COMPUTER-ASSISTED INSTRUCTION:

A SIMPLEX ALGORITHM LABORATORY

by

Donavon B. Lewis

A Thesis Submitted to the Faculty of the
COMMITTEE ON COMPUTER SCIENCE
In Partial Fulfillment of the Requirements
For the Degree of
MASTER OF SCIENCE
In the Graduate College
THE UNIVERSITY OF ARIZONA

1971
STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Donovan Skurni

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

ROBERT L. BAKER
Associate Professor of Systems Engineering
ACKNOWLEDGMENTS

The author wishes to express his gratitude to Dr. Robert L. Baker for his guidance, assistance and encouragement throughout this work. He also wishes to thank Dr. William R. Ferrel and Dr. James F. LaSalle for their comments and suggestions. Mrs. Martha L. Dawson deserves recognition for her skill in converting the manuscript into the present document. Last but not least, the author wants to express his deep appreciation to his wife Kuniko and his two children Kim and David, without whose tolerance and encouragement this work would not have been possible.

The author also wishes to thank the United States Air Force for permitting him to pursue his academic goals.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF ILLUSTRATIONS</td>
<td>v</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>vi</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>Scope of the Work</td>
<td>1</td>
</tr>
<tr>
<td>Computer-Assisted Instruction</td>
<td>2</td>
</tr>
<tr>
<td>Types of CAI</td>
<td>5</td>
</tr>
<tr>
<td>AFO Systems</td>
<td>5</td>
</tr>
<tr>
<td>ISO Systems</td>
<td>8</td>
</tr>
<tr>
<td>Responsive Environment Systems</td>
<td>10</td>
</tr>
<tr>
<td>2. LINEAR PROGRAMMING AND THE SIMPLEX ALGORITHM</td>
<td>13</td>
</tr>
<tr>
<td>Linear Programming Review</td>
<td>13</td>
</tr>
<tr>
<td>Simplex Algorithm Review</td>
<td>15</td>
</tr>
<tr>
<td>3. SIMPLEX ALGORITHM LABORATORY</td>
<td>20</td>
</tr>
<tr>
<td>4. CONCLUSIONS AND RECOMMENDATIONS</td>
<td>27</td>
</tr>
<tr>
<td>APPENDIX A: USER’S MANUAL</td>
<td>30</td>
</tr>
<tr>
<td>APPENDIX B: SAMPLE DIALOGUE</td>
<td>74</td>
</tr>
<tr>
<td>APPENDIX C: CONTROL PROGRAM LISTING</td>
<td>82</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>106</td>
</tr>
</tbody>
</table>
LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Optimal Solution</td>
<td>56</td>
</tr>
<tr>
<td>2.</td>
<td>Alternative Optimal Solution</td>
<td>58</td>
</tr>
<tr>
<td>3.</td>
<td>Unbounded Optimal Solution</td>
<td>60</td>
</tr>
<tr>
<td>4.</td>
<td>No Optimal Solution</td>
<td>60</td>
</tr>
</tbody>
</table>
ABSTRACT

The thesis presents the development of a Computer-Assisted Instruction (CAI) program called the Simplex Algorithm Laboratory (SAL). SAL, using a time-sharing computer, creates a responsive environment that allows the student to exercise initiative in experimenting with the simplex algorithm as it is applied to linear programming problems. The student is assumed to have a basic knowledge of the algorithm. Communication between the student and SAL is with a remote teletypewriter; the student needs no experience with computer languages. The student enters linear programming problems and "experiments" with the effects of the simplex algorithm. SAL automatically solves the problems and provides sensitivity analysis. The student can exercise his initiative by selecting many different problems and may obtain a better "feel" for the algorithm by observing immediate solutions to these problems.

The thesis contains a review of current trends in CAI, a brief introduction to linear programming and the simplex algorithm, and a description of SAL. A User's Manual, in sufficient detail to allow use of SAL, is included along with a listing of the FORTRAN IV control program. Several pages of sample dialogue illustrate the interaction between the student and SAL.
CHAPTER 1

INTRODUCTION

This thesis reports the development of a Computer-Assisted Instruction (CAI) program called the Simplex Algorithm Laboratory (SAL). SAL is designed to provide a student-computer interactive responsive environment to enable the student to experiment with the applications of the simplex algorithm to linear programming problems. The student can arbitrarily select a linear programming problem and ask SAL to provide the solution and the sensitivity analysis. The student has the option of observing each iteration of the algorithm or only the final results and can freely vary the problem parameters to study the effects of the changes on the solution. SAL is programmed in FORTRAN IV and has been implemented on the Control Data Corporation 6400 computer using the CDC INTERCOM 2 time-sharing system at The University of Arizona, Tucson.

