RP125

FINAL REPORT

# ITD UPDATE OF WINTER MAINTENANCE COMPLEMENT PREDICTION MODEL

Submitted to:

IDAHO TRANSPORTATION DEPARTMENT

ITD/UI Cooperative Transportation Research Program

National Center for Advanced Transportation Technology (NCATT)

University of Idaho

Donald F. Haber

December 12, 1996

# ABSTRACT

Two models that were developed for Idaho Transportation Department in 1989-1990, one for predicting Cost/Benefits for changes in Winter Level of Service and one for Winter Maintenance Complement determination were revised to included data up to 1994.

The Benefit/Cost computer prediction model was completely rewritten to include the new data on yearly traffic volume and yearly winter maintenance costs. The revised model software was written in Visual Basic and was developed to be compatible with the six maintenance districts computer facilities. An output from a hypothetical change in winter maintenance levels is illustrated.

The Average Storm Hours for ten winter seasons (1984-1994) for 42 foreman areas and the six districts were determined. The Average Storm Hours for the each of the six districts were compared with the overall state average value. This comparison was then used to recommend priorities for changes winter complements in the six districts.

ABSTRACT........................................................................................i

TABLE OF CONTENTS ........................................................................................ ii

LIST OF TABLES ........................................................................................ iii

LIST OF FIGURES ........................................................................................ iv

INTRODUCTION ........................................................................................ 1

SCOPE OF WORK........................................................................................ 1

REVISING THE B/C MODEL

Cost Algorithm........................................................................................ 2

Benefits Algorithm ........................................................................................ 3

New Computer Program........................................................................................ 3

WINTER COMPLEMENT REVISION

Basis for Analysis ........................................................................................ 6

Calculation of Average Storm Hours ........................................................................................ 6

Winter Complement Determination........................................................................................ 9

REFERENCES ........................................................................................ 10

APPENDICES

A. Summary of District and FA ASH Values ........................................................................................ 11

B. Visual Basis Program Listing ........................................................................................ 18

C. Flow Chart for B/C Algorithm ........................................................................................ 38

# LIST OF TABLES

TABLE 1: Sample Output of Cost/Benefit Program ........ 5

TABLE 2: Example of Storm Day Determination ........ 7

TABLE 3: Ten Year Average ASH Values ........ 9

# LIST OF FIGURES

FIGURE 1: Daily Winter Maintenance Hours and Storm Day Cut-off ................................8

## INTRODUCTION

In 1989 and 1990, two highway maintenance analysis models and computer software to run and display the models[1] were developed by Professor Donald Haber and graduate students at the U. of Idaho for the Maintenance Department of the Idaho Transportation Department(ITD) . One model was to assist the department in determining the proper winter maintenance complement level for the six maintenance district's. The other model was to predict a cost/benefit factor for changes in winter level of service (WLS) on any highway segment in the state.

Both models used data collected yearly by ITD on winter maintenance costs, ADT (average daily traffic volume), lane miles of WLS, and other road prism data. Since 1990, these models along with the computer software to run and display the model analyses, have helped IDT prioritize changes in WLS and have provided a neutral basis for prioritizing changes in the six district's full time winter maintenance complement levels.

Because of the importance of the yearly cost and traffic volume data to the models and since neither model has been updated since 1990, ITD contracted with the U. of Idaho to update and revise both models and the computer software. This update will include cost, traffic volume, and changes in highway prism data for the entire period from 1984 to 1994.

## SCOPE OF WORK

Revising both models and the B/C computer program, was the major basis for the scope of work. The first task was to integrate five years of road data from 1990 - 1994 with the 1984 -1989 data in the two models. This involved updating the data base to include yearly information on costs, traffic volume, and lane miles of specific WLS and other road prism data. Once the updated data base was determined, other specified tasks were:

1. update the cost algorithm,
2. update and change the benefits algorithm to reflect more recent ADT and labor cost estimates,

3. recalculate the Average Storm Hours (ASH) for each district,
4. revise the computer software for the B/C analysis and make it compatible with ITD software,
5. determine a winter complement recommendation using the updated historical information,
6. document the revised computer software and users manual..

## REVISING THE B/C COMPUTER PROGRAM

**The Cost Algorithm** The cost algorithm predicts the cost per mile for changes in WLS for specified highway segment . The algorithm is based on a regression analysis using road, terrain, and climate factors[1,2]. New values for these factors were determined using the revised road data base. The factors are:

Road Factors:

1. Total Lane Miles in Each Foreman Area,
2. Road Curvature and Gradient,
3. Level of Service

Terrain and Climate factors:

1. Terrain Type,
2. Elevation
3. Snow and Wind Factors,
4. Climate Factor and Temperature.

The value of most factors did not change significantly from the1989 data with the exception of total lane miles in each FA and the level of service for certain highway segments.

## Benefits Algorithm

The basis for the calculation of benefits from changes in WLS for a highway segment are: **1)** the value ($/min) of time saved due to better winter road conditions (if the WLS is increased) and **2)** the total change in delay time for all vehicles using the highway segment.

Vehicle delay time due to a change in WLS for a segment of highway was determined by:

$$Time\ saved = Trip\ Length * \left[\frac{1}{V_{old}} - \frac{1}{V_{new}}\right]$$

where, $V_{old}$ = Speed on previous level of service

$V_{new}$ = Speed on upgraded level of service

Since both $V_{old}$ and $V_{new}$ will vary from vehicle to vehicle, a simulation using the two vehicle speeds as random variables was used to approximate the time saved. The vehicle speed probability distributions for WLS were estimated using studies conducted by the Utah Department of Transportation[3,4]. The average daily traffic volume for the highway segment was then used to calculate the average total vehicle time saved daily.

The value of time savings was calculated assuming an average hourly labor rate. Two rates were used, one for commuters and one for general travelers. The hourly labor rates are input by the user when running the B/C program or default rates are used based on data obtained from "County Business Patterns - Idaho"[4].

## New Computer Program

The previous B/C computer program was formulated in the "Quick Basic" language. The revised program uses a "Visual Basic" formulation. All Districts have the ability to run this program in Visual Basic. The Visual Basic approach presents a much friendlier interface for the user then the previous approach. A complete overhaul of all subroutines was conducted, simplifying the program and improving the speed of presentation.

The output for a sample run of the Benefit/Cost computer program is shown in Table 1. Complete listing of the Visual Basic Program is found in Appendix B and the flow chart is given in Appendix C.

## TABLE 1

## SAMPLE OUTPUT OF THE COST/BENEFIT PROGRAM

Date of Analysis: 6/3/96 Time of Analysis: 10:12:07 AM

DISTRICT 2 CHANGE IN LEVEL OF SERVICE 3 TO 2

| SEGMENT | ROUTE | FA | MILES | ADT | BENEFITS | COSTS | B/C | BMP | EMP |
|---|---|---|---|---|---|---|---|---|---|
| 1800 | 3 | 220 | 2.2 | 1155 | $2959 | $516 | 5.74 | 26.8 | 29.0 |
| 1800 | 3 | 220 | 10.9 | 706 | $9124 | $2603 | 3.50 | 39.0 | 49.9 |
| 1850 | 6 | 240 | 13.2 | 585 | $9258 | $4446 | 2.08 | 9.9 | 23.0 |
| 1860 | 9 | 240 | 13.5 | 748 | $12151 | $4570 | 2.66 | 0.0 | 13.5 |
| 1870 | 8 | 240 | 11.7 | 1510 | $21319 | $3969 | 5.37 | 14.6 | 26.3 |
| 1870 | 3 | 220 | 10.0 | 697 | $8270 | $2389 | 3.46 | 26.3 | 36.3 |
| 1870 | 8 | 220 | 17.3 | 353 | $7232 | $4130 | 1.75 | 36.3 | 53.6 |
| 1880 | 99 | 220 | 11.7 | 627 | $8704 | $2793 | 3.12 | 0.0 | 11.7 |
| 1930 | 11 | 270 | 30.0 | 1256 | $33968 | $6813 | 4.99 | 0.0 | 30.0 |
| 1930 | 11 | 270 | 5.3 | 1400 | $6722 | $1210 | 5.56 | 30.0 | 35.3 |
| 1940 | 62 | 250 | 15.4 | 489 | $7454 | $3843 | 1.94 | 0.0 | 15.4 |
| 1950 | 162 | 250 | 23.1 | 611 | $13974 | $5760 | 2.43 | 0.0 | 23.1 |
| 1960 | 13 | 290 | 11.0 | 1637 | $17892 | $2383 | 7.51 | 0.0 | 11.0 |
| 1960 | 13 | 260 | 15.4 | 1490 | $30757 | $5223 | 5.89 | 11.0 | 26.4 |
| 1970 | 14 | 290 | 30.0 | 507 | $15111 | $6499 | 2.33 | 0.0 | 30.0 |

| TOTAL MILES | TOTAL TIME | TOTAL BENEFITS | TOTAL COSTS |
|---|---|---|---|
|  | (DAYS SAVED) |  |  |
| 252.3 | 1023.5 | $242,547 | $64,409 |

## WINTER COMPLEMENT REVISION

### Basis for Analysis

In this analysis, the size of the winter complement for each district or FA is determined solely by the average number of labor hours (ASH) used over a 24 hour period in fulfilling assigned winter maintenance levels during sever winter storms.

### Calculation of Average Storm Hours (ASH)

ASH is calculated using ten years of reported labor hours used on winter maintenance activities by the 42 Foreman Areas (FA) . Four specific maintenance activities were chosen as those activities most representative of winter storm maintenance. The activities covered  snow removal, sanding and storm patrolling, The daily man hours spent on these activities are reported to the ITD's Maintenance Management Information System (MMIS) where it was recorded on tape.

From the MMIS, the daily man hours recorded for the four activities were found for the winter seasons from 1984-1994. The average and standard deviation for daily labor hours were calculated for each winter season 1984-1994. A storm day was defined as the days where the labor hours spent on the snow related activities exceeded the winter daily average plus 1.5 times the standard deviation. For a day to be classified as a storm day, the labor hours had to be over 40% more than daily seasonal average. Table 2, shows the procedure used and Figure 1 illustrates the storm day determination for FA 120 for the 1993 winter season. The hours spent on each storm day was then averaged to obtain the ASH for a FA over one season. ASH values were calculated for each of the ten year winter period for the 42 FA's and the overall ten year average FA ASH was determined. The ten year district ASH value was calculated by averaging the ASH values for the FA's in that district. Average ASH values for the six districts and the 42 FA's for the ten year period are summarized in Table 3.

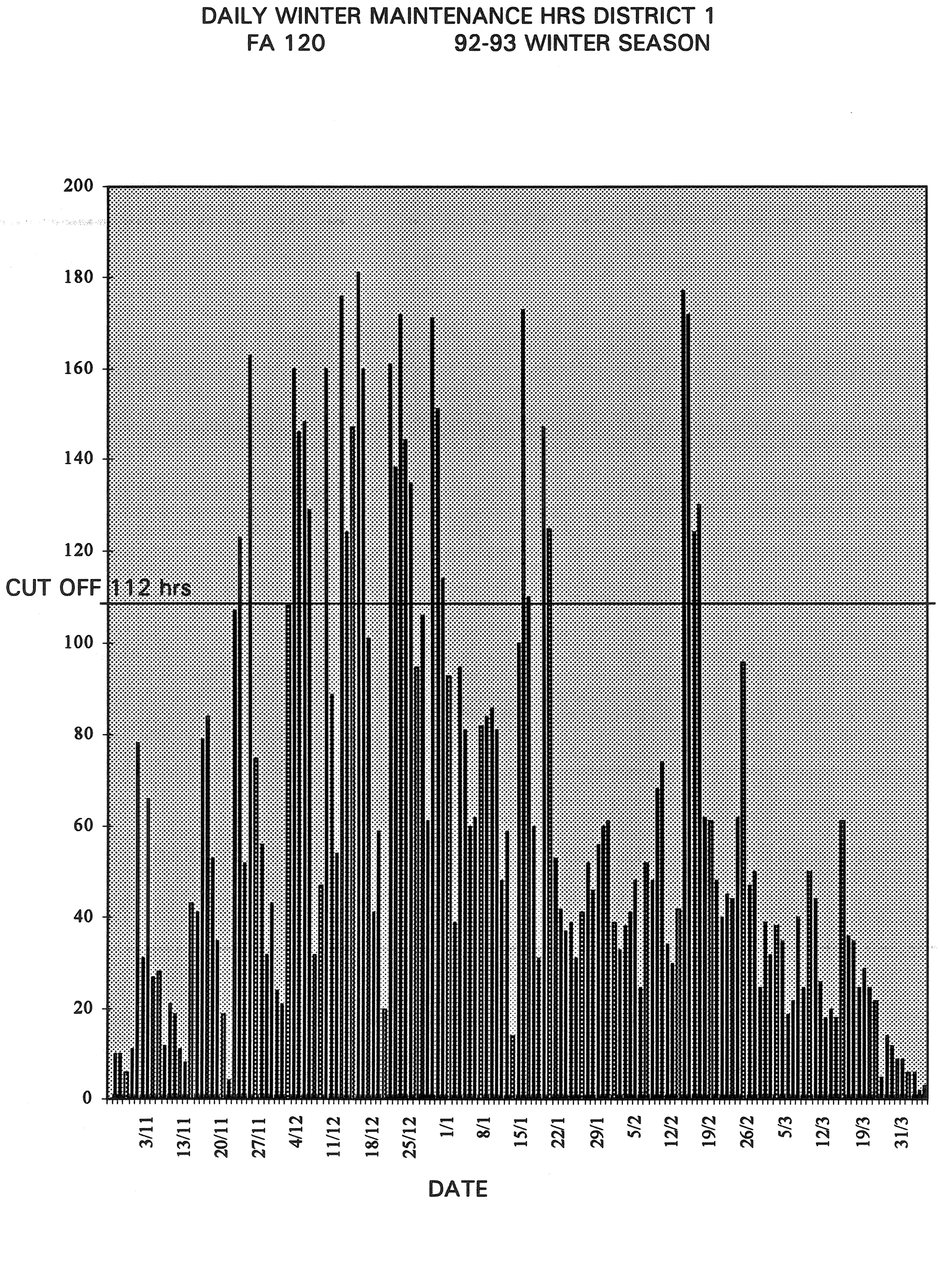
## TABLE 2

## STORM DAY DETERMINATION

DISTRICT 1 FA 12O

| 92-93 | | | STORM | | | | STORM | | |
|---|---|---|---|---|---|---|---|---|---|
| DATE | HRS | | DAY | DATE | HRS | | DAY | | |
| | | | 112 | | | | 84 | | |
| 9/10/92 | 10 | 0 | 0 | 10/31/93 | 4 | 0 | 0 | 92-93 | |
| 10/13/92 | 10 | 0 | 0 | 11/4/93 | 8 | 0 | 0 | Mean | 55 |
| 10/14/92 | 6 | 0 | 0 | 11/5/93 | 4 | 0 | 0 | SD | 38 |
| 10/15/92 | 11 | 0 | 0 | 11/6/93 | 5 | 0 | 0 | Sample Variance | 1457 |
| 10/29/92 | 78 | 0 | 0 | 11/7/93 | 4 | 0 | 0 | Sum | 8248 |
| 11/2/92 | 31 | 0 | 0 | 11/8/93 | 12 | 0 | 0 | Count | 151 |
| 11/3/92 | 66 | 0 | 0 | 11/12/93 | 2 | 0 | 0 | Mean + 1.5*SD = | 112 |
| 11/4/92 | 27 | 0 | 0 | 11/13/93 | 49 | 0 | 0 | Storm Days = | 11 |
| 11/8/92 | 28 | 0 | 0 | 11/14/93 | 25 | 0 | 0 | Ave. Ash = | 139 |
| 11/9/92 | 12 | 0 | 0 | 11/15/93 | 15 | 0 | 0 | Storm hr | 1532 |
| 11/10/92 | 21 | 0 | 0 | 11/16/93 | 37 | 0 | 0 | | |
| 11/11/92 | 19 | 0 | 0 | 11/17/93 | 61 | 0 | 0 | | |
| 11/12/92 | 11 | 0 | 0 | 11/18/93 | 83 | 0 | 0 | 93-94 | |
| 11/13/92 | 8 | 0 | 0 | 11/19/93 | 43 | 0 | 0 | Mean | 43 |
| 11/14/92 | 43 | 0 | 0 | 11/20/93 | 32 | 0 | 0 | SD | 28 |
| 11/15/92 | 41 | 0 | 0 | 11/21/93 | 118 | 118 | 1 | Sample Variance | 759 |
| 11/16/92 | 79 | 0 | 0 | 11/22/93 | 83 | 0 | 0 | Sum | 7071 |
| 11/17/92 | 84 | 0 | 0 | 11/23/93 | 61 | 0 | 0 | Count | 165 |
| 11/18/92 | 53 | 0 | 0 | 11/24/93 | 38 | 0 | 0 | Mean + 1.5*SD = | 84 |
| 11/19/92 | 35 | 0 | 0 | 11/26/93 | 25 | 0 | 0 | Storm Days = | 14 |
| 11/20/92 | 19 | 0 | 0 | 11/27/93 | 41 | 0 | 0 | Ave. Ash = | 102 |
| 11/21/92 | 4 | 0 | 0 | 11/28/93 | 63 | 0 | 0 | Storm hr | 1424 |
| 11/22/92 | 107 | 0 | 0 | 11/29/93 | 117 | 117 | 1 | | |
| 11/23/92 | 123 | 123 | 1 | 11/30/93 | 89 | 89 | 1 | | |
| 11/24/92 | 52 | 0 | 0 | 12/1/93 | 136 | 136 | 1 | | |
| 11/25/92 | 163 | 163 | 1 | 12/2/93 | 96 | 96 | 1 | | |
| 11/26/92 | 75 | 0 | 0 | 12/3/93 | 61 | 0 | 0 | | |
| 11/27/92 | 56 | 0 | 0 | 12/4/93 | 32 | 0 | 0 | | |
| 11/28/92 | 32 | 0 | 0 | 12/5/93 | 53 | 0 | 0 | | |

## FIGURE 1
## DAILY WINTER MAINTENANCE HOURS AND STORM DAY CUT-OFF



## Winter Complement Determination

Once the ten year district average ASH values was determined, the state ten year average ASH was calculated. Table 3, shows the deviations of each district's ASH values from the statewide average. Comparing the present district winter complement size with deviations from the state average ASH, a basis for assigning priorities for additions or cuts to the districts winter complements could be established.

Modifications to this approach could include that in certain winter storms regular winter maintenance personnel are used for over 8 hours a day and personnel outside of the regular winter maintenance complement may also be employed. However, even with these factors, the average ASH still could be valuable as a guide for storm maintenance labor requirements.

**TABLE 3**
**TEN YEAR AVERAGE ASH VALUES**

| DISTRICT | ASH | ASH/8 | %CHANGE |
|---|---|---|---|
| 1 | 699 | 87 | 14 |
| 2 | 533 | 67 | -13 |
| 3 | 636 | 80 | 6 |
| 4 | 579 | 72 | -4 |
| 5 | 604 | 75 | 0 |
| 6 | 557 | 70 | -8 |
| STATE AVE. | 601 | 75.1 | |
| TOTAL | 4208 | 526 | |

## REFERENCES

1. Haber, D. F. , Maloney, M., and Horn, D., Determination of a model to Predict Winter Maintenance Personnel Levels, Final Report , University of Idaho, Civil Engineering Department, September 1989.

2. Haber, D. F. and Limaye, U., Benefit Cost Analysis of Winter Maintenance Levels of the Idaho Transportation Department, Final Report, University of Idaho, Civil Engineering Department, December 1990.

3. McBride, J. C., et al, Economic Impact of Highway Snow and Ice Control Final Report, ESIC - User's Manual, Report No. FHWA-RD-77-95, December 1977.

4. McBride, J. C., et al, Economic Impact of Highway Snow and Ice Control Final Report, ESIC - User's Manual, Report No. FHWA-RD-77-96, December 1977.

5. County Business Patterns - Idaho, U.S. Department of Commerce, Bureau of the Census, Washington D. C., 1982-87

# APPENDIX A

## SUMMARY TABLE OF DISTRICT ASH VALUES FOR TEN YEARS

| | | | | | 88-94 | 84-88 | Ave. | |
|---|---|---|---|---|---|---|---|---|
| | District 1 | | DISTRICT 1<br>AVE ASH | | 746.2 | 627.9 | 698.9 | |
| | | | AVE TOTAL<br>STORM HRS | | 11853. | | | |
| | | 88-89 | 89-90 | 90-91 | 91-92 | 92-93 | 93-94 | Averages |
| 120 | Storm days | 15 | 13 | 15 | 12 | 13 | 14 | |
| | Ave Ash | 66.7 | 73.7 | 71.4 | 62.84 | 73.79 | 64.92 | 68.9 |
| | Storm | 1000.0 | 958.0 | 1071. | 754.1 | 959.2 | 908.9 | 941.9 |
| 130 | Storm Days | 20 | 20 | 11 | 15 | 19 | 16 | |
| | Ave Ash | 145.5 | 162.0 | 148.1 | 124.67 | 160.37 | 139.38 | 146.7 |
| | Storm Hrs | 2909.0 | 3240. | 1629. | 1870.1 | 3047.0 | 2230.1 | 2487.5 |
| 140 | Storm Days | 17 | 16 | 18 | 14 | 18 | 16 | |
| | Ave Ash | 129.3 | 133.9 | 119.8 | 100.07 | 120.11 | 112 | 119.2 |
| | Storm Hrs | 2197.9 | 2142. | 2156. | 1401.0 | 2162.0 | 1792.0 | 1975.2 |
| 150 | Storm Days | 17 | 15 | 15 | 10 | 11 | 14 | |
| | Ave Ash | 166.4 | 134.5 | 126.3 | 96.4 | 139.27 | 101.71 | 127.4 |
| | Storm Hrs | 2829.1 | 2017. | 1895. | 964.0 | 1532.0 | 1423.9 | 1776.8 |
| 160 | Storm Days | 14 | 14 | 16 | 13 | 13 | 13 | |
| | Ave Ash | 138.9 | 136.4 | 121.8 | 74.24 | 172.23 | 110.13 | 125.6 |
| | Storm Hrs | 1944.0 | 1909. | 1949. | 965.1 | 2239.0 | 1431.7 | 1739.7 |
| 170 | Storm Days | 37 | 18 | 14 | 14 | 15 | 13 | |
| | Ave Ash | 153.8 | 168.9 | 158.1 | 116.64 | 212.2 | 140.92 | 158.4 |
| | Storm Hrs | 5690.6 | 3040. | 2213. | 1633.0 | 3183.0 | 1832.0 | 2931.9 |
| | **TOTAL STORM HRS** | **16571** | **1330** | **1091** | **7587** | **13122** | **9619** | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | 88-94 | 84-88 | Ave | |
| | District 2 | | DISTRICT 2 | | 557.7 | 495.0 | 532.6 | |
| | | | AVE ASH | | | | | |
| | | | STORM HRS | | 7807.0 | | | |
| | | | AVE TOTAL | | | | | |
| | | 88-89 | 89-90 | 90-91 | 91-92 | 92-93 | 93-94 | averages |
| 220 | Storm Days | 15 | 19 | 17 | 8 | 13 | 13 | |
| | Ave Ash | 128.3 | 96.84 | 99.8 | 68 | 108.5 | 65.4 | 94.5 |
| | Storm Hrs | 1924.5 | 1840. | 1696. | 544.0 | 1410.5 | 850.2 | 1377.6 |
| 240 | Storm Days | 15 | 6 | 19 | 10 | 13 | 15 | |
| | Ave Ash | 117.7 | 105.7 | 98.3 | 67.1 | 119.2 | 70.3 | 96.4 |
| | Storm Hrs | 1765.5 | 634.2 | 1867. | 671.0 | 1549.6 | 1054.5 | 1257.1 |
| 250 | Storm Days | 17 | 5 | 16 | 9 | 11 | 12 | |
| | Ave Ash | 106.1 | 78.2 | 84.1 | 58.9 | 101.18 | 72.67 | 83.5 |
| | Storm Hrs | 1803.7 | 391.0 | 1346. | 530.1 | 1113.0 | 872.0 | 1009.3 |
| 260 | Storm Days | 16 | 17 | 21 | 13 | 15 | 12 | |
| | Ave Ash | 135.2 | 135.4 | 112.2 | 104.7 | 127.6 | 127.6 | 123.8 |
| | Storm Hrs | 2163.2 | 2301. | 2356. | 1361.1 | 1914.0 | 1531.2 | 1937.9 |
| 270 | Storm Days | 18 | 15 | 14 | 71 | 10 | 11 | |
| | Ave Ash | 82.6 | 84.2 | 79.6 | 9 | 91.6 | 74.5 | 70.3 |
| | Storm Hrs | 1486.8 | 1263. | 1114. | 639.0 | 916.0 | 819.5 | 1039.8 |
| 290 | Storm Days | 13 | 12 | 17 | 14 | 13 | 11 | |
| | Ave Ash | 106.9 | 85.2 | 87.2 | 63.4 | 102.3 | 90.9 | 89.3 |
| | Storm Hrs | 1389.7 | 1022. | 1482. | 887.6 | 1329.9 | 999.9 | 1185.3 |
| | TOTAL STORM HRS | 10533 | 7452 | 9863 | 4633 | 8233 | 6127 | |

| | 88-94 | 84-88 | Ave. |
|---|---|---|---|
| DISTRICT 3<br>AVE ASH | 535.7 | 577.9 | 552.6 |
| STORM HRS<br>AVE TOTAL | 7912.0 | | |

| | District 3 | 88-89 | 89-90 | 90-91 | 91-92 | 92-93 | 93-94 | averages |
|---|---|---|---|---|---|---|---|---|
| 320 | Storm Days | 20 | 18 | 7 | 14 | 20 | 18 | |
| | Ave Ash | 103.1 | 91.7 | 81 | 71.1 | 103.1 | 91.7 | 90.3 |
| | Storm Hrs | 2062 | 1650. | 567.0 | 995.4 | 2062.0 | 1650.6 | 1497.9 |
| 330 | Storm Days | 14 | 12 | 20 | 15 | 18 | 10 | |
| | Ave Ash | 106.1 | 61.1 | 74.4 | 52.5 | 112.6 | 99.5 | 84.4 |
| | Storm Hrs | 1485.4 | 733.2 | 1488. | 787.5 | 2026.8 | 995.0 | 1252.7 |
| 340 | Storm Days | 16 | 12 | 17 | 12 | 13 | 9 | |
| | Ave Ash | 98.5 | 76.3 | 64.2 | 70.4 | 129.1 | 107.1 | 90.9 |
| | Storm Hrs | 1576 | 915.2 | 1091. | 844.8 | 1678.3 | 963.9 | 1178.3 |
| 350 | Storm Days | 18 | 16 | 15 | 16 | 20 | 12 | |
| | Ave Ash | 104 | 91.7 | 96.1 | 46.1 | 140.1 | 112.6 | 98.4 |
| | Storm Hrs | 1872 | 1467. | 1441. | 737.6 | 2802.0 | 1351.2 | 1611.9 |
| 360 | Storm Days | 14 | 11 | 14 | 9 | 17 | 11 | |
| | Ave Ash | 115.1 | 73 | 94.3 | 58.9 | 121.5 | 105.1 | 94.7 |
| | Storm Hrs | 1611.4 | 803.0 | 1320. | 530.1 | 2065.5 | 1156.1 | 1247.7 |
| 370 | Storm Days | 12 | 15 | 16 | 15 | 16 | 14 | |
| | Ave Ash | 94.3 | 77.4 | 72.1 | 45 | 95.4 | 78.1 | 77.1 |
| | Storm Hrs | 1131.6 | 1161. | 1153. | 675.0 | 1526.4 | 1093.4 | 1123.5 |
| 390 | Storm Days | 13 | 14 | 13 | 16 | 14 | 11 | |
| | Ave Ash | 84.8 | 73.5 | 71.2 | 56.9 | 71.4 | 74.1 | 72.0 |
| | Storm Hrs | 1102.4 | 1029. | 925.6 | 910.4 | 999.6 | 815.1 | 963.7 |
| | **TOTAL STORM HRS** | **10841** | **7759** | **7987** | **5481** | **13161** | **8025** | |

**88-94 84-88 Ave.**

| | 88-94 | 84-88 | Ave. |
|---|---|---|---|
| DISTRICT 4<br>AVE ASH | 579.2 | 577.6 | 578.6 |
| AVE TOTAL<br>STORM HRS | 5338.1 | | |

| | District 4 | 88-89 | 89-90 | 90-91 | 91-92 | 92-93 | 93-94 | averages |
|---|---|---|---|---|---|---|---|---|
| 430 | Storm Days | 9 | 12 | 13 | 7 | 9 | 6 | |
| | Ave Ash | 158.1 | 121.8 | 156.8 | 132.1 | 216.3 | 154.3 | 156.6 |
| | Storm Hrs | 1422.9 | 1461. | 2038. | 924.7 | 1946.7 | 925.8 | 1453.4 |
| 450 | Storm Days | 10 | 11 | 10 | 7 | 12 | 10 | |
| | Ave Ash | 165 | 117.3 | 86.7 | 76.2 | 225 | 122.3 | 132.1 |
| | Storm Hrs | 1650.0 | 1290. | 867.0 | 533.4 | 2700.0 | 1223.0 | 1377.3 |
| 460 | Storm Days | 7 | 5 | 4 | 1 | 8 | 5 | |
| | Ave Ash | 118.6 | 79.4 | 115.5 | 83 | 142.1 | 115.4 | 109.0 |
| | Storm Hrs | 830.2 | 397.0 | 462.0 | 83.0 | 1136.8 | 577.0 | 581.0 |
| 480 | Storm Days | 11 | 17 | 19 | 15 | 17 | 18 | |
| | Ave Ash | 94.4 | 73.5 | 70.7 | 79.3 | 101.5 | 84.7 | 84.0 |
| | Storm Hrs | 1038.4 | 1249. | 1343. | 1189.5 | 1725.5 | 1524.6 | 1345.1 |
| 490 | Storm Days | 8 | 5 | 5 | 3 | 9 | 4 | |
| | Ave Ash | 93.75 | 77.6 | 117.2 | 66 | 128.7 | 102 | 97.5 |
| | Storm Hrs | 750.0 | 388.0 | 586.0 | 198.0 | 1158.3 | 408.0 | 581.4 |
| | **TOTAL STORM HRS** | **5692** | **4786** | **5297** | **2929** | **8667** | **4658** | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | **88-94** | **84-88** | **Ave.** | |
| | | | **DISTRICT 5 ASH** | | **582.6** | **636.0** | **603.9** | |
| | | | **AVE TOTAL STORM HRS** | | **8512.03** | | | |
| | District 5 | 88-89 | 89-90 | 90-91 | 91-92 | 92-93 | 93-94 | |
| **530** | Storm Days | 14 | 19 | 21 | 14 | 17 | 16 | |
| | Ave Ash | 118.8 | 85.5 | 94.7 | 88 | 113.8 | 90.75 | **98.6** |
| | Storm Hrs | 1663.2 | 1624. | 1988. | 1232.0 | 1934.6 | 1452.0 | **1649.2** |
| **540** | Storm Days | 16 | 12 | 14 | 7 | 14 | 13 | |
| | Ave Ash | 88.6 | 63 | 85.6 | 75.6 | 103.8 | 79.9 | **82.8** |
| | Storm Hrs | 1417.6 | 756.0 | 1198. | 529.2 | 1453.2 | 1038.7 | **1065.5** |
| **550** | Storm Days | 13 | 22 | 21 | 17 | 11 | 17 | |
| | Ave Ash | 81.7 | 60 | 62.1 | 60.2 | 82.7 | 63.5 | **68.4** |
| | Storm Hrs | 1062.1 | 1320. | 1304. | 1023.4 | 909.7 | 1079.5 | **1116.5** |
| **560** | Storm Days | 14 | 9 | 11 | 7 | 13 | 8 | |
| | Ave Ash | 68.5 | 46.2 | 61.6 | 44 | 91.9 | 60.9 | **62.2** |
| | Storm Hrs | 959.0 | 415.8 | 677.6 | 308.0 | 1194.7 | 487.2 | **673.7** |
| **570** | Storm Days | 9 | 11 | 10 | 11 | 13 | 9 | |
| | Ave Ash | 91.8 | 54.7 | 69.3 | 55.7 | 109.8 | 76 | **76.2** |
| | Storm Hrs | 826.2 | 601.7 | 693.0 | 612.7 | 1427.4 | 684.0 | **807.5** |
| **580** | Storm Days | 12 | 18 | 19 | 18 | 25 | 8 | |
| | Ave Ash | 98.8 | 86.6 | 90.8 | 76.8 | 118.7 | 125.8 | **99.6** |
| | Storm Hrs | 1185.6 | 1558. | 1725. | 1382.4 | 2967.5 | 1006.4 | **1637.7** |
| **590** | Storm Days | 12 | 17 | 20 | 17 | 18 | 15 | |
| | Ave Ash | 92.6 | 92.3 | 88.3 | 84.5 | 105.6 | 105.9 | **94.9** |
| | Storm Hrs | 1111.2 | 1569. | 1766. | 1436.5 | 1900.8 | 1588.5 | **1562.0** |
| | **TOTAL STORM HRS** | **8225** | **7846** | **9353** | **6524** | **11788** | **7336** | |

|  | 88-94 | 84-88 | Ave. |
|---|---|---|---|
| DISTRICT 6<br>AVE ASH | 571.5 | 534.4 | 556.7 |
| AVE TOTAL<br>STORM HRS | 7919.3 |  |  |

|  | District 6 | 88-89 | 89-90 | 90-91 | 91-92 | 92-93 | 93-94 | averages |
|---|---|---|---|---|---|---|---|---|
| 640 | Storm Days | 17 | 14 | 20 | 19 | 19 | 17 |  |
|  | Ave Ash | 105.6 | 78.4 | 92.2 | 92.4 | 138 | 93.5 | 100.0 |
|  | Storm Hrs | 1795.2 | 1097. | 1844. | 1755.6 | 2622.0 | 1589.5 | 1784.0 |
| 650 | Storm Days | 12 | 12 | 12 | 11 | 11 | 12 |  |
|  | Ave Ash | 122.3 | 97.1 | 92.4 | 92.7 | 148.2 | 105.8 | 109.8 |
|  | Storm Hrs | 1467.6 | 1165. | 1108. | 1019.7 | 1630.2 | 1269.6 | 1276.9 |
| 660 | Storm Days | 11 | 9 | 16 | 11 | 7 | 13 |  |
|  | Ave Ash | 103.1 | 85.4 | 81.8 | 107.4 | 114.8 | 108.4 | 100.2 |
|  | Storm Hrs | 1134.1 | 768.6 | 1308. | 1181.4 | 803.6 | 1409.2 | 1101.0 |
| 670 | Storm Days | 14 | 9 | 10 | 11 | 12 | 14 |  |
|  | Ave Ash | 59.6 | 50 | 39 | 41 | 71.2 | 40.9 | 50.3 |
|  | Storm Hrs | 834.4 | 450.0 | 390.0 | 451.0 | 854.4 | 572.6 | 592.1 |
| 680 | Storm Days | 11 | 13 | 14 | 11 | 13 | 15 |  |
|  | Ave Ash | 86.7 | 65.3 | 60.7 | 69.3 | 88.8 | 80.5 | 75.2 |
|  | Storm Hrs | 953.7 | 848.9 | 849.8 | 762.3 | 1154.4 | 1207.5 | 962.8 |
| 690 | Storm Days | 14 | 15 | 16 | 16 | 20 | 15 |  |
|  | Ave Ash | 186 | 101.9 | 119.7 | 98.2 | 187.1 | 123.7 | 136.1 |
|  | Storm Hrs | 2604.0 | 1528. | 1915. | 1571.2 | 3742.0 | 1855.5 | 2202.7 |
|  | TOTAL STORM HRS | 8789 | 5859 | 7417 | 6741 | 10807 | 7904 |  |

