWBEN(1)/TOTPWBEN:LPRINT
```

```
1860 LPRINT CHR$(27) "E" "FUEL COST SAVINGS" CHR$(27) "F" TAB(26);:LPRINT USING
"$$#########,";AWBEN(2);
1870 LPRINT TAB(48);:LPRINT USING "$$#########,";PWBEN(2);
1880 LPRINT TAB(67);:LPRINT USING "###.#_%";100*PWBEN(2)/TOTPWBEN:LPRINT
1890 LPRINT CHR$(27) "E" "ACC. COST SAVINGS" CHR$(27) "F" TAB(26);:LPRINT USING
"$$#########,";AWBEN(3);
1900 LPRINT TAB(48);:LPRINT USING "$$#########,";PWBEN(3);
1910 LPRINT TAB(67);:LPRINT USING "###.#_%";100*PWBEN(3)/TOTPWBEN:LPRINT
1920 LPRINT CHR$(27) "E" "TOTAL BENEFIT" CHR$(27) "F" TAB(26);:LPRINT USING "$$#
########,";TOTAWBEN;
1930 LPRINT TAB(48);:LPRINT USING "$$#########,";TOTPWBEN;
1940 LPRINT TAB(67);:LPRINT USING "###.#_%";100*(PWBEN(1)+PWBEN(2)+PWBEN(3))/TOT
PWBEN:LPRINT
1950 LPRINT CHR$(27) "E" "NET WORTH" CHR$(27) "F" TAB(26);:LPRINT USING "$$#####
####,";TOTAWBEN-TOTAW;
1960 LPRINT TAB(48);:LPRINT USING "$$#########,";TOTPWBEN-TOTPW:LPRINT
1970 LPRINT CHR$(27) "E" "BENEFIT/COST RATIO" CHR$(27) "F" TAB(28):LPRINT USING
"####.##";TOTAWBEN/TOTAW:LPRINT ´***BREAKEVEN ANAL***
1980 IF (TOTPWBEN<>0) AND (TOTPW<>0) GOTO 1990 ELSE 2070
1990 BENCH=100*(TOTPW-TOTPWBEN)/TOTPWBEN:COSTCH=100*(TOTPWBEN-TOTPW)/TOTPW
2000 LPRINT:IF BENCH>=0 GOTO 2010 ELSE 2020
2010 LPRINT CHR$(27) "E" "With costs as shown, BENEFITS MUST INCREASE";:LPRINT U
SING " ####.##_%";BENCH;:LPRINT " to bring B/C up to 1.0.":GOTO 2030
2020 LPRINT "With costs as shown, benefits MAY DECREASE";:LPRINT USING " ####.##
_%";BENCH;:LPRINT " to bring B/C down to 1.0."
2030 LPRINT:IF COSTCH>=0 GOTO 2040 ELSE 2050
2040 LPRINT "With benefits as shown, costs MAY INCREASE";:LPRINT USING " ####.##
_%";COSTCH;:LPRINT " to bring B/C down to 1.0.":GOTO 2070
2050 LPRINT "With benefits as shown, COSTS MUST DECREASE";:LPRINT USING " ####.#
#_%";COSTCH;:LPRINT " to bring B/C up to 1.0." CHR$(27) "F"
2070 CLS:LOCATE 10,13:COLOR 0,7:PRINT "PRINT JOB COMPLETED! READY TO RETURN TO M
AIN PROGRAM":COLOR 7,0
2080 PLAY "MB T250 O2 L8;ABO3CDEFGFEDCO2BA"
2082 LOCATE 12,9:COLOR 0,7:PRINT "PRESS `R´ TO RETURN TO MAIN ROUTINE AFTER PRIN
TER HAS STOPPED":COLOR 7,0:LOCATE 14,20,1:COLOR 9,0:PRINT "PLEASE WAIT UNTIL PRI
NTER IS FINISHED!!";:COLOR 7,0
2084 C$=INKEY$:IF C$="" GOTO 2084
2086 IF C$="R" OR C$="r" GOTO 2090 ELSE 2080
2090 KEY(1) OFF
2100 CHAIN "MAINDSS.EXE"
12500 BEEP:FOR DELAY=1 TO 1000:WAT=DELAY*2:NEXT DELAY:RETURN:´>>DELAY<<
27699 STOP
30000 ´ NON-NUMERIC INPUT TRAP
30010 C$="":Z%=0:DEC=0:W%=0:M%=0:NEG=1:R=0
30020 V$=INKEY$:IF V$="" GOTO 30020
30030 IF V$="\" GOTO 30040 ELSE 30070
30040 FOR BS=1 TO Z%+W%+DEC+R
30050 PRINT CHR$(29) CHR$(32) CHR$(29);
30060 NEXT BS: GOTO 30010
30070 IF V$=CHR$(13) GOTO 30450
30080 IF V$="-" THEN NEG=-1!:R=1:GOTO 30430
30090 IF DEC=1 THEN W%=W%+1:GOTO 30220
30100 IF V$="." THEN DEC=1 : GOTO 30430
30110 IF V$="0" GOTO 30330 ELSE 30120
30120 IF V$="1" GOTO 30340 ELSE 30130
30130 IF V$="2" GOTO 30350 ELSE 30140
30140 IF V$="3" GOTO 30360 ELSE 30150
30150 IF V$="4" GOTO 30370 ELSE 30160
30160 IF V$="5" GOTO 30380 ELSE 30170
30170 IF V$="6" GOTO 30390 ELSE 30180
```