Scope of the Work

This thesis contains four chapters: Introduction; Linear Programming and the Simplex Algorithm; Simplex Algorithm Laboratory; and Conclusions and Recommendations. There are three appendices: User's Manual; Sample Dialogue of Simplex Algorithm
Laboratory; and Control Program Listing. The remainder of Chapter 1 includes a brief review of programmed instruction and its relationship to CAI, a definition of CAI and some examples of the latest trends in development of CAI systems. Chapter 2 is a brief introductory discussion of linear programming and the simplex algorithm to provide the reader with the necessary background information needed for Chapter 3. Chapter 3 explains what SAL is and how it is used by the student. Detailed instructions for the actual use and control of SAL are provided in the User's Manual (see Appendix A, Chapters III and IV). A summary of the thesis and recommendations for extensions of SAL are contained in Chapter 4.

The User's Manual in Appendix A is a self-contained document providing all the information needed by the student to obtain and use SAL. Chapter I of the User's Manual contains an outline of the material contained in the manual.

Appendix B is a copy of a portion of an actual dialogue between the computer and a student using SAL. The responses made by the student have been underlined. Appendix C contains the complete FORTRAN IV listing of the SAL control program.

**Computer-Assisted Instruction**

Computer-assisted instruction (CAI) began as a natural extension of programmed instruction when the computer was used as the
vehicle for presenting the instructional material. It is appropriate, therefore, that a discussion of CAI begin with an introduction to programmed instruction.

Silvern and Silvern (1966, p. 1649) reported that programmed instruction is a learner-centered method of instruction without the presence or intervention of a human instructor. The subject matter is usually presented to the student in small increments and requires frequent responses which may be written, oral, or manipulative. The responses may be constructed, may be of the multiple-choice type, or may be of a variety of other styles and the student is immediately informed of the correctness of his response. Programmed instruction evolved between about 1920 and 1955 and gained momentum in the late 1950's and early 1960's.

Certain criteria have developed which are now generally accepted features of programmed instruction (Silvern and Silvern, 1966, p. 1649).

1. Instruction is provided without the presence or intervention of a human instructor.

2. Learner learns at his own rate (conventional group instruction, films, television, and other media and methods which do not allow learner control of rate do not satisfy this criterion).
3. Instruction is presented in small incremental steps requiring frequent responses by the learner; step size is a function of the subject-matter and the characteristics of the learner population.

4. There is a participative overt interaction, or two-way communication, between learner and instructional program.

5. Learner receives immediate feedback informing him of his progress.

6. Reinforcement is used to strengthen learning.

7. Sequence of lessons is carefully controlled and consistent.

8. Instructional program shapes and controls learner behavior.

Programmed texts or teaching machines are the usual methods of presenting the instructional material. When a computer is used for the presentation, the process may be termed computer-assisted instruction. However, this definition is too restrictive and does not reflect the current status of CAI.

A more appropriate definition of CAI is any instructional system having a two-way communication between a human learner and a computer in which there is a stimulus-response-feedback relationship producing learning. This expanded definition of CAI is used in this thesis. Communication between the learner and the computer is through a remote student console which is usually a teletypewriter or a cathode-ray tube (CRT). The student sits at the teletypewriter and performs two functions: he reads what is printed on the printout or
CRT, and he types responses on his keyboard for transmission to the computer.

Types of CAI

Computer-assisted instruction has, until the past few years, been of the general type labeled by Carbonell (1970b) as "ad hoc frame-oriented" (AFO) CAI systems. AFO systems are organized around blocks of material usually called frames. The frames are entered in advance by the teacher and include paragraphs of English text and questions to be answered. Alternative branching routines, based on anticipated student response, are used to select from among the frames. The questions are usually of the multiple-choice type. The student can exercise little or no initiative. An AFO system of any depth or complexity requires a comparatively large amount of effort on the part of the teacher in the preparation of the text material and the branching routines since every alternative must be programmed.