# APPENDIX B

## VISUAL BASIC PROGRAM

```
DECLARE FUNCTION comfort! (t!)
DECLARE FUNCTION comfort1! (t!)
DECLARE FUNCTION delay! (t!)
DECLARE FUNCTION delay1! (t!)

DECLARE SUB AVGCOST (nls!, ntlm!, nelev!, nash!, ncurves!, nsf!, nwf!, ACOST!)
DECLARE SUB deltime (va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, t
DECLARE SUB indata (va!(), rf!(), sigd!(), sigw!())
DECLARE SUB inflation (NoSimul, inflat!, yr$)
DECLARE SUB info (FILES$, t() AS ANY, S() AS ANY)
DECLARE SUB SCR5 (mi, j%, ii%, formn%(), S() AS ANY, nls!, ntlm!, nelev!, nash!,
DECLARE SUB scrn1 ()
DECLARE SUB scrn3 (lx%, lnew!, FILES$, work1, mi, D$, district%)
DECLARE SUB wadt1 (segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(),
DECLARE SUB wadt2 (segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(),

DIM va(1 TO 8) AS SINGLE
DIM rf(1 TO 8) AS SINGLE
DIM sigd(1 TO 8) AS SINGLE
DIM sigw(1 TO 8) AS SINGLE

DIM wsum(500), formn%(500), msum(500), yben(500), netc(500), TOTIME(500), r%(500
DIM bmfa(500), fmfa(500), st(500), l(500), segment(500), tlnm(500)


TYPE steady
       obs AS INTEGER
       fa AS INTEGER
       ls AS SINGLE
       ash AS SINGLE
       elev AS SINGLE
       tlm AS SINGLE
       curves AS INTEGER
       sf AS SINGLE
       wf AS SINGLE
       ls1tlm AS INTEGER
       ls2tlm AS INTEGER
       ls3tlm AS INTEGER
       ls4tlm AS INTEGER
       ls5tlm AS INTEGER
       tlm9596 AS INTEGER
       ls9596 AS SINGLE
END TYPE

TYPE trans
       obst AS INTEGER
       dist AS INTEGER
       yr12 AS INTEGER
       yash AS SINGLE
       deltsh AS SINGLE
       sii AS DOUBLE
       tsh AS SINGLE
END TYPE
DIM t(1 TO 42) AS trans, S(1 TO 38)  AS steady


10 FOR i% = 1 TO 2
   formn%(i%) = 0!: yben(i%) = 0: netc(i%) = 0: TOTIME(i%) = 0
   lx% = 0!: lnew = 0!: lold = 0!
   msum(i%) = 0!
   wsum(i%) = 0!
   tlnm(i%) = 0!
NEXT i%

FILES$ = "c:\highway"
CONST pi = 3.141592654#

        'SCRN1 INTRODUCES DELAY COST SIMULATION ESTIMATES!
CALL scrn1
        'SCRN3 CALLS FOR NECESSARY INPUT DATA!
CALL scrn3(lx%, lnew, FILES$, work1, mi, D$, district%)
        'INITIALIZES ARRAY VALUES!
CALL indata(va(), rf(), sigd(), sigw())
        'WADT CALLLS FOR INPUT AND CALCULATES WEIGHTED ADT AND MILES PER GIVEN M
 lold = lx%
 NoSimul = 2000

 CALL inflation(NoSimul, inflat, yr$)
 IF D$ = "D" OR D$ = "d" THEN 114
 CALL wadt1(segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(),
GOTO 115
114 CALL wadt2(segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(

        'LOOP TO CALCULATE WINTER AVERAGE DAILY TRAFFIC AND COST SIMULATION PER
CLS
 LOCATE 14, 10, 0
 PRINT "PROGRAM IS NOW DOING SIMULATIONS.  PLEASE CONTINUE TO WAIT !!"

'*******************************************************************
'get total benefits


115
'go into a subroutine & calculate the costs once
trip = mi
CALL deltime(va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1,

FOR j% = 1 TO fmn%
 CLS
        IF j% = 1 THEN
                CALL info(FILES$, t(), S())
        END IF

        IF st(j%) = 1 THEN          'use to get proper info found in sub deltime..
                ttime = ttimeS1
                ccost = ccostS1
                dcost = dcostS1
        ELSEIF st(j%) = 0 THEN
                ttime = ttimeS0
                ccost = ccostS0
                dcost = dcostS0
        END IF

         wntadt = 0
       wntadt = (wsum(j%) / msum(j%))
       IF msum(j%) <= 1 THEN 22
       acttrip = msum(j%)
'now proportion the above information...
'that is...
'cost for 2 miles = cost for "trip" miles * actual miles / "trip" miles used for
```

```
                ccost = acttrip / trip * ccost
                dcost = acttrip / trip * dcost
                ttime = acttrip / trip * ttime
'get total benefits
                ccost = ccost * wntadt          ' $ per year
                dcost = dcost * wntadt          ' $ per year
                ttime = ttime * wntadt          ' $ per year

            FOR ii% = 1 TO 37
                IF formn%(j%) = S(ii%).fa THEN
                        DAYS = S(ii%).ash / 8
                        EXIT FOR
                END IF
            NEXT ii%

                scost = DAYS * (dcost + ccost)
                yben(j%) = scost
                IF o$ = "d" THEN yben(j%) = -yben(j%)
                TOTIME(j%) = (ttime / 1440) * DAYS
22 NEXT j%

 ACOST = 0: dcost = 0

'*****************************************************************
'get the costs
 COLOR 0, 7, 0
 FOR mi% = 1 TO 60
    PRINT
 NEXT mi%
 FOR j% = 1 TO fmn%
   tlm = tlnm(j%)
   fmfa(j%) = bmfa(j%) + msum(j%)
   CALL info(FILES$, t(), S())
   CLS
  IF formn%(j%) <> 0 THEN
            FOR ii% = 1 TO 37
                IF S(ii%).fa% = formn%(j%) THEN mark1% = ii%
            NEXT ii%
        'SCR5 CALLS FOR CHANGES TO LEVEL OF SERVICE !
      CALL SCR5(mi, j%, mark1%, formn%(), S(), nls, ntlm, nelev, nash, ncurves,

        'AVGCOST CALCULATES AVERAGE ANNUAL MAINTENANCE COST PER MILEPOST SEGMENT
      CALL AVGCOST(nls, ntlm, nelev, nash, ncurves, nsf, nwf, ACOST)
            pcost = ACOST

      LOCATE 8, 52, 0
      PRINT "PROPOSED"
      LOCATE 9, 54, 0
      PRINT "COST"

      LOCATE 11, 51, 0
      PRINT USING "$$######"; ACOST
            CALL AVGCOST(ols, otlm, nelev, nash, ncurves, nsf, nwf, ACOST)
            curcost = ACOST
            netcost = pcost - curcost
      LOCATE 8, 62, 0
      PRINT "CURRENT"
      LOCATE 9, 63, 0
      PRINT "COST"
      LOCATE 11, 61, 0
      PRINT USING "$$######"; ACOST
      LOCATE 8, 72, 0
      PRINT "NET"
      LOCATE 9, 72, 0
      PRINT "COST"
      LOCATE 11, 69, 0
      PRINT USING "$$######"; netcost
      netc(j%) = netcost
  END IF
 NEXT j%


400
 CLS
 name$ = "filename"
 FILE$ = "c:\highway"
 LOCATE 8, 10, 0
 PRINT "Please specify the drive and subdirectory "
 LOCATE 9, 10, 0
 PRINT "for the output file    ["; FILE$; "]    ";
 LINE INPUT subd$
 subd$ = RTRIM$(LTRIM$(subd$))
 IF subd$ = "" THEN subd$ = FILE$
 LOCATE 12, 10, 0
 PRINT "Filename ["; name$; "]?  ";
 LINE INPUT filname$
 filname$ = RTRIM$(LTRIM$(filname$))
 IF filname$ = "" THEN filname$ = name$

  IF LEN(subd$) = 1 THEN
    subd$ = subd$ + ":\"
  ELSEIF RIGHT$(subd$, 1) <> "\" AND MID$(subd$, LEN(subd$) - 1, 1) <> ":" THEN
    subd$ = subd$ + "\"
  ELSE
    subd$ = subd$
  END IF

 filename$ = subd$ + filname$
 LOCATE 15, 10, 0
 PRINT "This is the path for the output file..."
 LOCATE 17, 25, 0
 PRINT filename$
 LOCATE 19, 10, 0
 INPUT "Is this the intended path?  (Y/N)   ", an$
 IF LCASE$(an$) <> "y" THEN 400
CLS


            TSUM = 0: TBEN = 0: TOTC = 0: TOT = 0

            OPEN filename$ FOR OUTPUT AS #5
            IF D$ = "d" OR D$ = "D" THEN 136
            GOTO 139
136         PRINT #5,
            PRINT #5, TAB(10); "DISTRICT"; district%; "CHANGE IN LEVEL OF SERVIC
            GOTO 138
139         PRINT #5,


            PRINT #5, TAB(10); "STATEWIDE CHANGE IN LEVEL OF SERVICE FROM"; lold
```

```
138         PRINT #5,
            PRINT #5, TAB(1); "SEGMENT ROUTE  FA   MILES  ADT  NET BENEFITS  NET
            PRINT #5,
          FOR j% = 1 TO fmn%
            IF j% > 1 THEN PRINT
            adt = wsum(j%) / msum(j%)
            IF netc(j%) = 0 THEN netc(j%) = .01
            IF yben(j%) > 0 THEN 66
            IF yben(j%) = 0 THEN 23
66          IF yben(j%) < 0 AND netc(j%) < 0 THEN bcratio = netc(j%) / yben(j%)
            IF yben(j%) > 0 AND netc(j%) > 0 THEN bcratio = yben(j%) / netc(j%)
            GOTO 62
61          bcratio = 0
62          PRINT #5, USING "######"; segment(j%);
            PRINT #5, USING "   ###"; r%(j%);
            PRINT #5, " "; formn%(j%);
            PRINT #5, " "; USING "###.#"; msum(j%);
            PRINT #5, " "; USING "#####."; adt;
            PRINT #5, " "; USING "$$#######."; yben(j%);
            PRINT #5, " "; USING "$$#######."; netc(j%);
            PRINT #5, "   "; USING "##.##"; bcratio;
            PRINT #5, "   "; USING "###.#"; bmfa(j%);
            PRINT #5, "   "; USING "###.#"; fmfa(j%)
            TSUM = msum(j%) + TSUM
            TBEN = TBEN + yben(j%)
            TOTC = TOTC + netc(j%)
            TOT = TOTIME(j%) + TOT
CLS



23  NEXT
            PRINT #5,
            PRINT #5,
            PRINT #5, " TOTAL MILES"; "      TOTAL TIME"; "      TOTAL BENEFITS
            IF o$ = "d" THEN
               PRINT #5, TAB(18); "(ADDITIONAL"
               PRINT #5, TAB(17); "DAYS OF TRAVEL)"
             ELSE
               PRINT #5, TAB(18); "(DAYS SAVED)"
            END IF
            PRINT #5,
            PRINT #5, "   "; USING "####.#"; TSUM;
            PRINT #5, "        "; USING "######.#"; TOT;
            PRINT #5, "         "; USING " $$######."; TBEN;
            PRINT #5, "        "; USING "$$######."; TOTC
            CLOSE #5
            CLS
            LOCATE 10, 10, 0
PRINT "Do you want to print file "; filename$;
INPUT yn$
CLS
IF yn$ = "Y" OR yn$ = "y" THEN SHELL "COPY " + filename$ + " PRN > NULL"
LOCATE 12, 10, 0
PRINT "Do you want to display file "; filename$;
INPUT yn$
CLS
IF yn$ = "Y" OR yn$ = "y" THEN SHELL "type " + filename$ + " | more"
PRINT
LOCATE 25, 10, 0
```

```
PRINT "Press any key to continue"
DO
LOOP WHILE INKEY$ = ""
CLS
LOCATE 13, 10, 0
            INPUT ; "DO YOU WISH TO RUN THE PROGRAM AGAIN? ", ii$
            IF ii$ = "Y" OR ii$ = "y" THEN
            CLS
            CLEAR
            GOTO 10
            END IF
102 END


SUB AVGCOST (nls, ntlm, nelev, nash, ncurves, nsf, nwf, ACOST)

ACOST = -104424.75# + 359.16969# * nash * nls - .03117893# * ntlm * nelev + .161

END SUB

FUNCTION comfort (t)
IF t < 2 THEN
        comfort = 0
ELSEIF t > 2 THEN
        comfort = .0833 * (t - 2)
END IF
END FUNCTION

FUNCTION comfort1 (t)
IF t < 1 THEN comfort1 = 0
IF t > 1 THEN comfort1 = .0833 * (t - 1)
END FUNCTION

FUNCTION delay (t)
IF t < 2 THEN delay = 0
IF t > 2 THEN delay = .16667 * (t - 2)
END FUNCTION

FUNCTION delay1 (t)
IF t < 1 THEN delay1 = 0
IF t > 1 THEN delay1 = .16667 * (t - 1)
END FUNCTION

SUB deltime (va() AS SINGLE, rf() AS SINGLE, sigd() AS SINGLE, sigw() AS SINGLE,

    RANDOMIZE TIMER

    IF lnew > lold THEN
        lnew1 = lold
        lold1 = lnew
        lnew = lnew1
        lold = lold1
        o$ = "d"
    END IF

    jbcS1 = (2 * lold - 1)                  'if st(j)=1  --> jb=0
    jbpS1 = (2 * lnew - 1)
    jbcS0 = (2 * lold - 1) + 1              'if st(j)=0  --> jb=1
    jbpS0 = (2 * lnew - 1) + 1
    ttimeS1 = 0: ttimeS0 = 0
    v1ctot = 0: v1ptot = 0: v2ctot = 0: v2ptot = 0
'Open "c:\itd\times3-2.txt" For Output As #6
'Print #6, "state value = 1"
'Print #6, mi; "miles"

        pnty = 2: pntx = 10: chop$ = "beef"
     FOR num = 1 TO NoSimul          'for a state value {st(j)} = 1
           veh = num
           COLOR 0, 3, 0
           IF veh = 1 THEN
             FOR v% = 1 TO 62
               PRINT
             NEXT v%
           END IF
           IF (50 * INT(veh / 50)) = veh THEN
             IF pnty = 22 THEN CLS : pnty = 2
             IF pntx = 13 THEN chop$ = "pork"
             IF pntx = 10 THEN chop$ = "beef"
             IF chop$ = "pork" THEN
                   pntx = pntx - 1
             ELSEIF chop$ = "beef" THEN
                   pntx = pntx + 1
             END IF
             IF (50 * INT(veh / 50)) = veh THEN
                pnty = pnty + 2
             END IF
             LOCATE pnty, pntx, 0
             PRINT "PROGRAM IS NOW DOING SIMULATIONS.  PLEASE CONTINUE TO WAIT !
           END IF

                    r1 = RND(1): r2 = RND(2)
                    z1 = SQR(-2 * LOG(r1)) * COS(2 * pi * r2)
                    z2 = SQR(-2 * LOG(r1)) * SIN(2 * pi * r2)
                    v1c = va(jbcS1) * rf(jbcS1) + sigw(jbcS1) * z1
                    v1p = va(jbpS1) * rf(jbpS1) + sigw(jbpS1) * z1
                    v2c = va(jbcS1) * rf(jbcS1) + sigw(jbcS1) * z2
                    v2p = va(jbpS1) * rf(jbpS1) + sigw(jbpS1) * z2
                    v1ctot = v1c + v1ctot
                    v1ptot = v1p + v1ptot
                    v2ctot = v2c + v2ctot
                    v2ptot = v2p + v2ptot
'                 If jbpS1 < jbcS1 Then v2c = v2p     '*************this line has
                    time1S1 = trip * ((1 / v1c) - (1 / v1p)) * 60 'min/car for a
                    time2S1 = trip * ((1 / v2c) - (1 / v2p)) * 60 'min/car for a
'                    If num < 100 Then Print #6, "time1", time1S1, "time2", time
                IF trip > 5 THEN GOTO line37
                    ccost1S1 = comfort1(time1S1)
                    ccost2S1 = comfort1(time2S1)
                    dcost1S1 = delay1(time1S1)
                    dcost2S1 = delay1(time2S1)
                GOTO line38
line37:
                    ccost1S1 = comfort(time1S1)
                    ccost2S1 = comfort(time2S1)
                    dcost1S1 = delay(time1S1)
                    dcost2S1 = delay(time2S1)
line38:
                    ttimeS1 = (time1S1 + time2S1) + ttimeS1
                    ccostS1 = (ccost1S1 + ccost2S1) * inflat + ccostS1
                    dcostS1 = (dcost1S1 + dcost2S1) * inflat + dcostS1
    NEXT num
    v1cave = v1ctot / NoSimul
    v1pave = v1ptot / NoSimul
    v2cave = v2ctot / NoSimul
    v2pave = v2ptot / NoSimul

    ttimeS1 = ttimeS1 / (2 * NoSimul)                           'min per "trip" mile
    ccostS1 = ccostS1 / (2 * NoSimul)                           '$ per "trip" miles
    dcostS1 = dcostS1 / (2 * NoSimul) * (work1 / 100)           '$ per "trip" miles
'            PRINT "average v1c", v1cave, "average v2c", v2cave
'            PRINT "average v1p", v1pave, "average v2p", v2pave
'            PRINT "total time for the given segment length "; "mi"; " =", ttime
'            PRINT "average comfort cost =", ccostS1
'            PRINT "average delay cost =", dcostS1
'INPUT sffsufs
'*************************************************************************
        CLS
        pnty = 2: pntx = 10: chop$ = "beef"
    FOR num = 1 TO NoSimul                   'for a state value {st(j)} = 0
           veh = num
           COLOR 0, 3, 0
           IF (50 * INT(veh / 50)) = veh THEN
             IF pnty = 22 THEN CLS : pnty = 2
             IF pntx = 13 THEN chop$ = "pork"
             IF pntx = 10 THEN chop$ = "beef"
             IF chop$ = "pork" THEN
                   pntx = pntx - 1
             ELSEIF chop$ = "beef" THEN
                   pntx = pntx + 1
             END IF
             IF (50 * INT(veh / 50)) = veh THEN
                pnty = pnty + 2
             END IF
             LOCATE pnty, pntx, 0
             PRINT "PROGRAM IS NOW DOING SIMULATIONS.  PLEASE CONTINUE TO WAIT !
           END IF

                    r1 = RND(1): r2 = RND(2)
                    z1 = SQR(-2 * LOG(r1)) * COS(2 * pi * r2)
                    z2 = SQR(-2 * LOG(r1)) * SIN(2 * pi * r2)
                    v1c = va(jbcS0) * rf(jbcS0) + sigw(jbcS0) * z1
                    v1p = va(jbpS0) * rf(jbpS0) + sigw(jbpS0) * z1
                    v2c = va(jbcS0) * rf(jbcS0) + sigw(jbcS0) * z2
                    v2p = va(jbpS0) * rf(jbpS0) + sigw(jbpS0) * z2
'                If jbpS0 < jbcS0 Then v2c = v2p     '*************this line has
                    time1S0 = trip * ((1 / v1c) - (1 / v1p)) * 60  'min/car for
                    time2S0 = trip * ((1 / v2c) - (1 / v2p)) * 60  'min/car for
                IF trip > 5 THEN GOTO line137
                    ccost1S0 = comfort1(time1S0)
                    ccost2S0 = comfort1(time2S0)
                    dcost1S0 = delay1(time1S0)
                    dcost2S0 = delay1(time2S0)
                GOTO line138
line137:
                    ccost1S0 = comfort(time1S0)
                    ccost2S0 = comfort(time2S0)
                    dcost1S0 = delay(time1S0)
                    dcost2S0 = delay(time2S0)
line138:
```

```
                    ttimeS0 = (time1S0 + time2S0) + ttimeS0
                    ccostS0 = (ccost1S0 + ccost2S0) * inflat + ccostS0
                    dcostS0 = (dcost1S0 + dcost2S0) * inflat + dcostS0
    NEXT num
    ttimeS0 = ttimeS0 / (2 * NoSimul)                       'min per "trip" mile
    ccostS0 = ccostS0 / (2 * NoSimul)                       '$ per "trip" miles
    dcostS0 = dcostS0 / (2 * NoSimul) * (work1 / 100)       '$ per "trip" miles

'            PRINT "average v1c", v1cave, "average v2c", v2cave
'            PRINT "average v1p", v1pave, "average v2p", v2pave
'            PRINT "total time for the given segment length "; "mi"; " =", ttime
'            PRINT "average comfort cost =", ccostS0
'            PRINT "average delay cost =", dcostS0
'INPUT sffsufs

    IF o$ = "d" THEN
        lold = lnew1
        lnew = lold1
    END IF
END SUB

SUB indata (va(), rf(), sigd(), sigw())

        va(1) = 50: rf(1) = .78: sigd(1) = 4.2: sigw(1) = 5.1
        va(2) = 41: rf(2) = .79: sigd(2) = 5.8: sigw(2) = 4.1

        va(3) = 50: rf(3) = .7:  sigd(3) = 4.2: sigw(3) = 5.1
        va(4) = 41: rf(4) = .75: sigd(4) = 5.8: sigw(4) = 4.7

        va(5) = 50: rf(5) = .58: sigd(5) = 4.2: sigw(5) = 4.2
        va(6) = 41: rf(6) = .58: sigd(6) = 5.8: sigw(6) = 4!

        va(7) = 50: rf(7) = .5: sigd(7) = 4.2: sigw(7) = 4!
        va(8) = 41: rf(8) = .5: sigd(8) = 5.8: sigw(8) = 4!



END SUB

SUB inflation (NoSimul, inflat, yr$)
yrnot:
    CLS
    LOCATE 3, 23, 0
    yr$ = "1991-92"
    IF LEN(yr$) <> 7 THEN GOTO yrnot
    yr1 = VAL(MID$(yr$, 3, 2))
    yr2 = VAL(MID$(yr$, 6, 2))
    wage1 = 5.104987 + ((yr1 - 77) * .319739)
    wage2 = 5.104987 + ((yr2 - 77) * .319739)
    wage = (wage1 + wage2) * .5
    LOCATE 11, 9, 0
    PRINT "Extrapolated average wage for year "; yr$; " is $ ";
    PRINT USING "##.##"; wage;
    PRINT " per hour"
    LOCATE 16, 22, 0
    INPUT ; "Do you want to change this (Y/N) ", jb$
    IF jb$ = "Y" OR jb$ = "y" THEN
                  CLS
                  LOCATE 10, 22, 0
                  INPUT "Input current Average Wage (in $/hour) = ", wage
    END IF

 inflat = wage / 5.26


COLOR 0, 7, 0
LOCATE 11, 34, 0
CLS
PRINT "PLEASE  WAIT "
LOCATE 13, 30, 0
PRINT "Simulating "; NoSimul; " Cars"

END SUB

SUB info (FILES$, t() AS trans, S() AS steady) STATIC

dirchange:  file1$ = FILES$ + "\nsteady.dat"
            file2$ = FILES$ + "\Trans.dat"
OPEN file1$ FOR INPUT AS #1

FOR ii% = 1 TO 37
      INPUT #1, S(ii%).obs, S(ii%).fa%, S(ii%).ls, S(ii%).ash, S(ii%).elev, S(ii
      NEXT ii%
CLOSE #1

OPEN file2$ FOR INPUT AS #2
FOR jj% = 1 TO 42
      INPUT #2, t(jj%).obst, t(jj%).dist, t(jj%).yr12, t(jj%).yash, t(jj%).delts
NEXT jj%
CLOSE #2


END SUB

SUB SCR5 (mi, j%, ii%, formn%(), S() AS steady, nls, ntlm, nelev, nash, ncurves,
jb226 = 0
nelev = S(ii%).elev
nash = S(ii%).ash
ncurves = S(ii%).curves
nsf = S(ii%).sf
nwf = S(ii%).wf

usl:
CLS
LOCATE 3, 20, 0
PRINT USING "####"; tlm;
PRINT " LANE-MILES OF ROUTE"; r%(j%); "CHANGED TO LEVEL "
LOCATE 4, 25, 0
PRINT "OF SERVICE (LOS)"; lnew
PRINT "FOR FORMAN AREA"; formn%(j%)
LOCATE 8, 7, 0
PRINT "                    CURRENT       PROPOSED"
LOCATE 9, 6, 0
PRINT "                    LANE-MILES    LANE-MILES"
LOCATE 11, 6, 0
PRINT "Level of service 1     "; : PRINT USING "###."; S(ii%).ls1tlm
LOCATE 13, 6, 0
PRINT "Level of service 2     "; : PRINT USING "###."; S(ii%).ls2tlm
LOCATE 15, 6, 0
PRINT "Level of service 3     "; : PRINT USING "###."; S(ii%).ls3tlm;
```

```
LOCATE 17, 6, 0
PRINT "Level of service 4      "; : PRINT USING "###."; S(ii%).ls4tlm;
LOCATE 19, 6, 0
PRINT "Level of service 5      "; : PRINT USING "###."; S(ii%).ls5tlm

otlm = S(ii%).ls1tlm + S(ii%).ls2tlm + S(ii%).ls3tlm + S(ii%).ls4tlm + S(ii%).ls
ols = (5 * S(ii%).ls1tlm + 4 * S(ii%).ls2tlm + 3 * S(ii%).ls3tlm + 2 * S(ii%).ls

LS1 = S(ii%).ls1tlm: LS2 = S(ii%).ls2tlm: LS3 = S(ii%).ls3tlm: LS4 = S(ii%).ls4t

IF lnew = 1 AND lold = 1 THEN LS1 = LS1
IF lnew = 1 AND lold = 2 THEN LS1 = LS1 + tlm: LS2 = LS2 - tlm
IF lnew = 1 AND lold = 3 THEN LS1 = LS1 + tlm: LS3 = LS3 - tlm
IF lnew = 1 AND lold = 4 THEN LS1 = LS1 + tlm: LS4 = LS4 - tlm
IF lnew = 1 AND lold = 5 THEN LS1 = LS1 + tlm: LS5 = LS5 - tlm

IF lnew = 2 AND lold = 1 THEN LS2 = LS2 + tlm: LS1 = LS1 - tlm
IF lnew = 2 AND lold = 2 THEN LS2 = LS2
IF lnew = 2 AND lold = 3 THEN LS2 = LS2 + tlm: LS3 = LS3 - tlm
IF lnew = 2 AND lold = 4 THEN LS2 = LS2 + tlm: LS4 = LS4 - tlm
IF lnew = 2 AND lold = 5 THEN LS2 = LS2 + tlm: LS5 = LS5 - tlm

IF lnew = 3 AND lold = 1 THEN LS3 = LS3 + tlm: LS1 = LS1 - tlm
IF lnew = 3 AND lold = 2 THEN LS3 = LS3 + tlm: LS2 = LS2 - tlm
IF lnew = 3 AND lold = 3 THEN LS3 = LS3
IF lnew = 3 AND lold = 4 THEN LS3 = LS3 + tlm: LS4 = LS4 - tlm
IF lnew = 3 AND lold = 5 THEN LS3 = LS3 + tlm: LS5 = LS5 - tlm

IF lnew = 4 AND lold = 1 THEN LS4 = LS4 + tlm: LS1 = LS1 - tlm
IF lnew = 4 AND lold = 2 THEN LS4 = LS4 + tlm: LS2 = LS2 - tlm
IF lnew = 4 AND lold = 3 THEN LS4 = LS4 + tlm: LS3 = LS3 - tlm
IF lnew = 4 AND lold = 4 THEN LS4 = LS4
IF lnew = 4 AND lold = 5 THEN LS4 = LS4 + tlm: LS5 = LS5 - tlm

IF lnew = 5 AND lold = 1 THEN LS5 = LS5 + tlm: LS1 = LS1 - tlm
IF lnew = 5 AND lold = 2 THEN LS5 = LS5 + tlm: LS2 = LS2 - tlm
IF lnew = 5 AND lold = 3 THEN LS5 = LS5 + tlm: LS3 = LS3 - tlm
IF lnew = 5 AND lold = 4 THEN LS5 = LS5 + tlm: LS4 = LS4 - tlm
IF lnew = 5 AND lold = 5 THEN LS5 = LS5

'  CHECK TO SEE IF THERE ARE ANY ROUNDING ERRORS DURING THE CALCULATIONS

        IF LS1 > -1 AND LS1 < 0 THEN LS1 = ABS(LS1)
LOCATE 11, 43, 0: PRINT USING "###."; LS1
        IF LS2 > -1 AND LS2 < 0 THEN LS2 = ABS(LS2)
LOCATE 13, 43, 0: PRINT USING "###."; LS2
        IF LS3 > -1 AND LS3 < 0 THEN LS3 = ABS(LS3)
LOCATE 15, 43, 0: PRINT USING "###."; LS3
        IF LS4 > -1 AND LS4 < 0 THEN LS4 = ABS(LS4)
LOCATE 17, 43, 0: PRINT USING "###."; LS4
        IF LS5 > -1 AND LS5 < 0 THEN LS5 = ABS(LS5)
LOCATE 19, 43, 0: PRINT USING "###."; LS5


ntlm = LS1 + LS2 + LS3 + LS4 + LS5
nls = (5 * LS1 + 4 * LS2 + 3 * LS3 + 2 * LS4 + LS5) / ntlm

END SUB

SUB scrn1
```

```
CLS
COLOR 7, 0, 0
FOR mi% = 1 TO 60
  PRINT
NEXT mi%
LOCATE 8, 24, 0
PRINT "DELAY & DISCOMFORT COST ANALYSIS"
LOCATE 11, 39, 0
PRINT "By"
LOCATE 13, 31, 0
PRINT "Dr. Donald  F. Haber"
LOCATE 14, 40, 0
PRINT "&"
LOCATE 15, 33, 0
PRINT "Umesh  S. Limaye'"
LOCATE 24, 27, 0
PRINT "Press any key to continue..."
DO
LOOP WHILE INKEY$ = ""

END SUB

SUB scrn3 (lx%, lnew, FILES$, work1, mi, D$, district%)
 CLS
 LOCATE 5, 30, 0
 PRINT "DATA  INPUT  SCREEN"
 LOCATE 9, 10, 0
 PRINT "This program will calculate the benefits and costs due to a"
 LOCATE 10, 10, 0
 PRINT "change in winter maintenance levels.  The benefits accrue from"
 LOCATE 11, 10, 0
 PRINT "a decrease in time of travel only and not from decreased accidents."
116 LOCATE 15, 10, 0
 PRINT "Do you wish to calculate the benefits and costs statewide or for"
 LOCATE 16, 10, 0
 INPUT "only one district?  Input 'd' for district or 's' for statewide.  ", D$
 IF D$ = "d" OR D$ = "D" THEN 117
 IF D$ = "s" OR D$ = "S" THEN 91
 CLS
GOTO 116

'**** STATEWIDE INFORMATION

91 CLS
 LOCATE 8, 10, 0
 PRINT "This program can calculate the benefits and costs statewide. "
89 LOCATE 11, 10, 0
 PRINT "What level of service do you wish to change statewide?"
 LOCATE 13, 13, 0
 INPUT "Enter 1,2,3,4 or 5.   ", lx%
 IF lx% = 1 OR lx% = 2 OR lx% = 3 OR lx% = 4 OR lx% = 5 THEN 90
 CLS
 LOCATE 8, 10, 0
 PRINT "Please enter the level of service again.                        "
GOTO 89
90 LOCATE 16, 10, 0
 PRINT "Please input the new level of service.   "
 LOCATE 18, 13, 0
 INPUT "Enter 1,2,3,4 or 5.   ", lnew
 IF lnew = 1 OR lnew = 2 OR lnew = 3 OR lnew = 4 OR lnew = 5 THEN 92
 CLS
 LOCATE 13, 10, 0
 PRINT "Please enter the new level of service again."
GOTO 90

'**** DISTRICT INFORMATION

117 CLS
118 LOCATE 8, 10, 0
 INPUT "Which one district is desired (1-6)    ", district%
 IF district% = 1 OR district% = 2 OR district% = 3 OR district% = 4 OR district
 CLS
 LOCATE 5, 10, 0
 PRINT "Please enter the district again."
GOTO 118
112
 LOCATE 11, 10, 0
 PRINT "What level of service do you wish to change district wide?"
 LOCATE 13, 13, 0
 INPUT "Enter 1,2,3,4 or 5.    ", lx%
 IF lx% = 1 OR lx% = 2 OR lx% = 3 OR lx% = 4 OR lx% = 5 THEN 110
 CLS
 LOCATE 8, 10, 0
 PRINT "Please enter the level of service again."
GOTO 112
110 LOCATE 16, 10, 0
 PRINT "Please input the new level of service"
 LOCATE 18, 13, 0
 INPUT "Enter 1,2,3,4 or 5.    ", lnew
 IF lnew = 1 OR lnew = 2 OR lnew = 3 OR lnew = 4 OR lnew = 5 THEN 92
 CLS
 LOCATE 13, 10, 0
 PRINT "Please enter the new level of service again."
GOTO 110

'**** DISTRICT AND STATEWIDE INFORMATION

92
 IF lnew = lx% THEN
   CLS
   LOCATE 5, 10, 0
   PRINT "The current los and new los are equal and will result in a"
   LOCATE 6, 10, 0
   PRINT "benefit cost ratio of 0."
   LOCATE 8, 10, 0
   PRINT "Please enter the values again. "
   IF D$ = "d" OR D$ = "D" THEN 112
   IF D$ = "s" OR D$ = "S" THEN 89
 END IF

CLS
 LOCATE 9, 10, 0
 PRINT "Input the maximum segment length for change (the default is 30 mi)"
 LOCATE 10, 15, 0
 INPUT "", mi
 IF mi = 0 THEN mi = 30
 LOCATE 11, 10, 0
 PRINT "The maximum segment length for change has been set to"; mi
 LOCATE 12, 10, 0
 PRINT "miles."
 LOCATE 14, 15, 0
 INPUT "Is this the intended value?  (Y/N)   ", intended$
 IF LCASE$(intended$) = "y" THEN 94
GOTO 92

94 LOCATE 18, 10, 0
 PRINT "Input the percentage of commuters (do not use % sign it is automatic) "
 LOCATE 19, 15, 0
 INPUT "", work1
 IF work1 <= 100 AND work1 >= 0 THEN 93
 CLS
 LOCATE 12, 10, 0
 PRINT "The percentage of commuters must be between 0 and 100."
GOTO 94
93 LOCATE 19, 17, 0
 PRINT "%"


931
 CLS
 FILES$ = "c:\highway"
 LOCATE 11, 10, 0: PRINT "Please enter the directory and subdirectory"
 LOCATE 13, 10, 0: PRINT "where the data file is located  ["; FILES$; "] ";
    LINE INPUT files1$
    IF files1$ <> "" THEN FILES$ = files1$
    FILES$ = LTRIM$(RTRIM$(FILES$))

  trigger$ = "ere"
  IF LEN(FILES$) = 1 THEN
    FILES$ = FILES$ + ":"
    trigger$ = "yep"
   ELSEIF RIGHT$(FILES$, 1) = ":" THEN
     trigger$ = "yep"
   ELSEIF RIGHT$(FILES$, 1) = "\" AND MID$(FILES$, LEN(FILES$) - 1, 1) = ":" THE
    FILES$ = MID$(FILES$, 1, LEN(FILES$) - 1)
    trigger$ = "yep"
   ELSEIF RIGHT$(FILES$, 1) = "\" THEN
    FILES$ = MID$(FILES$, 1, LEN(FILES$) - 1)
    trigger$ = "nope"
   ELSE
    FILES$ = FILES$
    trigger$ = "nope"
  END IF

 LOCATE 15, 10, 0
 PRINT "This is the directory and subdirectory for the data file..."
 LOCATE 17, 25, 0
  IF trigger$ = "yep" THEN
    FFILES$ = FILES$ + "\"
   ELSEIF trigger$ = "nope" THEN
    FFILES$ = FILES$
  END IF
 PRINT FFILES$
 LOCATE 19, 10, 0
 INPUT "Is this the intended path?  (Y/N)   ", ans$
 IF LCASE$(ans$) <> "y" THEN 931

 CLS
END SUB
```

```
SUB wadt1 (segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), l

'ROUTINE TO FIND COMPUTE MILEAGE WEIGHTED ADT AND FOREMAN AREA FOR STATEWIDE ANA
'THE DATA FILE IS ASSUMED TO BE SORTED BY SEGMENT CODE AND BEGINNING MILEPOST
'ALLOWANCE IS MADE FOR UPTO 10 FOREMAN AREAS TO BE CROSSED
'MAIN VARIABLES ARE AS FOLLOWS:
'             fmn%      = FOREMAN AREA COUNTER
'             FORMN(I) = FOREMAN AREA (I)
'             MSUM(I)  = CUMULATIVE MILEAGE SUM FOR FOREMAN AREA (I)
'             WSUM(I)  = CUMULATIVE WEIGHTED SUM OF ADT'S FOR FOREMAN AREA (I)

CLS
LOCATE 10, 15, 0
PRINT "PROGRAM IS SEARCHING FOR DESIRED ROAD SEGMENTS AND LOS"
LOCATE 12, 15, 0
PRINT "THE PROGRAM WILL ALSO PERFORM SIMULATIONS DURING THIS PERIOD"
LOCATE 14, 15, 0
PRINT "PLEASE WAIT !!"

'OPEN DATA FILE OF INPUT
OPEN FILES$ + "\newuofi.dat" FOR INPUT AS #3

INPUT #3, route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes

'SET INITIAL FOREMAN AREA AND COUNTER
fmn% = 0
miles = 0

41 IF lx% <> ll% GOTO 43
   GOTO 42
43 IF EOF(3) THEN 49
   INPUT #3, route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
   GOTO 41
42 fmn% = 1
        formn%(fmn%) = fa%
        r%(fmn%) = route%
        st(fmn%) = state
        l(fmn%) = ll%
        segment(fmn%) = segment%
        msum(fmn%) = 0
        wsum(fmn%) = 0
        tlnm(fmn%) = 0                              '99999
        bmfa(fmn%) = bm
44 miles = (em - bm)
      IF (msum(fmn%) + miles) > mi THEN 45
      msum(fmn%) = msum(fmn%) + miles
      wsum(fmn%) = wsum(fmn%) + (miles) * adt
      st(fmn%) = state
      tlnm(fmn%) = tlnm(fmn%) + lanes * miles         '99999
      em1 = em
GOTO 46
45
      m2 = (msum(fmn%) + miles) - mi
      m1 = mi - msum(fmn%)
      msum(fmn%) = mi
      wsum(fmn%) = wsum(fmn%) + m1 * adt
      tlnm(fmn%) = tlnm(fmn%) + lanes * m1          '99999
      m2counter = 1
444
      yonder$ = "howdy"
```

```
            IF m2 > mi THEN
               fmn% = fmn% + 1
               formn%(fmn%) = fa%
               m2 = m2 - mi
               msum(fmn%) = mi
               wsum(fmn%) = mi * adt
               tlnm(fmn%) = lanes * mi         '99999
               r%(fmn%) = route%
               st(fmn%) = state
               segment(fmn%) = segment%
                  IF m2counter = 1 THEN
                    bmfa(fmn%) = bm + m1
                  ELSEIF m2counter > 1 THEN
                   bmfa(fmn%) = bmfa(fmn% - 1) + mi
                  END IF
               m2counter = m2counter + 1
               bm = bmfa(fmn%)
               m1 = mi
               yonder$ = "howdy"
            ELSEIF m2 <= mi THEN
               fmn% = fmn% + 1
               formn%(fmn%) = fa%
               msum(fmn%) = m2
               wsum(fmn%) = m2 * adt
               tlnm(fmn%) = tlnm(fmn%) + lanes * m2          '99999
               r%(fmn%) = route%
               st(fmn%) = state
               segment(fmn%) = segment%
               bmfa(fmn%) = bm + m1
               em1 = em
               yonder$ = "doody"
             END IF
IF yonder$ = "howdy" THEN 444

GOTO 46
47    fmn% = fmn% + 1
      formn%(fmn%) = fa%
      r%(fmn%) = route%
      st(fmn%) = state
      l(fmn%) = ll%
      segment(fmn%) = segment%
      bmfa(fmn%) = bm
      miles = (em - bm)
    IF miles < mi THEN 48
GOTO 45
48  msum(fmn%) = msum(fmn%) + miles
    wsum(fmn%) = wsum(fmn%) + (miles) * adt
    tlnm(fmn%) = tlnm(fmn%) + lanes * miles         '99999
    em1 = em
46 IF EOF(3) THEN 49
    INPUT #3, route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
    IF ll% <> lx% THEN 46
    IF r%(fmn%) = route% AND formn%(fmn%) = fa% AND segment% = segment(fmn%) AND
    msum(fmn% + 1) = 0
    wsum(fmn% + 1) = 0
    tlnm(fmn% + 1) = 0
GOTO 47

49 CLOSE #3                        'CLOSE INPUT FILE
```