```
30180 IF V$="7" GOTO 30400 ELSE 30190
30190 IF V$="8" GOTO 30410 ELSE 30200
30200 IF V$="9" GOTO 30420
30210 BEEP:BEEP:GOTO 30020
30220 IF V$="0" THEN D(W%)=0:GOTO 30430
30230 IF V$="1" THEN D(W%)=1:GOTO 30430
30240 IF V$="2" THEN D(W%)=2:GOTO 30430
30250 IF V$="3" THEN D(W%)=3:GOTO 30430
30260 IF V$="4" THEN D(W%)=4:GOTO 30430
30270 IF V$="5" THEN D(W%)=5:GOTO 30430
30280 IF V$="6" THEN D(W%)=6:GOTO 30430
30290 IF V$="7" THEN D(W%)=7:GOTO 30430
30300 IF V$="8" THEN D(W%)=8:GOTO 30430
30310 IF V$="9" THEN D(W%)=9:GOTO 30430
30320 W%=W%-1:GOTO 30210
30330 Z%=Z%+1:B(Z%)=0:GOTO 30430
30340 Z%=Z%+1:B(Z%)=1:GOTO 30430
30350 Z%=Z%+1:B(Z%)=2:GOTO 30430
30360 Z%=Z%+1:B(Z%)=3:GOTO 30430
30370 Z%=Z%+1:B(Z%)=4:GOTO 30430
30380 Z%=Z%+1:B(Z%)=5:GOTO 30430
30390 Z%=Z%+1:B(Z%)=6:GOTO 30430
30400 Z%=Z%+1:B(Z%)=7:GOTO 30430
30410 Z%=Z%+1:B(Z%)=8:GOTO 30430
30420 Z%=Z%+1:B(Z%)=9:GOTO 30430
30430 PRINT V$;
30440 GOTO 30020
30450 XC=0:YC=0
30460 FOR Y%=1 TO Z%
30470 XC=XC+(B(Y%)*10^(Z%-Y%))
30480 NEXT Y%
30490 FOR M%=1 TO W%
30500 YC=YC+D(M%)*10^(-M%)
30510 NEXT M%
30520 NEWNUM=(XC+YC)*NEG:NEG=1
30530 RETURN
30540 'RETURN MAIN
32099 STOP
32100 REM SCREEN FOOTER
32105 LOCATE 20,28:COLOR 0,7:PRINT CHR$(25)"PRINTING THIS FILE " CHR$(25):COLOR
7,0
32110 LOCATE 22,15:PRINT PLACE$:PRINT TAB(15) ACT$:PRINT TAB(15) "OPENS IN ";:PR
INT USING "####";YOP;:PRINT "   LIFE: ";:PRINT USING "##_ _Y_R_S_.";NYEAR;
32120 RETURN 'END FOOTER
33999 STOP
35000 BEEP: LOCATE 8,18:COLOR 0,7:PRINT "  PRINTING INTERRUPTED. CHOOSE OPTION:
    ":COLOR 7,0
35010 LOCATE 10,20:PRINT "1> RESUME PRINTING":LOCATE 11,20:PRINT "2> RESTART PRI
NTING":LOCATE 12,20:PRINT "3> EXIT PRINT ROUTINE"
35020 LOCATE 13,33,1:COLOR 0,7:PRINT "ENTER CHOICE: ";:COLOR 7,0
35025 GOSUB 30000:C=NEWNUM
35030 LOCATE 8,18:PRINT SPC(55):LOCATE 10,20:PRINT SPC(20):LOCATE 11,20:PRINT SP
C(20):LOCATE 12,20:PRINT SPC(22):LOCATE 13,33:PRINT SPC(15)
35035 IF C>3 GOTO 35000
35040 IF C=1 THEN RETURN
35050 IF C=2 THEN RETURN 10
35060 RETURN 2090
```

APPENDIX D
PROGRAM LISTING

PROG

```
10 CLS:KEY OFF
20 REM yjh capital budgeting---for projects with one benefit
30 DIM C(50),B(50),N$(50),XTEMP(50),X(50),II(50),MM(50),CZ(50),BZ(50),NZ$(50),XC
N$(50),XOC(50),XOB(50)
40 CLS:PRINT SPC(27) "CAPITAL BUDGETING ROUTINE"
45 PRINT SPC(25) "NORTHWESTERN UNIVERSITY - 1985":PRINT
48 PRINT TAB(28) "1 RETRIEVE FROM FILE"
50 PRINT TAB(28) "2 DEFINE NEW PROJECT LIST"
55 PRINT TAB(28) "3 QUIT"
60 LOCATE 8,25,1:COLOR 0,7:PRINT "ENTER NUMBER OF DESIRED ACTION ";:COLOR 7,0
70 C$=INKEY$: IF C$="" GOTO 70
80 IF C$="1" GOTO 3000
90 IF C$="2" GOTO 120
95 IF C$<>"3" GOTO 100 ELSE LOCATE 14,25,1:COLOR 0,7:PRINT "VERIFY QUIT BY ENTEF
ING `Q`";:COLOR 7,0:BEEP
96 Q$=INKEY$:IF Q$="" GOTO 96
97 IF Q$="Q" OR Q$="q" THEN STOP ELSE 40
100 LOCATE 15,20: COLOR 0,7:PRINT "PLEASE ENTER CORRECT INSTRUCTION!";:BEEP
110 COLOR 7,0: GOTO 40
120 REM ********************** INPUT ROUTINE *********************
125 CLS
160 CLS:INPUT"Number of projects: ",N:PRINT:IF N<51 GOTO 170
162 COLOR 0,7:LOCATE 6,9:PRINT "NO MORE THAN 50 PROJECTS PERMITTED!":COLOR 7,0:E
EEP:FOR DELAY=1 TO 1000:XWAIT=DELAY^2:NEXT DELAY:GOTO 160
170 XON=N:FOR I=1 TO N
180 PRINT"Name of project ";:PRINT USING "##_: ";I;
190 INPUT"",N$(I):XON$(I)=N$(I)
200 PRINT"Cost of project ";:PRINT USING "##_:_ _$";I;
210 INPUT "",C(I):XOC(I)=C(I)
220 PRINT  "Benefit of project ";:PRINT USING "##_:_ _$";I;
230 INPUT "",B(I):XOB(I)=B(I)
240 PRINT
250 NEXT I
260 PRINT :INPUT "Total Budget: $",BB
265 XOBUDG=BB
270 REM HERE WE ECHO BACK THE INPUTS AND ALLOW USER TO CHANGE THEM
280 CLS:LOCATE 1,1:PRINT " Project #" SPC(5) "Project Name" SPC(10) "Cost" SPC(1
0) "Benefit" SPC(10) "Net Worth"
290 FOR I=1 TO XON
300 LOCATE 2+I,5:PRINT I:LOCATE 2+I,16:PRINT LEFT$(XON$(I),10) SPC(1);:LOCATE 2+
I,30:PRINT USING "$$#########,";XOC(I);:LOCATE 2+I,47:PRINT USING "$$#########,"
;XOB(I);:LOCATE 2+I,65:PRINT USING "$$#########,";XOB(I)-XOC(I)
310 NEXT I
320 PRINT:PRINT "TOTAL BUDGET = ";:PRINT USING "$$#########,";XOBUDG
330 PRINT SPC(30);:COLOR 0,7:PRINT "CHANGE INPUT (Y,N)?";:COLOR 7,0
340 C$=INKEY$: IF C$="" GOTO 340
350 IF C$="N" OR C$="n" OR C$=CHR$(13) GOTO 480
355 IF C$="Y" OR C$="y" GOTO 360 ELSE 356
356 PRINT SPC(20);:COLOR 0,7:PRINT "PLEASE ENTER CORRECT INSTRUCTION!";:BEEP
357 COLOR 7,0:FOR DELAY=1 TO 1200:XWAIT=DELAY^2:NEXT DELAY:GOTO 270
360 PRINT CHR$(13) SPC(25):COLOR 0,7:PRINT "CHANGE PROJECT OR BUDGET (P,B)?";:CC
LOR 7,0
370 C$=INKEY$: IF C$="" GOTO 370
380 IF C$="B" OR C$="b" GOTO 460
385 IF C$="p" OR C$="P" GOTO 390 ELSE  GOTO 360
390 PRINT:PRINT:INPUT "Project #: ",X:PRINT
400 PRINT"Name of project ";:PRINT USING "##_: ";X;
410 INPUT "",N$(X):XON$(X)=N$(X)
420 PRINT"Cost of project ";:PRINT USING "##_:_ _$";X;
430 INPUT "",C(X):XOC(X)=C(X)
```