AFO Systems

The best known of the AFO systems are ELIZA (Taylor 1967) and PLANIT (Feingold 1967). ELIZA was developed by Professor Joseph Weizenbaum of the MIT Department of Electrical Engineering. ELIZA provides a means for a two-way conversation in natural English (including numbers) between a student and a time-shared computer. The student may select a topic from several available topics, called
scripts, and ask the computer questions about that topic. The computer scans the student's question for keywords that it uses as a basis for selecting the response to the question. The selection of the response is determined by the success (or failure) of the keywords to match predetermined patterns. Alternative patterns are tested until a match is obtained or until the pattern alternatives are exhausted. The computer program is based on a symmetric list processing language called SLIP embedded in the machine language MAD.

**ELIZA** is different from most AFO systems because of its ability to accept natural English statements or questions from the student. It does, however, have the AFO characteristic features of 1) requiring detailed and explicit programming and entering of the text material and branching alternatives and 2) restricting the student's initiative by providing only a limited number of preselected logical paths through the text material. Taylor (1967) estimated that a programmer requires about one hour to produce that part of the script that will engage the student's attention for one minute.

**PLANIT**, designed at System Development Corporation, comprises both a language and a program developed for use with time-shared computers. The system has four modes of operation: Lesson building, editing, execution, and calculation. The teacher can use all four modes when preparing the lesson. The student has access only to the execute mode and the calculate mode. The distinguishing features of **PLANIT**
are: 1) the reduction of some of the burden of lesson preparation by the teacher through characteristics of the language and 2) the calculation mode wherein the student may obtain immediate numerical answers to problems he may be required to solve. PLANIT was originally used for teaching statistics and has been extended to such areas as spelling, vocabulary, and computer programming.

The magnitude of the demands on the teacher, the lack of initiative available to the student, and the lack of significant contribution by the computer have stimulated interest in other types of computer-assisted instruction. Several recent contributions to CAI have two things in common that distinguish them from the conventional AFO systems. The new approaches more effectively use the computational and logical capability of the computer and they provide the student with more freedom to exercise initiative. Although the recent contributions are similar in these traits, they have different operation characteristics that allow them to be separated into two categories.

One category has been defined by Carbonell (1970b) as an information-structure-oriented (ISO) system. Wexler (1970b) presents a system that also can be included in this category. Both of these systems of the ISO type will be discussed in the following paragraphs. The other category will be referred to in this thesis as systems that provide a "responsive environment" for the student. Rosenberg (1965) and Feurzeig, Wexelblat and Rosenberg (1970) have described systems
of the latter type and these two systems will be discussed after the
ISO systems have been covered. The main subject of this thesis, the
Simplex Algorithm Laboratory, falls in the responsive environment
category and will be discussed in Chapter 3 of the thesis and in more
detail in Chapters III and IV of the User's Manual, Appendix A.

ISO Systems

The SCHOLAR system (Carbonell 1970a, 1970b) is a set of
computer programs of the ISO type. SCHOLAR is based on a complex
but well-defined information structure in the form of a network of
facts, concepts, and procedures. The elements of the network are
units of information defining words and events in the form of multilevel
tree lists. Elements of each list act as pointers to interconnected
related elements and lists and create, in a sense, a semantic network.
The subject material must have an inherent structural pattern compat-
ible with the multilevel tree concept.

The SCHOLAR system has an executive program that probes the
semantic network in order to generate the material to be presented, the
questions to be asked, the answers to its own questions, and conditional
branching to program selected (not predetermined) sequences based on
the responses of the student. The executive program is almost com-
pletely independent of the subject material being presented.
The data base of SCHOLAR reflects basic "knowledge" about the subject being treated and allows the system to exhibit certain traits characteristic of artificial intelligence such as answering questions not specifically anticipated and carrying on a mixed-initiative contextual dialogue with a human in a subset of English.

Wexler (1970a, 1970b) developed a program that also uses the structure of the subject material to generate statements, questions and remedial sequences. The student is allowed to exhibit initiative by asking questions.

The system developed by Wexler is designed to treat non-numeric problem areas such as geography, biology, and history. The underlying principle of this approach is the arrangement of the subject material into information nets. An information net is composed of a set of objects that are grouped into classes. A net object represents a basic piece of factual information that is stored as a value string or as a character string. Links are formed between objects in the net. The conventions for the significance of the links between objects is specified by the teacher. A trace, defined as a collection of net objects and a derivational path linking them, is formed when the information net is searched for a particular type of data. The traces are controlled by skeleton patterns that define dynamic search mechanisms.