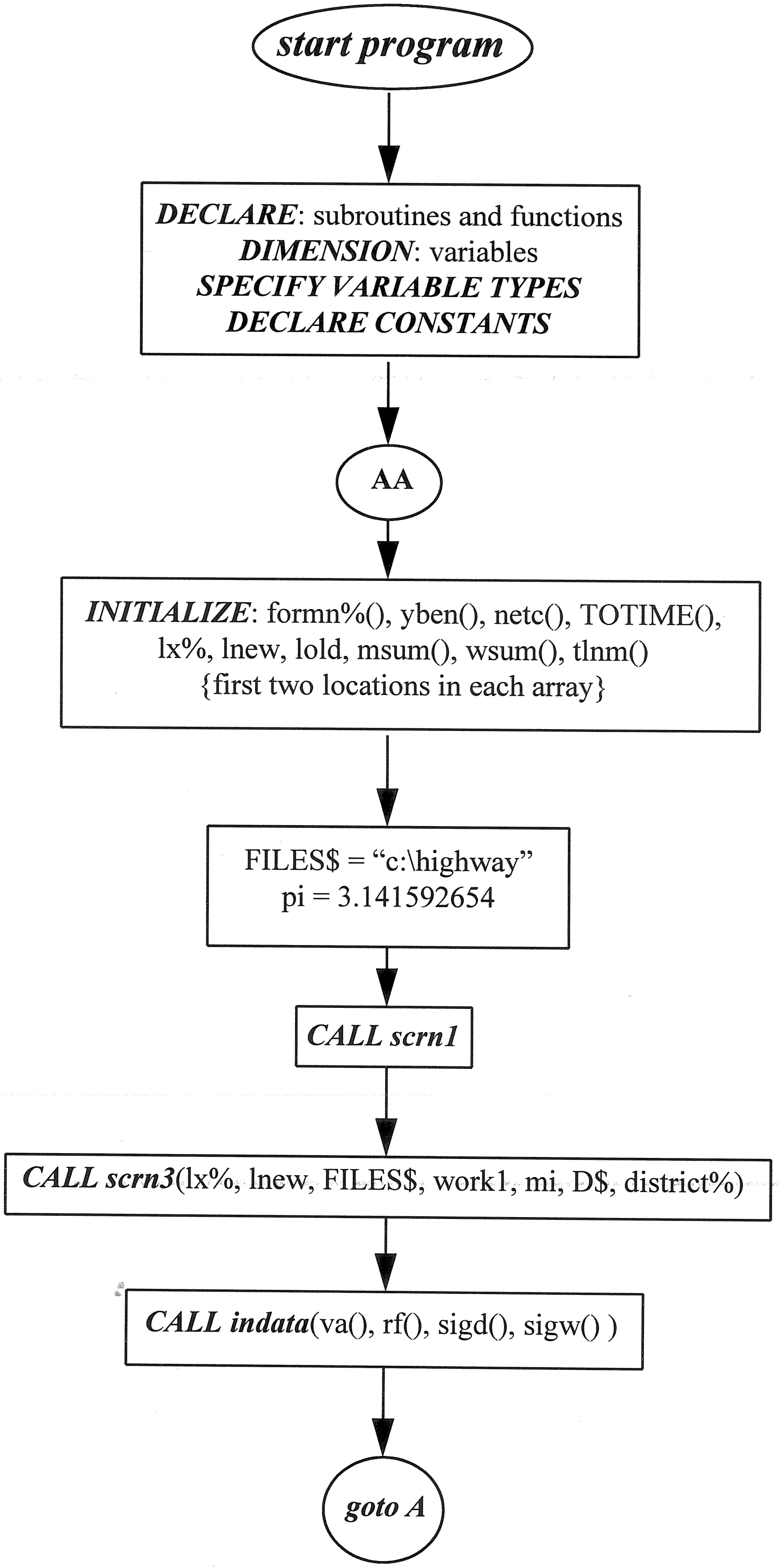
```
END SUB

SUB wadt2 (segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), l
'ROUTINE TO FIND COMPUTE MILEAGE WEIGHTED ADT AND FOREMAN AREA FOR A SINGLE DIST
'THE DATA FILE IS ASSUMED TO BE SORTED BY SEGMENT CODE AND BEGINNING MILEPOST
'ALLOWANCE IS MADE FOR UPTO 10 FOREMAN AREAS TO BE CROSSED
'MAIN VARIABLES ARE AS FOLLOWS:
'             fmn%      = FOREMAN AREA COUNTER
'             FORMN(I) = FOREMAN AREA (I)
'             MSUM(I)  = CUMULATIVE MILEAGE SUM FOR FOREMAN AREA (I)
'             WSUM(I)  = CUMULATIVE WEIGHTED SUM OF ADT'S FOR FOREMAN AREA (I)

CLS
LOCATE 10, 15, 0
PRINT "PROGRAM IS SEARCHING FOR DESIRED ROAD SEGMENTS AND LOS"
LOCATE 12, 15, 0
PRINT "THE PROGRAM WILL ALSO PERFORM SIMULATIONS DURING THIS PERIOD"
LOCATE 14, 15, 0
PRINT "PLEASE WAIT !!"
fmn% = 0
miles = 0

'OPEN DATA FILE OF INPUT
OPEN FILES$ + "\newuofi.dat" FOR INPUT AS #3

INPUT #3, route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
'SET INITIAL FOREMAN AREA AND COUNTER

132     IF lx% <> ll% GOTO 129
        IF district% <> INT(fa% / 100) GOTO 129
        GOTO 131

129     IF EOF(3) THEN 122
        INPUT #3, route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
        GOTO 132

131 fmn% = 1
        formn%(fmn%) = fa%
        r%(fmn%) = route%
        st(fmn%) = state
        l(fmn%) = ll%
        segment(fmn%) = segment%
        msum(fmn%) = 0
        wsum(fmn%) = 0
        tlnm(fmn%) = 0
        bmfa(fmn%) = bm

126   miles = (em - bm)
      IF (msum(fmn%) + miles) > mi THEN 151
      msum(fmn%) = msum(fmn%) + miles
      wsum(fmn%) = wsum(fmn%) + (miles) * adt
      tlnm(fmn%) = tlnm(fmn%) + lanes * miles          '99999
      st(fmn%) = state
      em1 = em
      GOTO 119
151
      m2 = (msum(fmn%) + miles) - mi
      m1 = mi - msum(fmn%)
      msum(fmn%) = mi
      wsum(fmn%) = wsum(fmn%) + m1 * adt
      tlnm(fmn%) = tlnm(fmn%) + lanes * m1          '99999
      m2counter = 1
555
      yonder$ = "howdy"
            IF m2 > mi THEN
               fmn% = fmn% + 1
               formn%(fmn%) = fa%
               m2 = m2 - mi
               msum(fmn%) = mi
               wsum(fmn%) = mi * adt
               tlnm(fmn%) = lanes * mi                          '9999
               r%(fmn%) = route%
               st(fmn%) = state
               segment(fmn%) = segment%
                  IF m2counter = 1 THEN
                    bmfa(fmn%) = bm + m1
                  ELSEIF m2counter > 1 THEN
                   bmfa(fmn%) = bmfa(fmn% - 1) + mi
                  END IF
               m2counter = m2counter + 1
               bm = bmfa(fmn%)
               m1 = mi
               yonder$ = "howdy"
            ELSEIF m2 <= mi THEN
               fmn% = fmn% + 1
               formn%(fmn%) = fa%
               msum(fmn%) = m2
               wsum(fmn%) = m2 * adt
               tlnm(fmn%) = tlnm(fmn%) + lanes * m2             '99999
               r%(fmn%) = route%
               st(fmn%) = state
               segment(fmn%) = segment%
               bmfa(fmn%) = bm + m1
               em1 = em
               yonder$ = "doody"
             END IF
IF yonder$ = "howdy" THEN 555

GOTO 119
125   fmn% = fmn% + 1
      formn%(fmn%) = fa%
      r%(fmn%) = route%
      st(fmn%) = state
      l(fmn%) = ll%
      segment(fmn%) = segment%
      bmfa(fmn%) = bm
      miles = (em - bm)
   IF miles < mi THEN 153
      GOTO 151
153  msum(fmn%) = msum(fmn%) + miles
     wsum(fmn%) = wsum(fmn%) + (miles) * adt
     tlnm(fmn%) = tlnm(fmn%) + lanes * miles          '99999
     em1 = em
119 IF EOF(3) THEN 122
    INPUT #3, route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
    IF ll% <> lx% THEN 119
    IF district% <> INT(fa% / 100) THEN 119
    IF r%(fmn%) = route% AND formn%(fmn%) = fa% AND segment% = segment(fmn%) AND
    msum(fmn% + 1) = 0
    wsum(fmn% + 1) = 0
    tlnm(fmn% + 1) = 0
    GOTO 125

122 CLOSE #3                                'CLOSE INPUT FILE
END SUB
```

# APPENDIX C

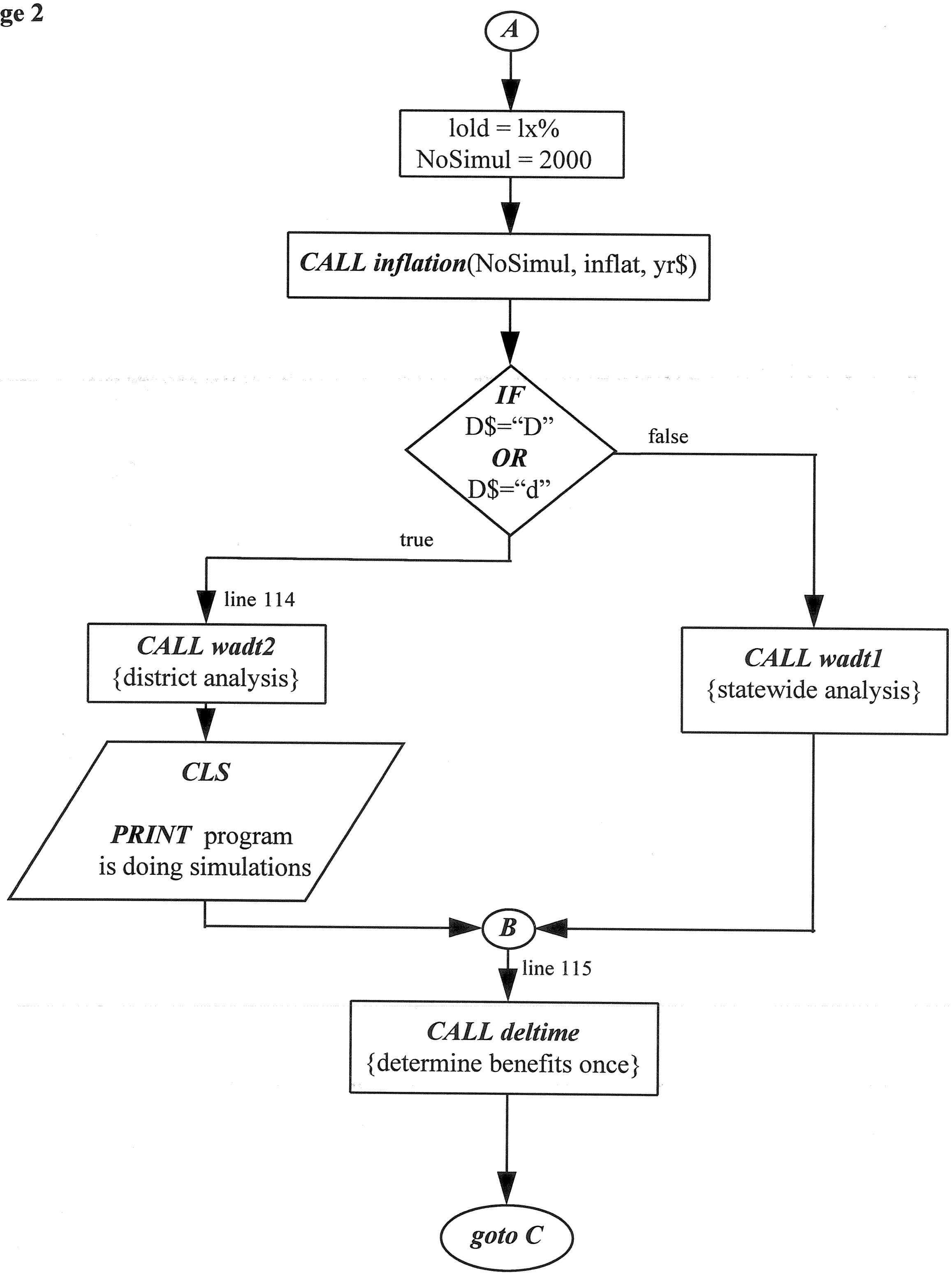
## FLOW CHART FOR BENEFIT / COST

## VISUAL BASIC PROGRAM

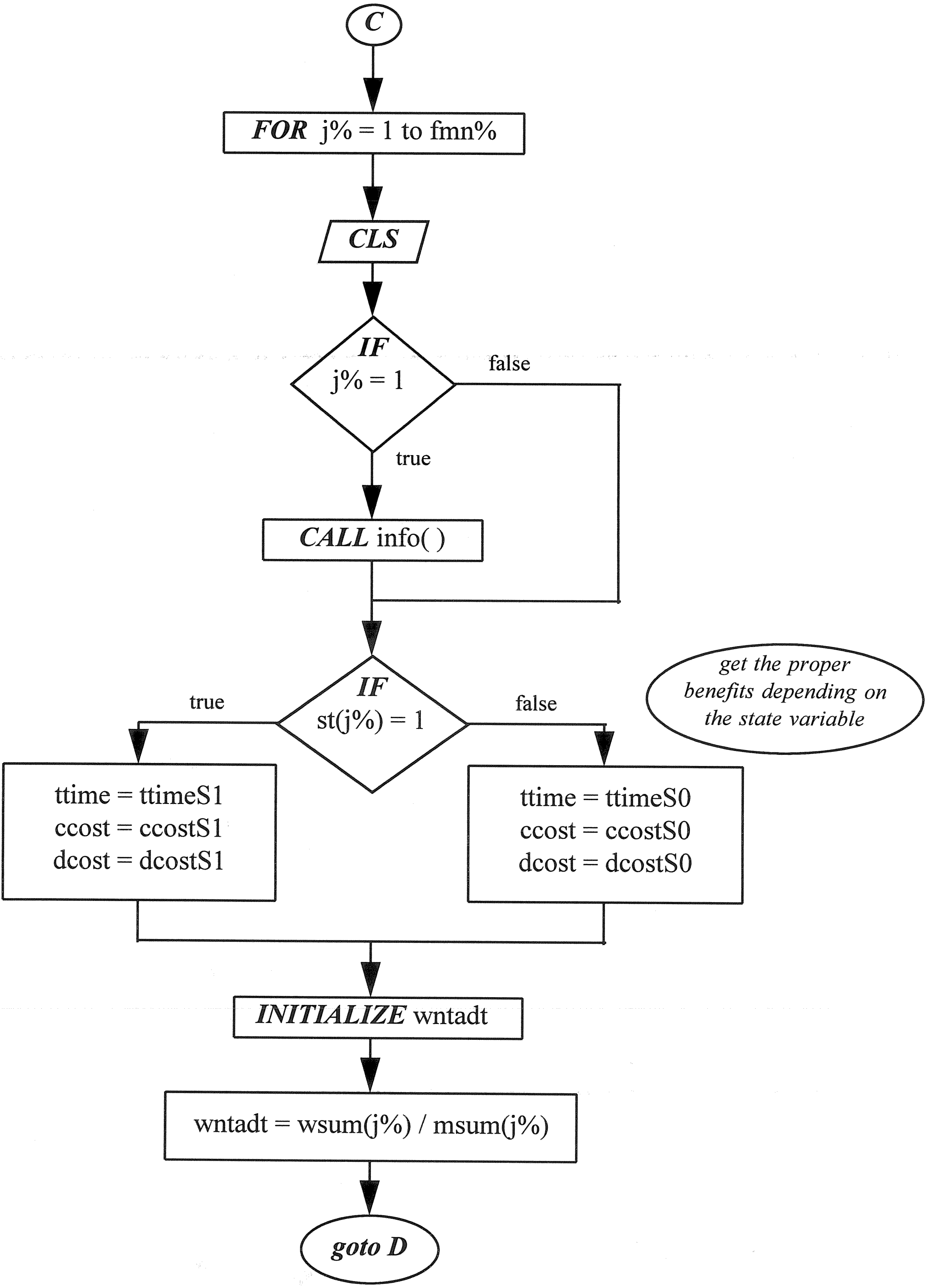
## Main Program
**page 1**


start program
DECLARE: subroutines and functions
DIMENSION: variables
SPECIFY VARIABLE TYPES
DECLARE CONSTANTS
AA
INITIALIZE: formn%(), yben(), netc(), TOTIME(),
lx%, lnew, lold, msum(), wsum(), tlnm()
{first two locations in each array}
FILES$ = "c:\highway"
pi = 3.141592654
CALL scrn1
CALL scrn3(lx%, lnew, FILES$, work1, mi, D$, district%)
CALL indata(va(), rf(), sigd(), sigw() )
goto A

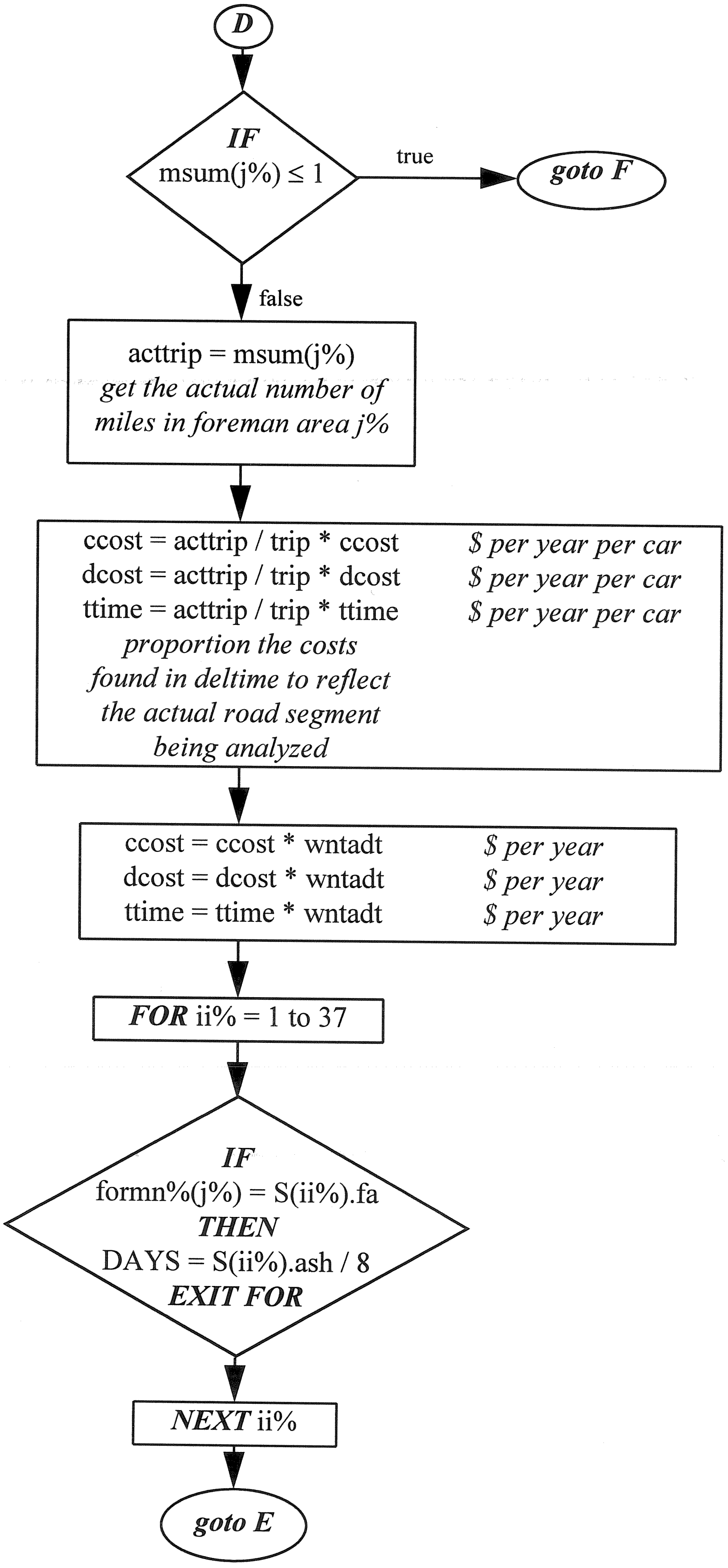
## Main Program
**page 2**



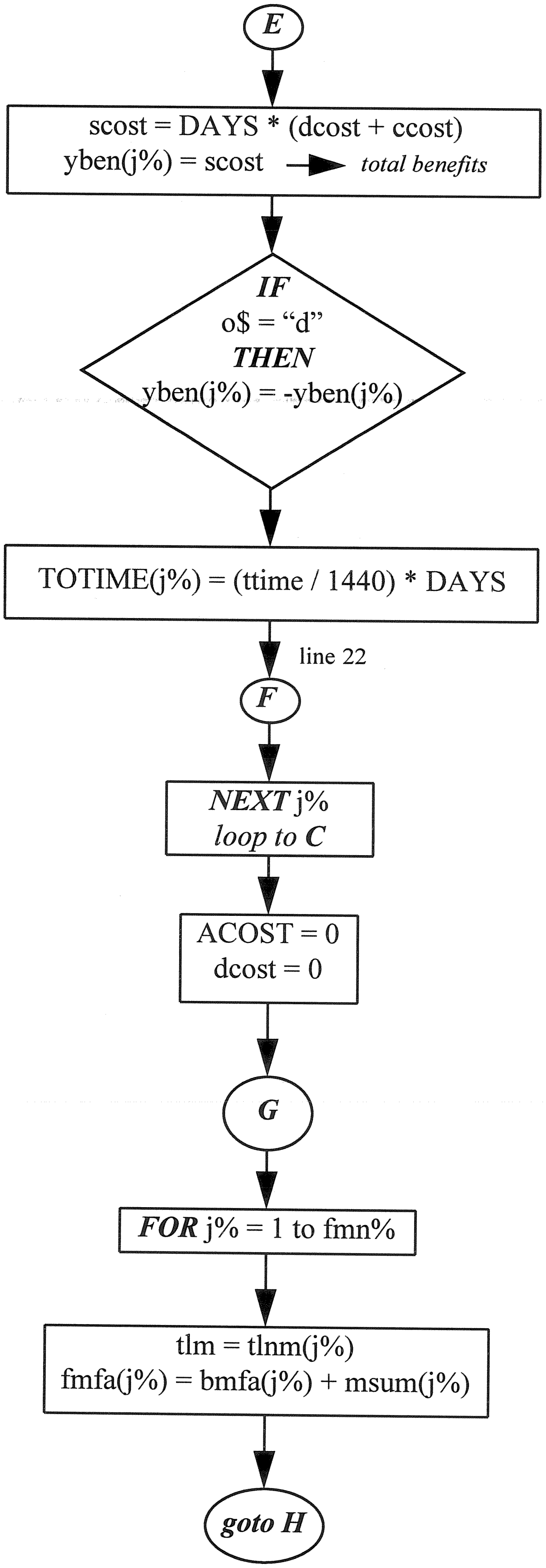
## Main Program
**page 3**



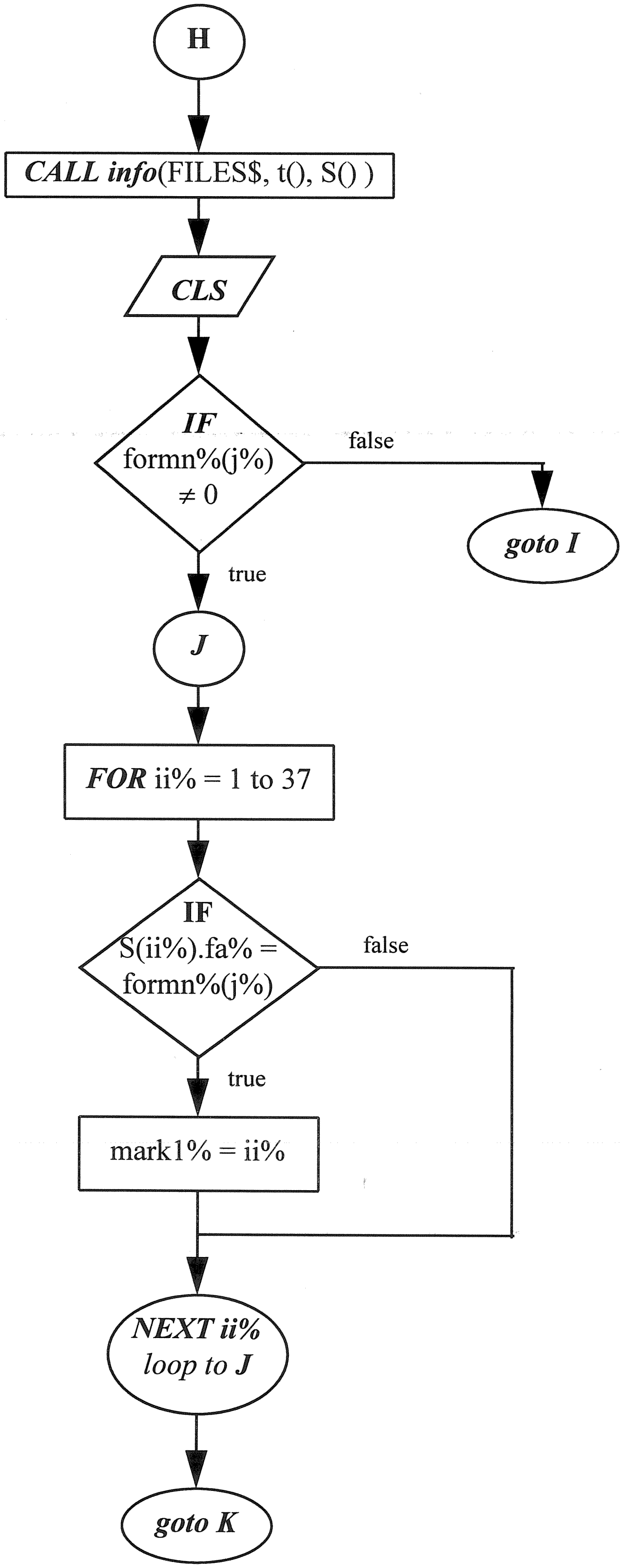
**ain Program**
**page 4**



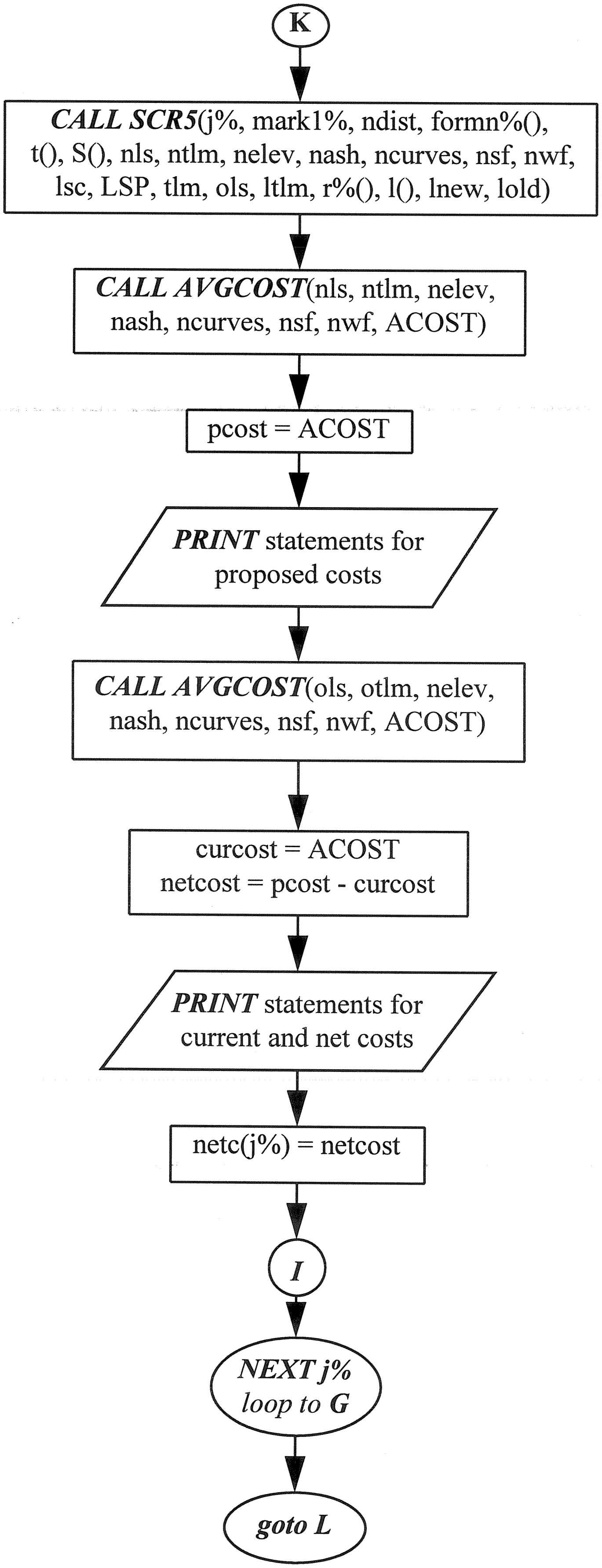
## Main Program
**page 5**



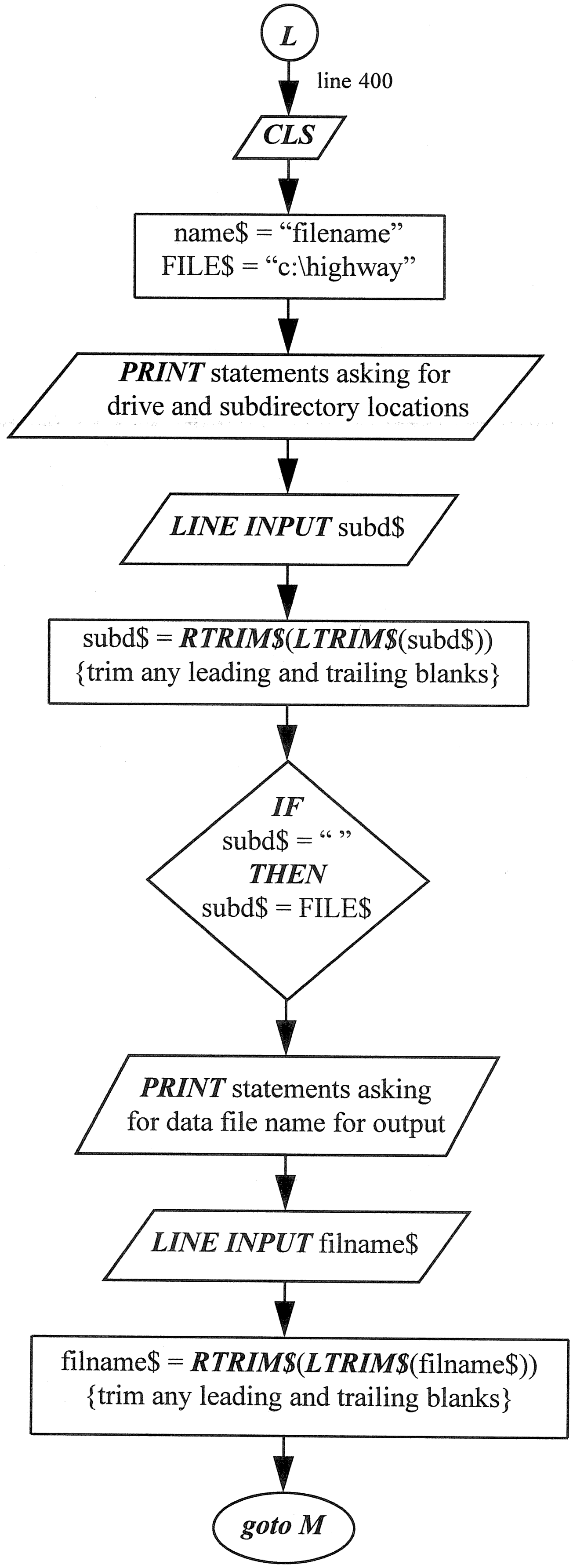
## Main Program
**page 6**



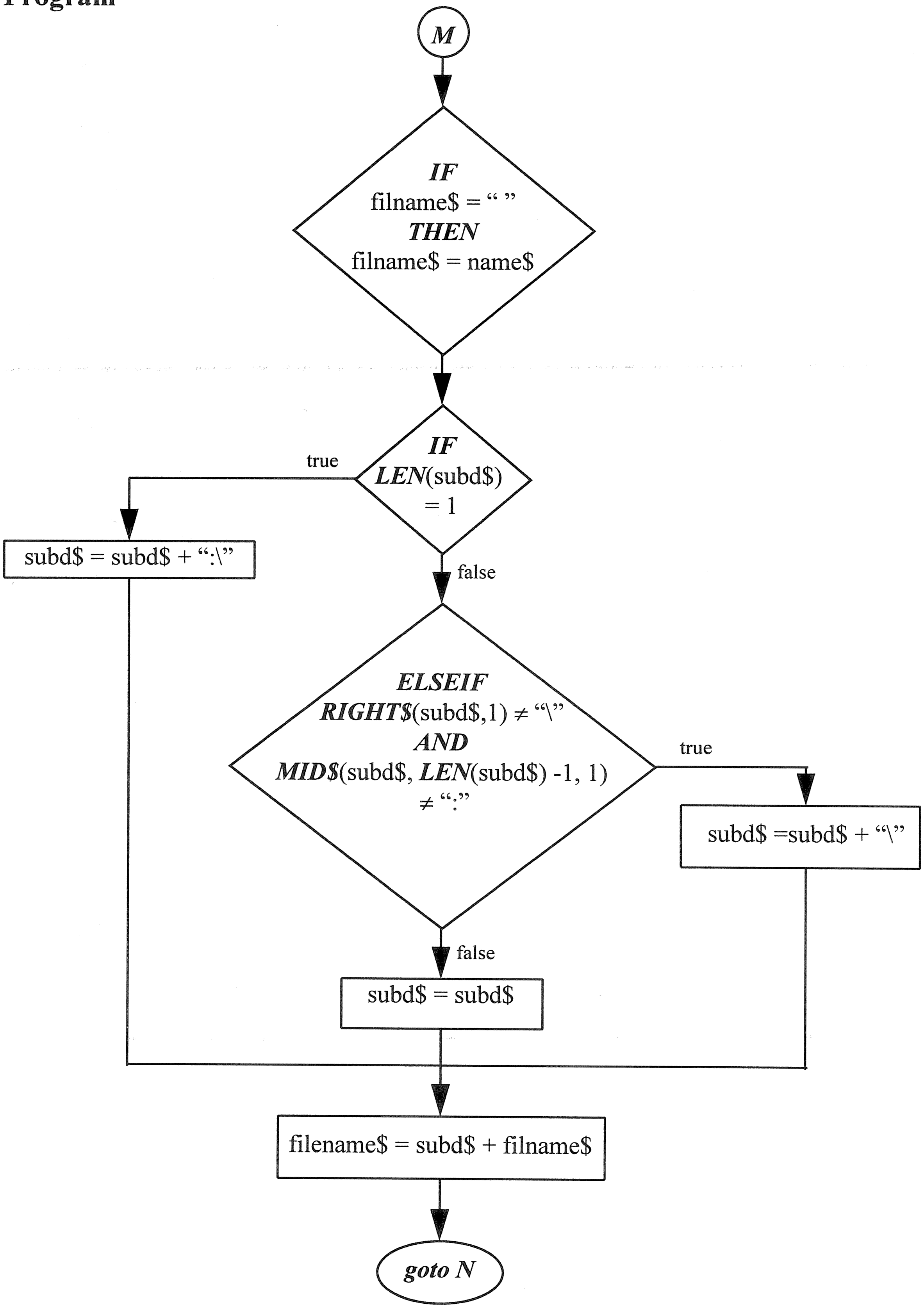
## Main Program
**page 7**


K
CALL SCR5(j%, mark1%, ndist, formn%(), t(), S(), nls, ntlm, nelev, nash, ncurves, nsf, nwf, lsc, LSP, tlm, ols, ltlm, r%(), l(), lnew, lold)
CALL AVGCOST(nls, ntlm, nelev, nash, ncurves, nsf, nwf, ACOST)
pcost = ACOST
PRINT statements for proposed costs
CALL AVGCOST(ols, otlm, nelev, nash, ncurves, nsf, nwf, ACOST)
curcost = ACOST
netcost = pcost - curcost
PRINT statements for current and net costs
netc(j%) = netcost
I
NEXT j% loop to G
goto L

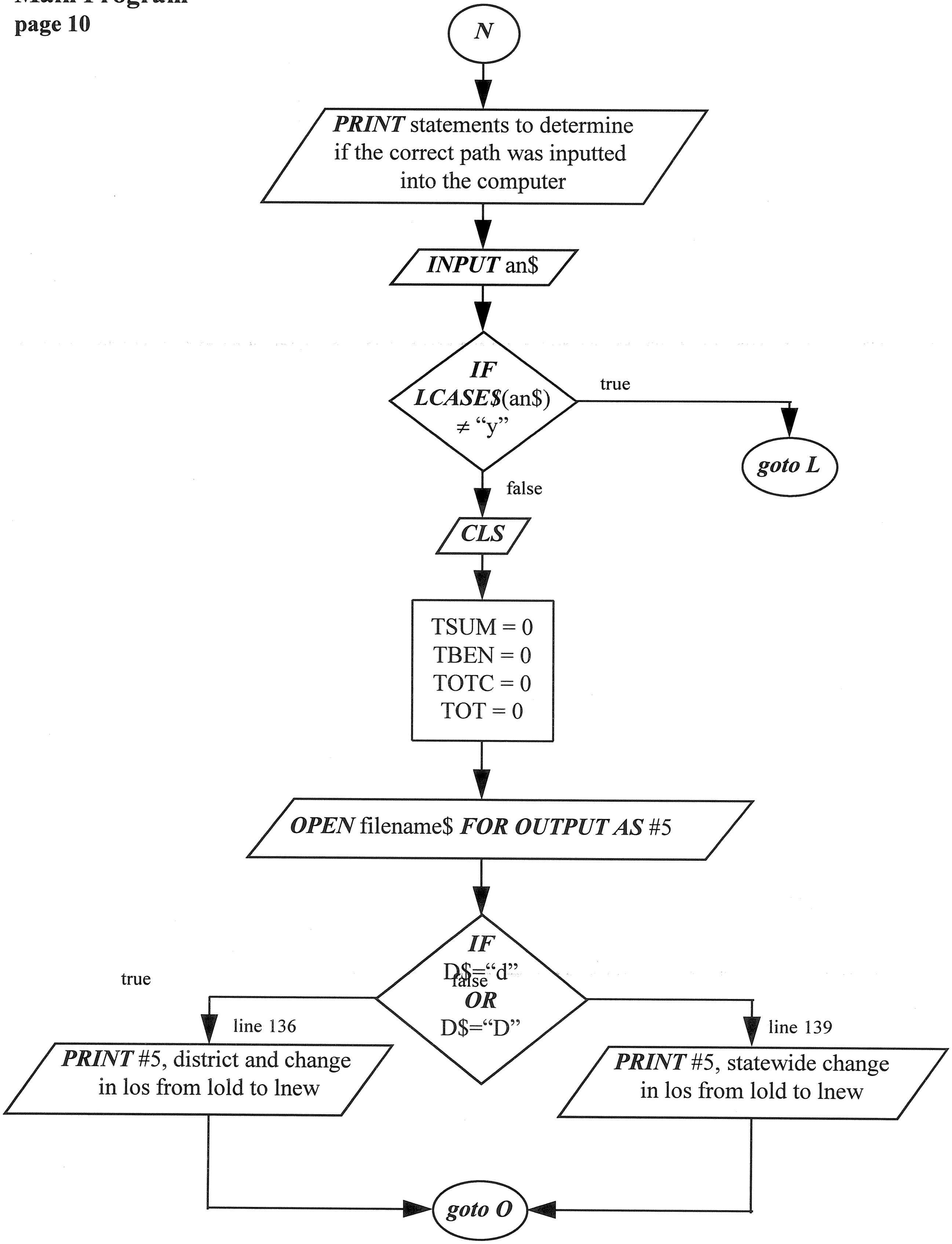
## Main Program
**page 8**


L
line 400
CLS
name$ = "filename"
FILE$ = "c:\highway"
PRINT statements asking for drive and subdirectory locations
LINE INPUT subd$
subd$ = RTRIM$(LTRIM$(subd$))
{trim any leading and trailing blanks}
IF subd$ = " " THEN subd$ = FILE$
PRINT statements asking for data file name for output
LINE INPUT filname$
filname$ = RTRIM$(LTRIM$(filname$))
{trim any leading and trailing blanks}
goto M