```
440 PRINT "Benefit of project ";:PRINT USING "##_: _ _$";X;
450 INPUT "",B(X):XOB(X)=B(X):GOTO 270
460 PRINT :PRINT:INPUT "Total Budget: $",BB:XOBUDG=BB
470 PRINT:GOTO 270
480 COLOR 0,7:LOCATE N+7,25:PRINT "HOW MANY PROJECTS ARE REQUIRED? ";:COLOR 7,0:
·INPUT "",M
490 'PRINT:PRINT "HOW MANY REQUIRED PROJECTS? ";:COLOR 7,0:INPUT "",M
500 PRINT :PRINT :FOR I=1 TO M
510 INPUT"REQUIRED PROJECT #";MM(I)
520 NEXT I
530 FOR I=1 TO M
540 CZ(I)=C(MM(I)):BZ(I)=B(MM(I)):NZ$(I)=N$(MM(I))
550 FOR J=MM(I) TO N+1-I
560 C(J)=C(J+1):B(J)=B(J+1):N$(J)=N$(J+1)
570 NEXT J
580 NEXT I
590 REM ******************* computing process ***************************
600 FOR I=1 TO M
610 BB=BB-CZ(I)
620 NEXT I
625 IF BB<0 GOTO 4000
630 N=N-M
640 NN=2^N
650 MAX=0
660 FOR I=1 TO NN
670 XTEMP(1)=I\2
680 X(1)=I-XTEMP(1)*2
690 TEMP=X(1)*C(1):TEMP2=(B(1)-C(1))*X(1)
700 FOR J=2 TO N
710 XTEMP(J)=XTEMP(J-1)\2
720 X(J)=XTEMP(J-1)-XTEMP(J)*2
730 TEMP=TEMP+X(J)*C(J):TEMP2=TEMP2+(B(J)-C(J))*X(J)
740 NEXT J
750 IF TEMP>BB GOTO 910
760 IF TEMP2<MAX GOTO 910
770 MAX=TEMP2
780 REM ******************* output ***************************************
790 FOR L=1 TO 50
800 II(L)=0
810 NEXT L
820 PRINT
830 PRINT SPC(31) "CURRENT SOLUTION "
840 PRINT"PROJECTS CHOSEN:"
850 FOR K=1 TO N
860 IF X(K)=0 GOTO 890
870 PRINT"PROJECT ";N$(K)
880 II(K)=X(K)
890 NEXT K
900 PRINT "TOTAL BENEFIT:";MAX
905 LOCATE 25,50:PRINT "computing.....":LOCATE 25,70: PRINT TIME$
910 NEXT I
915 PLAY "MF T100 MS O5 L16;EEE"
920 REM ******************* output table ********************************
930 CLS:LOCATE 1,1:PRINT "I.ALTERNATIVES CHOSEN":LOCATE 1,50:PRINT "BUDGET = ";:
PRINT USING "$$#########,";XOBUDG
940 LOCATE 2,1:PRINT "Project Name" SPC(12) "Cost" SPC(12) "Benefit" SPC(12) "Ne
t Worth"
950 L=3
960 FOR I=1 TO N
970 IF II(I)=0 GOTO 1000
```

```
980 LOCATE L+1,5 :PRINT LEFT$(N$(I),10) ;:LOCATE L+1,17:PRINT USING "$$#########
,";C(I);:LOCATE L+1,36:PRINT USING "$$#########,";B(I);:LOCATE L+1,56:PRINT USIN
G "$$#########,";B(I)-C(I)
990 L=L+1
1000 NEXT I
1010 FOR I=1 TO M
1020 LOCATE 2+L,5:PRINT LEFT$(NZ$(I),10) ;:LOCATE 2+L,17:PRINT USING "$$########
#,";CZ(I);:LOCATE 2+L,36:PRINT USING "$$#########,";BZ(I);:LOCATE 2+L,56:PRINT U
SING "$$#########,";BZ(I)-CZ(I)
1030 L=L+1
1040 NEXT I
1050 PRINT "____________________________________________________________________
_"
1060 CT=0:BT=0:NT=0
1070 FOR I=1 TO N
1080 CT=CT+C(I)*II(I):BT=BT+B(I)*II(I):NT=NT+(B(I)-C(I))*II(I)
1090 NEXT I
1100 FOR I=1 TO M
1110 CT=CT+CZ(I):BT=BT+BZ(I):NT=NT+BZ(I)-CZ(I)
1120 NEXT I
1130 LOCATE 2+L,5: PRINT "Total":LOCATE 2+L,17:PRINT USING "$$#########,";CT;:LO
CATE 2+L,36:PRINT USING "$$#########,";BT;:LOCATE 2+L,56:PRINT USING "$$########
#,";NT
1140 'PRINT
1145 PRINT
1150 PRINT "II.OTHER ALTERNATIVES"
1155 'PRINT
1160 PRINT "Project Name" SPC(12) "Cost" SPC(12) "Benefit" SPC(12) "Net Worth"
1170 FOR I=1 TO N
1175 L=CSRLIN
1180 IF II(I)=1 GOTO 1200
1190 LOCATE L,5 :PRINT LEFT$(N$(I),10) ;:LOCATE L,17:PRINT USING "$$#########,";
C(I);:LOCATE L,36:PRINT USING "$$#########,";B(I);:LOCATE L,56:PRINT USING "$$##
#######,";B(I)-C(I)
1195 'L=L+1
1200 NEXT I
1210 PRINT "____________________________________________________________________
_"
1220 LOCATE 25,29,1:COLOR 0,7:PRINT"PRINT HARD COPY (Y,N)?";:COLOR 7,0
1230 C$=INKEY$: IF C$="" GOTO 1230
1250 IF C$="Y" OR C$="y" GOTO 1260 ELSE 1255
1255 LOCATE 25,27:COLOR 0,7:PRINT "REVISE THIS PROGRAM (Y,N)?";:COLOR 7,0
1256 C$=INKEY$:IF C$=""GOTO 1256
1257 IF C$="Y" OR C$="y" GOTO 1258 ELSE 1660
1258 BB=XOBUDG:N=XON:FOR REX=1 TO N:C(REX)=XOC(REX):B(REX)=XOB(REX):N$(REX)=XON$
(REX):NEXT REX:GOTO 270
1260 LOCATE 25,1,1:COLOR 0,7:BEEP:PRINT"    TURN ON PRINTER:  POSITION PAPER AT
TOP OF PAGE; HIT ANY KEY WHEN READY!  ";:COLOR 7,0
1270 C$=INKEY$:IF C$="" GOTO 1270
1275 LOCATE 25,1:PRINT SPC(78)
1280 LPRINT CHR$(27) "E" CHR$(14);"           INVESTMENT PROGRAM" CHR$(27) "F"
1290 LPRINT TAB(30) "BUDGET = ";:LPRINT USING "$$#########,";XOBUDG
1300 LPRINT
1310 LPRINT CHR$(27) "E" " I.ALTERNATIVES CHOSEN"
1320 LPRINT
1330 LPRINT " Project Name" SPC(10) "Cost" SPC(12) "Benefit" SPC(12) "Net Worth"
 CHR$(27) "F"
1335 LPRINT
1340 FOR I=1 TO N
1350 IF II(I)=0 GOTO 1370
```