The system has three modes of operation: teacher, student, and dialogue. The teacher mode is used by the teacher to define and
input the information net and the skeleton patterns. The student uses the student mode to receive instruction in the form of information, questions, and remedial assistance. The dialogue mode is used by the student to ask the system a question or to ask for information on a specific topic. This system is similar to SCHOLAR but is more restrictive in answering questions.

**Responsive Environment Systems**

ENPORT (Rosenberg 1965, Rosenberg, Feurzeig, and Wexelblat 1970) is a digital computer program that provides a responsive environment used in CAI to simulate physical phenomena. Physical systems, including mechanical, electrical, hydraulic, acoustic, and mixed kinds are modeled by the student using bond-graphs. A bond-graph is a graphical notation for representing physical and engineering components and their interconnections. The student enters the bond-graph description and the defining parameters of the system. ENPORT accepts the input from the student and simulates the system. The results of the simulation are displayed on an oscilloscope or lineprinter.

The student is free to choose both the system he wishes to have simulated and the parameters defining that system. Thus, he can investigate the behavior of a variety of systems under a variety of conditions.
An inherent problem with a program like ENPORT is that of assisting the student in his use of the program. To assist the student, the program must place more restrictions on the interaction which thereby reduces the potential for student initiative. A program has been developed to monitor the interactions of a student with ENPORT and to assist the student when requested. The monitor program is limited to assisting the student in the construction of bond-graphs related to relatively small and simple systems. The problem of monitoring larger and more complex systems is still not solved.

A Simple Instructional Monitor (SIMON) developed by Feurzeig in 1969 and explained by Feurzeig, Wexelblat, and Rosenberg (1970) is a program that tries to detect, diagnose, and overcome a student's difficulty in solving a problem in a responsive environment. SIMON uses a general purpose programming language called Telcomp (Myer 1967) for the interaction between the student and the monitor program. The student is presented with a description of a problem and a list of relevant variables. The problem has been previously entered into SIMON by the teacher who also supplies a "good" program solution. The problem must be expressable in mathematical terms and may simulate the operations of biological, physical, economic and other processes. The student tries to develop a Telcomp program to solve the problem by relating and combining the variables as he models the process. The student has no access to the "good" program except
that he can input variables of his choosing and observe the effects on the good program. He can also test his own program by inputing selected variables. When he is satisfied with his program, the student can ask SIMON to check it. SIMON will compare the students program against the good program by running test data through both and comparing the results. When asked, SIMON will help the student identify errors in his program.

As mentioned earlier, the discussion of the Simplex Algorithm Laboratory, also a responsive environment system, will begin in Chapter 3. Chapter 2 will deal next with background information on linear programming and the simplex algorithm.
CHAPTER 2

LINEAR PROGRAMMING AND THE
SIMPLEX ALGORITHM

This chapter is designed to provide a quick review of linear programming and the simplex algorithm and to familiarize the reader with the notation and terminology used in this thesis in the presentation of the Simplex Algorithm Laboratory (SAL). It is assumed the reader has a basic knowledge of linear programming and the simplex algorithm. The reader who has no prior knowledge in this area should skip this chapter and refer to Chapter V of the User's Manual (Appendix A) which will provide the necessary background information.

A brief summary of linear programming and the simplex algorithm will be given with comments on the terminology and notation that is used in this thesis and SAL.

**Linear Programming Review**

A linear programming problem is a mathematical model using linear relationships to deal with the efficient use or allocation of limited resources to meet specified objectives. The linear relation describing the objective is called the "objective function." The linear equations or inequalities specifying the limitations and conditions on
the problem are called the "constraints." A solution that both satisfies the constraints and optimizes the objective function is called the "optimal solution."

The goal may be to maximize or to minimize an objective function. This thesis will deal only with maximization problems. No generality is lost because the minimum of an objective function is equal to minus the maximum of the negative of the objective function.

A general canonical statement of the linear programming problem is

\[
\text{maximize } \sum_{j=1}^{n} c_j x_j
\]

subject to

\[
\sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, \ldots, m
\]

\[
x_j \geq 0 \quad \text{for } j = 1, 2, \ldots, n
\]

Constraint relationships may also be expressed by equalities or inequalities of opposite direction from the canonical form given. Any constraint relationship can be reduced to the canonical form but the conversion is not necessary when using SAL since SAL will accept problems with constraints expressed by \( \leq, \geq, \) or \( = \).