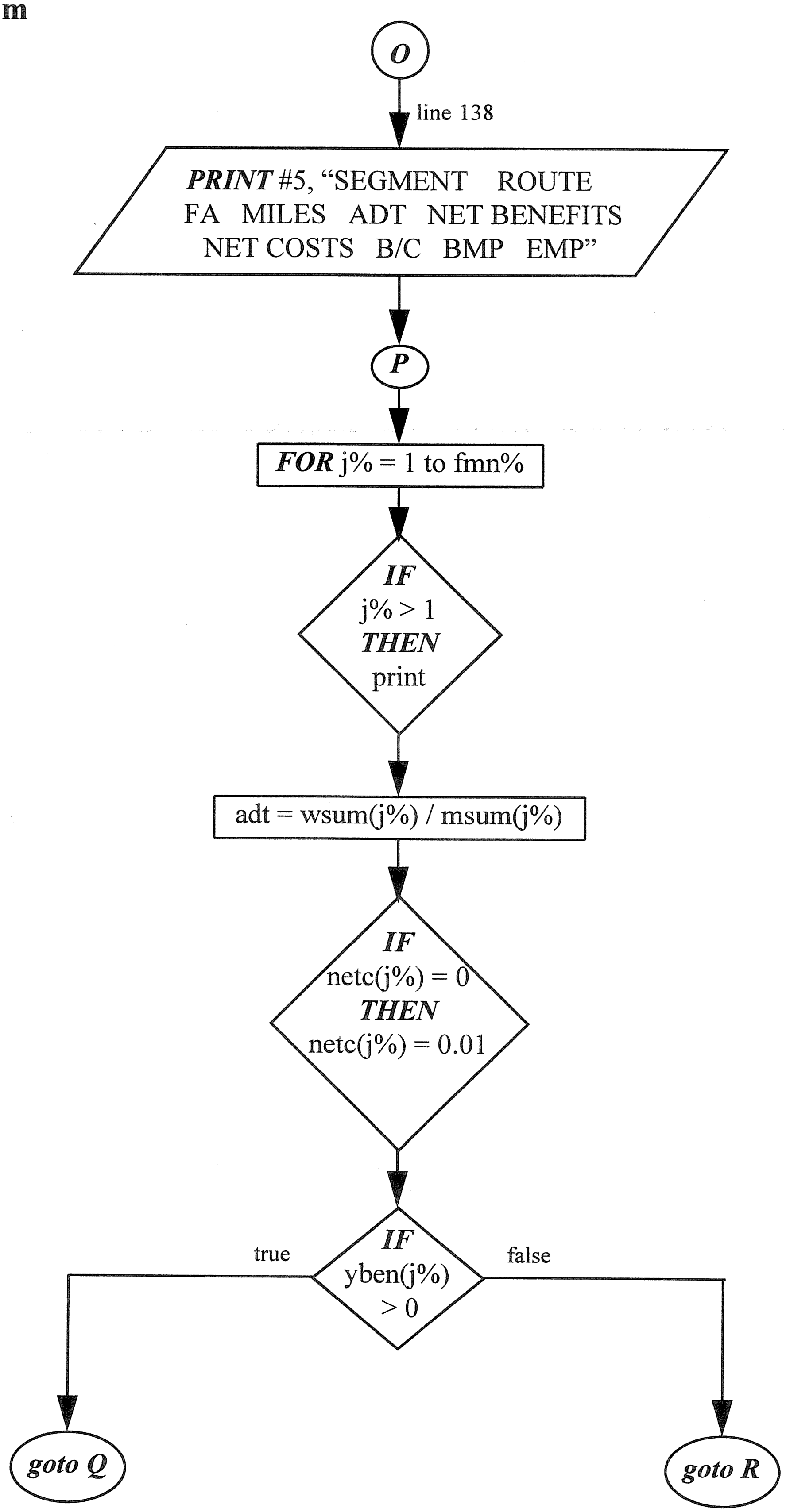
## Main Program
**page 9**



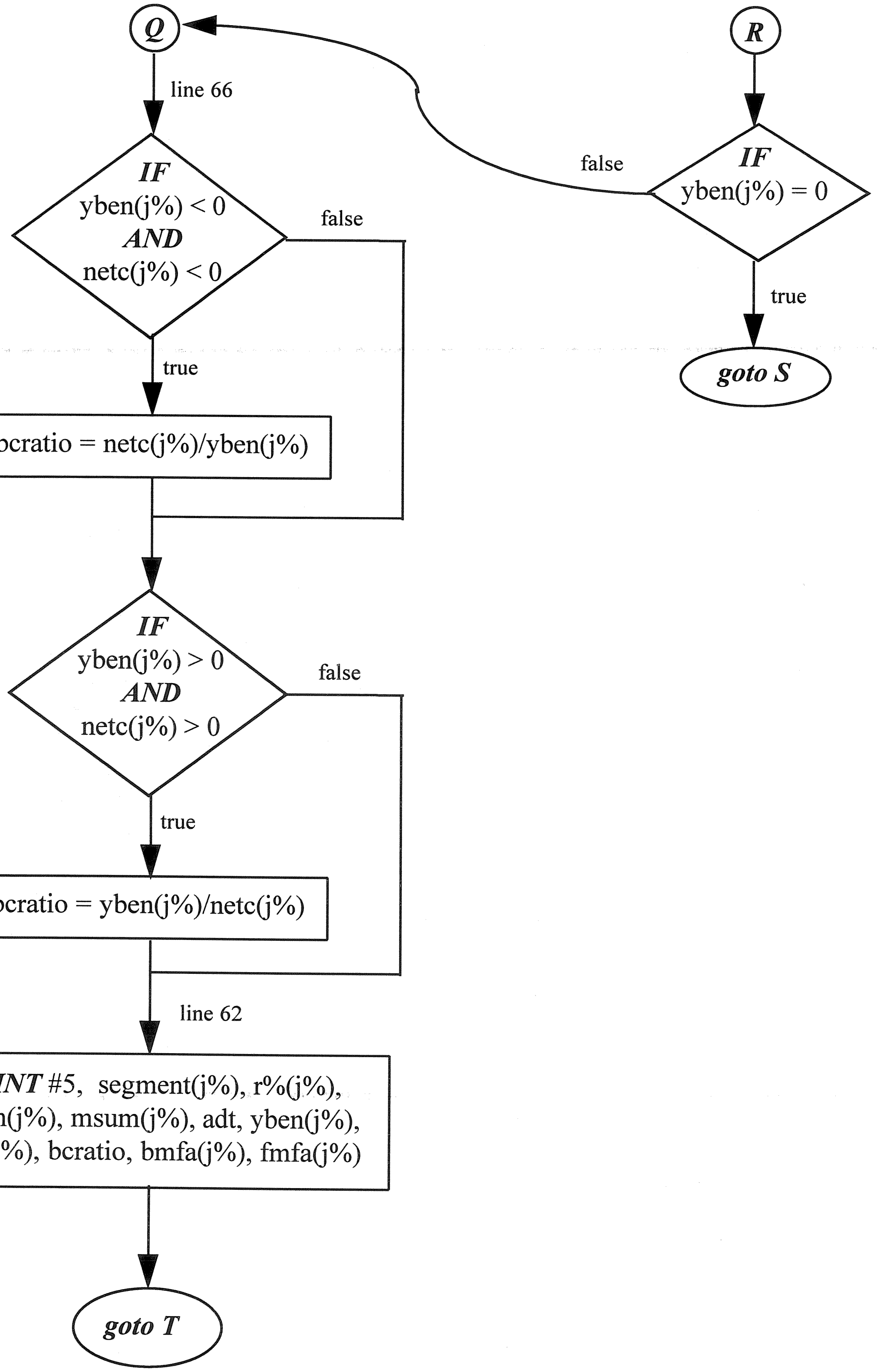
## Main Program
### page 10


N
PRINT statements to determine if the correct path was inputted into the computer
INPUT an$
IF LCASE$(an$) ≠ “y”
true
goto L
false
CLS
TSUM = 0
TBEN = 0
TOTC = 0
TOT = 0
OPEN filename$ FOR OUTPUT AS #5
IF D$=“d” OR D$=“D”
false
true
line 136
PRINT #5, district and change in los from lold to lnew
line 139
PRINT #5, statewide change in los from lold to lnew
goto O

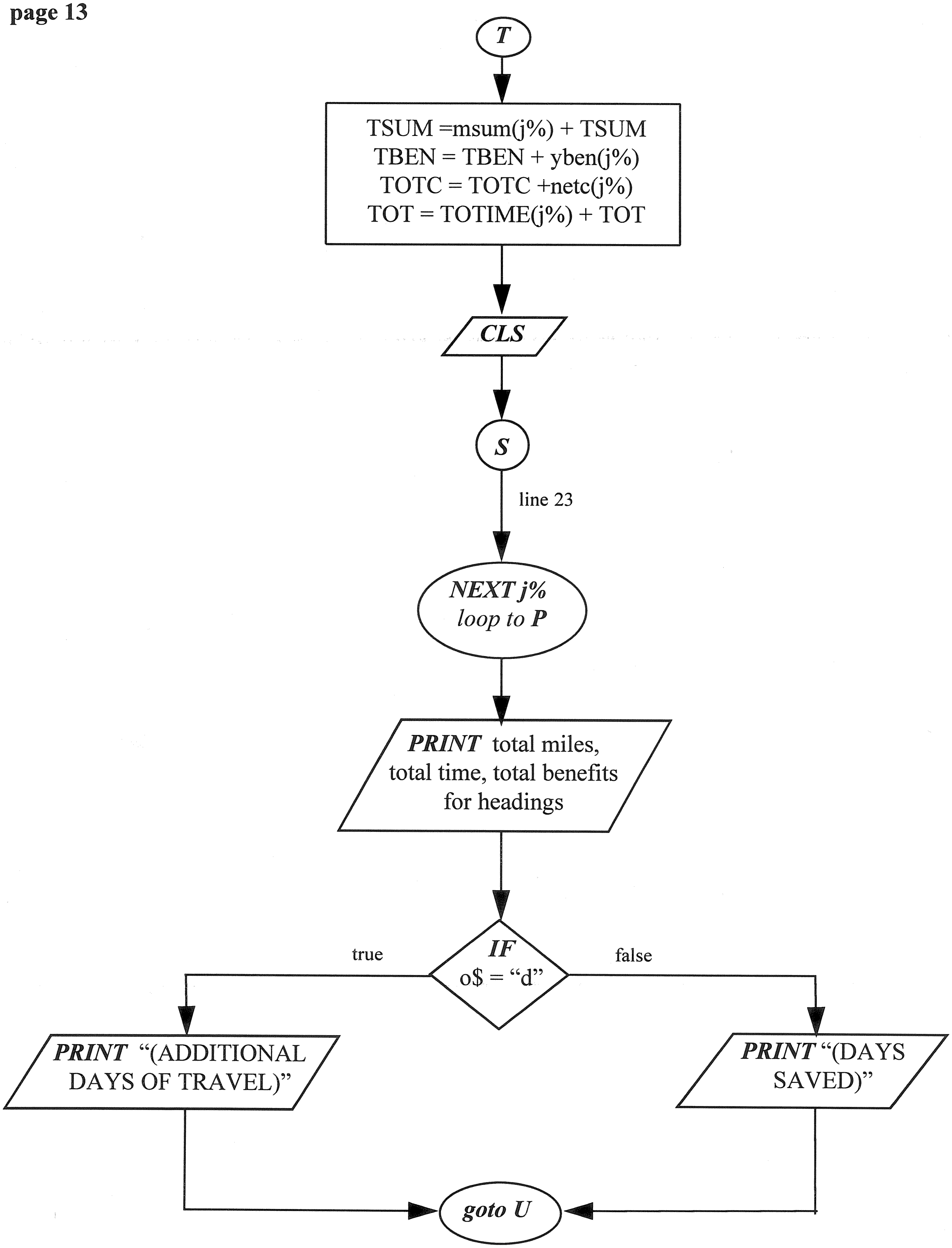
**Main Program**
**page 11**


O
line 138
PRINT #5, "SEGMENT ROUTE FA MILES ADT NET BENEFITS NET COSTS B/C BMP EMP"
P
FOR j% = 1 to fmn%
IF j% > 1 THEN print
adt = wsum(j%) / msum(j%)
IF netc(j%) = 0 THEN netc(j%) = 0.01
IF yben(j%) > 0
true
false
goto Q
goto R

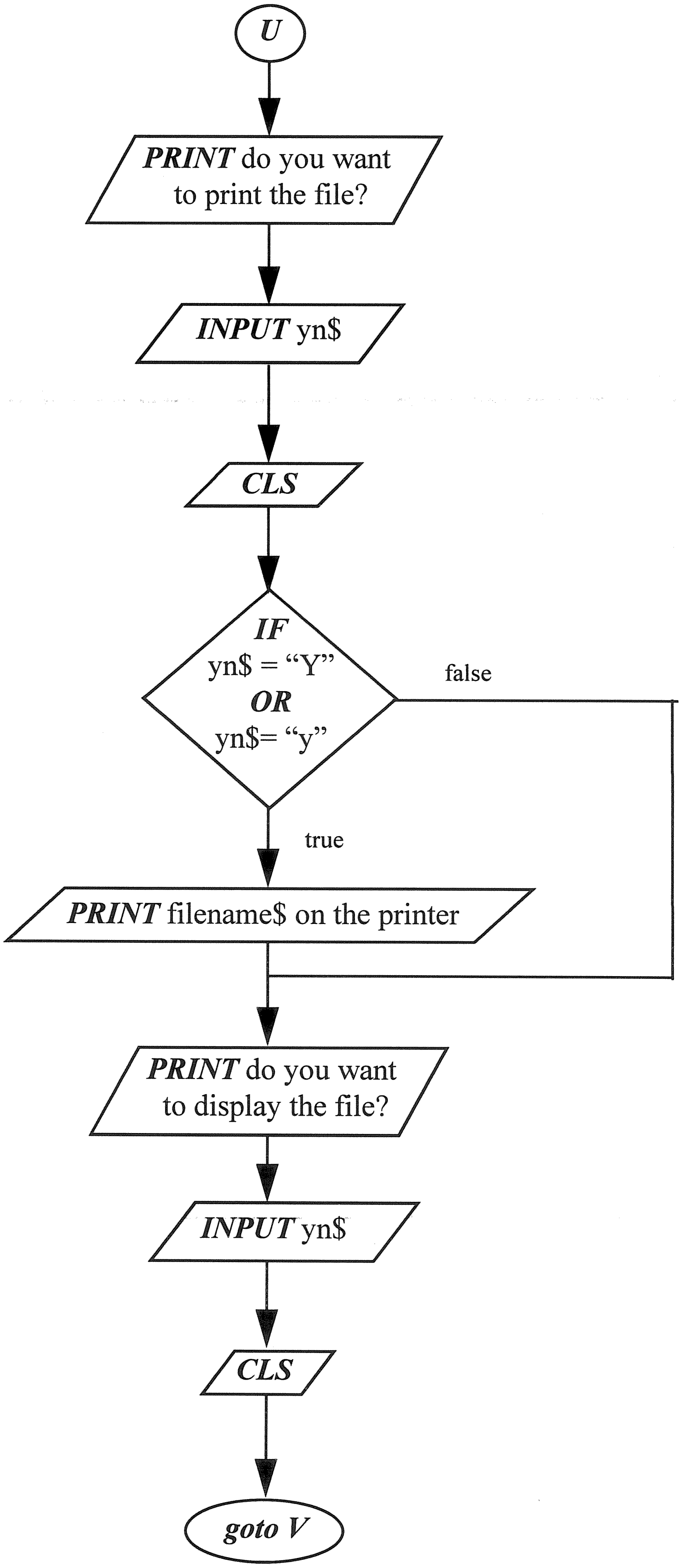
## Main Program
### page 12


Q
line 66
IF
yben(j%) < 0
AND
netc(j%) < 0
false
true
bcratio = netc(j%)/yben(j%)
IF
yben(j%) > 0
AND
netc(j%) > 0
false
true
bcratio = yben(j%)/netc(j%)
line 62
PRINT #5, segment(j%), r%(j%), formn(j%), msum(j%), adt, yben(j%), netc(j%), bcratio, bmfa(j%), fmfa(j%)
goto T
R
IF
yben(j%) = 0
false
true
goto S

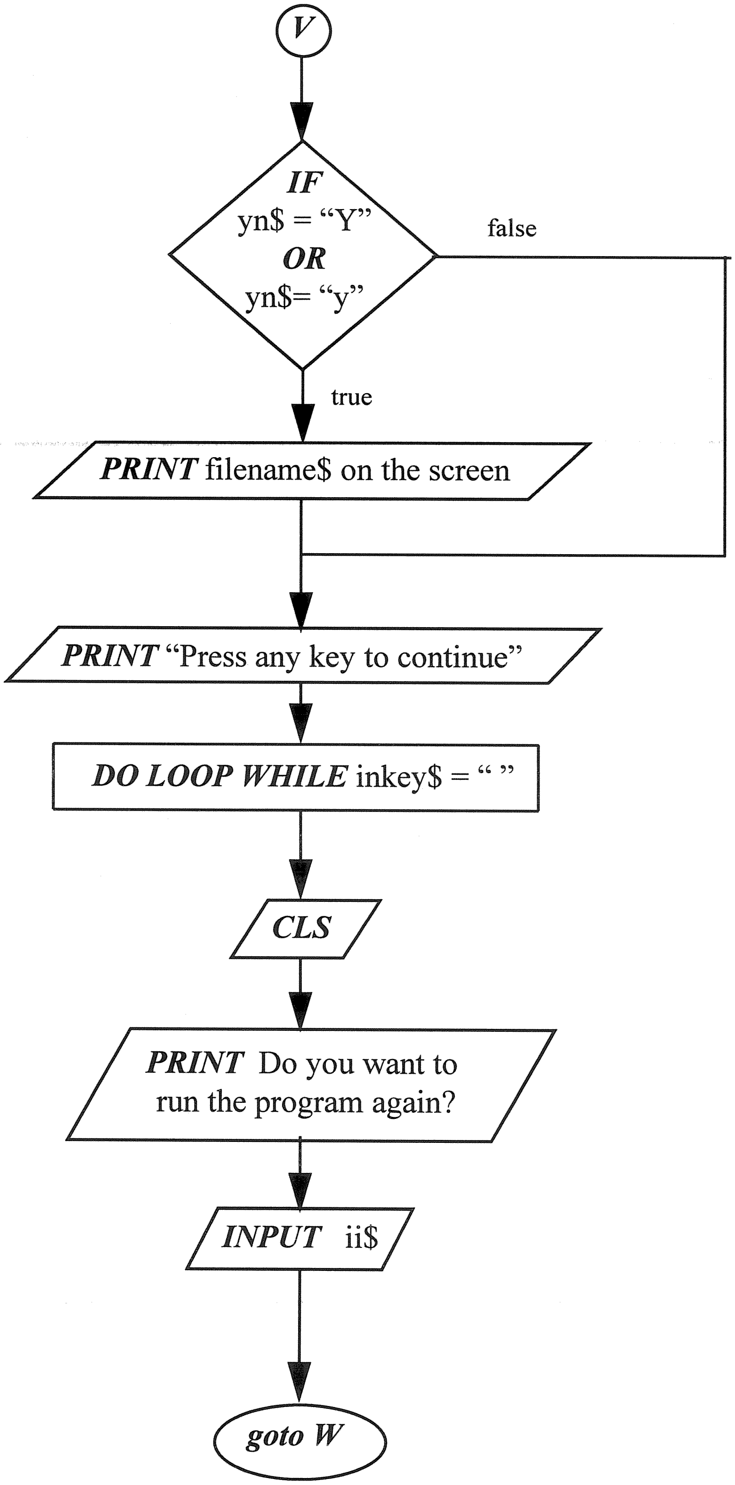
## Main Program
**page 13**



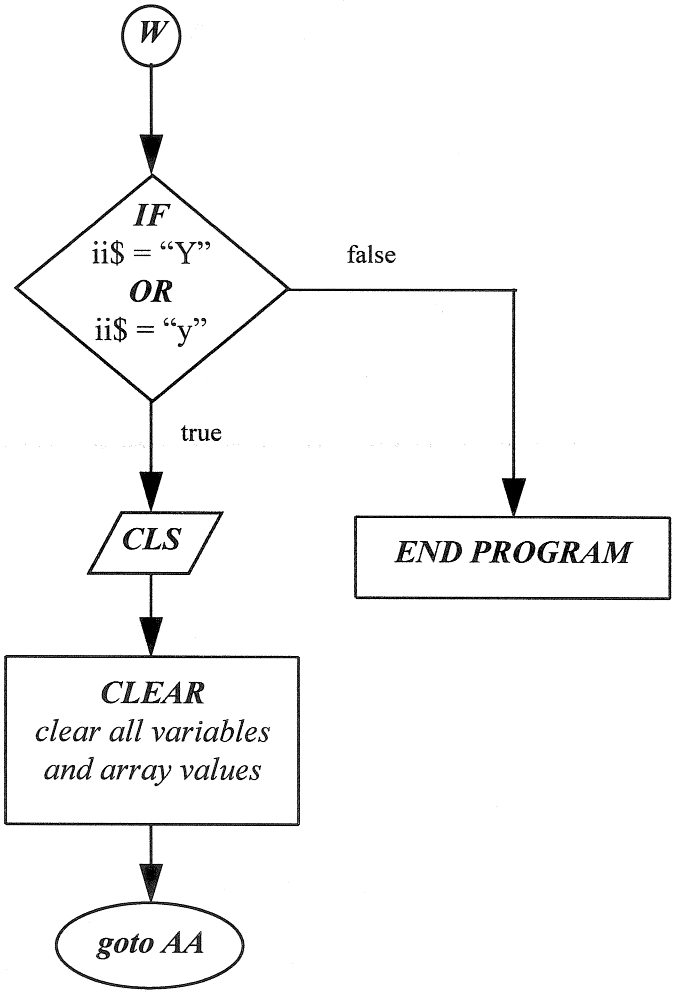
## Main Program
**page 14**


U
PRINT do you want to print the file?
INPUT yn$
CLS
IF yn$ = "Y" OR yn$= "y"
false
true
PRINT filename$ on the printer
PRINT do you want to display the file?
INPUT yn$
CLS
goto V

## Main Program
### page 15

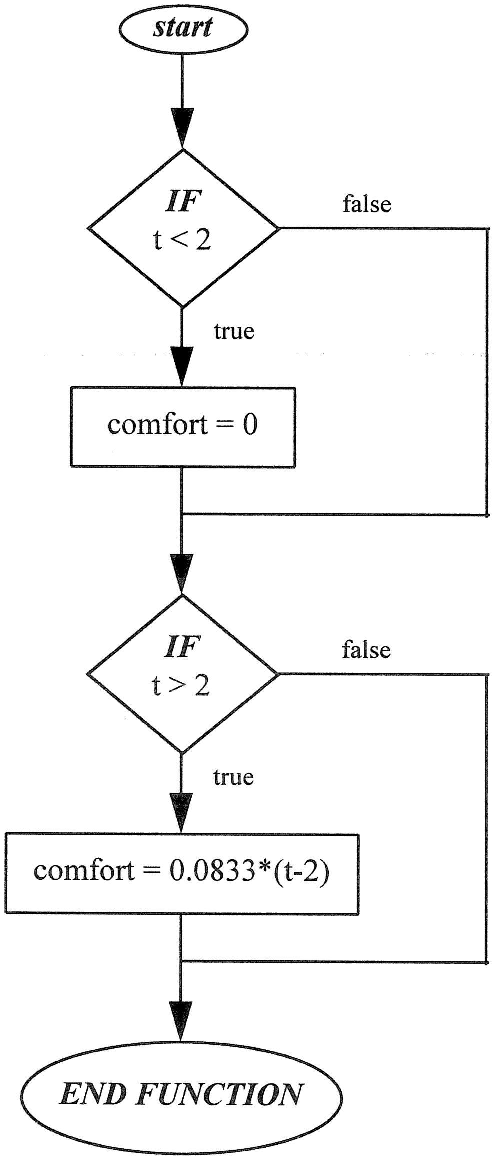

V
IF
yn$ = "Y"
OR
yn$= "y"
false
true
PRINT filename$ on the screen
PRINT "Press any key to continue"
DO LOOP WHILE inkey$ = " "
CLS
PRINT Do you want to run the program again?
INPUT ii$
goto W

## Main Program
**page 16**



## Function comfort(t)

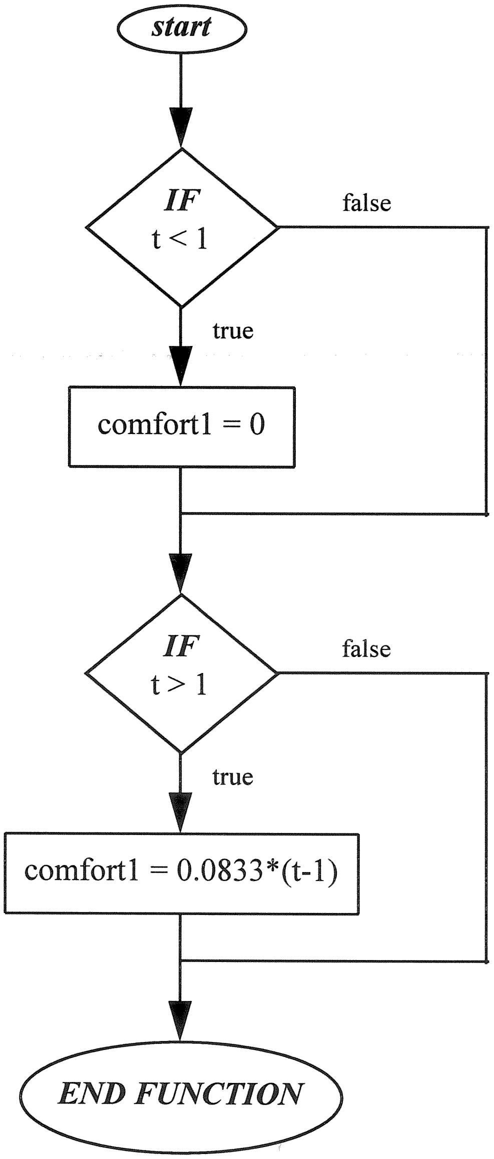
### page 1



Where:

t = time   found in main program

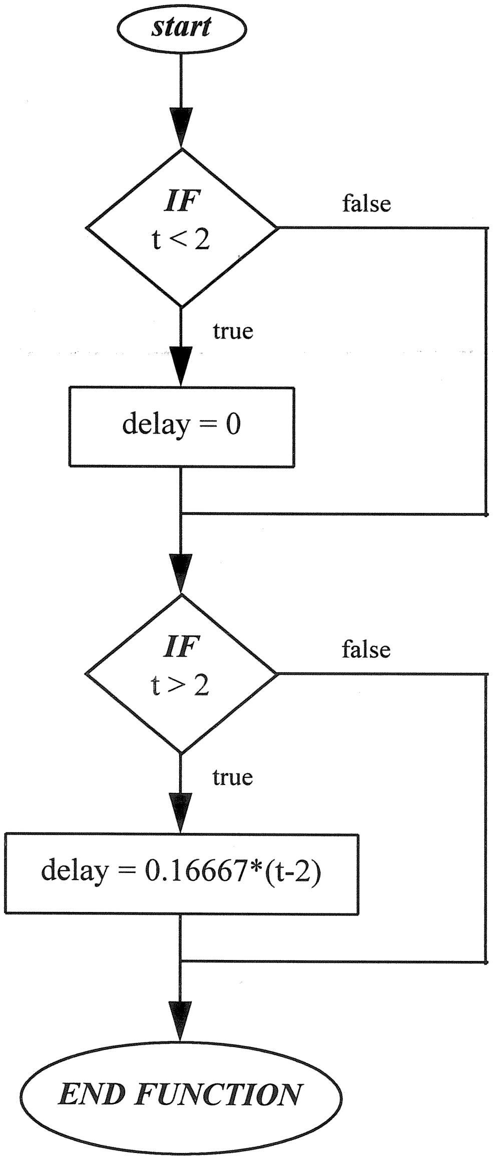
## Function comfort1(t)
### page 1



Where:

t = time found in main program

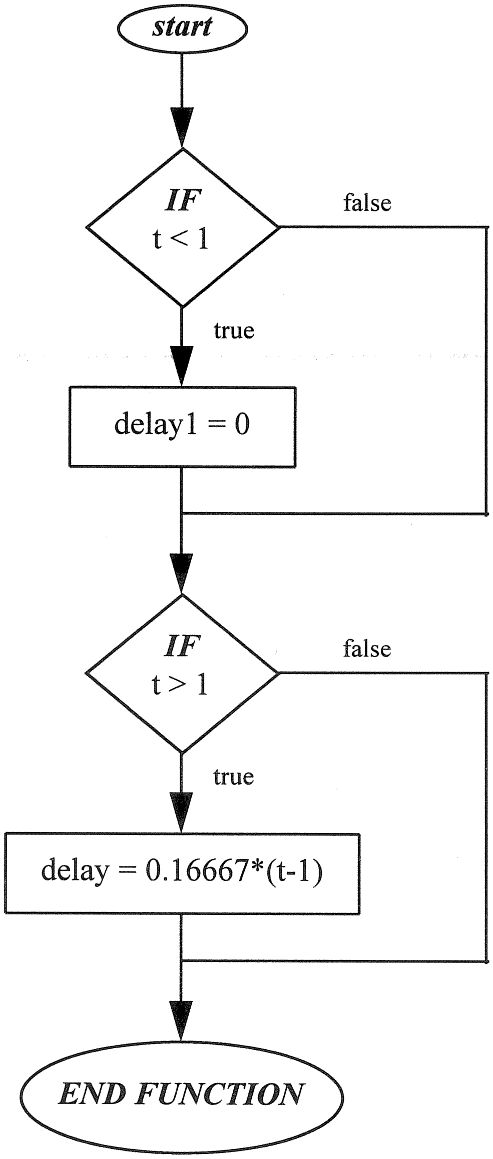
**Function**
**delay(t)**
**page 1**



Where:

t = time   found in main program

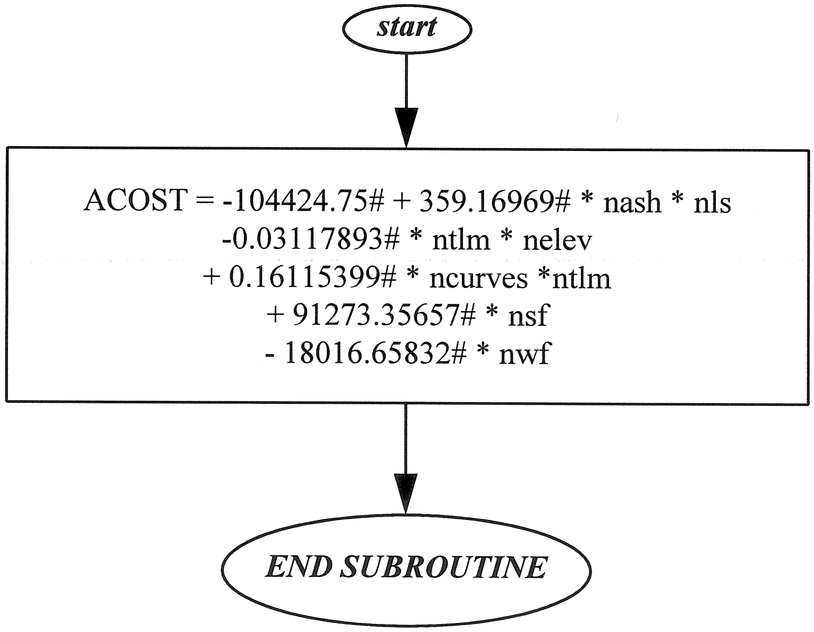
**Function**
**delay1(t)**
**page 1**



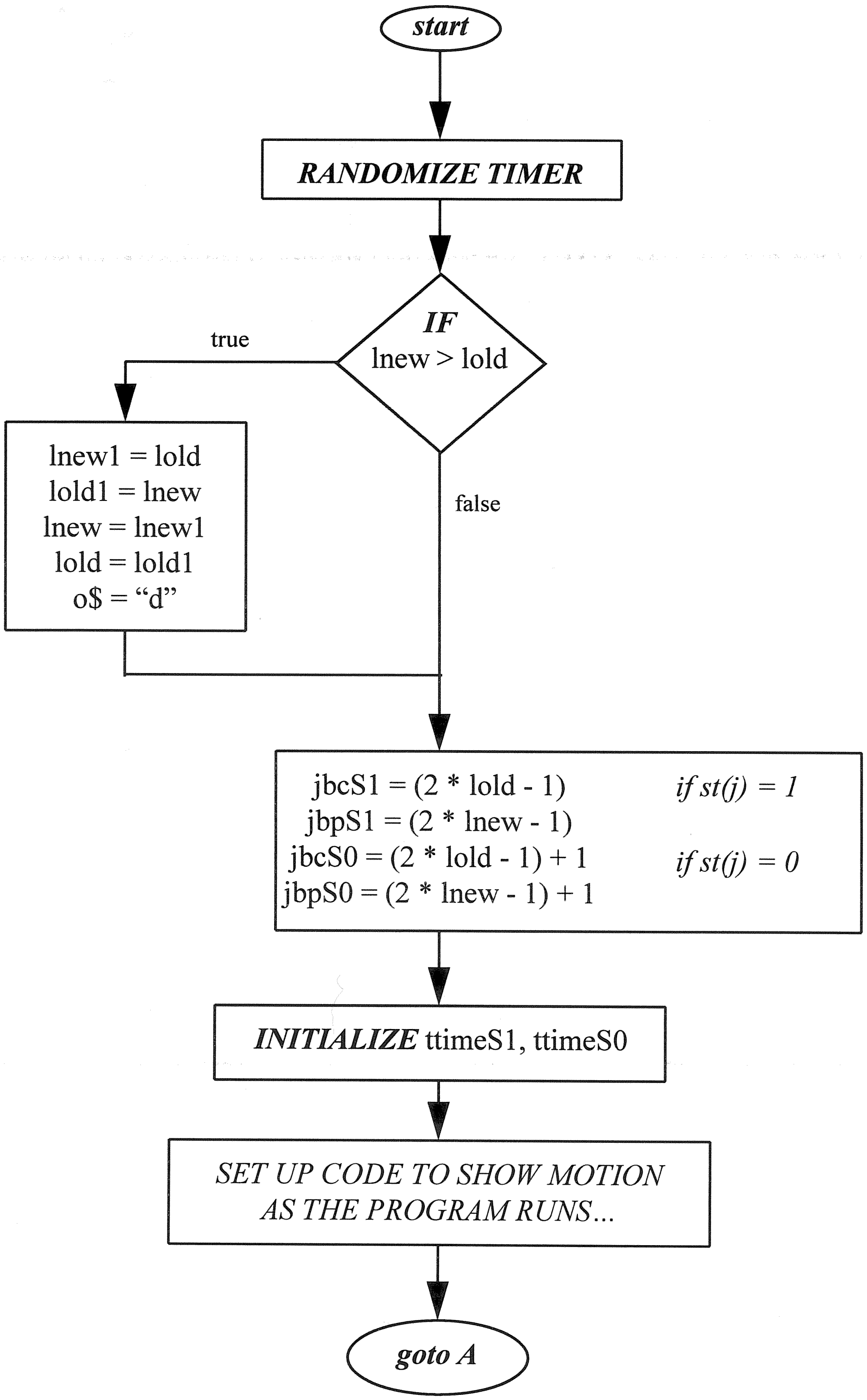
Where:

t = time found in main program

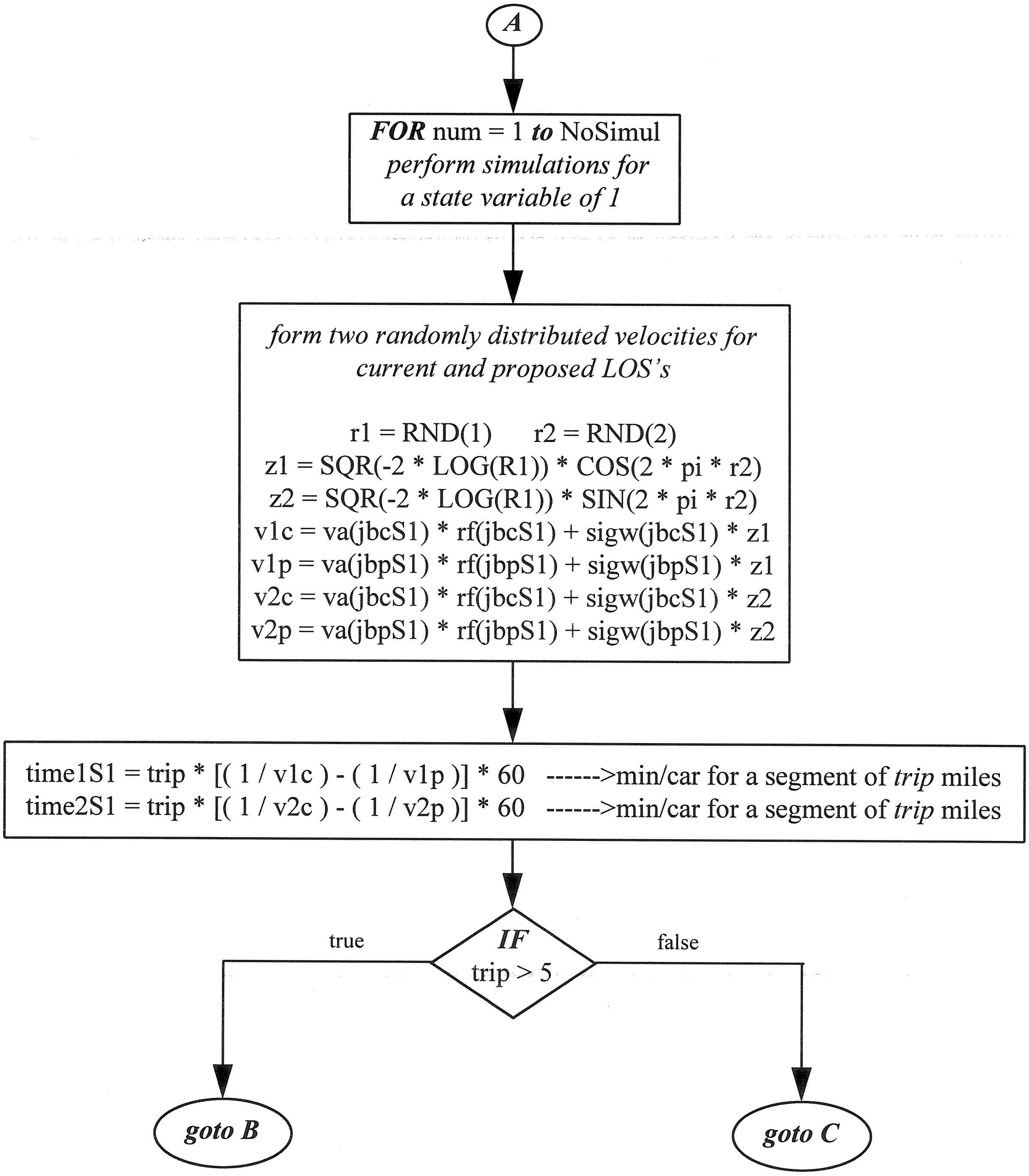
**Subroutine**
**AVGCOST**(nls, ntlm, nelev, nash, ncurves, nsf, nwf, ACOST)
**page 1**


start
ACOST = -104424.75# + 359.16969# * nash * nls
-0.03117893# * ntlm * nelev
+ 0.16115399# * ncurves *ntlm
+ 91273.35657# * nsf
- 18016.65832# * nwf
END SUBROUTINE

**Subroutine**
**deltime(**va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1, ccostS1, dcostS1, ttimeS0, ccostS0, dcostS0, NoSimul, inflat, work1 )
**page 1**



**Subroutine**
**deltime(**va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1, ccostS1, dcostS1, ttimeS0, ccostS0, dcostS0, NoSimul, inflat, work1 )
**page 2**



***FOR*** num = 1 ***to*** NoSimul
*perform simulations for a state variable of 1*

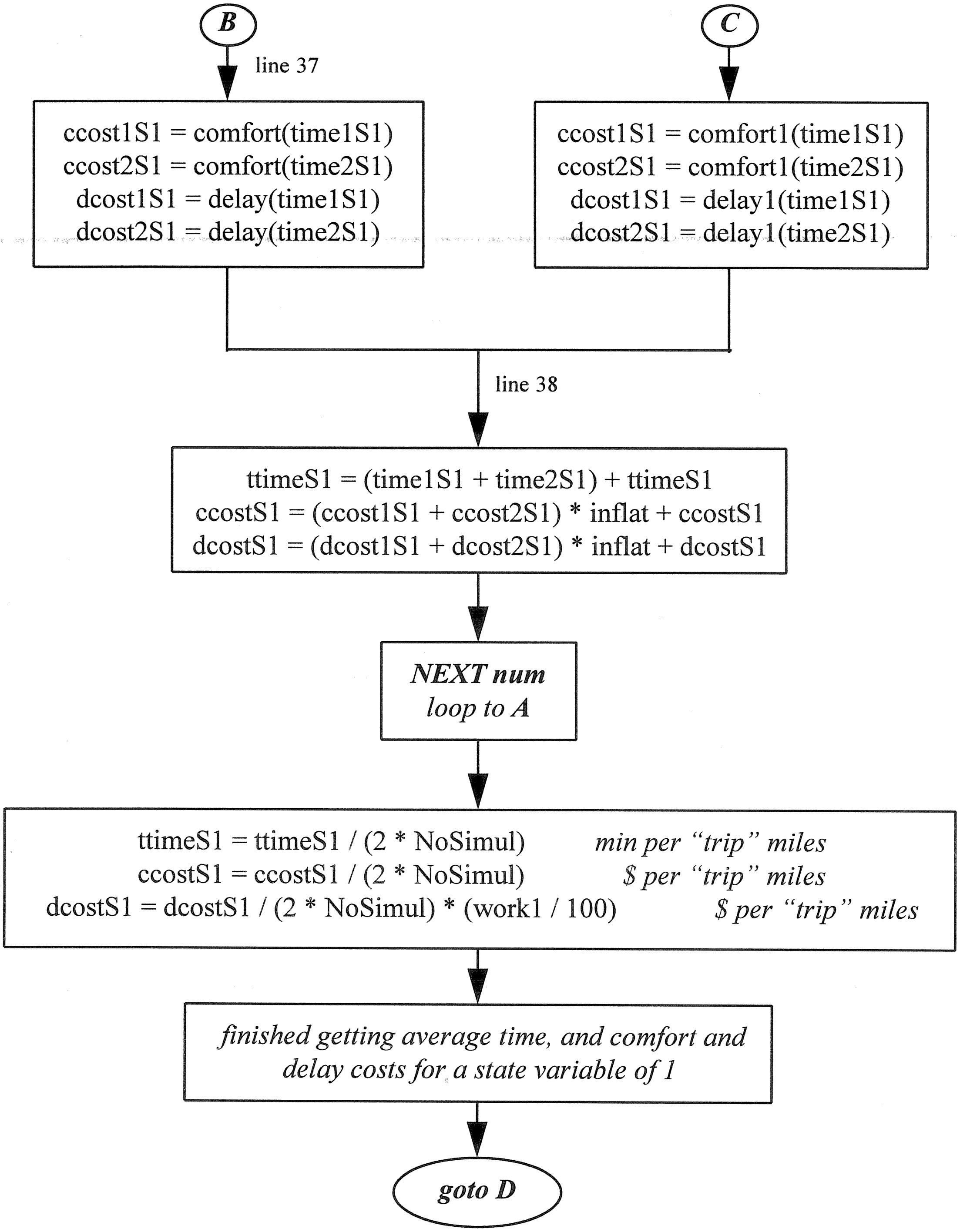
*form two randomly distributed velocities for current and proposed LOS's*

```
r1 = RND(1)      r2 = RND(2)
z1 = SQR(-2 * LOG(R1)) * COS(2 * pi * r2)
z2 = SQR(-2 * LOG(R1)) * SIN(2 * pi * r2)
v1c = va(jbcS1) * rf(jbcS1) + sigw(jbcS1) * z1
v1p = va(jbpS1) * rf(jbpS1) + sigw(jbpS1) * z1
v2c = va(jbcS1) * rf(jbcS1) + sigw(jbcS1) * z2
v2p = va(jbpS1) * rf(jbpS1) + sigw(jbpS1) * z2
```

```
time1S1 = trip * [( 1 / v1c ) - ( 1 / v1p )] * 60  ------>min/car for a segment of trip miles
time2S1 = trip * [( 1 / v2c ) - ( 1 / v2p )] * 60  ------>min/car for a segment of trip miles
```