```
1360 LPRINT SPC(5) LEFT$(N$(I),10);:LPRINT TAB(17);:LPRINT USING "$$#########,";
C(I);:LPRINT TAB(35):LPRINT USING "$$#########,";B(I);:LPRINT TAB(55):LPRINT US:
NG "$$#########,";B(I)-C(I)
1370 NEXT I
1380 FOR I=1 TO M
1390 LPRINT SPC(5) LEFT$(NZ$(I),10);:LPRINT TAB(17);:LPRINT USING "$$#########,'
;CZ(I);:LPRINT TAB(35);:LPRINT USING "$$#########,";BZ(I);:LPRINT TAB(55):LPRIN1
 USING "$$#########,";BZ(I)-CZ(I)
1400 NEXT I
1410 LPRINT "____________________________________________________________________
____"
1420 CT=0:BT=0:NT=0
1430 FOR I=1 TO N
1440 CT=CT+C(I)*II(I):BT=BT+B(I)*II(I):NT=NT+(B(I)-C(I))*II(I)
1450 NEXT I
1460 FOR I=1 TO M
1470 CT=CT+CZ(I):BT=BT+BZ(I):NT=NT+BZ(I)-CZ(I)
1480 NEXT I
1490 LPRINT SPC(5) "Total";:LPRINT TAB(17):LPRINT USING "$$#########,";CT;:LPRIN
T TAB(35):LPRINT USING "$$#########,";BT;:LPRINT TAB(55):LPRINT USING "$$#######
##,";NT
1500 LPRINT
1510 LPRINT
1520 LPRINT CHR$(27) "E" " II.OTHER ALTERNATIVES"
1530 LPRINT
1540 LPRINT " Project Name" SPC(10) "Cost" SPC(12) "Benefit" SPC(12) "Net Worth"
 CHR$(27) "F"
1545 LPRINT
1550 FOR I=1 TO N
1560 IF II(I)=1 GOTO 1580
1570 LPRINT SPC(5) LEFT$(N$(I),10);:LPRINT TAB(17);:LPRINT USING "$$#########,";
C(I);:LPRINT TAB(35):LPRINT USING"$$#########,";B(I);:LPRINT TAB(55):LPRINT USIN
G "$$#########,";B(I)-C(I)
1580 NEXT I
1590 LPRINT "____________________________________________________________________
____"
1600 CT=0:BT=0:NT=0
1610 FOR I=1 TO N
1620 IF II(I)=1 GOTO 1640
1630 CT=CT+C(I):BT=BT+B(I):NT=NT+(B(I)-C(I))
1640 NEXT I
1650 LPRINT SPC(5) "Total";:LPRINT TAB(17):LPRINT USING "$$#########,";CT;:LPRIN
T TAB(35):LPRINT USING "$$#########,";BT;:LPRINT TAB(55):LPRINT USING "$$#######
##,";NT
1655 LOCATE 25,27:COLOR 0,7:PRINT "REVISE THIS PROGRAM (Y,N)?";:COLOR 7,0
1656 C$=INKEY$:IF C$=""GOTO 1656
1657 IF C$="Y" OR C$="y" GOTO 270
1660 REM ****FILER ROUTINE****
1665 CLS:LOCATE 3,10,1:COLOR 0,7:PRINT "DO YOU WANT TO FILE PROJECT CHARACTERIST
ICS ON DISK (Y,N)?";: COLOR 7,0
1670 C$=INKEY$: IF C$="" GOTO 1670
1680 IF C$="Y" OR C$="y" GOTO 2000 ELSE 40
2000 REM ************************ AUTOFILE SETUP ROUTINE *******************
2010 LOCATE 5,10,1:COLOR 0,7:PRINT "TYPE DRIVE LETTER (A, B OR C):FILE NAME, THE
N HIT <RETURN>";:COLOR 7,0: LOCATE 7,35,1: INPUT "",NAM$
2015 ON ERROR GOTO 5000
2020 OPEN NAM$ FOR INPUT AS #1:CLOSE #1
2100 BEEP:LOCATE 9,15:COLOR 0,7:PRINT "THIS FILE NAME AREADY IN USE!  CHOOSE ANC
THER NAME.":COLOR 7,0:FOR DELAY=1 TO 1200:XWAIT = DELAY^2:NEXT DELAY
2110 CLS: GOTO 1660
```

```
2400 WRITE #1,XON
2410 FOR I=1 TO XON
2420 WRITE #1,XON$(I),XOC(I),XOB(I)
2430 NEXT I
2440 WRITE #1,XOBUDG
2450 CLOSE #1:GOTO 40
3000 REM ********************** RETRIEVE FROM FILE ****************************
3010 CLS:LOCATE 1,21,1:COLOR 0,7: PRINT "DO YOU WANT TO SEE DISKETTE DIRECTORY?"
;:COLOR 7,0
3020 LOCATE 3,21: PRINT "ENTER `A´ TO SEE LEFT DRIVE DIRECTORY;"
3030 LOCATE 5,21: PRINT "ENTER `B´ TO SEE RIGHT DRIVE DIRECTORY;"
3035 LOCATE 7,21: PRINT "ENTER `C´ TO SEE ACTIVE FIXED DISK DIRECTORY;"
3040 LOCATE 9,21:PRINT "ENTER <RETURN> ONLY FOR NO DIRECTORY":LOCATE 10,1,1
3050 C$=INKEY$: IF C$="" GOTO 3050
3060 IF C$="A" OR C$="a" GOTO 3090
3070 IF C$="B" OR C$="b" GOTO 3100
3075 IF C$="C" OR C$="c" GOTO 3105
3080 IF C$=CHR$(13) GOTO 3110 ELSE BEEP:GOTO 3010
3090 FILES "a:*.*":GOTO 3110
3100 FILES "b:*.*": GOTO 3110
3105 FILES "C:*.*": GOTO 3110
3110 LOCATE 22,14,1: COLOR 0,7:PRINT "TYPE DRIVE LETTER (A, B or C):FILENAME and
 HIT <RETURN>";:COLOR 7,0
3111 LOCATE 23,35,1:INPUT "",NAM$
3120 ON ERROR GOTO 3150
3130 OPEN NAM$ FOR INPUT AS #1
3140 GOTO 3200
3150 IF ERR=53 THEN PRINT "FILE DOES NOT EXIST; TRY AGAIN": GOTO 3190
3160 IF ERR=64 THEN PRINT "BAD FILE NAME: TRY AGAIN":GOTO 3190
3170 IF ERR=68 OR ERR=71 THEN PRINT "DEVICE UNAVAILABLE: CHECK IT OUT!":GOTO 319
0
3180 IF ERR=72 THEN PRINT "DISK MEDIA ERROR; TRY AGAIN"
3190 PRINT "UNIMAGINEABLE ERROR!!"
3192 BEEP:FOR I=1 TO 1000: W=2*I:NEXT I:CLOSE #1:GOTO 3000
3195 IF EOF(1) THEN PRINT "THIS FILE IS EMPTY!":CLOSE #1:GOTO 3192
3200 INPUT #1,N:XON=N
3210 FOR I=1 TO N
3220 INPUT #1, N$(I),C(I),B(I)
3225 XON$(I)=N$(I):XOC(I)=C(I):XOB(I)=B(I)
3230 NEXT I
3240 INPUT #1,BB:XOBUDG=BB
3350 CLOSE #1:GOTO 280
4000 CLS: LOCATE 5,13:COLOR 0,7: PRINT "TOTAL BUDGET IS LESS THAN REQUIRED, NO F
EASIBLE SOLUTION!!": COLOR 7,0
4001 LOCATE 8,25,1:COLOR 0,7: PRINT "QUIT OR START OVER (Q,S)?";:COLOR 7,0
4002 C$=INKEY$: IF C$="" GOTO 4002
4004 IF C$="q" OR C$="Q" GOTO 4010
4005 IF C$="s" OR C$="S" GOTO 40 ELSE 4000
4010 STOP
5000 IF ERR=53 THEN OPEN NAM$ FOR OUTPUT AS #1:RESUME 2400
5010 IF ERR=64 THEN PRINT SPC(27) "BAD FILE NAME; TRY AGAIN.":RESUME 5500
5020 IF ERR=68 OR ERR=71 THEN PRINT SPC(23) "DEVICE UNAVAILABLE; CHECK IT OUT!":
RESUME 5500
5030 IF ERR=72 THEN PRINT SPC(26) "DISK MEDIA ERROR; TRY AGAIN.":RESUME 5500
5040 PRINT "UNMENTIONABLE ERROR: PLEASE TROUBLESHOOT."
5500 FOR DELAY = 1 TO 1500:XWAIT=DELAY^2:NEXT DELAY:RESUME 0:GOTO 2000
```

APPENDIX E
PROGRAM LISTING

VIEW