A linear programming problem may have one of four possible solutions: 1) an optimal solution, 2) an alternative optimal solution, 3) an unbounded optimal solution, and 4) no feasible solution. An optimal solution exists when the constraints are satisfied and the
objective function is maximized. An alternative optimal solution exists when the objective function is maximized at a particular value for two or more different solutions satisfying the constraints. When an alternative solution occurs, there are an infinite number of possible solutions available by taking weighted-averages of the appropriate extreme point solutions. An unbounded solution exists when some variable in the objective function may be increased without limit and still satisfy the constraints. No feasible solution occurs when the constraint relationships have no solution.

**Simplex Algorithm Review**

The simplex algorithm will be explained by using a specific linear programming problem as an example. The algorithm will then be expressed in general terms in summary form. An example of a problem is:

maximize \[ 4x_1 + 5x_2 + 9x_3 + 11x_4 \]

subject to \[ 1x_1 + 1x_2 + 1x_3 + 1x_4 \leq 15 \]
\[ 7x_1 + 5x_2 + 3x_3 + 2x_4 \leq 120 \]
\[ 3x_1 + 5x_2 + 10x_3 + 15x_4 \leq 100 \]
\[ x_1 \geq 0, \ x_2 \geq 0, \ x_3 \geq 0, \ x_4 \geq 0 \]

One convenient form of representing the above is to let \( x_0 = 0 \) be the value of the objective function and rewrite the above system by adding slack variables (surplus variables are appropriate if the direction of the inequality is reversed) to the inequality constraints.
\[ \begin{align*}
  x_0 - 4x_1 - 5x_2 - 9x_3 - 11x_4 &= 0 \\
  1x_1 + 1x_2 + 1x_3 + 1x_4 + 1x_5 &= 15 \\
  7x_1 + 5x_2 + 3x_3 + 2x_4 + 1x_6 &= 120 \\
  3x_1 + 5x_2 + 10x_3 + 15x_4 + 1x_7 &= 100
\end{align*} \]

The variables \( x_5, x_6, \) and \( x_7 \) are the slack variables and all variables \( x_1 \) through \( x_7 \) are \( \geq 0. \)

The four equations above in eight unknowns represent an underdetermined system of simultaneous equations. Setting four of the unknowns to zero will allow the remaining four to uniquely determine a solution to the system. The simplex algorithm is one computational technique which will select the appropriate variables to be at zero level as well as select the values for the other variables such that the solution will also maximize the objective function value \( x_0. \)

The simplex algorithm uses the system of equations just mentioned but a condensed format is more convenient for presenting the system. The condensed simplex tableau shown below is used in the SAL presentation.

<table>
<thead>
<tr>
<th>ITERATION NUMBER</th>
<th>BASIS</th>
<th>CURRENT VALUES</th>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( x_3 )</th>
<th>( x_4 )</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( x_0 )</td>
<td>0</td>
<td>-4</td>
<td>-5</td>
<td>-9</td>
<td>-11</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>( x_5 )</td>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>( x_6 )</td>
<td>120</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>( x_7 )</td>
<td>100</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>4</td>
</tr>
</tbody>
</table>
The variables that are not set to zero and that are in the current solution are called "basic variables" or "the basis" and are listed vertically under BASIS. The current values of the basic variables are shown immediately to their right under the heading CURRENT VALUES. The variables that are set to zero and that are not in the current solution are listed in the heading row and are called "nonbasic" variables. Row 1 refers to the objective function and Rows 2, 3, and 4 refer to the constraint equations.

Simplex Criterion I of the simplex algorithm states that if there are nonbasic variables having a negative coefficient in Row 1, select one such variable with the most negative coefficient. If all nonbasic variables have positive or zero coefficients in Row 1, an optimal solution has been reached. Criterion I specifies that $x_4$ should be entered into the basis.

Simplex Criterion II states that (a) For each row, take the ratios of the current right-hand side to the coefficients of the entering variable (ignore ratios with zero or negative numbers in the denominator). (b) Select the minimum ratio and exit the corresponding basic variable by setting it equal to zero. Criterion II specifies that $x_7$ should leave the basis (ratios are $\ldots$, 15, 60, and 6.66 for Rows 1, 2, 3, and 4, respectively). $x_4$ is entered into the basis and $x_7$ is removed and set equal to zero by a "change-of-basis" calculation or a "pivot operation" as follows:
Row 4: divide Row 4 by 15 to create a coefficient of 1 for $x_4$.
Row 1: multiply new Row 4 by 11 and add to Row 1,
Row 2: multiply new Row 4 by -1 and add to Row 2,
Row 3: multiply new Row 4 by -2 and add to Row 3.