***IF***
trip > 5

**Subroutine**
**deltime(**va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1, ccostS1, dcostS1, ttimeS0, ccostS0, dcostS0, NoSimul, inflat, work1 )
**page 3**



*B*

line 37

ccost1S1 = comfort(time1S1)
ccost2S1 = comfort(time2S1)
dcost1S1 = delay(time1S1)
dcost2S1 = delay(time2S1)

*C*

ccost1S1 = comfort1(time1S1)
ccost2S1 = comfort1(time2S1)
dcost1S1 = delay1(time1S1)
dcost2S1 = delay1(time2S1)

line 38

ttimeS1 = (time1S1 + time2S1) + ttimeS1
ccostS1 = (ccost1S1 + ccost2S1) * inflat + ccostS1
dcostS1 = (dcost1S1 + dcost2S1) * inflat + dcostS1

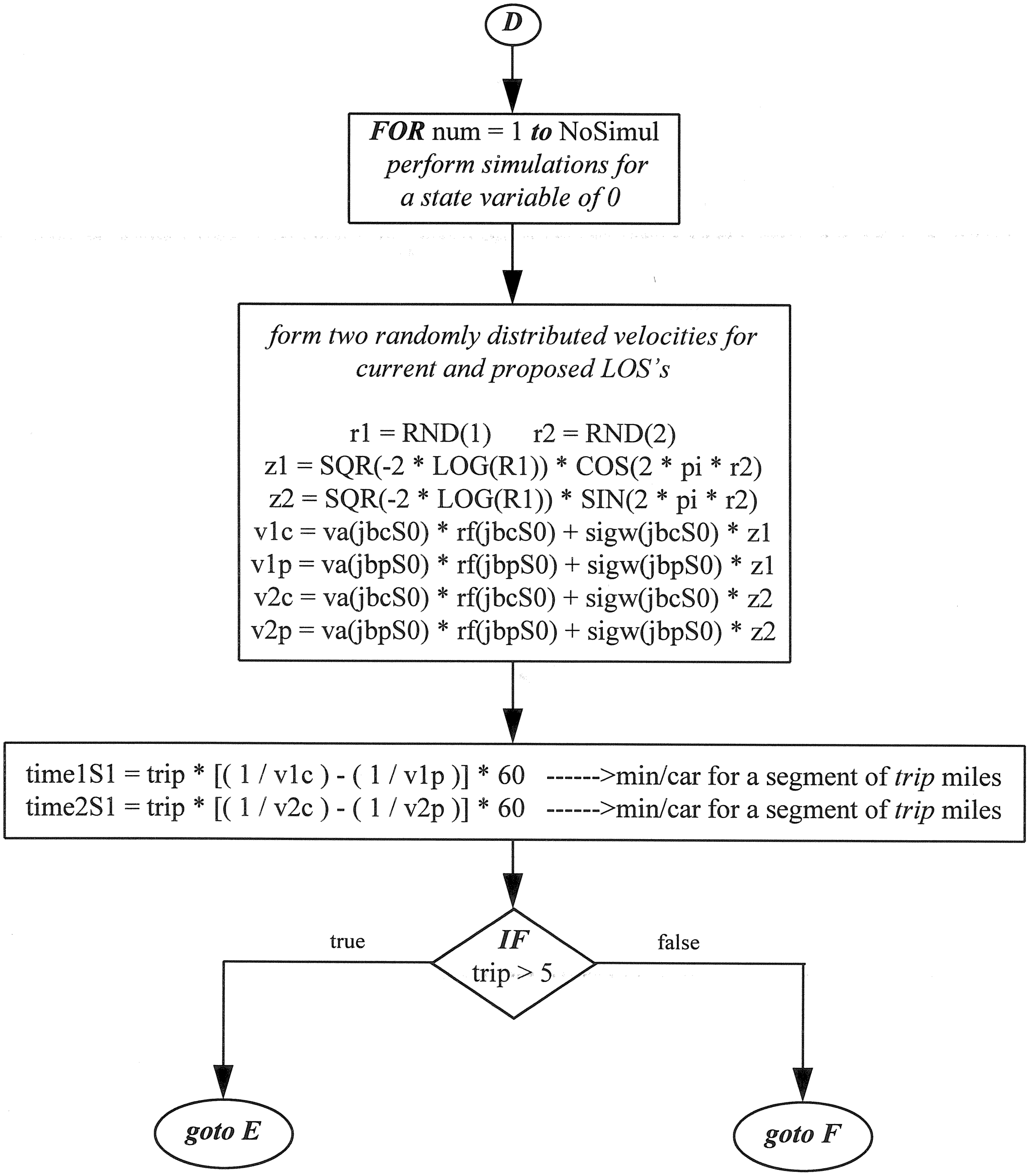
***NEXT num***
*loop to **A***

ttimeS1 = ttimeS1 / (2 * NoSimul) *min per "trip" miles*
ccostS1 = ccostS1 / (2 * NoSimul) *$ per "trip" miles*
dcostS1 = dcostS1 / (2 * NoSimul) * (work1 / 100) *$ per "trip" miles*

*finished getting average time, and comfort and delay costs for a state variable of 1*

***goto D***

**Subroutine**
**deltime**(va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1, ccostS1, dcostS1, ttimeS0, ccostS0, dcostS0, NoSimul, inflat, work1 )
**page 4**



***FOR*** num = 1 ***to*** NoSimul
*perform simulations for a state variable of 0*

*form two randomly distributed velocities for current and proposed LOS's*

```
r1 = RND(1)      r2 = RND(2)
z1 = SQR(-2 * LOG(R1)) * COS(2 * pi * r2)
z2 = SQR(-2 * LOG(R1)) * SIN(2 * pi * r2)
v1c = va(jbcS0) * rf(jbcS0) + sigw(jbcS0) * z1
v1p = va(jbpS0) * rf(jbpS0) + sigw(jbpS0) * z1
v2c = va(jbcS0) * rf(jbcS0) + sigw(jbcS0) * z2
v2p = va(jbpS0) * rf(jbpS0) + sigw(jbpS0) * z2
```

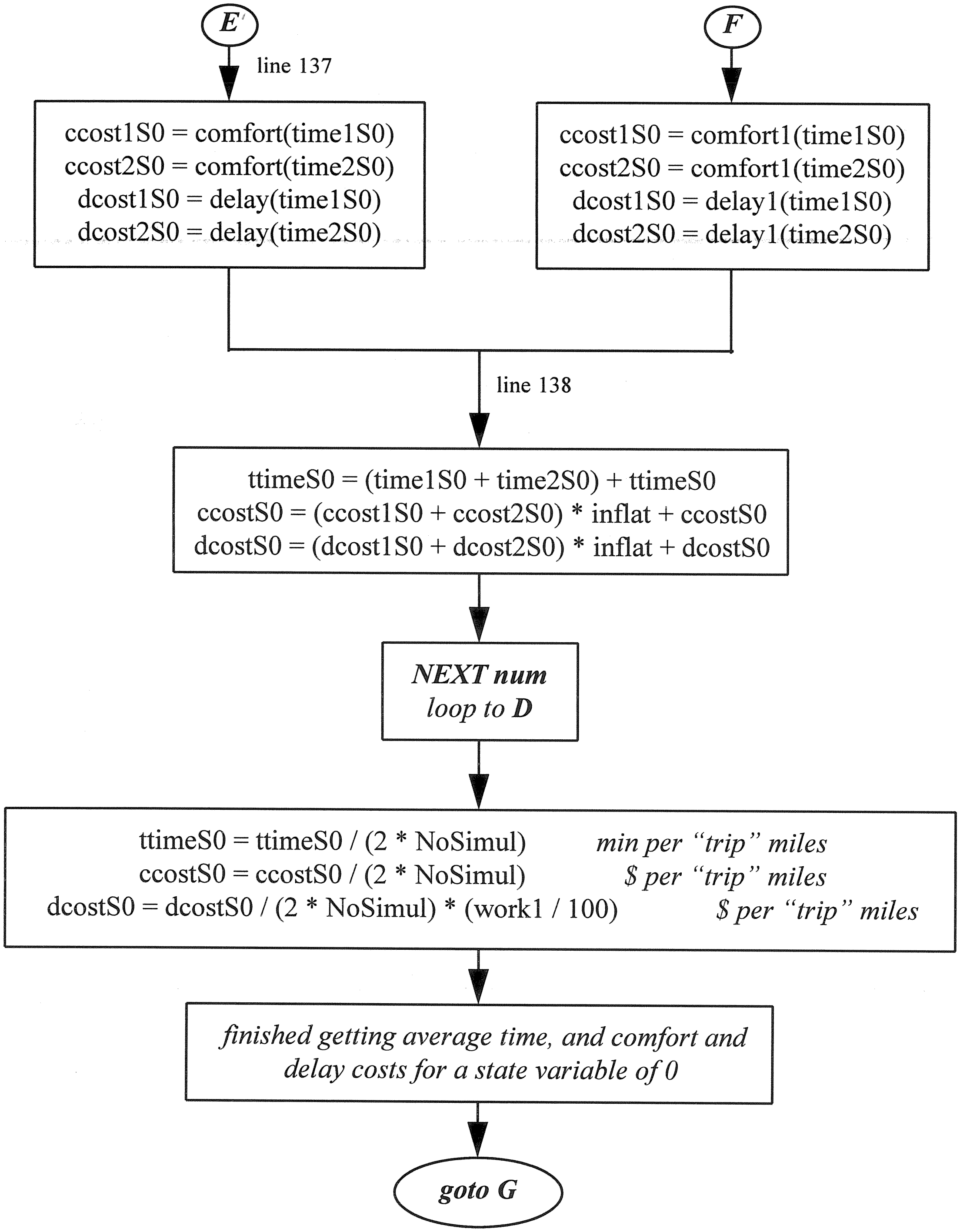
```
time1S1 = trip * [( 1 / v1c ) - ( 1 / v1p )] * 60   ------>min/car for a segment of trip miles
time2S1 = trip * [( 1 / v2c ) - ( 1 / v2p )] * 60   ------>min/car for a segment of trip miles
```

***IF*** trip > 5

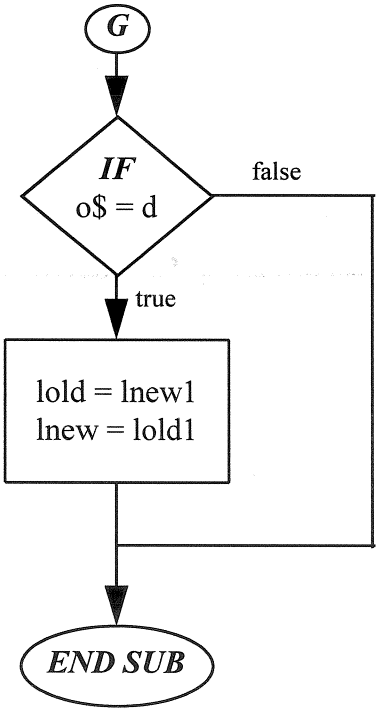
**Subroutine**

**deltime(**va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1, ccostS1, dcostS1, ttimeS0, ccostS0, dcostS0, NoSimul, inflat, work1 )

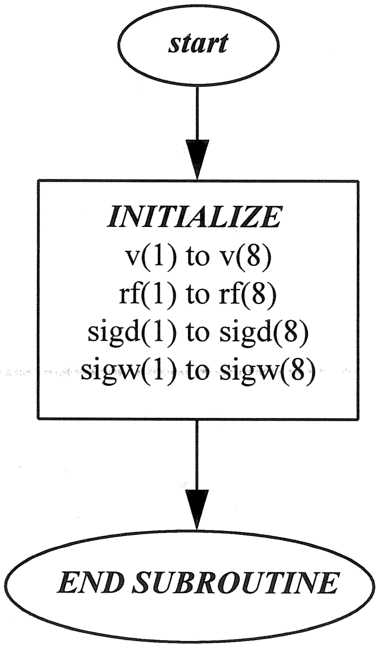
**page 5**


E
F
line 137
ccost1S0 = comfort(time1S0)
ccost2S0 = comfort(time2S0)
dcost1S0 = delay(time1S0)
dcost2S0 = delay(time2S0)
ccost1S0 = comfort1(time1S0)
ccost2S0 = comfort1(time2S0)
dcost1S0 = delay1(time1S0)
dcost2S0 = delay1(time2S0)
line 138
ttimeS0 = (time1S0 + time2S0) + ttimeS0
ccostS0 = (ccost1S0 + ccost2S0) * inflat + ccostS0
dcostS0 = (dcost1S0 + dcost2S0) * inflat + dcostS0
NEXT num
loop to D
ttimeS0 = ttimeS0 / (2 * NoSimul) min per "trip" miles
ccostS0 = ccostS0 / (2 * NoSimul) $ per "trip" miles
dcostS0 = dcostS0 / (2 * NoSimul) * (work1 / 100) $ per "trip" miles
finished getting average time, and comfort and delay costs for a state variable of 0
goto G

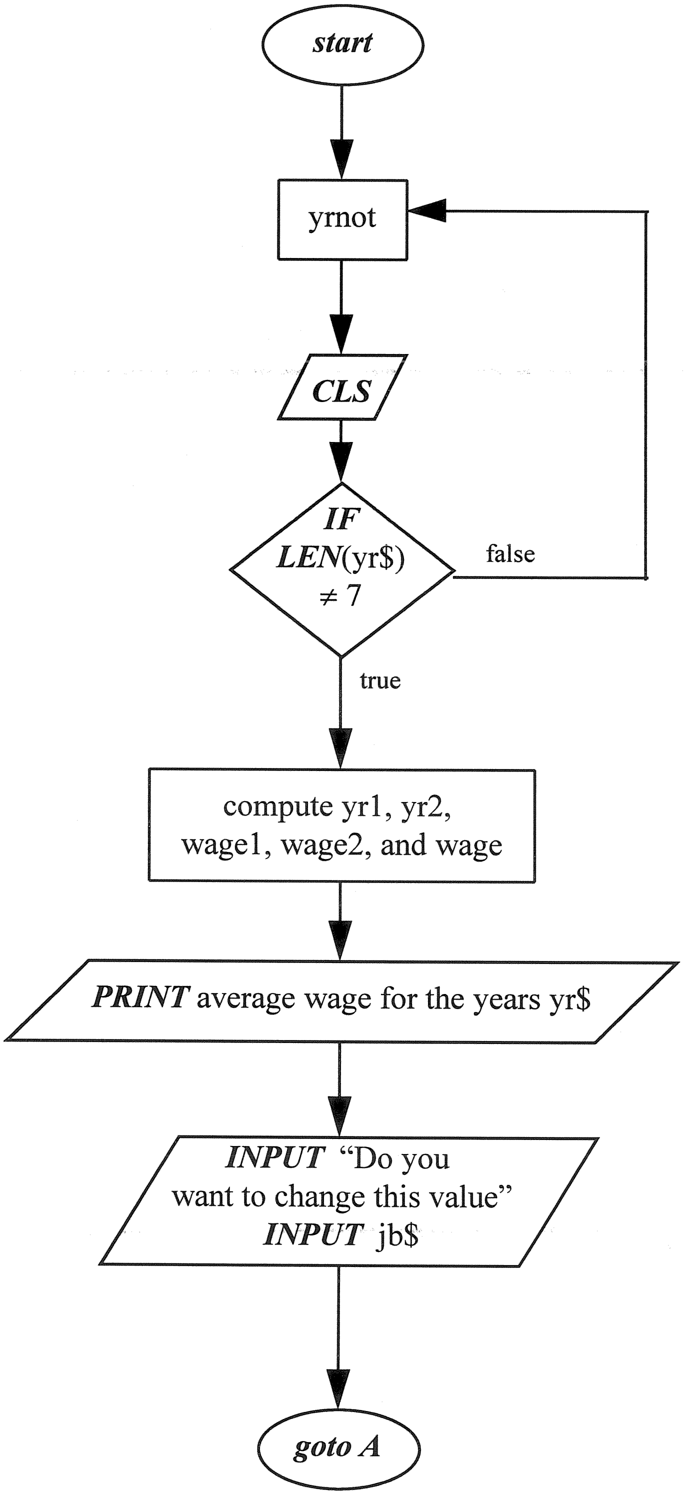
**Subroutine**
**deltime(**va!(), rf!(), sigd!(), sigw!(), lold, lnew, trip, pi, o$, ttimeS1, ccostS1, dcostS1, ttimeS0, ccostS0, dcostS0, NoSimul, inflat, work1 )
**page 6**



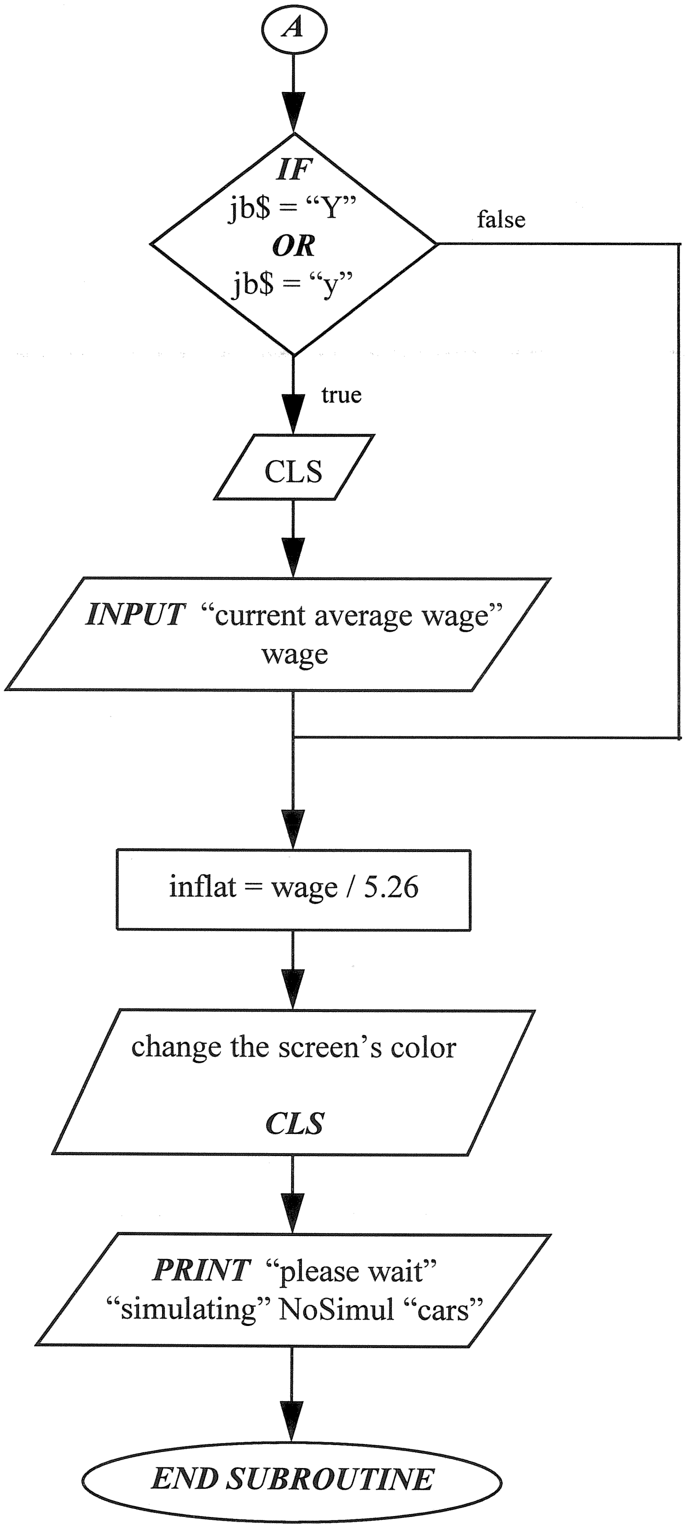
**Subroutine**
**indata(**va(), rf(), sigd(), sigw())
**page 1**



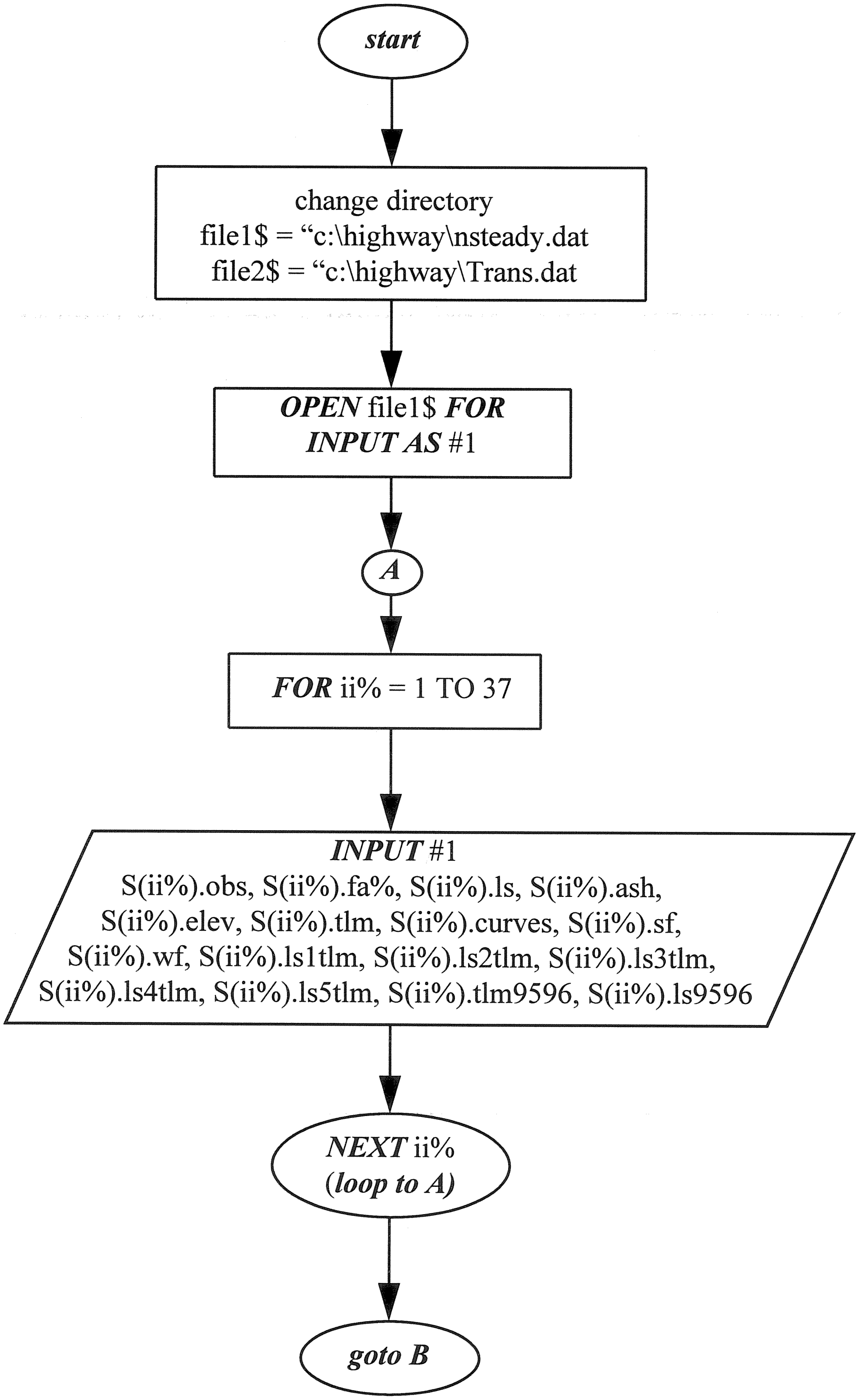
**Subroutine**
**inflation(**NoSimul, inflat, yr$)
**page 1**



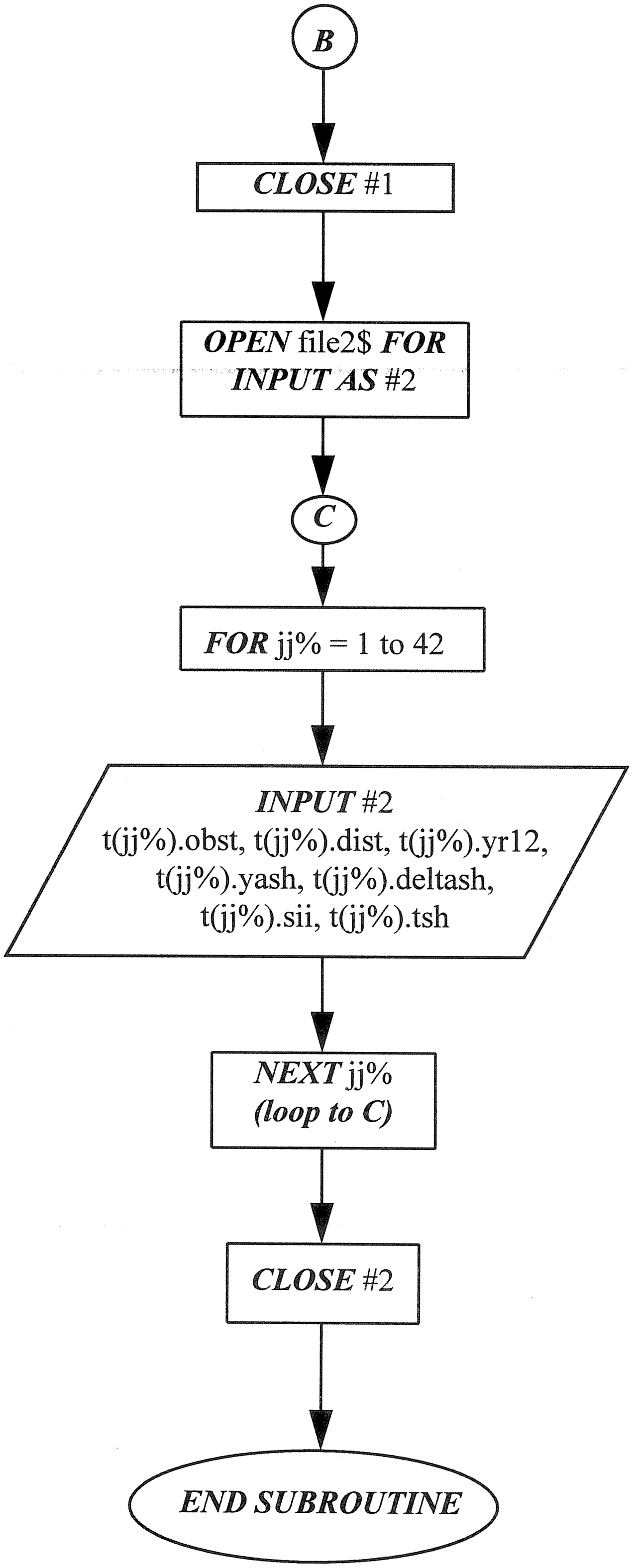
**Subroutine**
**inflation(**NoSimul, inflat, yr$)
**page 2**



**Subroutine**
**info(**FILES$, t() AS trans, S() AS steady)
**page 1**


*start*
change directory
file1$ = “c:\highway\nsteady.dat
file2$ = “c:\highway\Trans.dat
***OPEN*** file1$ ***FOR***
***INPUT AS*** #1
*A*
***FOR*** ii% = 1 TO 37
***INPUT*** #1
S(ii%).obs, S(ii%).fa%, S(ii%).ls, S(ii%).ash,
S(ii%).elev, S(ii%).tlm, S(ii%).curves, S(ii%).sf,
S(ii%).wf, S(ii%).ls1tlm, S(ii%).ls2tlm, S(ii%).ls3tlm,
S(ii%).ls4tlm, S(ii%).ls5tlm, S(ii%).tlm9596, S(ii%).ls9596
***NEXT*** ii%
**(*loop to A*)**
***goto B***

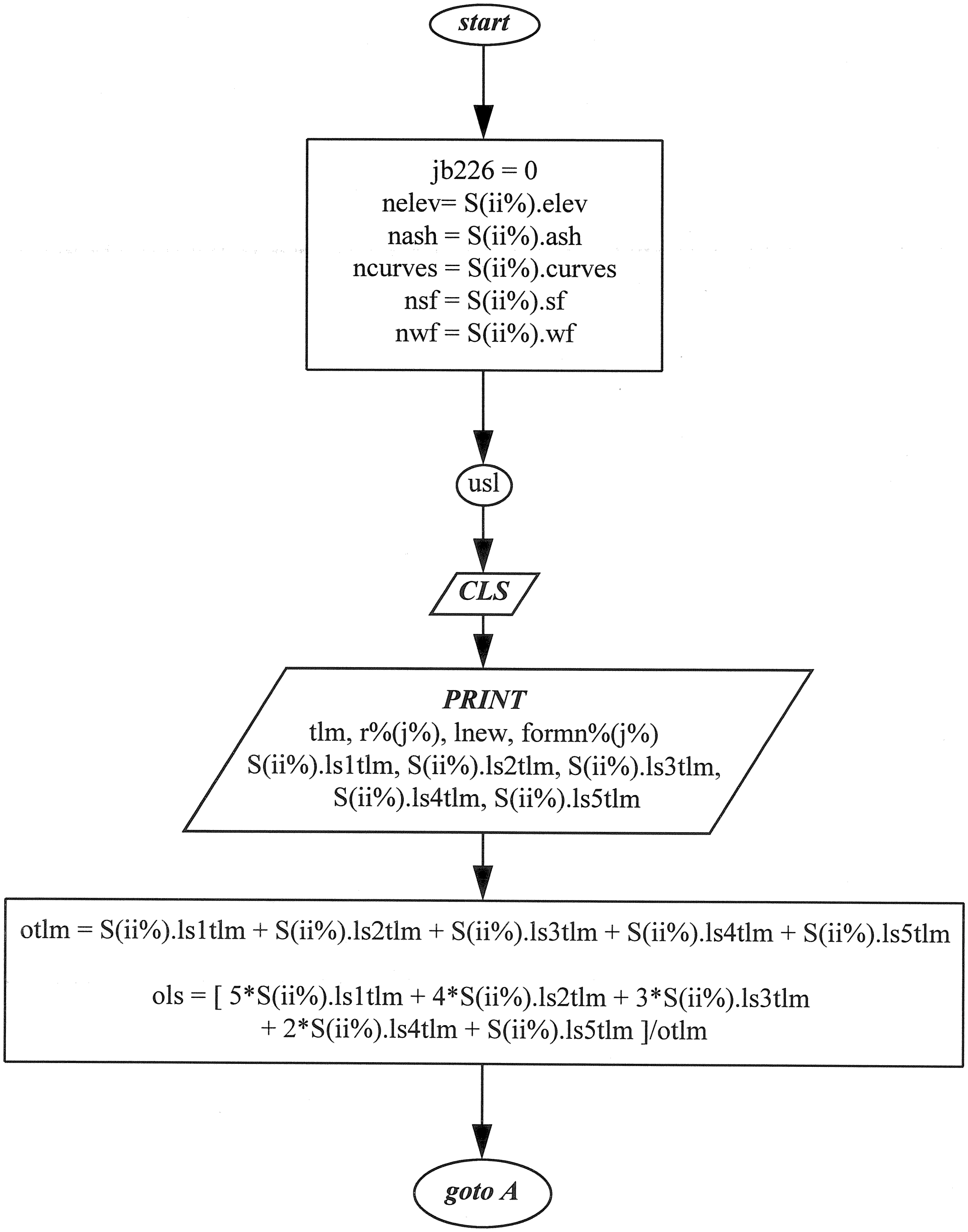
**Subroutine**
**info(**FILES$, t() AS trans, S() AS steady**)**
**page 2**



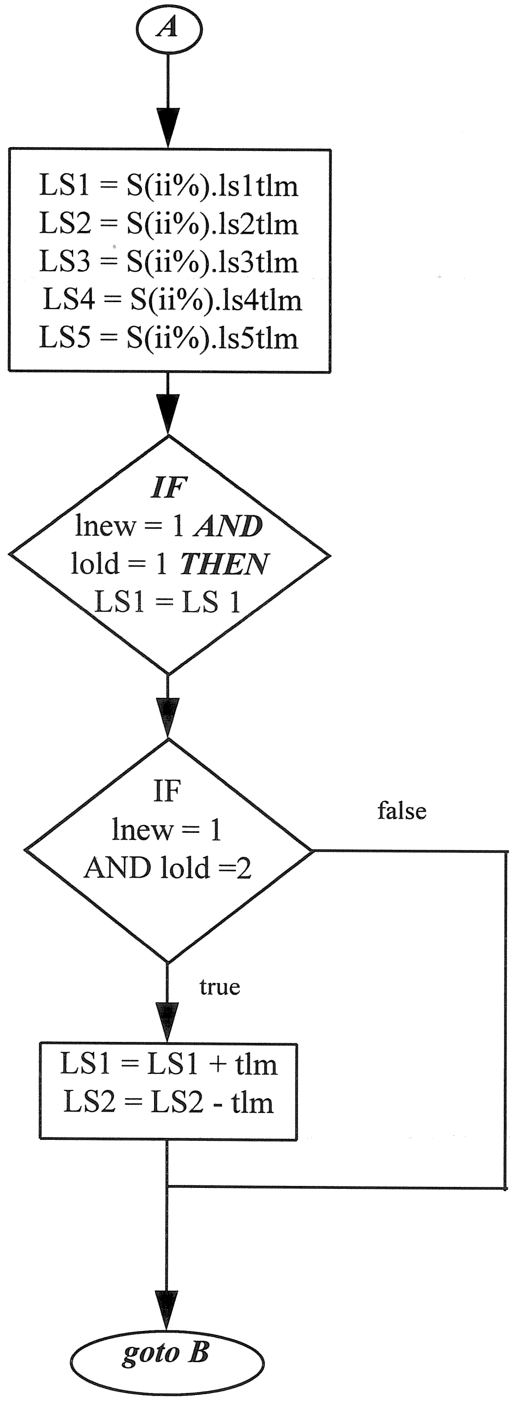
**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

**page 1**


start
jb226 = 0
nelev= S(ii%).elev
nash = S(ii%).ash
ncurves = S(ii%).curves
nsf = S(ii%).sf
nwf = S(ii%).wf
usl
CLS
PRINT
tlm, r%(j%), lnew, formn%(j%)
S(ii%).ls1tlm, S(ii%).ls2tlm, S(ii%).ls3tlm,
S(ii%).ls4tlm, S(ii%).ls5tlm
otlm = S(ii%).ls1tlm + S(ii%).ls2tlm + S(ii%).ls3tlm + S(ii%).ls4tlm + S(ii%).ls5tlm
ols = [ 5*S(ii%).ls1tlm + 4*S(ii%).ls2tlm + 3*S(ii%).ls3tlm
+ 2*S(ii%).ls4tlm + S(ii%).ls5tlm ]/otlm
goto A

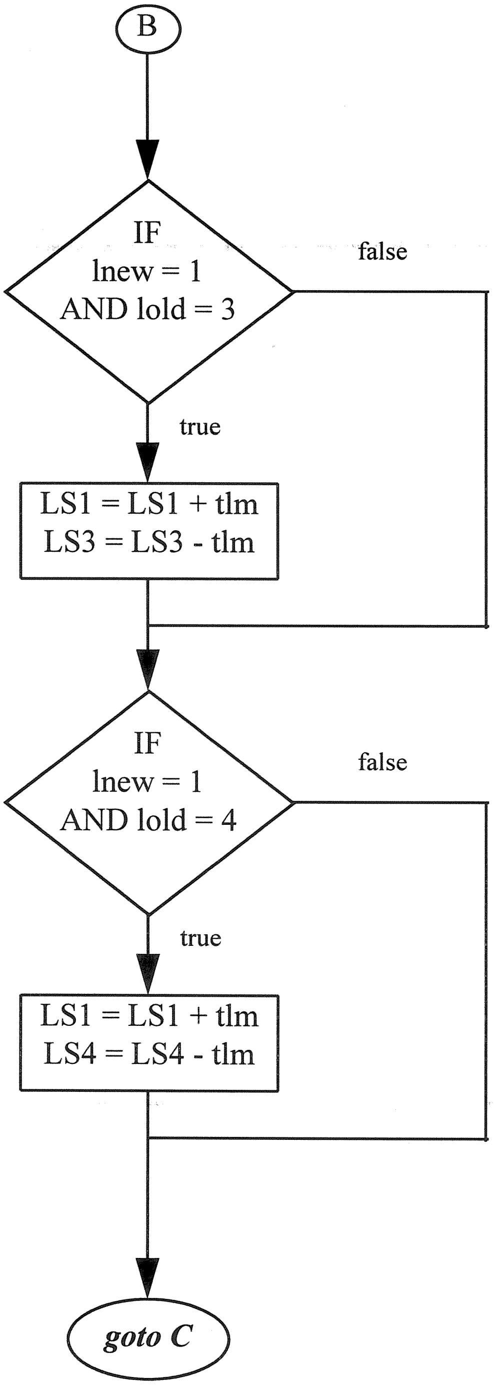
**Subroutine**
**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**
**page 2**