```
10 REM ---- FILE VIEW PROGRAM -------------------------------------------
20 CLEAR
30 DIM CASHF(100),PWBEN(3),AWBEN(3),TAG$(7),MPG(3),YR(7),COST(7),LIFE(7),ANCOST(
7),PWCOST(7),OATTSAV(2),PKTTSAV(2),OPTTSAV(2),OLDPKTT(2),NEWPKTT(2),OLDOPTT(2),N
EWOPTT(2),OLDOATT(2),NEWOATT(2)
40 DIM BENEFIT(3,3),BENVAL(3,3),FAT(4),INJ(4),PDO(4),TRAF(4),CLASFAT(4,2),CLASIN
J(4,2),CLASPDO(4,2),PF%(7),IDLE(3),REDTOTC(2),REDFATC(2),REDINJC(2),REDPDOC(2),C
LAS$(2)
100 REM ---- STARTUP ROUTINE ---------------------------------------------
110 CLS: KEY OFF: ON ERROR GOTO 370
120 LOCATE 11,31: PRINT "FILE VIEW PROGRAM"
130 LOCATE 14,28: PRINT "NORTHWESTERN UNIVERSITY"
141 XX = 1
142 FOR I = 41 TO 65
143 LOCATE 6,I: COLOR 0,7: PRINT " ": COLOR 7,0
144 LOCATE 6,I-XX: COLOR 0,7: PRINT " ": COLOR 7,0
145 XX = XX + 2
146 NEXT I
147 FOR I = 7 TO 18
148 LOCATE I,16: COLOR 0,7: PRINT "  ": COLOR 7,0
149 LOCATE I,64: COLOR 0,7: PRINT "  ": COLOR 7,0
150 NEXT I
151 XX = 49
152 FOR I = 16 TO 40
153 LOCATE 19,I: COLOR 0,7: PRINT " ": COLOR 7,0
154 LOCATE 19,I+XX: COLOR 0,7: PRINT " ": COLOR 7,0
155 XX = XX -2
156 NEXT I
157 FOR DELAY = 1 TO 500: WAT = DELAY*2: NEXT DELAY
158 XX = 1
159 FOR I = 41 TO 65
160 LOCATE 19,I: PRINT " "
161 LOCATE 19,I-XX: PRINT " "
162 XX = XX + 2
163 FOR DELAY = 1 TO 20: WAT = DELAY*2: NEXT DELAY
164 NEXT I
165 FOR I = 1 TO 12
166 LOCATE 19-I,16: PRINT "  "
167 LOCATE 19-I,64: PRINT "  "
168 FOR DELAY = 1 TO 20: WAT = DELAY*2: NEXT DELAY
169 NEXT I
170 XX = 49
171 FOR I = 16 TO 40
172 LOCATE 6,I: PRINT " "
173 LOCATE 6,I+XX: PRINT " "
174 XX = XX - 2
175 FOR DELAY = 1 TO 20: WAT = DELAY*2: NEXT DELAY
176 NEXT I
177 FOR DELAY = 1 TO 500: WAT = DELAY*2: NEXT DELAY: BEEP
200 REM ---- ROUTINE TO RETRIEVE FROM FILE ---------------------------------
210 CLS: LOCATE 1,20: COLOR 0,7: PRINT "CALL THE FILE THAT YOU WANT TO LOOK AT"
   RETURN HERE FOR NEW FILE
220 LOCATE 4,10: COLOR 7,0: PRINT "DO YOU WANT TO SEE DISKETTE DIRECTORY BEFORE
DOING THIS ?"
230 LOCATE 6,20: COLOR 0,7: PRINT " ": COLOR 7,0: LOCATE 6,21: PRINT " ENTER `A´
 TO SEE LEFT DRIVE DIRECTORY"
240 LOCATE 8,20: COLOR 0,7: PRINT " ": COLOR 7,0: LOCATE 8,21: PRINT " ENTER `B´
 TO SEE RIGHT DRIVE DIRECTORY"
245 LOCATE 10,20: COLOR 0,7: PRINT " ": COLOR 7,0: LOCATE 10,21: PRINT " ENTER
C´ TO SEE FIXED DRIVE ACTIVE DIRECTORY"
```