**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

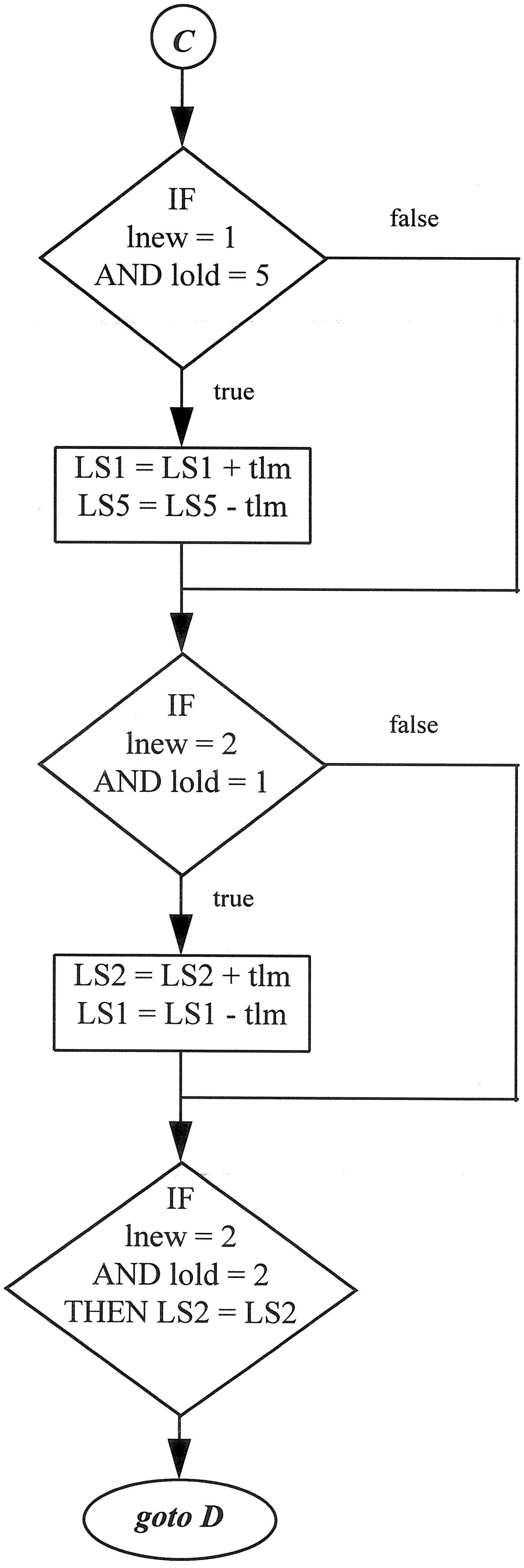
**page 3**



**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

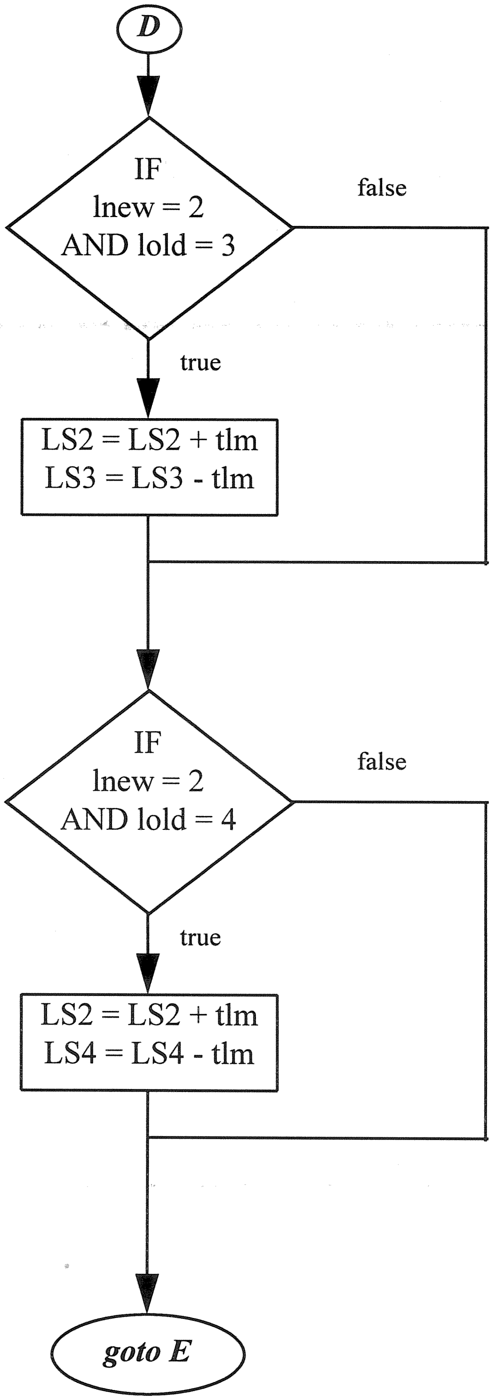
**page 4**



**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

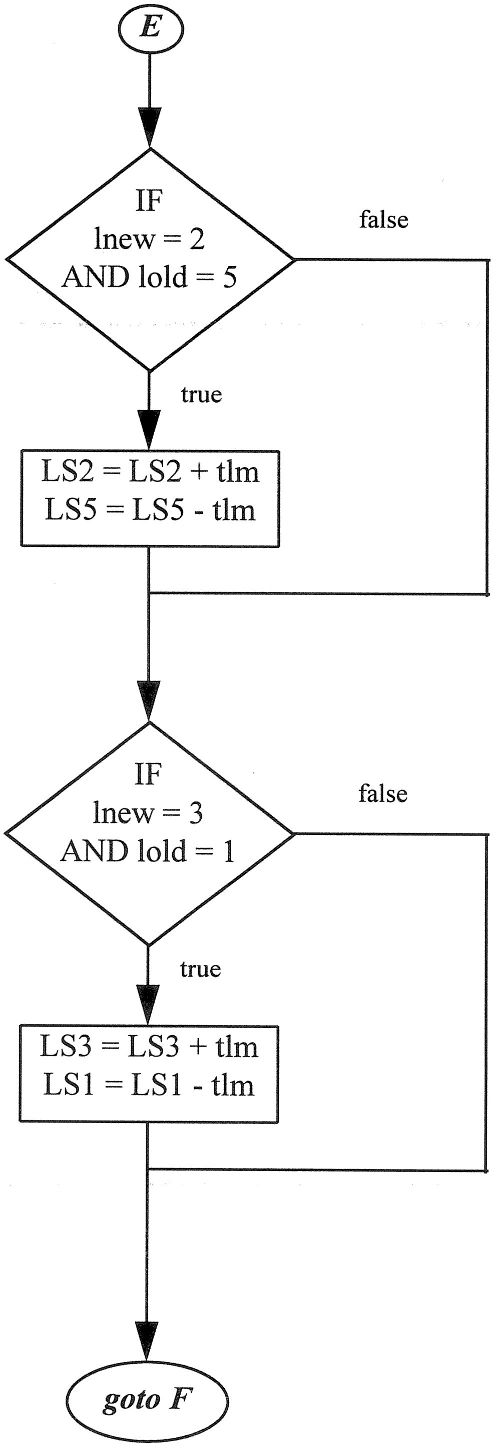
**page 5**



**Subroutine**

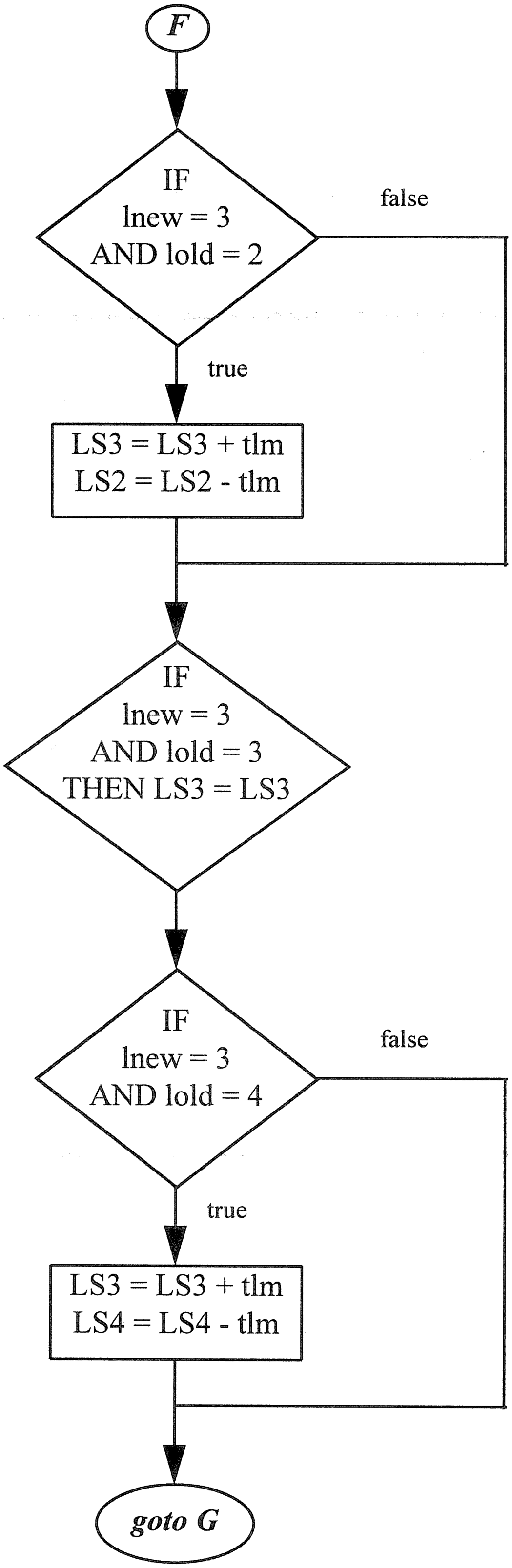
**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

**page 6**



**Subroutine**
**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los,
tlm, ols, otlm, r%(), l(), lnew, lold**)**

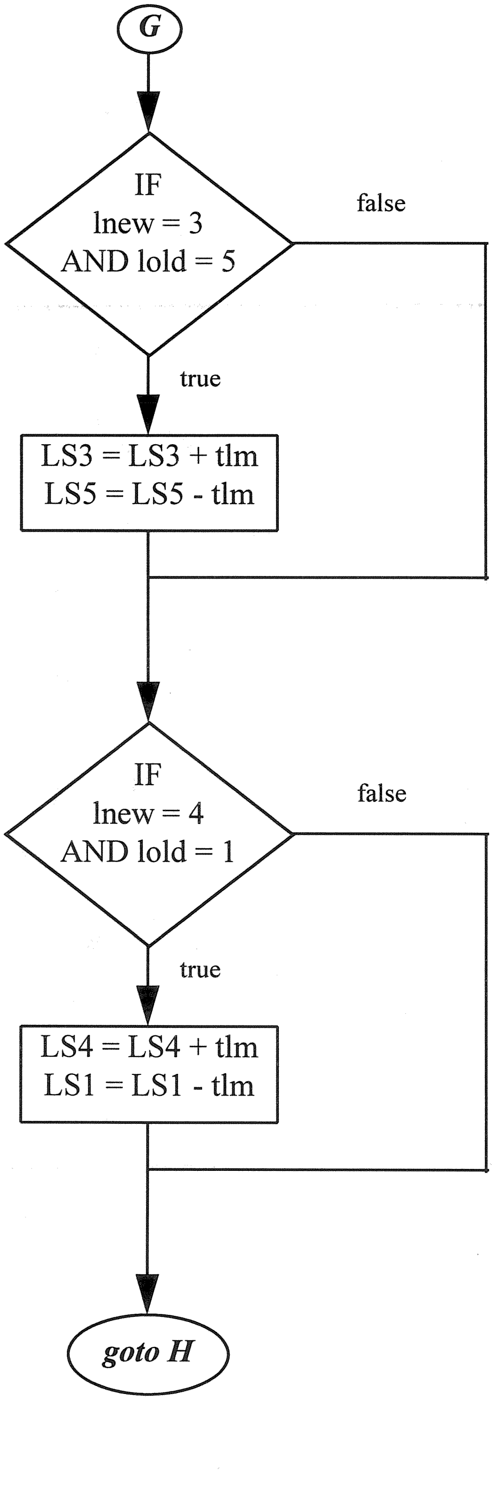
**page 7**



**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold)

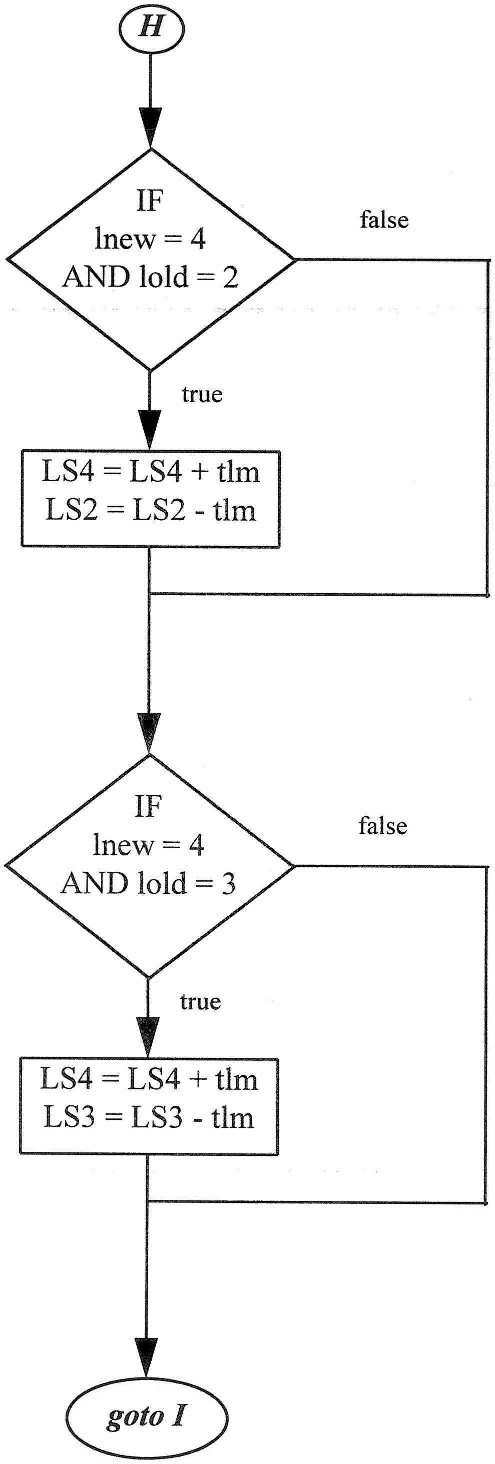
**page 8**



**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

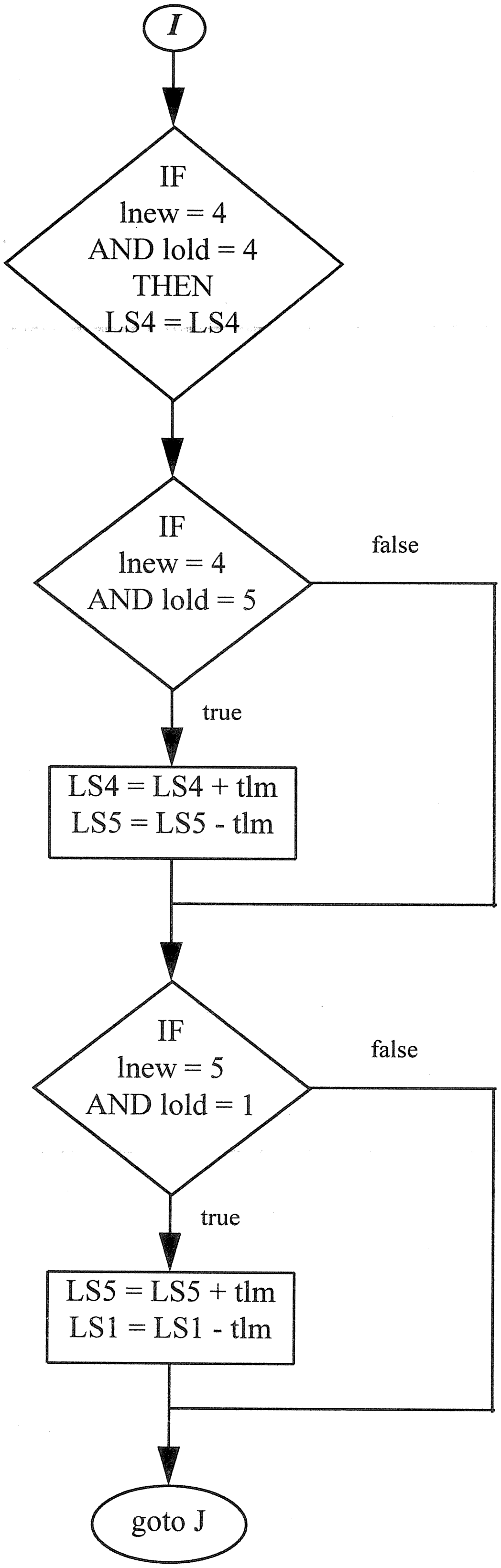
**page 9**



**Subroutine**

**SCR5(**j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold**)**

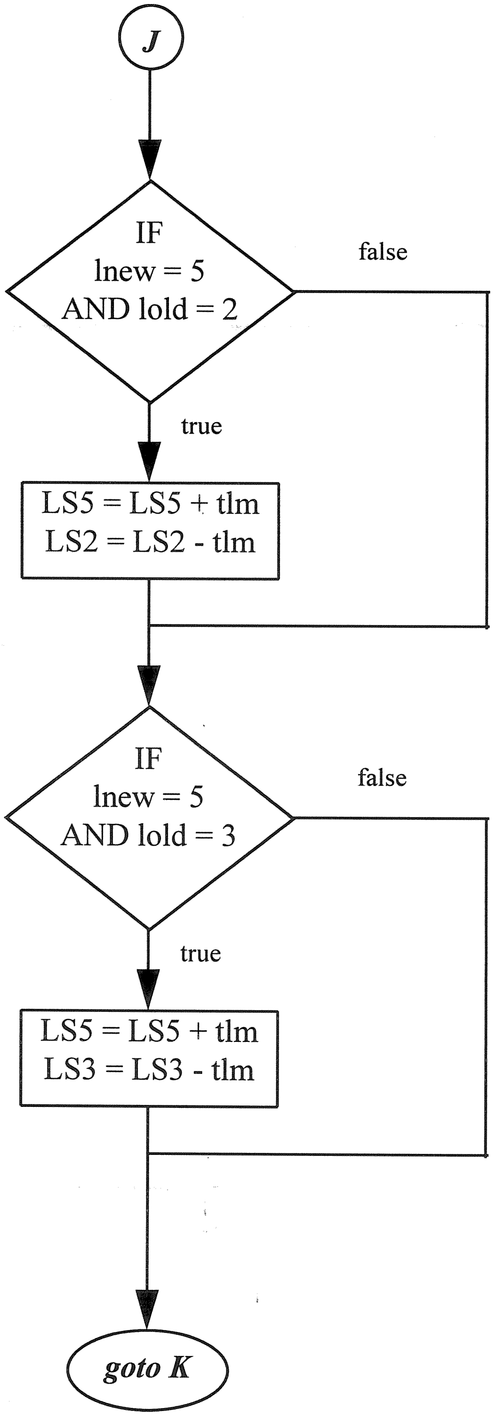
**page 10**



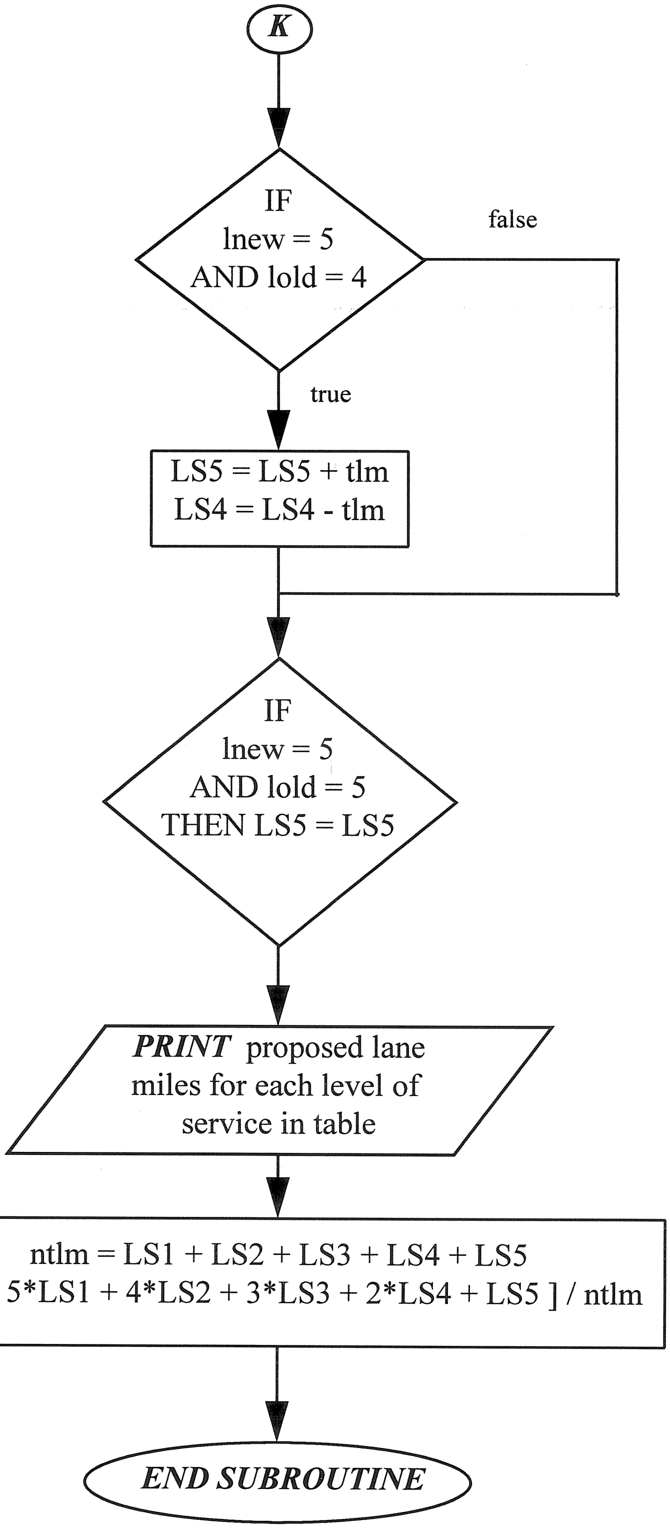
**Subroutine**

**SCR5**(j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold)

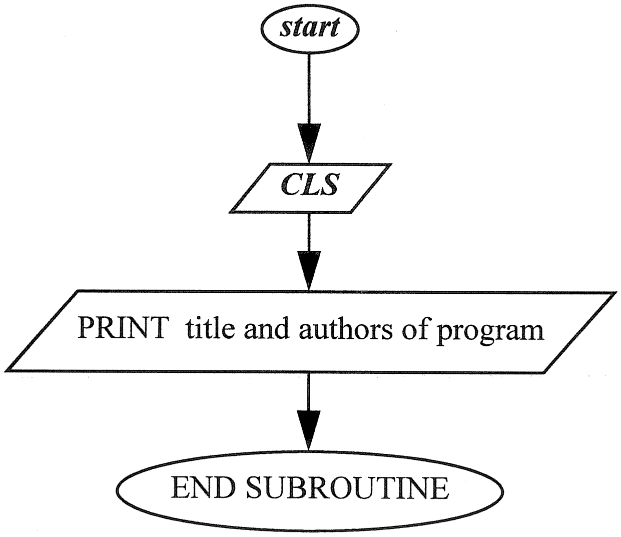
**page 11**



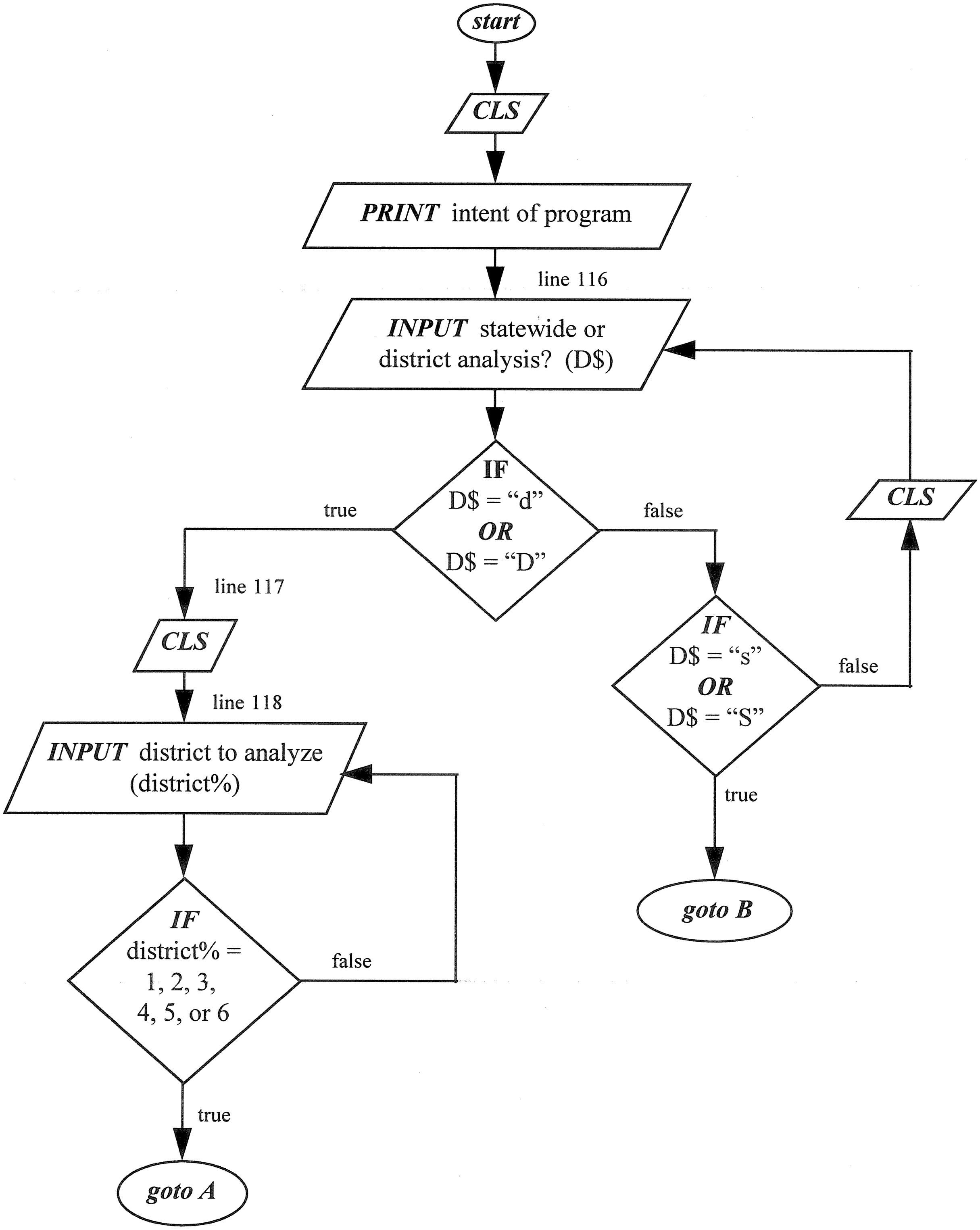
**Subroutine**
**SCR5**(j%, ndist, formn%, t(), S(),nls, ntlm, nelev, nash, ncurves,nsf, nwf, los, tlm, ols, otlm, r%(), l(), lnew, lold)
**page 12**


K
IF lnew = 5 AND lold = 4
false
true
LS5 = LS5 + tlm
LS4 = LS4 - tlm
IF lnew = 5 AND lold = 5 THEN LS5 = LS5
PRINT proposed lane miles for each level of service in table
ntlm = LS1 + LS2 + LS3 + LS4 + LS5
nls = [ 5*LS1 + 4*LS2 + 3*LS3 + 2*LS4 + LS5 ] / ntlm
END SUBROUTINE

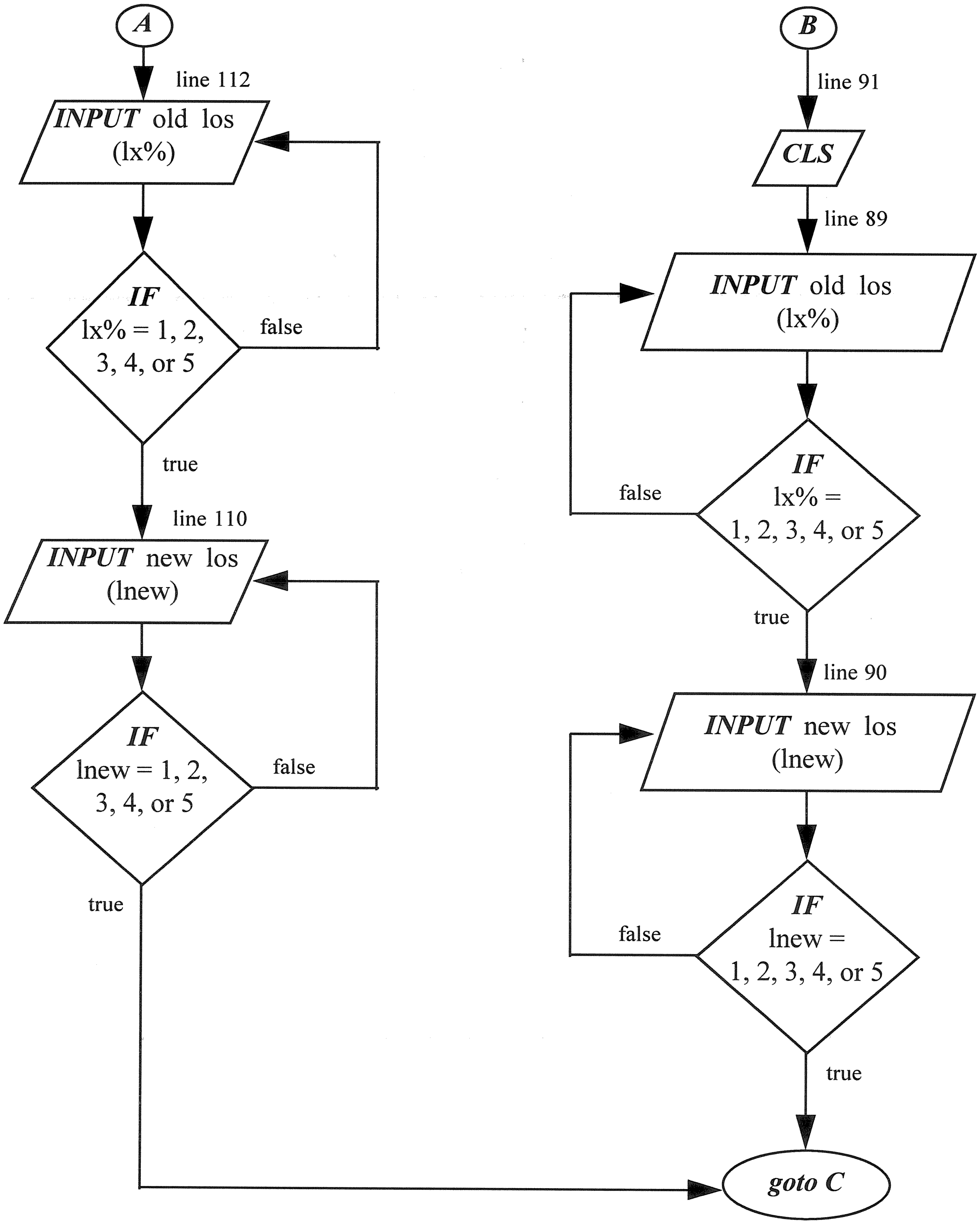
## Subroutine
## scrn1( )
### page 1



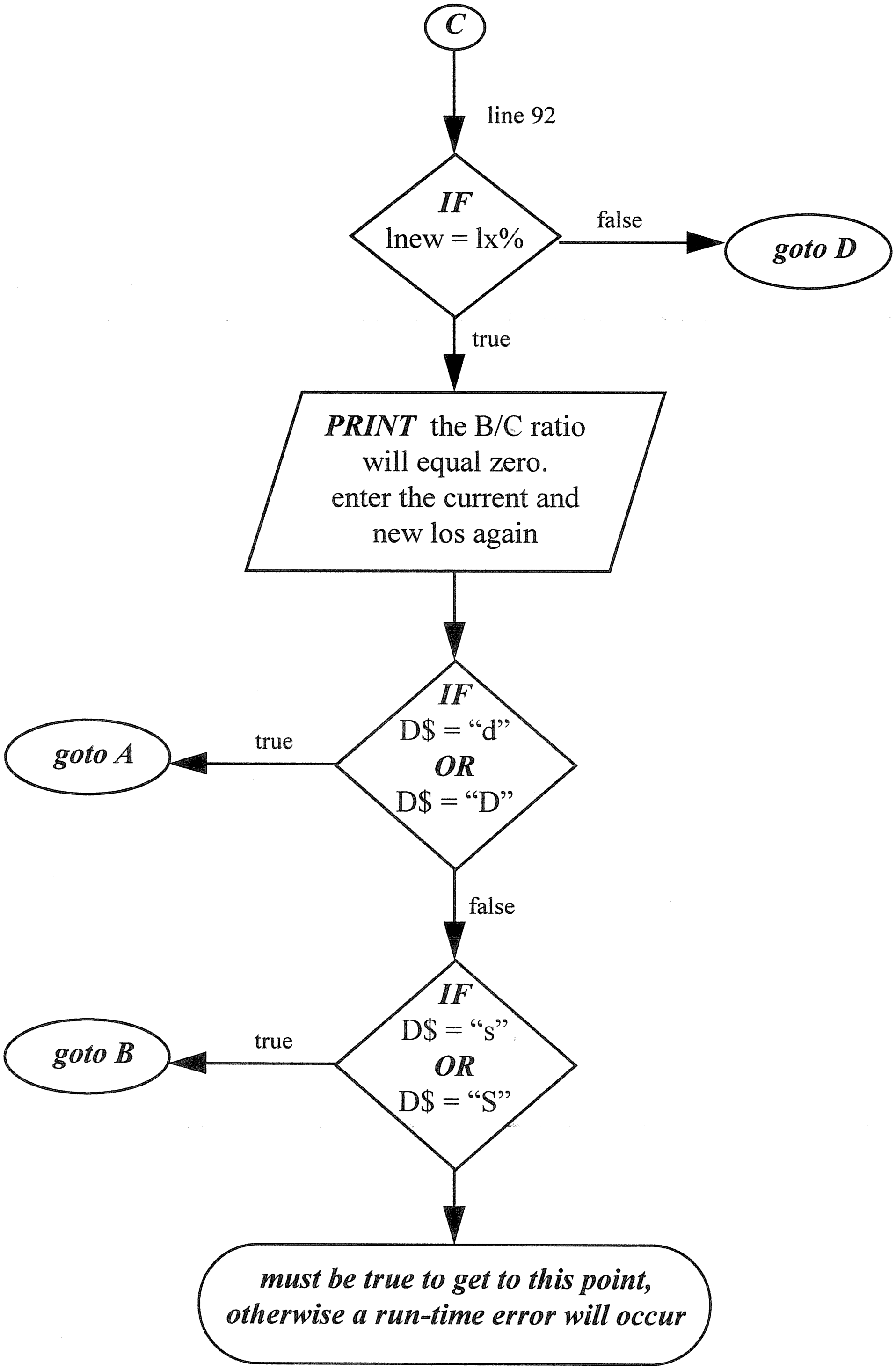
**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 1**



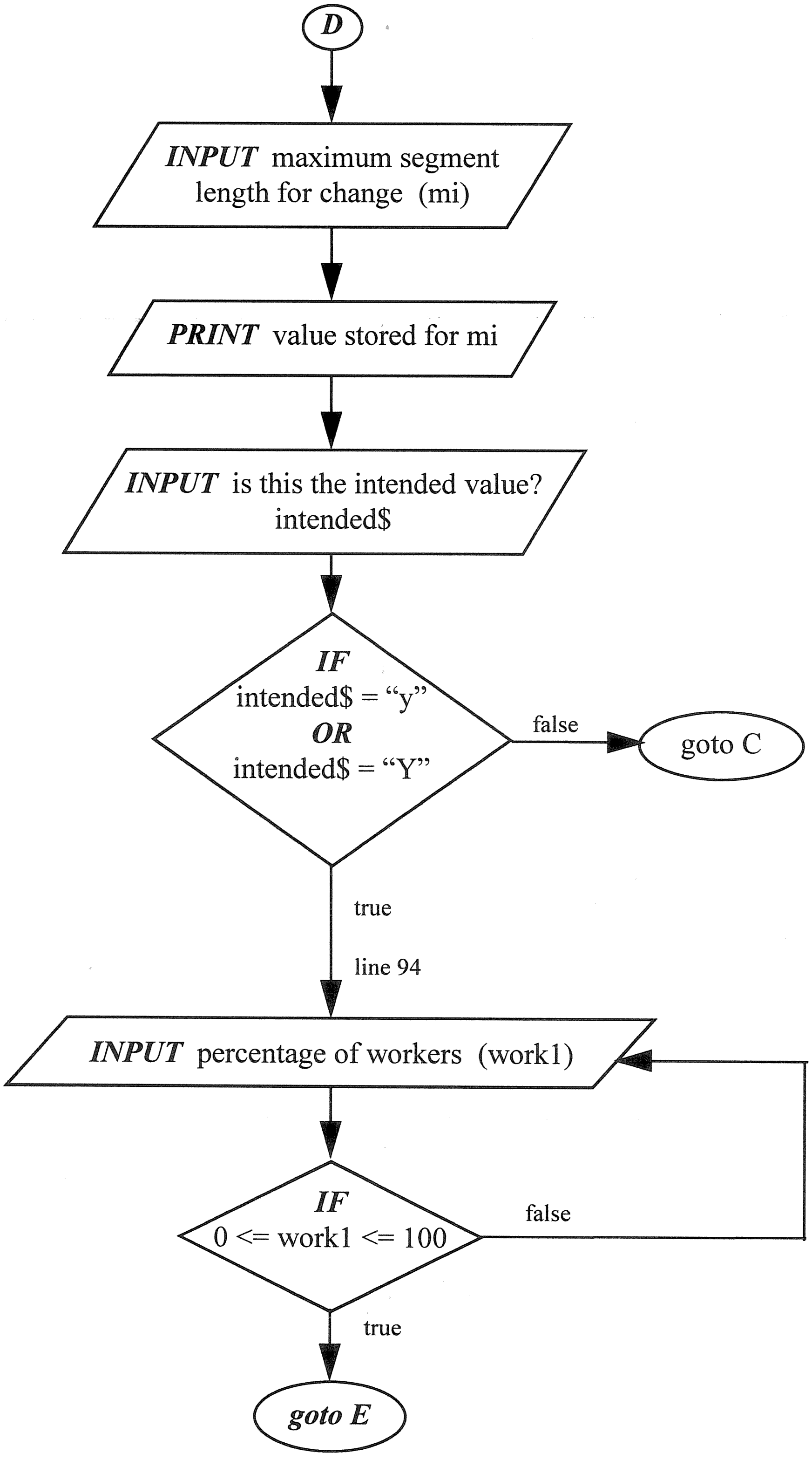
**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 2**



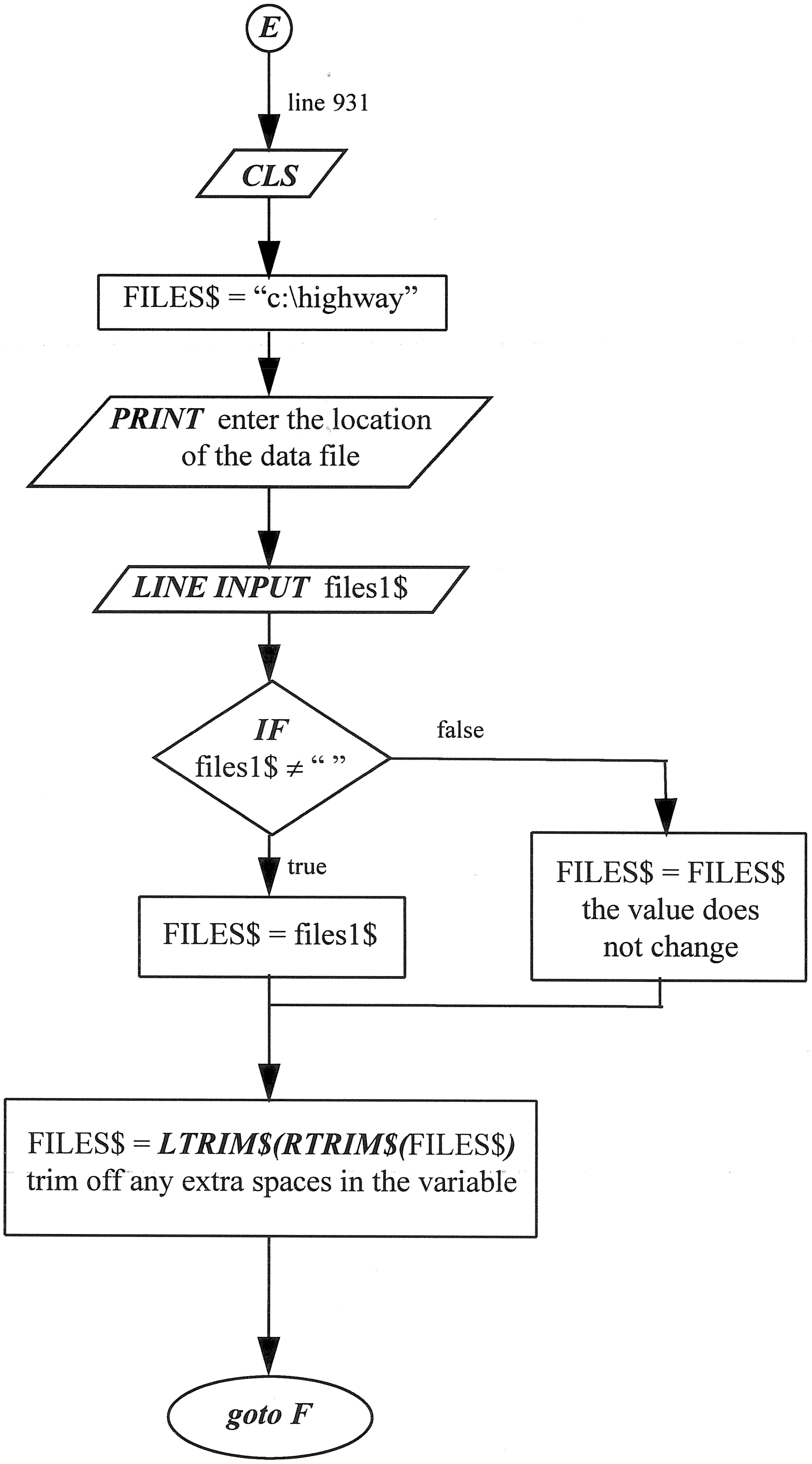
**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 3**