```
250 LOCATE 12,20: COLOR 0,7: PRINT " ": COLOR 7,0: LOCATE 12,21: PRINT " ENTER <
RETURN> ONLY FOR NO DIRECTORY": LOCATE 14,1,1
260 C$ = INKEY$: IF C$ = "" THEN GOTO 260
270 IF C$ = "a" OR C$ = "A" THEN GOTO 300
280 IF C$ = "b" OR C$ = "B" THEN GOTO 310
285 IF C$ = "C" OR C$ = "c" THEN GOTO 315
290 IF C$ = CHR$(13) THEN GOTO 320 ELSE BEEP: GOTO 210
300 FILES "A:*.*": GOTO 320
310 FILES "B:*.*": GOTO 320
315 FILES "C:*.*": GOTO 320
320 LOCATE 22,13,1: PRINT "TYPE ": COLOR 0,7: LOCATE 22,19: PRINT "DRIVE LETTER
(A OR B):FILENAME": COLOR 7,0: LOCATE 22,51: PRINT "and HIT <RETURN>"
330 LOCATE 23,35: INPUT "", NAM$
340 ON ERROR GOTO 370
350 CLOSE #1: OPEN NAM$ FOR INPUT AS #1
360 GOTO 430
370 CLS: LOCATE 2,10,0
375 IF ERR=53 THEN PRINT "FILE DOES NOT EXIST: TRY AGAIN !": RESUME 420
380 IF ERR=64 THEN PRINT "BAD FILE NAME: TRY AGAIN !": RESUME 420
390 IF ERR=68 OR ERR=71 THEN PRINT "DEVICE UNAVAILABLE: CHECK IT OUT !": RESUME
420
400 IF ERR=72 THEN PRINT "DISK MEDIA ERROR: TRY AGAIN !": RESUME 420
410 PRINT "ERROR # " ERR " HAS BEEN DETECTED !": RESUME 420
420 CLOSE #1: BEEP: FOR DELAY = 1 TO 999: WAT = DELAY*2: NEXT DELAY: GOTO 200
430 IF EOF(1) THEN PRINT "THIS FILE IS EMPTY": CLOSE #1: GOTO 420
440 CLOSE #1: OPEN NAM$ FOR INPUT AS #1: GOSUB 3000
450 CLOSE #1: OPEN NAM$ FOR INPUT AS #1: GOSUB 4000  'subroutine for input
1000 REM ---- ROUTINE TO DISPLAY THE PROJECT -------------------------------
1010 CLS: COLOR 0,7: PRINT SPC(79);: LOCATE 1,30: PRINT "PROJECT DESCRIPTION": C
OLOR 7,0
1020 LOCATE 3,1:COLOR .7: PRINT "PROJECT LOCATION";: COLOR 7,0: PRINT " : "
1030 LOCATE 4,1:COLOR .7: PRINT "PROPOSED ACTION";: COLOR 7,0: PRINT " : "
1040 LOCATE 6,5:COLOR .7: PRINT "OPENING YEAR";: COLOR 7,0: PRINT " : "
1050 LOCATE 6,45: COLOR .7: PRINT "PROJECT LIFE";: COLOR 7,0: PRINT " (years) :
"
1060 LOCATE 7,5: COLOR .7: PRINT "AADT";: COLOR 7,0: PRINT " : "
1070 LOCATE 7,45: COLOR .7: PRINT "FUTURE AADT";: COLOR 7,0: PRINT " : "
1080 LOCATE 9,1: PRINT "< Discount Rate =        >"
1090 LOCATE 9,40: COLOR 0,7: PRINT "  PRESENT   "
1100 LOCATE 9,60: PRINT " ANNUALIZED "
1110 LOCATE 10,40: PRINT "   VALUE    "
1120 LOCATE 10,60: PRINT "   VALUE    "
1130 COLOR 7,0: LOCATE 11,5: COLOR .7: PRINT "COSTS": COLOR 7,0
1135 LOCATE 12,7: PRINT "> Total Costs"
1140 LOCATE 14,5: COLOR .7: PRINT "BENEFITS": COLOR 7,0
1150 LOCATE 15,7: PRINT "> Time Savings"
1160 LOCATE 16,7: PRINT "> Fuel Savings"
1170 LOCATE 17,7: PRINT "> Accident Reduction"
1180 LOCATE 19,5: PRINT "NET WORTH"
1190 LOCATE 20,5: PRINT "B/C RATIO"
1200 LOCATE 21,1: FOR I = 1 TO 79: PRINT "-";: NEXT I
1500 REM ---- SCREEN ------------------------------------------------
1510 LOCATE 3,20: PRINT LEFT$(PLACE$,59)
1520 LOCATE 4,19: PRINT LEFT$(ACT$,60)
1530 LOCATE 6,20: PRINT USING "####"; YOP
1540 LOCATE 6,68: PRINT USING "###"; NYEAR
1550 LOCATE 7,12: PRINT USING "######"; AADT
1560 LOCATE 7,59: PRINT USING "#######"; FUTAADT
1570 LOCATE 9,19: PRINT USING "##.# %"; DISCO*100
1580 LOCATE 12,39: PRINT USING "$$#########,"; TOTPW
```

```
1590 LOCATE 12,59: PRINT USING "$$#########,"; TOTAW
1600 LOCATE 15,39: PRINT USING "$$#########,"; PWBEN(1)
1610 LOCATE 15,59: PRINT USING "$$#########,"; AWBEN(1)
1620 LOCATE 16,39: PRINT USING "$$#########,"; PWBEN(2)
1630 LOCATE 16,59: PRINT USING "$$#########,"; AWBEN(2)
1640 LOCATE 17,39: PRINT USING "$$#########,"; PWBEN(3)
1650 LOCATE 17,59: PRINT USING "$$#########,"; AWBEN(3)
1660 LOCATE 19,39: PRINT USING "$$#########,"; TOTPWBEN-TOTPW
1670 LOCATE 19,59: PRINT USING "$$#########,"; TOTAWBEN-TOTAW
1680 LOCATE 20,52: PRINT USING "####.##"; TOTPWBEN/TOTPW
2500 REM ---- MENU -----------------------------------------------------------
2510 LOCATE 22,1: COLOR 16,7: PRINT " MENU ": COLOR 7,0
2520 LOCATE 22,10: PRINT "1) See next project"
2530 LOCATE 22,50: PRINT "2) See previous project"
2540 LOCATE 23,10: PRINT "3) Print hard copy of current project";
2550 LOCATE 23,50: PRINT "4) Return to the `list´";
2560 LOCATE 24,10: PRINT "5) Quit";
2570 LOCATE 24,44: COLOR 0,7: PRINT "ENTER THE DESIRED ACTION";: COLOR 7,0: LOCA
TE 24,69,1
2580 C$ = INKEY$: IF C$ = "" THEN GOTO 2580
2590 IF C$ = "1" THEN GOSUB 5000
2600 IF C$ = "2" THEN GOSUB 6000
2610 IF C$ = "3" THEN GOSUB 7000
2620 IF C$ = "4" THEN GOTO 440
2630 IF C$ = "5" THEN GOSUB 9000
2640 BEEP: GOTO 2570
3000 REM ---- SUBROUTINE TO DISPLAY THE LIST ---------------------------------
3010 CLS: LOCATE 1,1,0: COLOR 0,7: PRINT SPC(79): LOCATE 1,26: PRINT "LIST OF PR
OJECTS IN THE FILE": COLOR 7,0
3020 LOCATE 3,1: COLOR .7: PRINT "Project No.";: LOCATE 3,22: COLOR .7: PRINT "L
ocation";: LOCATE 3,50: PRINT "Action";: LOCATE 3,73: PRINT "Date": COLOR 7,0: P
RINT
3025 REM-- mm = number of the project, pp = number of the page
3030 MM = 0: PP = 1
3050 GOSUB 8000
3140 MM = MM + 1
3150 PRINT SPC(3) MM;: LOCATE ,15: PRINT LEFT$(PLACE$,22);: LOCATE ,43: PRINT LE
FT$(ACT$,22);: LOCATE ,70: PRINT XDATE$
3160 IF EOF(1) THEN 3200 ELSE 3161
3161 IF MM >= PP*15 THEN 3162 ELSE 3050
3162 LOCATE 23,28: COLOR 0,7: PRINT "See next entries (Y,N) ?";: COLOR 7,0: LOCA
TE 23,53,1
3163 C$ = INKEY$: IF C$ = "" THEN 3163
3164 IF C$ = "n" OR C$ = "N" THEN 3200 ELSE 3165
3165 IF C$ = "y" OR C$ = "Y" THEN 3166 ELSE 3162
3166 PP = PP + 1: CLS: LOCATE 1,1,0: COLOR 0,7: PRINT SPC(79): LOCATE 1,26: PRIN
T "LIST OF PROJECTS IN THE FILE": COLOR 7,0
3167 LOCATE 3,1: COLOR .7: PRINT "Project No.";: LOCATE 3,24: COLOR .7: PRINT "L
ocation";: LOCATE 3,49: PRINT "Action";: LOCATE 3,73: PRINT "Date": COLOR 7,0: P
RINT
3168 GOTO 3050
3200 CLOSE #1: LOCATE 23,24: COLOR 0,7: PRINT "Is this the correct file (Y,N) ?"
;: COLOR 7,0: LOCATE 23,57,1
3210 C$ = INKEY$: IF C$ = "" THEN 3210
3220 IF C$ = "n" OR C$ = "N" THEN 3230 ELSE 3240
3230 RETURN 210
3240 IF C$ = "y" OR C$ = "Y" THEN 3250
3245 BEEP: GOTO 3200
3250 LOCATE 23,24: PRINT SPC(45);: LOCATE 23,23: COLOR 0,7: PRINT "ENTER PROJECT
 NO. and HIT <RETURN>";: COLOR 7,0: LOCATE 23,58: INPUT "",NN
```

```
3260 IF NN < 1 THEN BEEP: GOTO 3250
3270 IF NN > MM THEN BEEP: GOTO 3250
3275 LOCATE 23,23: COLOR 7,0: PRINT SPC(50);: LOCATE 23,31: COLOR 16,7: PRINT "
  PLEASE  WAIT   ";: COLOR 7,0: LOCATE,,0
3280 RETURN
4000 REM ---- INPUT SUBROUTINE ----------------------------------------------
4010 FOR Z = 1 TO NN
4020 GOSUB 8000
4150 NEXT Z
4160 RETURN
5000 REM ---- SUBROUTINE TO GET NEW RECORD ----------------------------------
5010 NN = NN + 1  'RESET THE RECORD NUMBER
5020 IF EOF(1) THEN 5030 ELSE 5050
5030 BEEP: CLS: LOCATE 12,25: COLOR 0,7: PRINT " NO MORE RECORD IN THIS FILE ":
COLOR 7,0: LOCATE 14,27: PRINT "RETURN TO CURRENT PROJECT": LOCATE ,,0
5040 FOR DELAY = 1 TO 600: WAT = DELAY*2: NEXT DELAY: NN = NN - 1: GOTO 5140
5050 GOSUB 8000
5140 RETURN 1010
6000 REM ---- SUBROUTINE TO GET PREVIOUS RECORD -----------------------------
6010 NN = NN - 1
6020 IF NN <= 0 THEN 6030 ELSE 6045
6030 BEEP: CLS: LOCATE 12,29: COLOR 0,7: PRINT " NO PREVIOUS RECORD ": COLOR 7,0
: LOCATE 14,27: PRINT "RETURN TO CURRENT PROJECT": LOCATE,,0
6040 FOR DELAY = 1 TO 600: WAT = DELAY*2: NEXT DELAY: NN = NN + 1: GOTO 6060
6045 LOCATE 25,44: COLOR 7,0: PRINT SPC(30);: LOCATE 25,50: COLOR 16,7: PRINT "
  PLEASE  WAIT   ";: LOCATE,,0: COLOR 7,0
6050 CLOSE #1: OPEN NAM$ FOR INPUT AS #1: GOSUB 4000
6060 RETURN 1010
7000 REM ---- SUBROUTINE TO PRINT HARD COPY ---------------------
7010 CLS: LOCATE 11,15: COLOR 0,7: PRINT " TURN ON PRINTER AND POSITION PAPER TO
 TOP OF PAGE "
7020 LOCATE 13,16: PRINT " "
7030 LOCATE 15,16: PRINT " ": COLOR 7,0
7040 LOCATE 13,18: PRINT "ENTER `P' WHEN READY"
7050 LOCATE 15,18: PRINT "HIT ANY OTHER KEYS TO RETURN TO SCREEN DISPLAY": LOCAT
E ,,0
7060 C$ = INKEY$
7070 IF C$ = "" THEN 7060
7080 IF C$ = "p" OR C$ = "P" THEN 7100 ELSE 7090
7090 RETURN 1010
7100 ON ERROR GOTO 10000
7105 LPRINT CHR$(13)
7110 CLS: LOCATE 12,32: COLOR 16,7: PRINT "    PRINTING    ": COLOR 7,0
7120 LOCATE 14,16: PRINT "RETURN TO SCREEN DISPLAY WHEN PRINTING FINISHED"
7130 LPRINT TAB(22) CHR$(27) "E" CHR$(14) "PROJECT EVALUATION" CHR$(27) "F"
7140 LPRINT TAB(26) CHR$(14) "SUMMARY REPORT" CHR$(13) CHR$(13) CHR$(13)
7150 LPRINT SPC(2) CHR$(27) "E" "DATE : " CHR$(27) "F" XDATE$ CHR$(13)
7160 LPRINT SPC(2) CHR$(27) "E" "PROJECT LOCATION : " CHR$(27) "F" LEFT$(PLACE$,
56) CHR$(13)
7170 LPRINT SPC(2) CHR$(27) "E" "PROPOSED ACTION : " CHR$(27) "F" LEFT$(ACT$,56)
 CHR$(13) CHR$(13)
7180 LPRINT SPC(2) CHR$(27) "E" "LENGTH, MILES : " CHR$(27) "F";: LPRINT USING "
###.##";LENG
7190 LPRINT CHR$(13) SPC(2) CHR$(27) "E" "OPENING YEAR : " CHR$(27) "F";: LPRINT
 USING "####";YOP;
7200 LPRINT SPC(25) CHR$(27) "E" "PROJECT LIFE, YRS : " CHR$(27) "F";: LPRINT US
ING "###";NYEAR
7210 LPRINT CHR$(13) SPC(2) CHR$(27) "E" "AADT : " CHR$(27) "F";: LPRINT USING "
######";AADT;
7220 LPRINT SPC(31) CHR$(27) "E" "FUTURE AADT : " CHR$(27) "F";: LPRINT USING "#
```