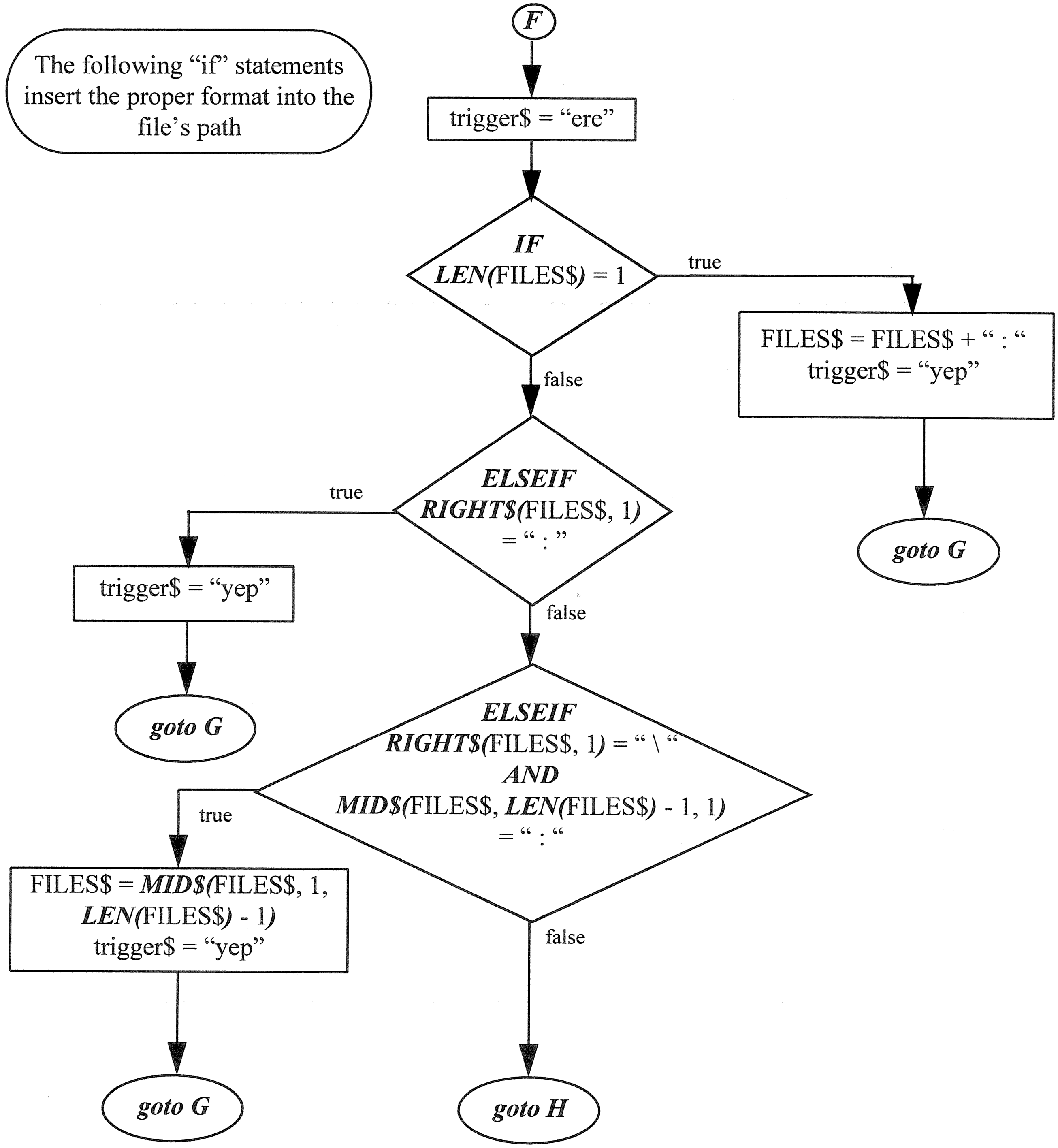
**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 4**


D
*INPUT* maximum segment length for change (mi)
*PRINT* value stored for mi
*INPUT* is this the intended value? intended$
*IF* intended$ = "y" *OR* intended$ = "Y"
false
goto C
true
line 94
*INPUT* percentage of workers (work1)
*IF* 0 <= work1 <= 100
false
true
*goto E*

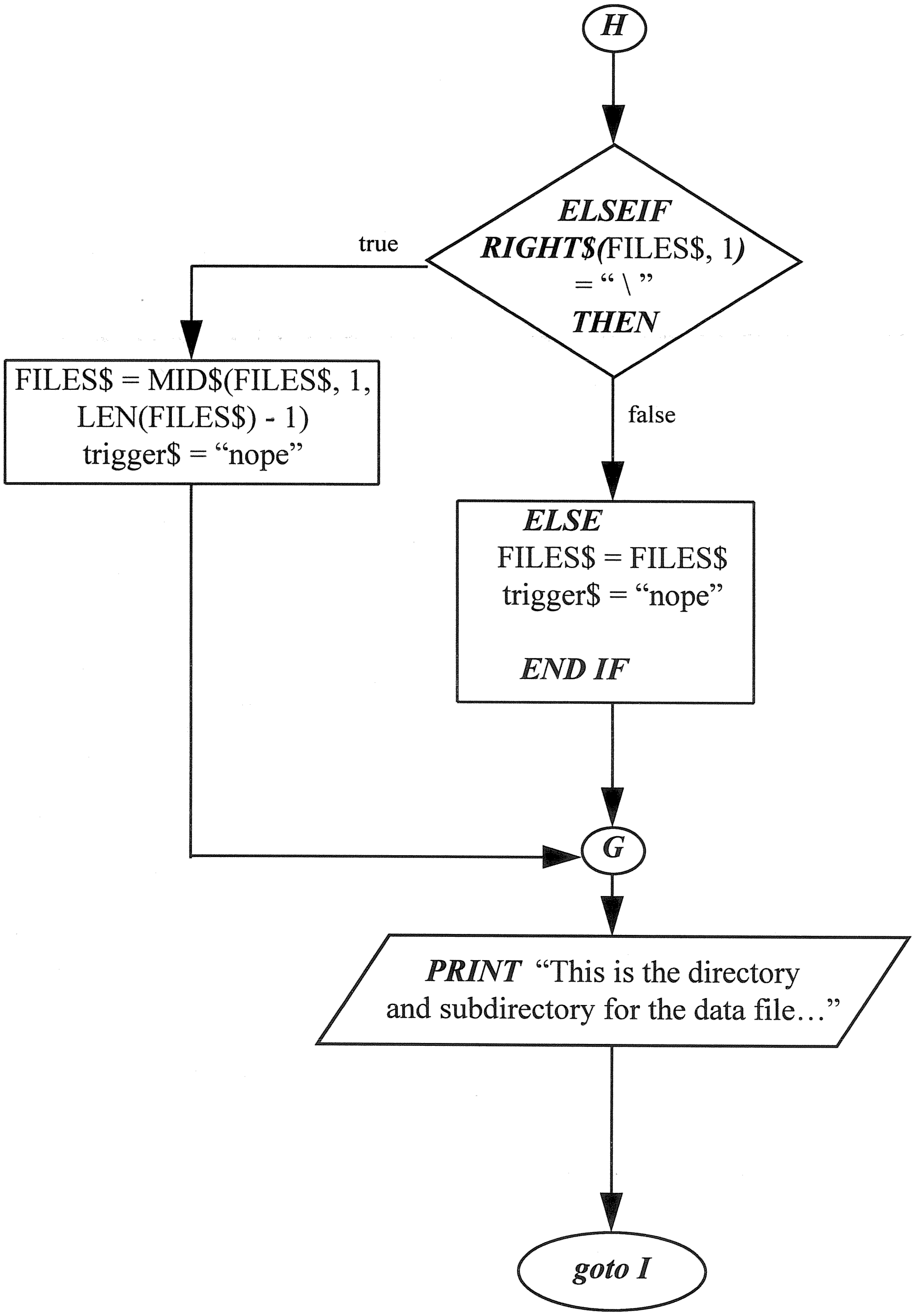
**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 5**


E
line 931
CLS
FILES$ = "c:\highway"
PRINT enter the location of the data file
LINE INPUT files1$
IF files1$ ≠ " "
false
true
FILES$ = files1$
FILES$ = FILES$ the value does not change
FILES$ = LTRIM$(RTRIM$(FILES$) trim off any extra spaces in the variable
goto F

**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 6**


F
The following "if" statements insert the proper format into the file's path
trigger$ = "ere"
IF LEN(FILES$) = 1
true
FILES$ = FILES$ + " : "
trigger$ = "yep"
goto G
false
ELSEIF RIGHT$(FILES$, 1) = " : "
true
trigger$ = "yep"
goto G
false
ELSEIF RIGHT$(FILES$, 1) = " \ " AND MID$(FILES$, LEN(FILES$) - 1, 1) = " : "
true
FILES$ = MID$(FILES$, 1, LEN(FILES$) - 1)
trigger$ = "yep"
goto G
false
goto H

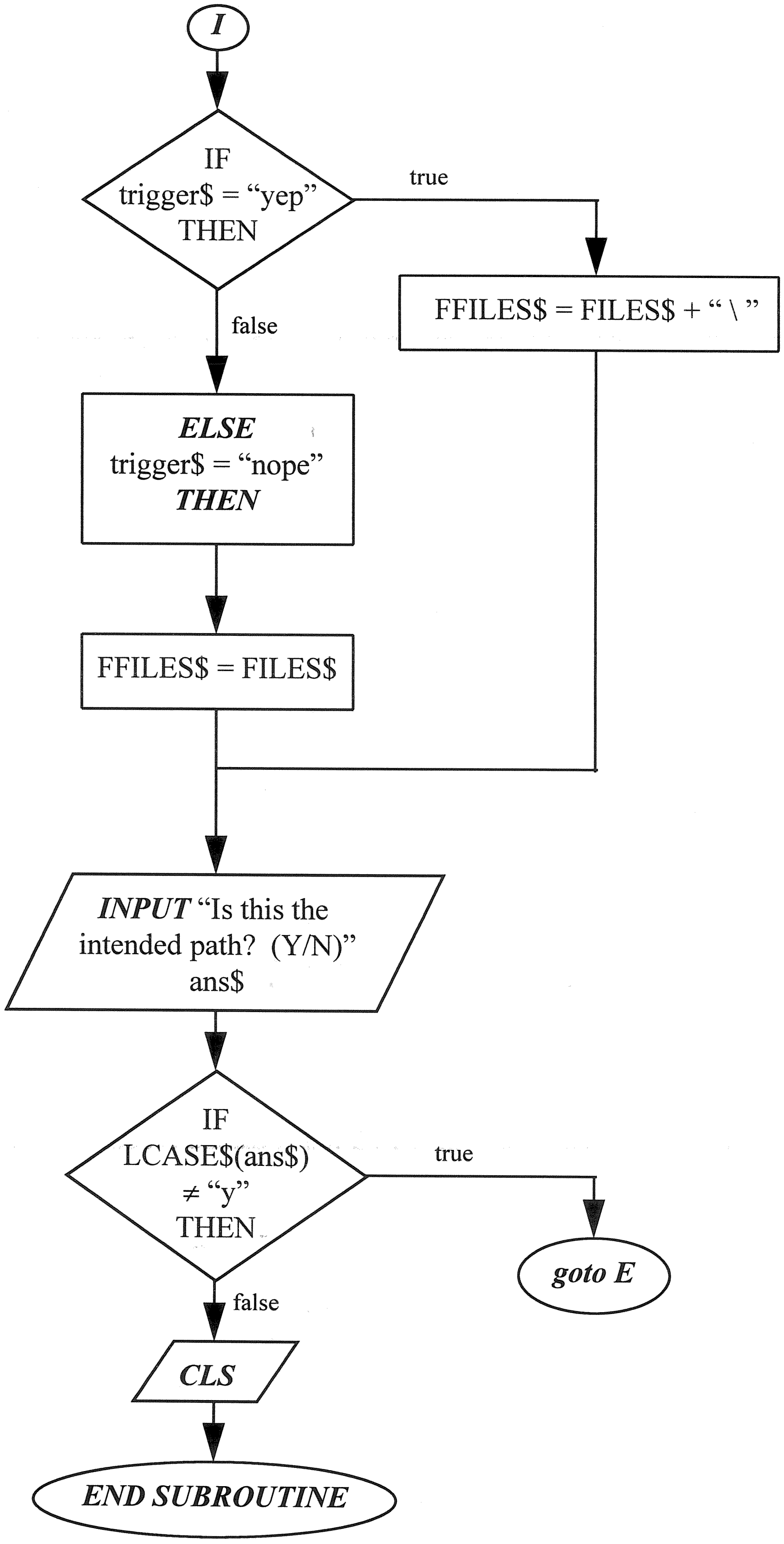
**Subroutine**
**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)
**page 7**



**Subroutine**

**scrn3**(lx%, lnew, FILES$, work1, mi, D$, district%)

**page 8**

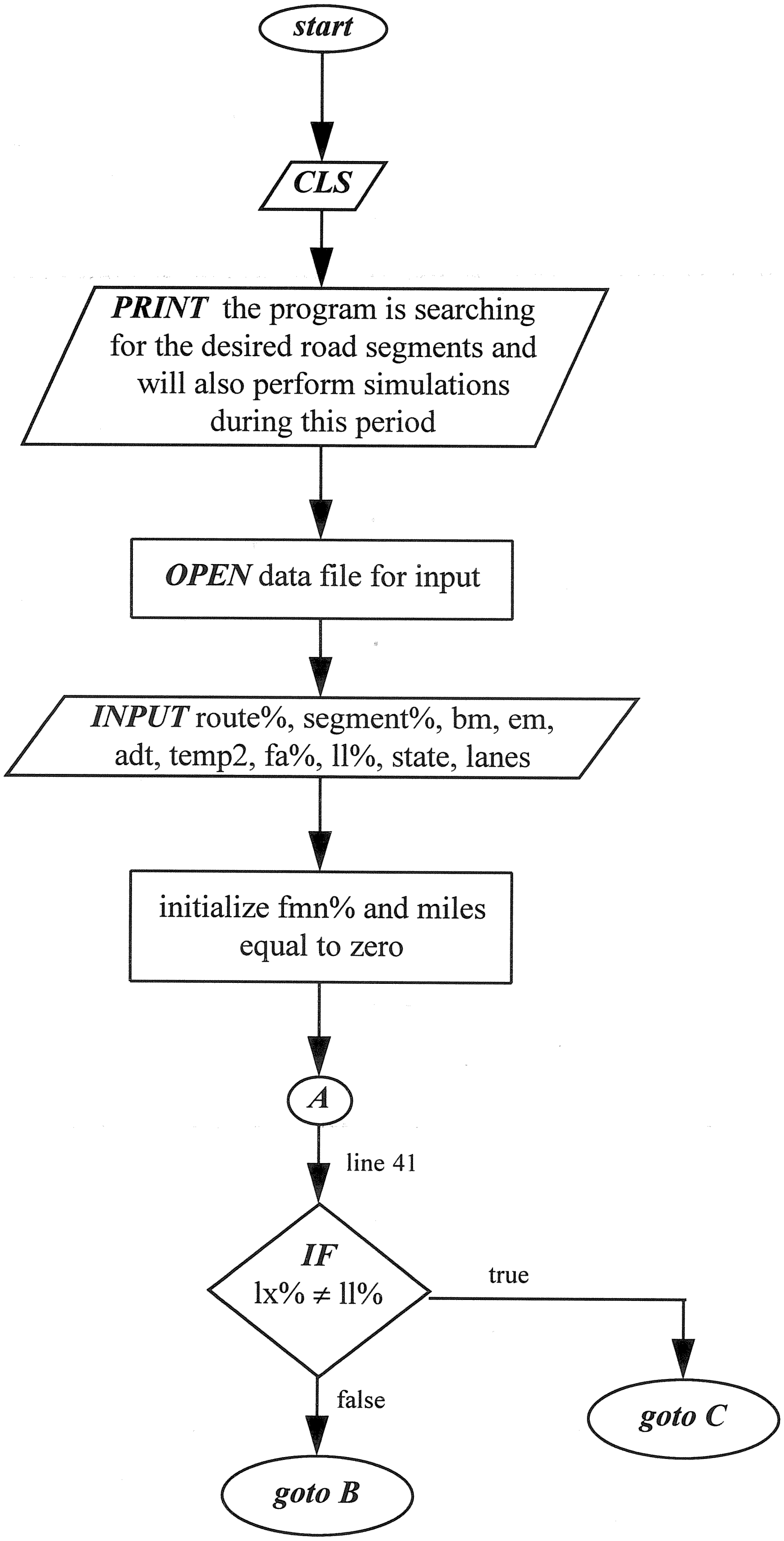


**Subroutine**

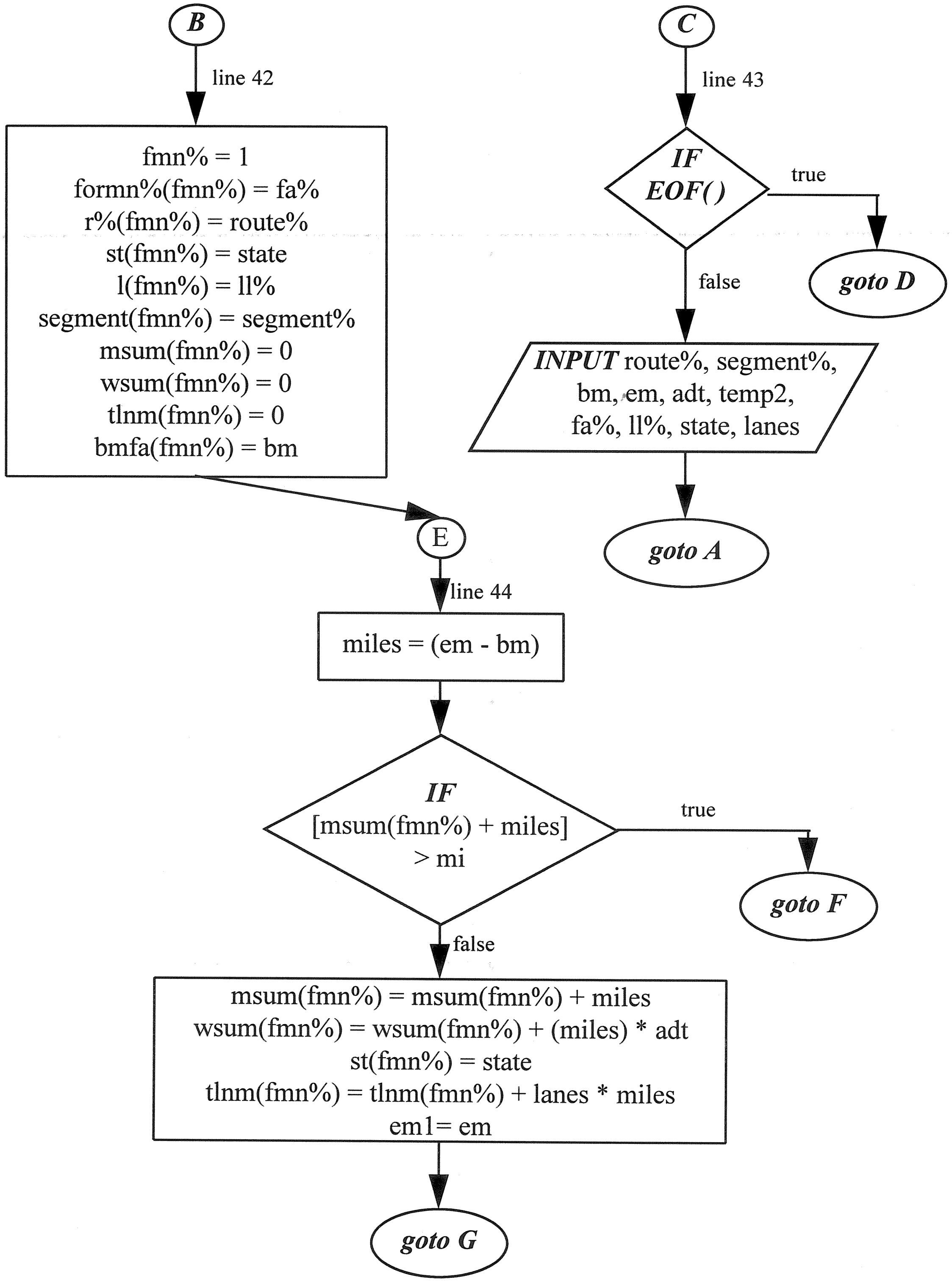
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%,
mi, FILES$, tlnm() )

***--statewide analysis--***

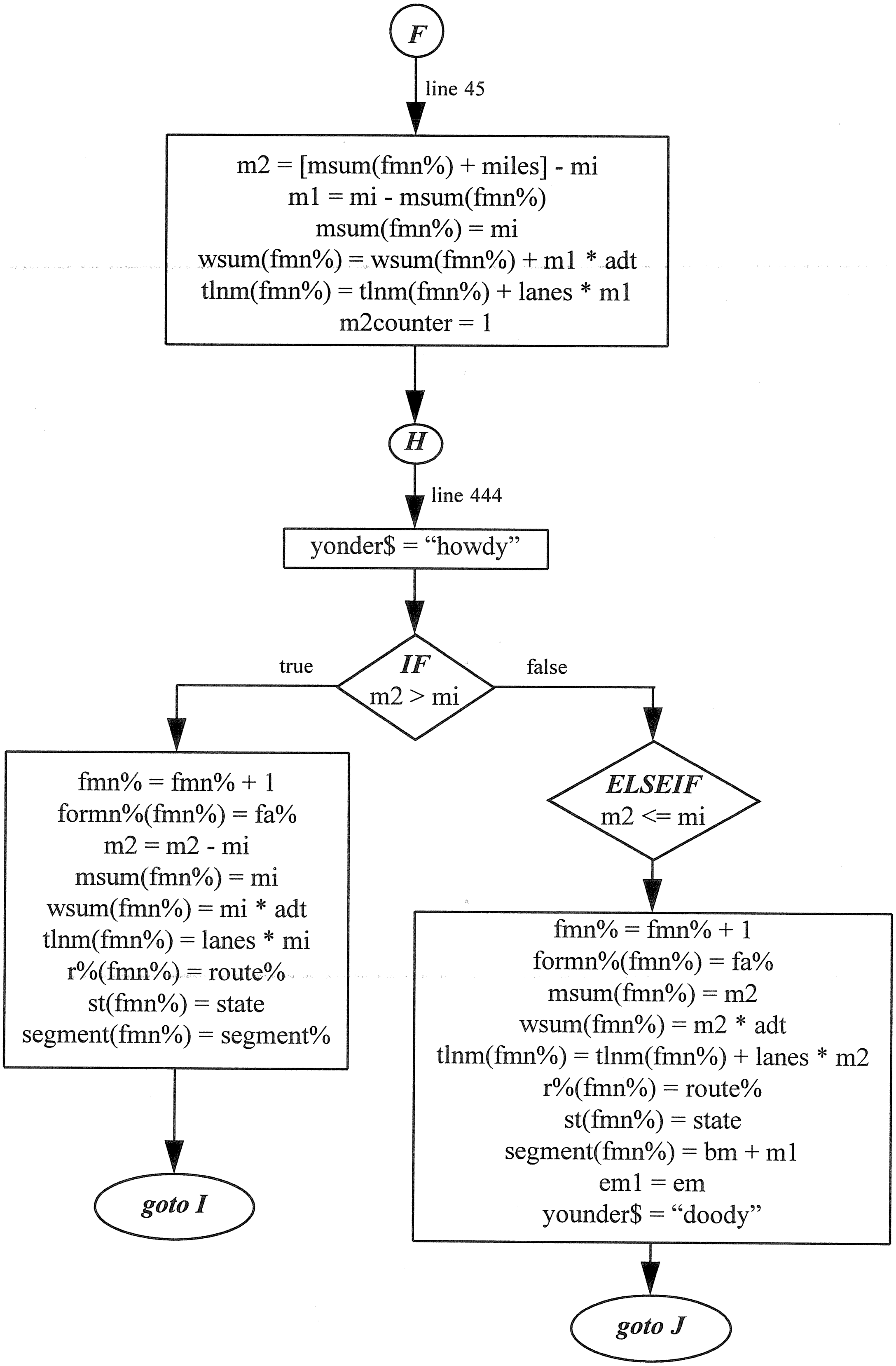
**page 1**


start
CLS
PRINT the program is searching for the desired road segments and will also perform simulations during this period
OPEN data file for input
INPUT route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
initialize fmn% and miles equal to zero
A
line 41
IF lx% ≠ ll%
true
false
goto C
goto B

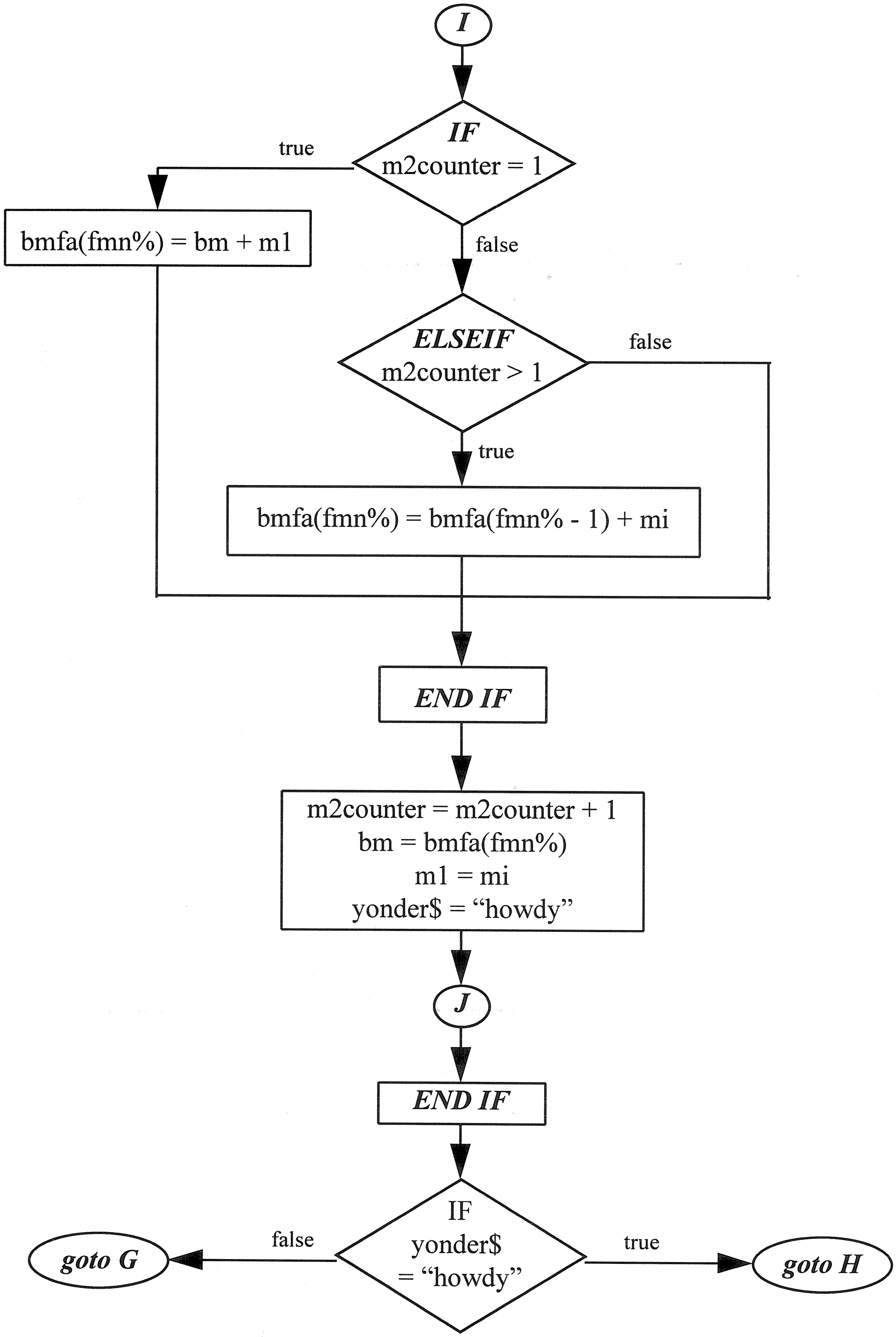
**Subroutine**
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%,
mi, FILES$, tlnm() )
**page 2**



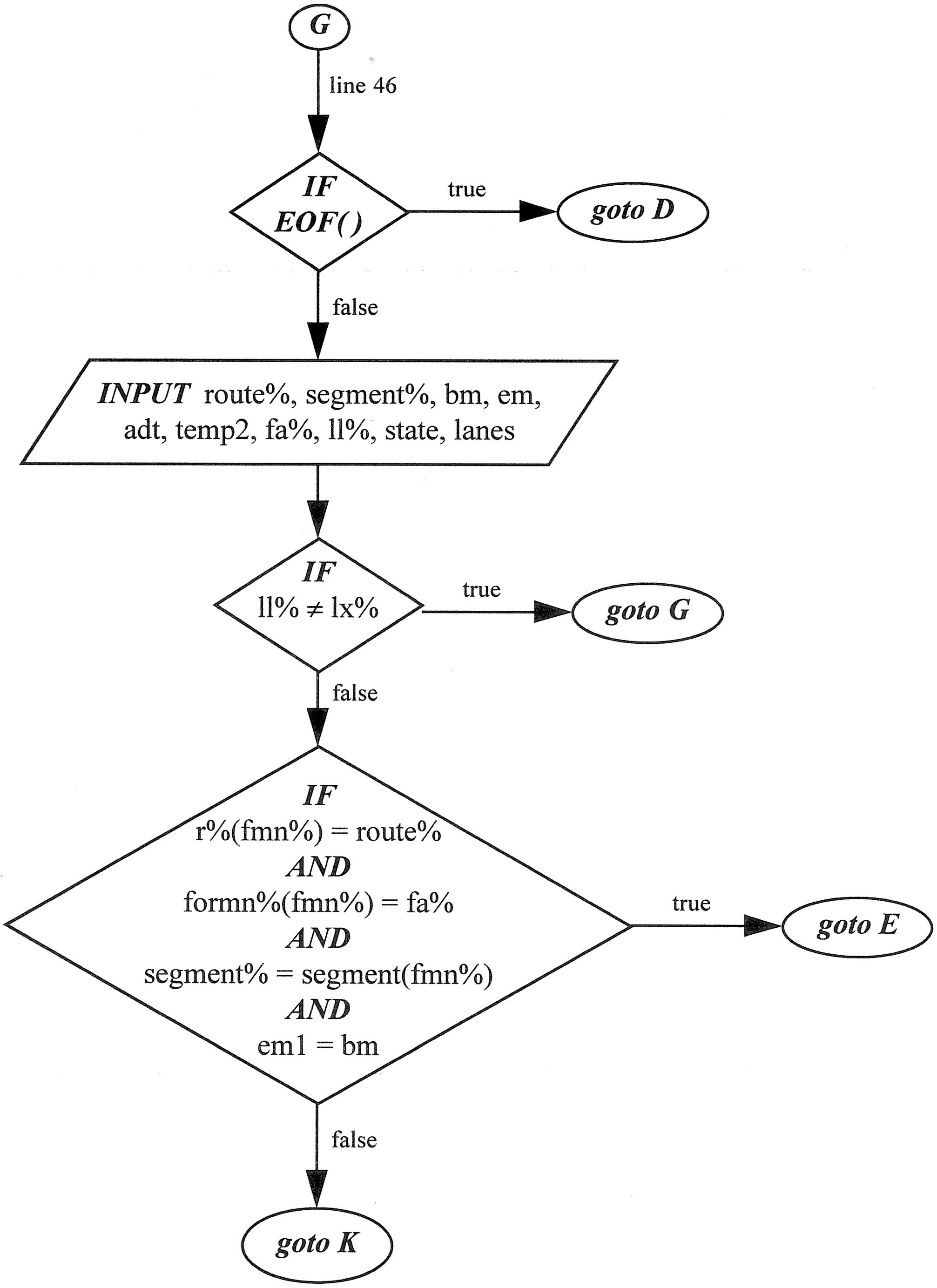
**Subroutine**
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, FILES$, tlnm() )
**page 3**



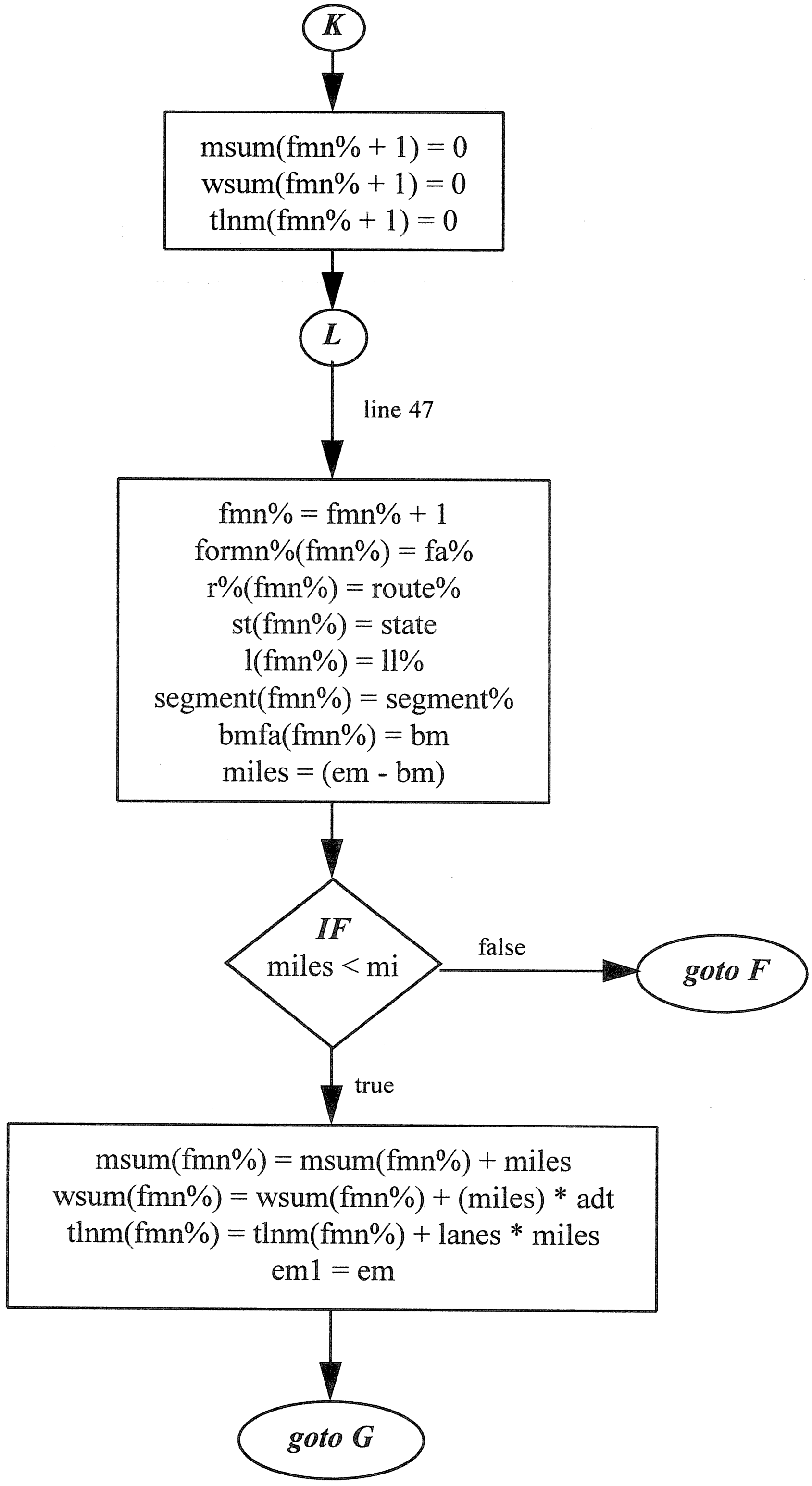
**Subroutine**
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%,
mi, FILES$, tlnm() )
**page 4**



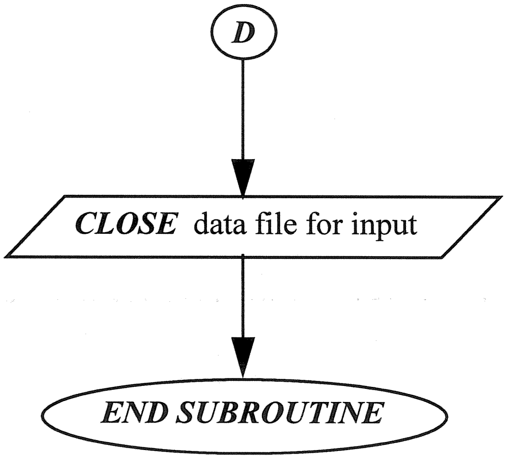
**Subroutine**
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, FILES$, tlnm() )
**page 5**



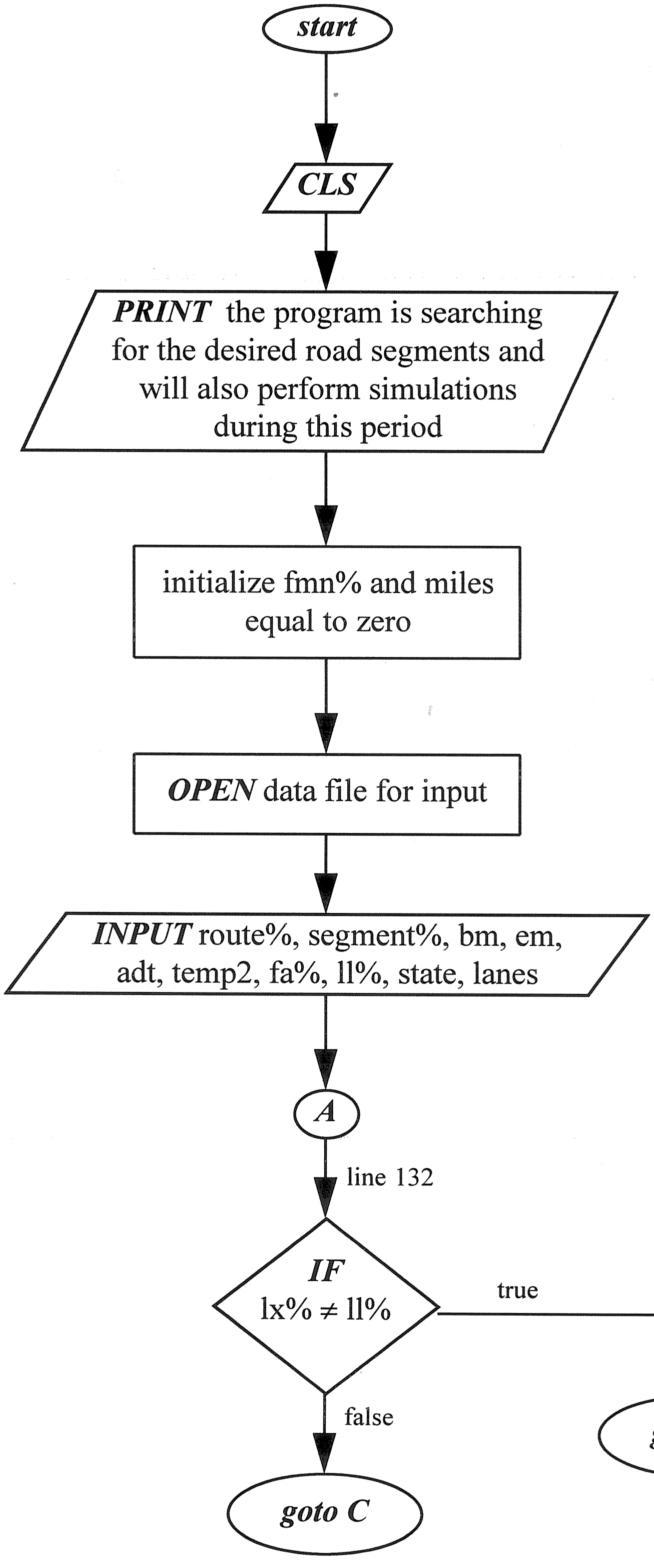
**Subroutine**
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, FILES$, tlnm() )
**page 6**