```
######";FUTAADT
7230 LPRINT CHR$(13) CHR$(13) CHR$(13) SPC(29) CHR$(27) "E" "DISCOUNT RATE : " C
HR$(27) "F";: LPRINT USING "##.# %";DISCO*100
7240 LPRINT CHR$(13) SPC(39) CHR$(27) "E" "PRESENT" SPC(13) "ANNUALIZED"
7250 LPRINT SPC(40) "VALUE" SPC(16) "VALUE" CHR$(13)
7260 LPRINT SPC(7) "COSTS" CHR$(27) "F"
7270 TAG$(1) = "DESIGN": TAG$(2) = "R-O-W": TAG$(3) = "CONSTRUCTION": TAG$(4) =
"MAINTENANCE": TAG$(5) = "OPERATION": TAG$(6) = "OTHER": TAG$(7) = "OTHER"
7280 FOR J = 1 TO 7
7290 LPRINT SPC(9) TAG$(J);
7300 LPRINT TAB(35);: LPRINT USING "$$##########,";PWCOST(J);
7310 LPRINT TAB(56);: LPRINT USING "$$##########,";ANCOST(J)
7320 NEXT J
7330 LPRINT CHR$(13) SPC(9) CHR$(27) "E" "TOTAL" CHR$(27) "F";: LPRINT TAB(37);:
 LPRINT USING "$$##########,";TOTPW;: LPRINT TAB(58);: LPRINT USING "$$##########,
";TOTAW
7340 LPRINT CHR$(13) CHR$(13) SPC(7) CHR$(27) "E" "BENEFITS" CHR$(27) "F"
7350 TAG$(1) = "TIME SAVINGS": TAG$(2) = "FUEL SAVINGS": TAG$(3) = "ACCIDENT RED
UCTION"
7360 FOR J = 1 TO 3
7370 LPRINT SPC(9) TAG$(J);
7380 LPRINT TAB(35);: LPRINT USING "$$##########,"; PWBEN(J);
7390 LPRINT TAB(56);: LPRINT USING "$$##########,"; AWBEN(J)
7400 NEXT J
7410 LPRINT CHR$(13) SPC(9) CHR$(27) "E" "TOTAL" CHR$(27) "F";: LPRINT TAB(37);:
 LPRINT USING "$$##########,";TOTPWBEN;: LPRINT TAB(58);: LPRINT USING "$$########
##,";TOTAWBEN
7420 LPRINT CHR$(13) CHR$(13) SPC(7) CHR$(27) "E" "NET WORTH" CHR$(27) "F" TAB(3
7);: LPRINT USING "$$##########,";TOTPWBEN-TOTPW;: LPRINT TAB(58);: LPRINT USING
"$$##########,";TOTAWBEN-TOTAW
7430 LPRINT CHR$(13) CHR$(13) SPC(7) CHR$(27) "E" "BENEFIT/COST RATIO" CHR$(27)
"F" TAB(50);: LPRINT USING "####.##";TOTPWBEN/TOTPW
7435 ON ERROR GOTO 370
7440 RETURN 1010
8000 REM ---- INPUT -------------------------------------------------------------
8002 INPUT #1,LENG,PLACE$
8004 INPUT #1,ACT$
8010 INPUT #1, NYEAR, AADT, YAADT, YOP, PCAR, PLTRUK, PHTRUK, ANGROW, FUTAADT, X
DATE$, XTIME$, PWTFC,PWXTFC, XFRACT, YX, TOTCASHF, MPG(1), MPG(2), MPG(3), GASPR
I, FLOWCON%, VNO%
8020 FOR L = 1 TO 7: INPUT #1, YR(L), COST(L), LIFE(L), ANCOST(L), PWCOST(L): NE
XT L
8030 INPUT #1, INFLATE,SHARE,WCAR,WLTRUK,WHTRUK,PEAKPCT,OLDPKSP,NEWPKSP,OLDOPSP,
NEWOPSP,OLDOASP,NEWOASP,ACCYEARS,ACCLASS,XAADT,SAVACF,SAVACI,SAVACP,CARSHARE,LTS
HARE,HTSHARE
8040 FOR X% = 1 TO 2: INPUT #1, OLDPKTT(X%), NEWPKTT(X%), OLDOPTT(X%), NEWOPTT(X
%), OLDOATT(X%), NEWOATT(X%): NEXT X%
8050 FOR TYPE = 1 TO 3: INPUT #1, PWBEN(TYPE), AWBEN(TYPE): FOR BENFLAG = 1 TO 3
: INPUT #1, BENEFIT(TYPE,BENFLAG), BENVAL(TYPE,BENFLAG): NEXT BENFLAG:NEXT TYPE
8060 FOR I = 1 TO (YOP-YX)+1+NYEAR: INPUT #1, CASHF(I): NEXT I
8070 FOR I = 1 TO ACCYEARS: INPUT #1, FAT(I), INJ(I), PDO(I), TRAF(I)
8080 FOR J = 1 TO ACCLASS: INPUT #1, CLASFAT(I,J), CLASINJ(I,J), CLASPDO(I,J)
8090 NEXT J: NEXT I
8100 INPUT #1, DISCO, PF%(6), PF%(1), PF%(2), PF%(3), PF%(4), PF%(5), REDTOT, RE
DFAT, REDINJ, REDPDO, FRAT, IRAT, PDORAT, NEWFRAT, NEWIRAT, NEWPRAT, TOTPWBEN, T
OTAWBEN
8110 FOR J = 1 TO ACCLASS: INPUT #1, REDTOTC(J), REDFATC(J), REDINJC(J), REDPDOC
(J), CLAS$(J): NEXT J
8120 INPUT #1, TOTAW, TOTPW, IDLE(1), IDLE(2), IDLE(3)
8125 FOR X%=1 TO 2:OPTTSAV(X%)=OLDOPTT(X%)-NEWOPTT(X%):PKTTSAV(X%)=OLDPKTT(X%)-N
```

```
EWPKTT(X%):OATTSAV(X%)=OLDOATT(X%)-NEWOATT(X%):NEXT X%
8130 RETURN
9000 REM ---- SUBROUTINE TO QUIT -----------------------------------------------
9010 BEEP: CLS: LOCATE 12,25: COLOR 0,7: PRINT " VERIFY QUIT BY ENTERING `Q´ ";:
 COLOR 7,0
9020 LOCATE 14,17: PRINT "HIT ANY OTHER KEYS TO RETURN TO CURRENT PROJECT": LOCA
TE.,0
9030 C$=INKEY$: IF C$ = "" THEN 9030
9040 IF C$ = "q" OR C$ = "Q" THEN 9045 ELSE 9050
9045 CLOSE #1: CLS: LOCATE 12,37: PRINT "BYE...": FOR I = 1 TO 7: PRINT: NEXT I:
 KEY ON: STOP
9050 RETURN 1010
10000 REM ---- PRINTING ERROR TRAP ------------------------------------------
10010 BEEP: CLS: LOCATE 2,10: PRINT "ERR # " ERR " HAS BEEN DETECTED !"
10020 IF ERR = 24 OR ERR = 27 OR ERR = 25 OR ERR = 57 OR ERR = 68 THEN 10030 ELS
E 10040
10030 LOCATE 4,10: PRINT "PLEASE CHECK THE PRINTER"
10040 RESUME 10050
10050 FOR DELAY = 1 TO 500: WAT = DELAY*2: NEXT DELAY
10060 ON ERROR GOTO 370
10070 GOTO 7010
```

APPENDIX F
List of Files on DSS Delivery Diskette

```
Disk Volume: DSSDELIVERY                    12-23-88    5:59 pm     Page 1


Path: \
      13 tagged files using 283869 bytes

      DSSTEST1.          1152  .a..    7-29-86  12:02 pm
      DSSTEST2.          2176  .a..    8-22-85   1:32 pm
      PROGEX  .           256  .a..    7-05-88  12:23 pm
      AUTOEXEC.BAT         21  .a..    7-22-85   2:34 pm
      T       .BAT        607  .a..   12-23-88  11:50 am
      STARTUP .DSS        128  .a..    7-29-86   2:07 pm
      INSTALL .EXE      23296  .a..    7-22-85   2:23 pm
      MAINDSS .EXE     114719  .a..    6-24-88   9:42 am
      PRINTDSS.EXE      56815  .a..    6-24-88   9:19 pm
      PROG    .EXE      42176  .a..    6-23-88   9:56 pm
      VIEW    .EXE      41732  .a..    6-21-88  10:21 pm
      CONFIG  .SYS         23  .a..    1-01-80  12:04 am
      RAMDISK .SYS        768  .a..    1-01-80  12:05 am
```

KEY TO FILES ON DSS DELIVERY DISKETTE

__Program Files__

1.	MAINDSS.EXE	=	Main DSS program
2.	PRINTDSS.EXE	=	DSS print routine
3.	PROG.EXE	=	Capital budgeting program
4.	VIEW.EXE	=	Program to view DSS project files
5.	INSTALL.EXE	=	Program to install DSS default parameters

__System Management Files__

6.	T.BAT	=	Batch program to transfer files to user's floppy diskettes
7.	AUTOEXEC.BAT	=	Autoexecution file to start DSS from floppy diskette
8.	CONFIG.SYS	=	Configuration file to create RAMDISK for floppy diskette systems
9.	RAMDISK.SYS	=	Program to define RAMDISK

__Data Files__

10.	STARTUP.DSS	=	File of default DSS parameters, initial version supplied, rewritten with INSTALL.EXE
11.	DSSTEST1	=	Example DSS project data files
12.	DSSTEST2	=	Example DSS project data files
13.	PROGEX	=	Example PROG data file