**Subroutine**
**wadt1(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%,
mi, FILES$, tlnm() )
**page 7**



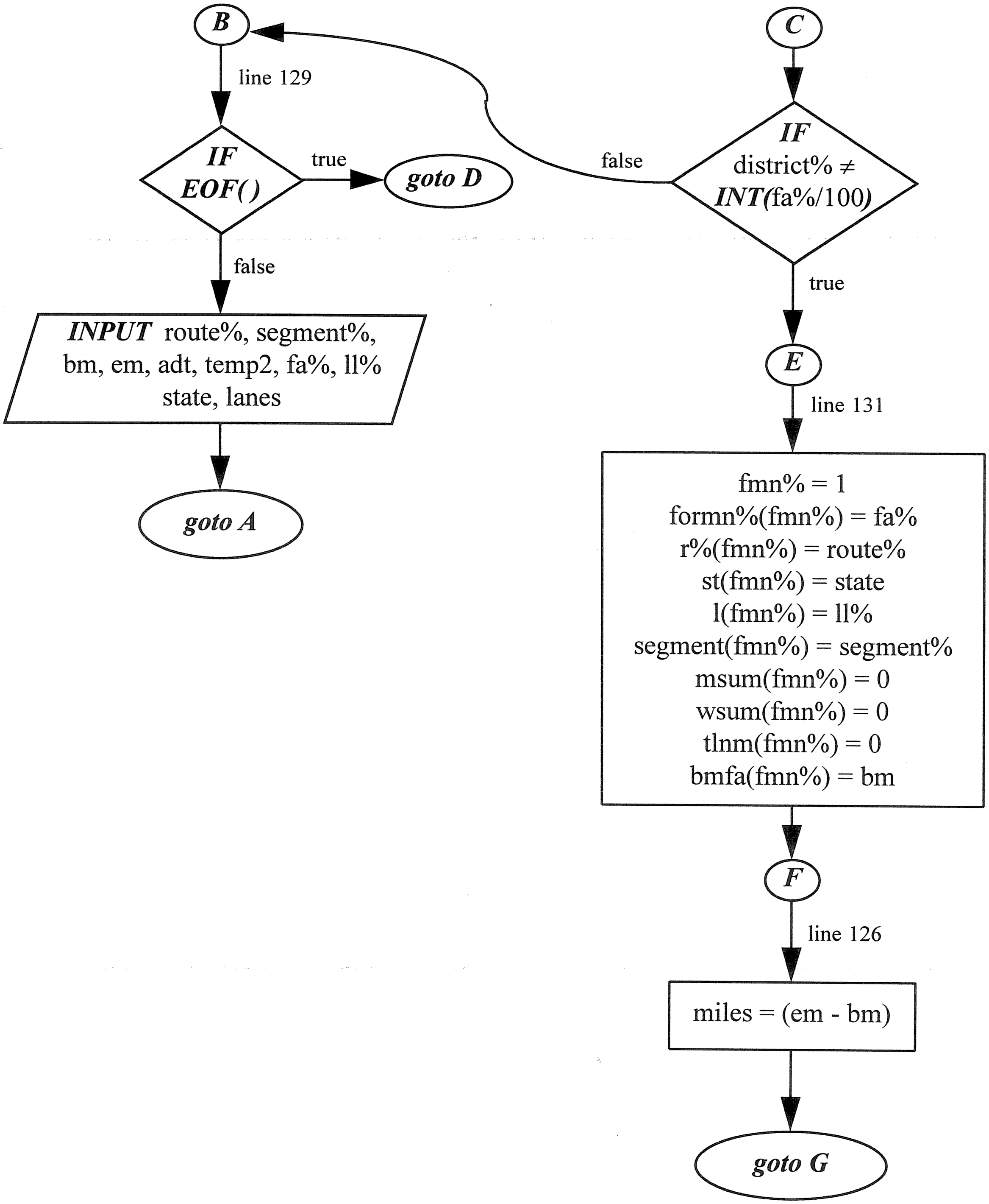
**Subroutine**
**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, district% , FILES$, tlnm() )
***--district analysis--***
**page 1**


start
CLS
PRINT the program is searching for the desired road segments and will also perform simulations during this period
initialize fmn% and miles equal to zero
OPEN data file for input
INPUT route%, segment%, bm, em, adt, temp2, fa%, ll%, state, lanes
A
line 132
IF lx% ≠ ll%
true
false
goto B
goto C

**Subroutine**

**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, district%, FILES$, tlnm() )

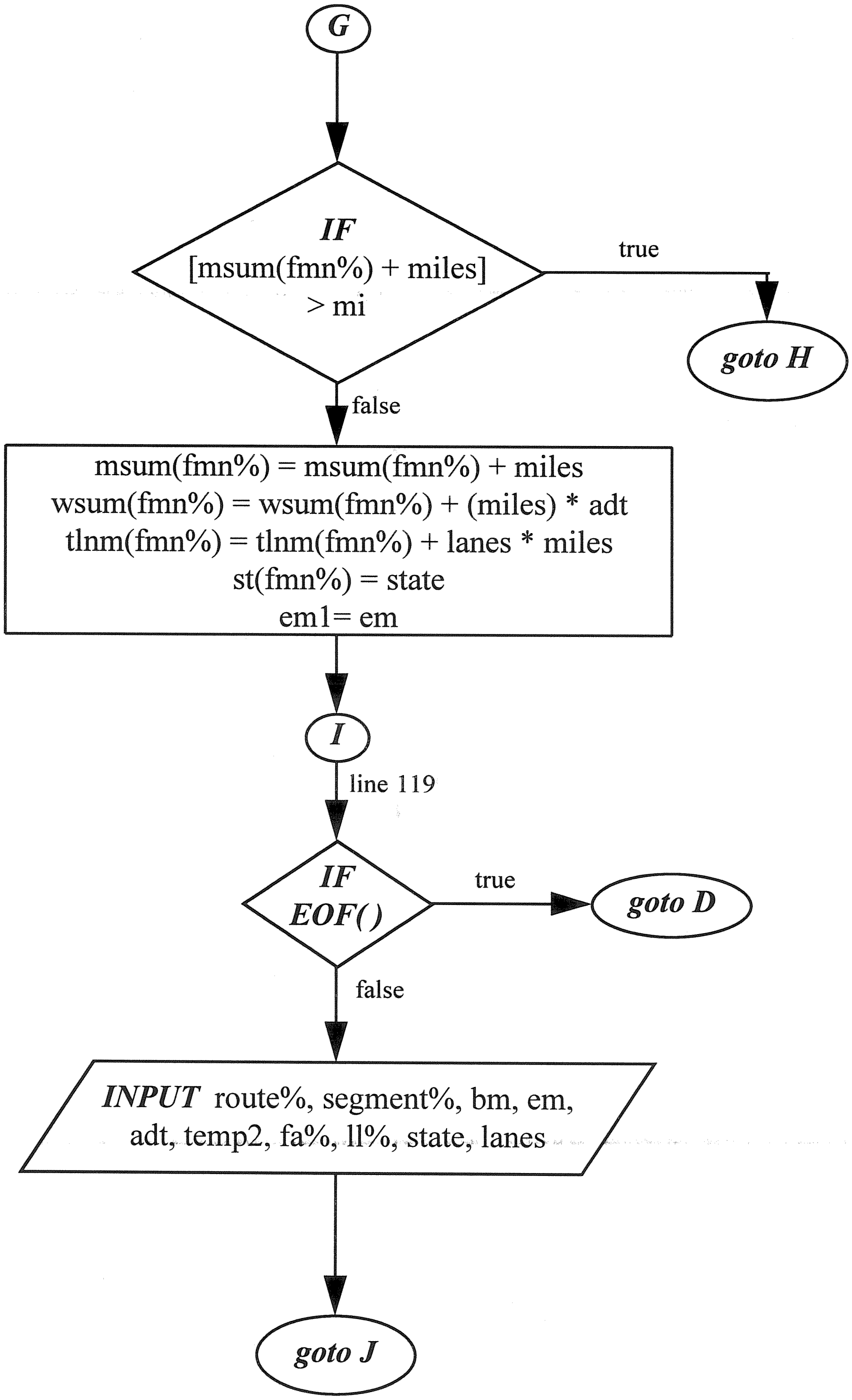
**page 2**



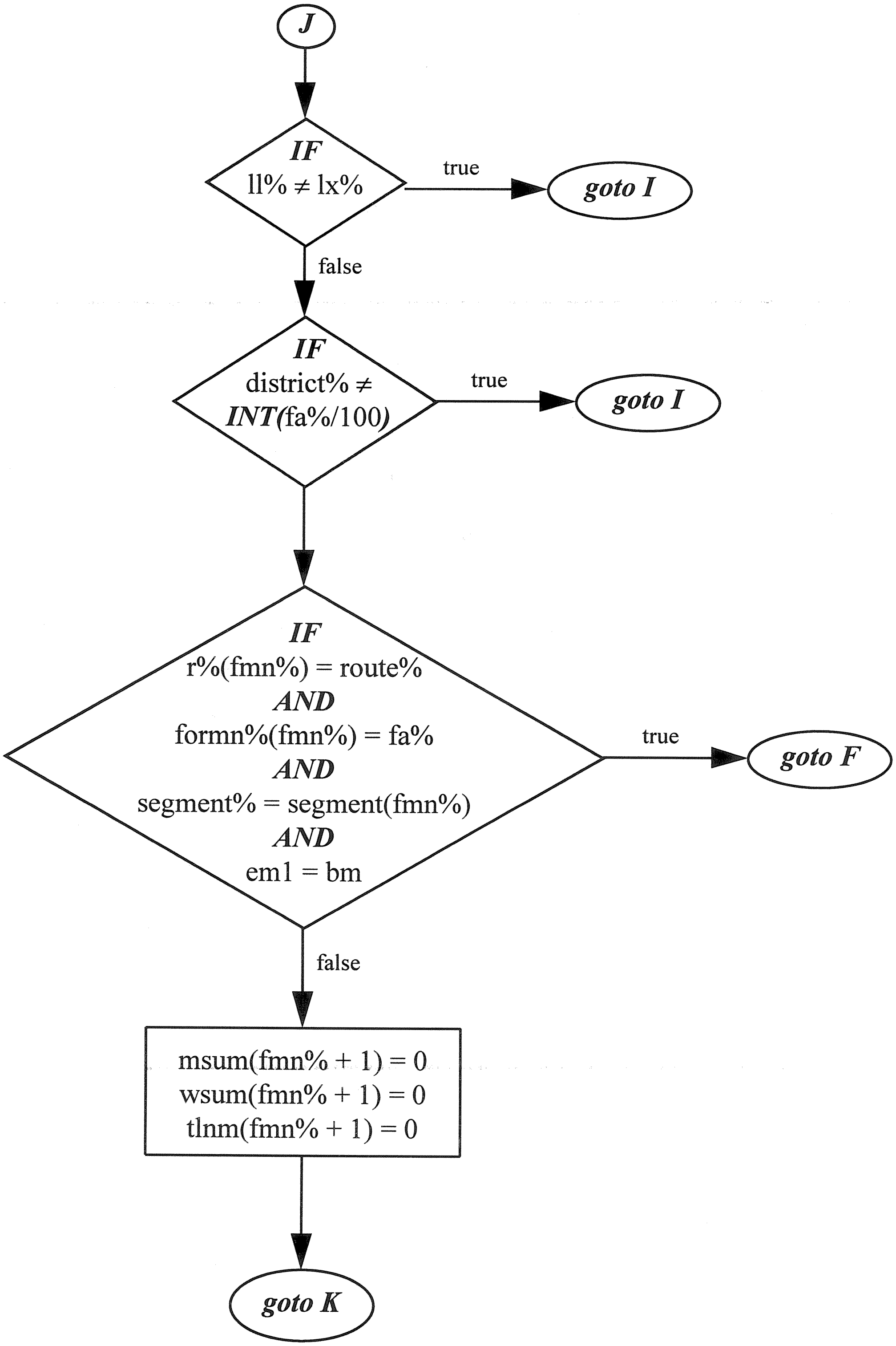
**Subroutine**

**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, district%, FILES$, tlnm() )

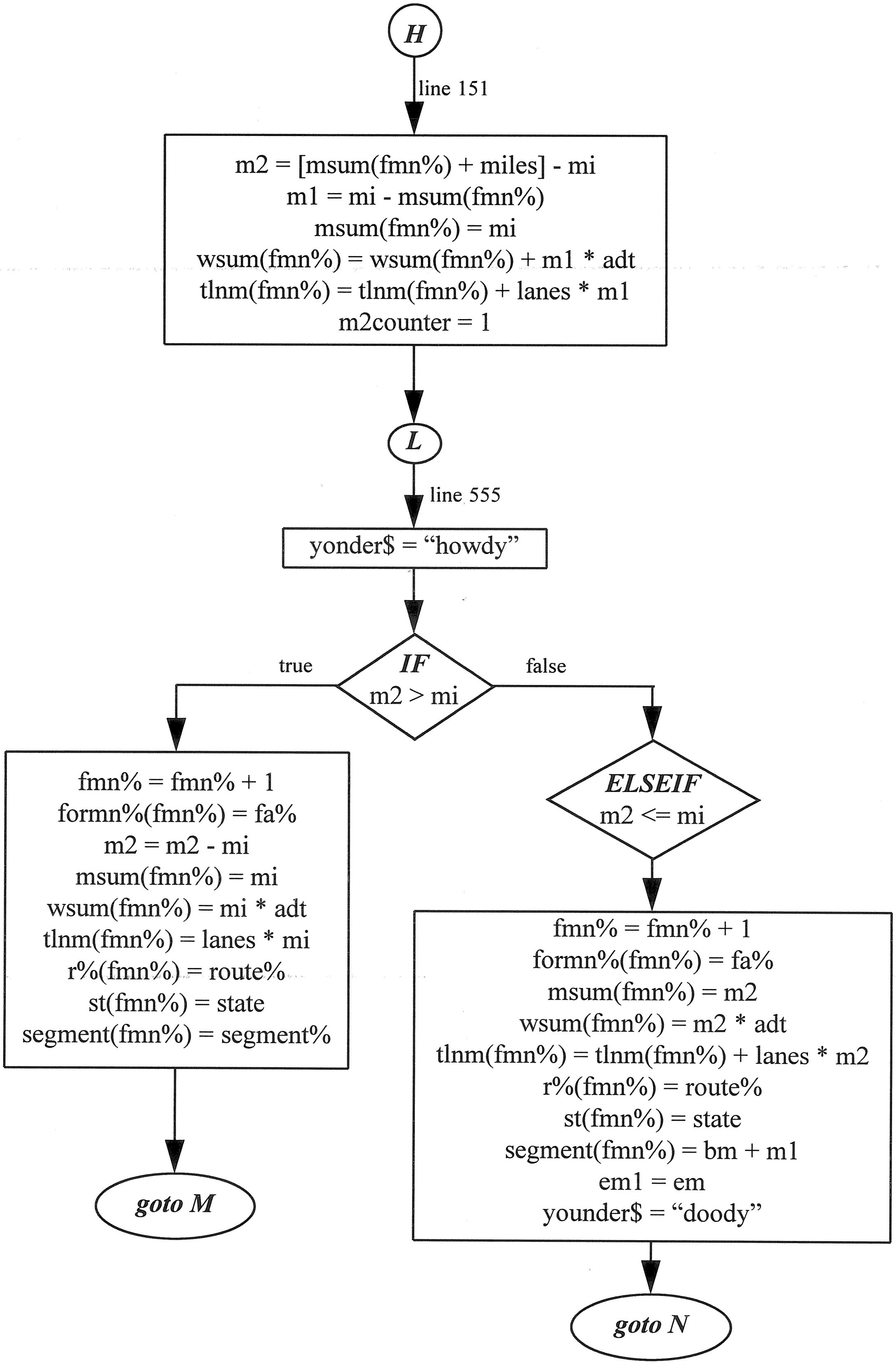
**page 3**


G
IF
[msum(fmn%) + miles]
> mi
true
goto H
false
msum(fmn%) = msum(fmn%) + miles
wsum(fmn%) = wsum(fmn%) + (miles) * adt
tlnm(fmn%) = tlnm(fmn%) + lanes * miles
st(fmn%) = state
em1= em
I
line 119
IF
EOF( )
true
goto D
false
INPUT route%, segment%, bm, em,
adt, temp2, fa%, ll%, state, lanes
goto J

**Subroutine**
**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%,
mi, district%, FILES$, tlnm() )
**page 4**



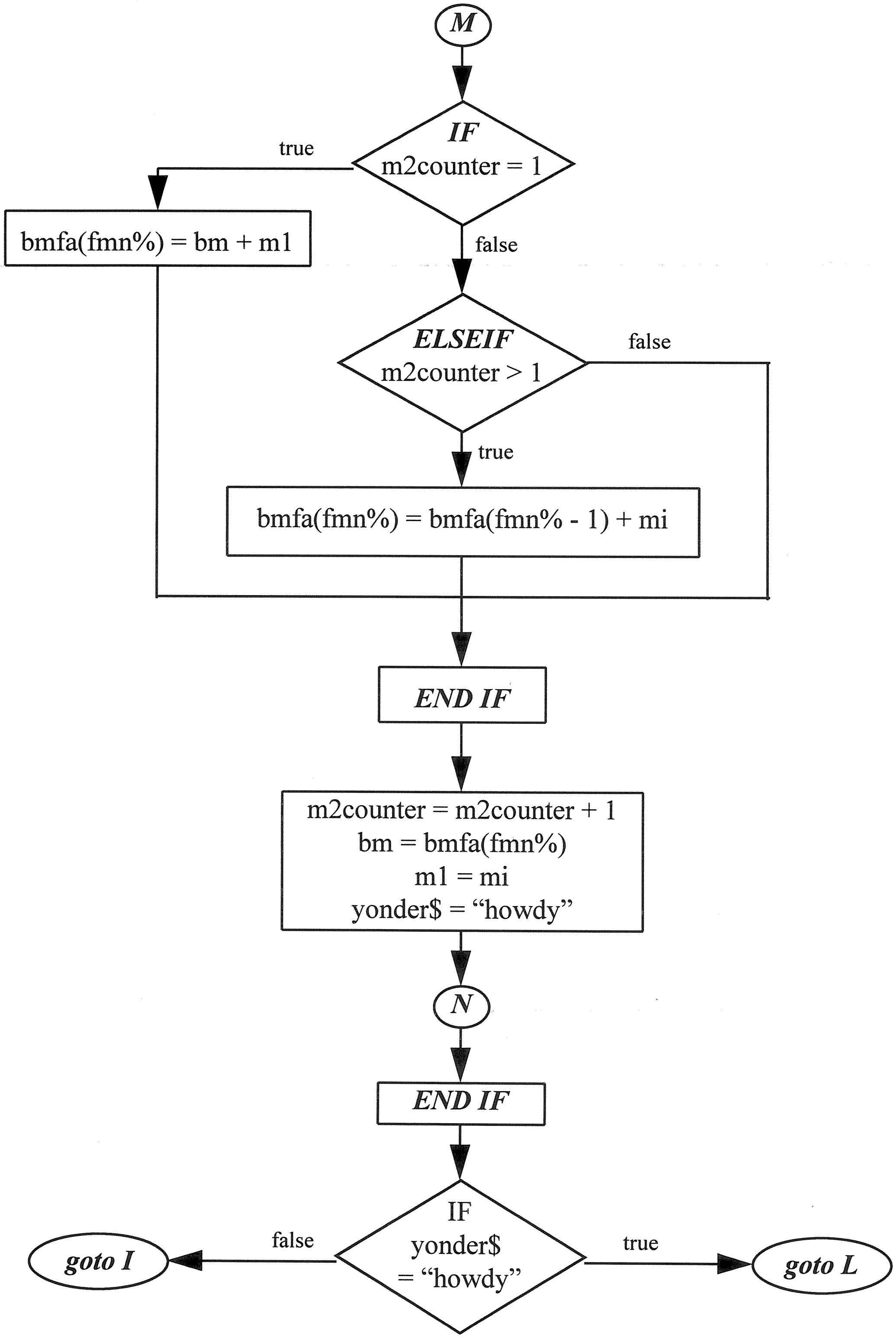
**Subroutine**
**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, district%, FILES$, tlnm() )
**page 5**



**Subroutine**

**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, district%, FILES$, tlnm() )

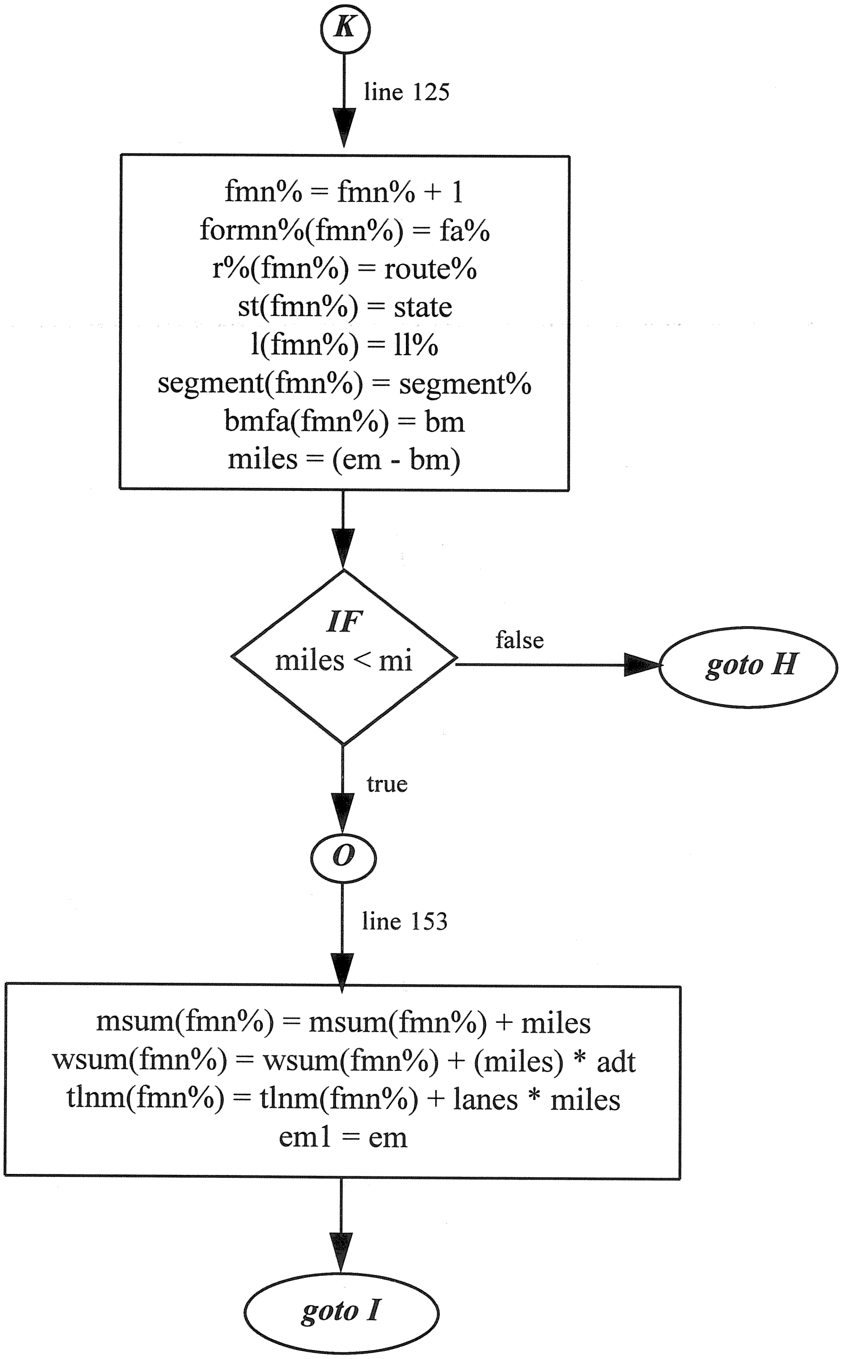
**page 6**



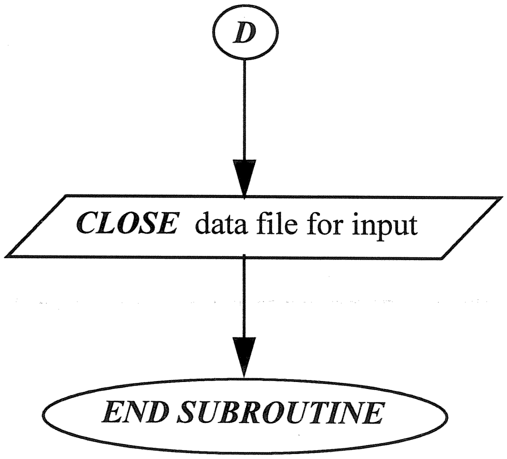
**Subroutine**

**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%, mi, district%, FILES$, tlnm() **)**

**page 7**



**Subroutine**
**wadt2(**segment(), fmn%, formn%(), msum(), wsum(), r%(), bmfa(), l(), st(), lx%,
mi, district%, FILES$, tlnm() **)**
**page 8**



**<u>HIGH.BAS</u>** variable list

**main program**

ACOST a cost determined from the AVGCOST subroutine—may either be a proposed cost or a current cost

adt average daily traffic volume—as defined in the program, it equals the weighted sum of mileage divided by the sum of mileage

an$ string variable used to check if the program stored the correct path for the output file

bcratio the benefit cost ratio for a roadway

bmfa() beginning milepost for the nth foreman area

car counter in a "FOR-LOOP" to run the proper number of simulations

ccost the costs due to decreased comfort resulting from a lower los

ccost1 the difference in comfort costs from the current to the proposed costs given one normally distributed random variable—the differences arise because of varying road conditions

ccost2 the difference in comfort costs from the current to the proposed costs given a different normally distributed random variable—the differences arise because of varying road conditions

chop$ string variable used to move text across the screen

curcost the estimated cost for a foreman area given the current lane-miles and los

D$ identifies if the analysis is for a district or the entire state

DAYS the number of days a foreman are will generally spend on winter maintenance—annual storm hours divided by 8

dcost the costs due to an increased delay resulting from a lower los

dcost1 the difference in delay costs from the current to the proposed costs given one normally distributed random variable—the differences arise because of varying road conditions

**<u>HIGH.BAS</u>** variable list

**main program**
page 2

dcost2 the difference in comfort costs from the current to the proposed costs given one normally distributed random variable—the differences arise because of varying road conditions

district% variable inputted by user for the district that is to be analyzed

FILE$ path for the output file

filename$ name of the output file for simulated data

FILES$ location of the data files used for input

filname$ final name of the output file for simulated data

fmfa%() ending mile marker for the nth foreman area

fmn% counter used within arrays to keep track of specific road segments and their los, traffic flow, costs, et cetera

formn%() array used to store the foreman areas of the nth observation

ii$ string variable used as a flag to either run the program again or exit entirely

inflat defined as the average hourly wage divided by 5.26

jb variable dependent on the “state” variable found in “newuofi.dat” that specifies if a road is a state highway. Important when distributing speeds along roadways for cost determination

jbc this variable identifies a specific value in two arrays that yield the mean velocity and standard deviation for highway speed given the current los.

jbp this variable identifies a specific value in two arrays that yield the mean velocity and standard deviation for highway speed given the proposed los

l() level of service for the $n^{th}$ foreman area

lnew the new los for a road segment

**HIGH.BAS** variable list

**main program**
page 3

lnew1 used to switch the proposed and current los if the proposed los is lower (i.e. changing a road segment from 3 to 4) to obtain a valid B/C ratio

lold the previous los for a road segment

lold1 used to switch the proposed and current los if the old los is higher (i.e. changing a road segment from 3 to 4) to obtain a valid B/C ratio

lx% the old los that is under consideration for change

mark1% identifies which observation in "nsteady.dat" contains relevant data for changes to be made in SCR5

mi variable inputted by user specifying the maximum segment length for change

msum() array that stores the cumulative mileage sum for the nth foreman area

name$ default name for the output data file

nash annual storm hours—identical to S(ii%).ash

ncurves variable relating the number of curves on a road—identical to S(ii%).curves

nelev variable relating the elevation of a road segment—identical to S(ii%).elev

netc() net costs for the nth foreman area under consideration

netcost temporary variable that takes the difference between the proposed cost and current cost for a particular foreman area

nls weighted los after proposed changes are made to a road segment

NoSimul number of simulations the computer will undertake—set at 500 by the programmers

nsf snow factor—identical to S(ii%).sf

**HIGH.BAS** variable list

**main program**
page 4

| | |
|---|---|
| ntlm | total lane miles after proposed changes are made to a road segment |
| nwf | wind factor—identical to S(ii%).wf |
| o$ | flag tripped if the proposed los is lower than the current los (i.e. changing a road segment from 3 to 4) |
| ols | weighted los before proposed changes are made to a road segment |
| otlm | total lane miles before proposed changes are made to a road segment |
| pcost | proposed cost given a changed los |
| pi | the geometric constant π |
| pntx | gives x coordinate when making text move during the program |
| pnty | gives y coordinate when making text move during the program |
| r%() | route corresponding to the nth foreman area |
| r1 | an evenly distributed random variable between 0 and 1 |
| r2 | an evenly distributed random variable between 0 and 1 |
| rf() | array used to store reduction factor for mean snow speed |
| S(ii%).ash | average storm hours for a year—from "nsteady.dat" |
| S(ii%).fa% | foreman area corresponding to the (ii%)th observation in "nsteady.dat" |
| scost | sum of the delayed and comfort costs times the number of days |
| segment() | array that store a road's segment number (i.e. 1540) for the nth observation |
| sigd() | array used to store dry road standard deviation |
| sigw() | array used to store wet road standard deviation |

**<u>HIGH.BAS</u>** variable list

**main program**
page 5

| | |
|---|---|
| subd$ | drive and subdirectory for the output file |
| st() | array that stores the “state” value from “newuofi.dat” |
| t() | one of many transient variables |
| TBEN | sum of all benefits for a particular foreman area |
| time1 | the difference in travel times between the current and proposed los |
| time2 | the difference in travel times between the current and proposed los |
| tlm | total lane miles in a road segment being analyzed—miles of road times number of lanes |
| tlnm() | array that stores the total lane miles for the nth observation (or nth foreman area) |
| TOT | total time saved with the proposed changes |
| TOTC | total costs incurred during the change |
| TOTIME() | total time saved for the nth observation (or nth foreman area) |
| trip | equal to the sum of mileage for the nth foreman area |
| TSUM | total miles changed |
| ttime | averaged total time saved given the 500 simulations |
| v1c | randomly generated velocity with the current los |
| v1p | randomly generated velocity with the proposed los |
| v2c | a different randomly generated velocity with the current los |
| v2p | a different randomly generated velocity with the proposed los |
| va() | array used to store mean dry road speed |

**HIGH.BAS** variable list

**main program**
page 6

veh identical to "car" but used to avoid contaminating a counting mechanism

wntadt weighted average of daily winter traffic

work identical to "work1" but used to avoid contamination of inputted values

work1 percentage of commuters on the roadway

wsum() array that stores the cumulative weighted sum of ADTs for the nth foreman area

yben() equals "DAYS" times "scost"—total benefits of increasing the los on a road segment

yn$ string variable used as a flag to print and/or display the program's output

yr$ string variable used to compute inflation for the given years

z1 a randomly generated z statistic value

z2 a randomly generated z statistic value

**<u>HIGH.BAS</u>** variable list

subroutine **AVGCOST**

| | |
|---|---|
| ACOST | temporary variable that stores the estimated cost of maintenance given the variables below |
| nash | annual storm hours (found in “nsteady.dat”) |
| ncurves | variable relating the curves along a road segment |
| nelev | variable relating the elevation of a particular segment of road. (found in “nsteady.dat”) |
| nls | a weighted level of service (los) |
| nsf | a “snow factor” for a road segment |
| ntlm | total lane miles in a district |
| nwf | a “wind factor” for a road segment |

**HIGH.BAS** variable list

subroutine **indata**

| | |
|---|---|
| rf() | array used to store reduction factor for mean snow speed |
| sigd() | array used to store dry road standard deviation |
| sigw() | array used to store wet road standard deviation |
| va() | array used to store mean dry road speed |

**<u>HIGH.BAS</u>** variable list

subroutine **inflation**

inflat average hourly wage divided by 5.26

jb$ string variable used to change the average wage estimated by wage for the years yr$

wage average wage for the years yr$--mathematical average of wage1 and wage2

wage1 variable used to compute the wage during year yr1

wage2 variable used to compute the wage during year yr2

yr$ string variable used to compute inflation for the given years

yr1 variable consisting of the third and fourth numbers in yr$

yr2 variable consisting of the sixth and seventh numbers in yr$

**<u>HIGH.BAS</u>** variable list

subroutine **info**

| | |
|---|---|
| file1$ | path for the data file “nsteady.dat” |
| file2$ | path for the data file “Trans.dat” |
| FILES$ | subdirectory location for the data files “nsteady.dat” and “Trans.dat” |
| S(ii%).ash | average storm hours for a year—from “nsteady.dat” |
| S(ii%).curves | figure relating the number of curves on a road—from “nsteady.dat” |
| S(ii%).elev | figure relating the elevation of the segment being considered—from “nsteady.dat” |
| S(ii%).fa% | foreman area corresponding to the (ii%)th observation in “nsteady.dat” |
| S(ii%).ls | weighted los for the foreman area in previous years |
| S(ii%).ls1tlm | total lane miles with an los of 1 in a particular foreman area |
| S(ii%).ls2tlm | total lane miles with an los of 2 in a particular foreman area |
| S(ii%).ls3tlm | total lane miles with an los of 3 in a particular foreman area |
| S(ii%).ls4tlm | total lane miles with an los of 4 in a particular foreman area |
| S(ii%).ls5tlm | total lane miles with an los of 5 in a particular foreman area |
| S(ii%).ls9596 | weighted los for a particular foreman area in the winter of 1995-96 |
| S(ii%).obs | the (ii%)th observation in “nsteady.dat” |
| S(ii%).sf | snow factor for foreman area ii% |
| S(ii%).tlm | total lane miles in foreman area (ii%) in previous years |
| S(ii%).tlm9596 | total lane miles in a foreman area during the years 1995-96 |
| S(ii%).wf | wind factor for a foreman area |
| t(jj%).deltsh | explains transient variations in total storm hours |

**<u>HIGH.BAS</u>** variable list

subroutine **info**
page 2

t(jj%).dist identifies which district the row of data represents

t(jj%).obst the (jj%)th observation in “Trans.dat”

t(jj%).sii statewide inflation index—relates varying values of commodities

t(jj%).tsh total storm hours for peak storms

t(jj%).yash yearly average storm hours

t(jj%).yr12 gives the winter relating to a particular row of data. For example, the winter of 1988-89 is “8889” in “Trans.dat”

**HIGH.BAS** variable list

subroutine **SCR5**

formn%(j%) a specific foreman area

l() level of service for the nth foreman area

lnew the new los for a road segment

lold the previous los for a road segment

LS1 total lane miles in the given district with los 1

LS2 total lane miles in the given district with los 2

LS3 total lane miles in the given district with los 3

LS4 total lane miles in the given district with los 4

LS5 total lane miles in the given district with los 5

mi maximum segment length for change on a road

nash annual storm hours—identical to S(ii%).ash

ncurves variable relating the number of curves on a road—identical to S(ii%).curves

nelev variable relating the elevation of a road segment—identical to S(ii%).elev

nls weighted los after proposed changes are made to a road segment

nsf snow factor—identical to S(ii%).sf

ntlm total lane miles after proposed changes are made to a road segment

nwf wind factor—identical to S(ii%).wf

ols weighted los before proposed changes are made to a road segment

otlm total lane miles before proposed changes are made to a road segment

**HIGH.BAS** variable list

subroutine **SCR5**
page 2

| | |
|---|---|
| r%(j%) | route corresponding to the (j%)th foreman area |
| S(ii%).ash | annual storm hours |
| S(ii%).elev | variable relating the elevation of a road segment |
| S(ii%).curves | variable relating the number of curves on a road |
| S(ii%).sf | snow factor for a road segment |
| S(ii%).wf | wind factor for a road segment |
| S(ii%).ls1tlm | total lane miles of los 1 in the (ii%)th foreman area—given in "nsteady.dat" |
| S(ii%).ls2tlm | total lane miles of los 2 in the (ii%)th foreman area—given in "nsteady.dat" |
| S(ii%).ls3tlm | total lane miles of los 3 in the (ii%)th foreman area—given in "nsteady.dat" |
| S(ii%).ls4tlm | total lane miles of los 4 in the (ii%)th foreman area—given in "nsteady.dat" |
| S(ii%).ls5tlm | total lane miles of los 5 in the (ii%)th foreman area—given in "nsteady.dat" |
| tlm | total lane miles in the road segment being analyzed |

**HIGH.BAS** variable list

subroutine **scrn3**

ans$ variable used to check if a designated path is the correct one

D$ identifies if the analysis is for a district or the entire state

district% the district that is to be analyzed—input at run time

files1$ temporary string variable used to store the location of the data file

FILES$ string variable used to store the final location of the data files

FFILES$ string variable used to check if the data file location is correct

intended$ string variable to check if the intended maximum segment length for change stored is what the user intended

lnew the new or proposed los

lx% the old los that is under consideration for change

mi maximum segment length for change

trigger$ string variable used for data file location checks

work1 percentage of commuters on the roadway

**<u>HIGH.BAS</u>** variable list

subroutine **wadt1**

| | |
|---|---|
| adt | average daily traffic volume |
| bm | temporary variable that stores the current road segment's beginning milepost |
| bmfa() | beginning milepost for the nth foreman area |
| em | temporary variable that stores the current road segment's ending milepost |
| em1 | temporary variable that stores the current road segment's ending milepost. *em1* is used to check if two road segments are continuous—i.e. the ending milepost of the previous segment is the beginning milepost of the current segment |
| FILES$ | string variable used to store the final location of the data files |
| fa% | temporary variable for the foreman area of inputted information |
| fmn% | counter used within arrays to keep track of specific road segments and their los, traffic flow, costs, et cetera |
| formn%() | array used to store the foreman areas of the nth observation |
| l() | array used to store the los of the nth observation |
| lanes | number of lanes on a road segment—inputted from "newuofi.dat" |
| ll% | temporary variable for the los of a specific road segment—inputted from "newuofi.dat" |
| lx% | the old los that is under consideration for change |
| m1<br>of | temporary variable that assesses how many miles a road segment is short the maximum segment length for change |
| m2 | temporary variable that assesses how many extra miles a road segment has compared to the maximum length for segment change |
| m2counter | variable used to cycle through road segments that are longer than the maximum segment length for change |

**HIGH.BAS** variable list

subroutine **wadt1**
page 2

mi	variable inputted by user specifying the maximum segment length for change

miles	the mileage of a road segment (or multiple road segments if continuous)

msum()	array that stores the cumulative mileage sum for the nth foreman area

r%()	array that stores the route for the nth observation

route%	temporary variable for the route of a specific road segment—inputted from “newuofi.dat”

segment()	array that store a road’s segment number (i.e. 1540) for the nth observation

segment%	temporary variable for a road’s segment number (i.e. 1540)—inputted from “newuofi.dat”

st()	array that stores the “state” value from “newuofi.dat”

state	temporary variable that stores the “state” value for a particular road segment—inputted from “newuofi.dat”

temp2	temporary variable indicating the temperature for a particular road segment

tlnm()	array that stores the total lane miles for the nth observation

wsum()	array that stores the cumulative weighted sum of ADTs for the nth foreman area

yonder$	string variable used to cycle through road segments that are longer than the maximum segment length for change—serves as a flag to (not)continue the breakdown of a long road segment

**HIGH.BAS** variable list

subroutine **wadt2**

| | |
|---|---|
| adt | average daily traffic volume |
| bm | temporary variable that stores the current road segment's beginning milepost |
| bmfa() | beginning milepost for the $n^{th}$ foreman area |
| district% | variable inputted by user for the district that is to be analyzed |
| em | temporary variable that stores the current road segment's ending milepost |
| em1 | temporary variable that stores the current road segment's ending milepost. *em1* is used to check if two road segments are continuous—i.e. the ending milepost of the previous segment is the beginning milepost of the current segment |
| FILES$ | location of the data files used for input |
| fa% | temporary variable for the foreman area of inputted information |
| fmn% | counter used within arrays to keep track of specific road segments and their los, traffic flow, costs, et cetera |
| formn%() | array used to store the foreman areas of the nth observation |
| l() | array used to store the los of the nth observation |
| lanes | number of lanes on a road segment—inputted from "newuofi.dat" |
| ll% | temporary variable for the los of a specific road segment—inputted from "newuofi.dat" |
| lx% | the old los that is under consideration for change |
| m1 | temporary variable that assesses how many miles a road segment is short of the maximum segment length for change |
| m2 | temporary variable that assesses how many extra miles a road segment has compared to the maximum length for segment change |

**HIGH.BAS** variable list

subroutine **wadt2**
page 2

m2counter variable used to cycle through road segments that are longer than the maximum segment length for change

mi variable inputted by user specifying the maximum segment length for change

miles the mileage of a road segment (or multiple road segments if continuous)

msum() array that stores the cumulative mileage sum for the nth foreman area

r%() array that stores the route for the nth observation

route% temporary variable for the route of a specific road segment—inputted from "newuofi.dat"

segment() array that store a road's segment number (i.e. 1540) for the nth observation

segment% temporary variable for a road's segment number (i.e. 1540)—inputted from "newuofi.dat"

st() array that stores the "state" value from "newuofi.dat"

state temporary variable that stores the "state" value for a particular road segment—inputted from "newuofi.dat"

temp2 temporary variable indicating the temperature for a particular road segment

tlnm() array that stores the total lane miles for the nth observation

wsum() array that stores the cumulative weighted sum of ADTs for the nth foreman area

yonder$ string variable used to cycle through road segments that are longer than the maximum segment length for change—serves as a flag to (not)continue the breakdown of a long road segment