The results of the pivot operation are:

<table>
<thead>
<tr>
<th>ITERATION NUMBER</th>
<th>BASIS</th>
<th>CURRENT VALUES</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_7$</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$x_0$</td>
<td>73.3</td>
<td>-1.80</td>
<td>-1.33</td>
<td>-1.67</td>
<td>.733</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$x_5$</td>
<td>8.33</td>
<td>.800</td>
<td>.667</td>
<td>.333</td>
<td>-.067</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>$x_6$</td>
<td>107</td>
<td>6.60</td>
<td>4.33</td>
<td>1.67</td>
<td>-.133</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>$x_4$</td>
<td>6.67</td>
<td>.200</td>
<td>.333</td>
<td>.667</td>
<td>.067</td>
<td>4</td>
</tr>
</tbody>
</table>

This process is continued until Criterion I specifies an optimal solution has been reached. The algorithm may be summarized in the following four steps:

Step 1. Select an initial basis.

Step 2. Apply Criterion I. If the current trial solution is not optimal, proceed to Step 3. Otherwise, stop.

Step 3. Apply Criterion II.

Step 4. Make a change-of-basis and return to Step 2.

Certain ambiguities may arise during application of the algorithm as it is stated above. An "alternative optimal solution" exists if there is a zero valued coefficient for any nonbasic variable in Row 1 of the final iteration. An "unbounded optimal solution" exists if there is no positive coefficient in any Row for the entering variable when applying
Criterion II. The selection of an initial basis may not be obvious if an all-slack basis can not be used as was done in the above example. One technique, the Big M method (Wagner 1969), uses artificial variables in the initial basis. The Big M method is used by SAL.

Sensitivity analysis, also called postoptimality analysis, uses the information available at the final iteration to determine the allowable variations, called ranging analysis, on certain coefficients and constants of the original problem before the current optimal solution is effected. SAL provides ranging analysis for variables in the objective function and constants of the constraint equations.

The next chapter contains a general description of the Simplex Algorithm Laboratory. The reader who is still uncertain of the material discussed in this chapter may want to refer to Chapter V of the User's Manual, Appendix A, before proceeding.
CHAPTER 3

SIMPLEX ALGORITHM LABORATORY

The Simplex Algorithm Laboratory was designed to allow the user to gain a more thorough understanding of the simplex algorithm through experimentation. The program was written in FORTRAN IV for the CDC 6400 computer at The University of Arizona. The system was implemented using a remote teletype console connected to the computer via the CDC INTERCOM 2 time-sharing system.

The User's Manual attached to this thesis (Appendix A) contains a complete description of the program and instructions on how to use it. The student sits at a remote terminal and gains access to the computer and to the Simplex Algorithm Laboratory (SAL) by following the procedure outlined in the User's Manual. The computer communicates with the student by printing English messages and text material on the student's terminal. The student responds by typing single-word or number replies on the keyboard and depressing the return key which then transmits the reply to the computer.

The SAL is divided into four Sections 1) Brief introduction, 2) Responsive environment, 3) Sensitivity analysis, and 4) Critique and evaluation; these will be discussed in order. Sections 2 and 3 are interrelated. This relationship will be explained in conjunction with
Section 3. Section 1 is a frame oriented presentation of the simplex algorithm. The treatment is in the conventional increments of text followed by questions asked of the student. Student response determines the subsequent frames as predetermined by the program. The text of Section 1 is not a complete instructional program and does not contain enough information to adequately present the major concepts of the simplex algorithm. The student totally unfamiliar with the simplex algorithm should first read Chapter V in the User's Manual, or any introductory text on the subject, before starting the laboratory.

Justification for using an abridged treatment in Section 1 rather than a complete text is based on the following:

1. A basic understanding of the simplex algorithm can be effectively obtained from a text book or lecture.

2. The abridged treatment is sufficient to refresh one's memory of the salient features of the algorithm.

3. Some introductory material is necessary to allow the student to learn the procedures for the control he must exercise in the following sections. Thus, the abridged text serves two purposes.

By the time the student has reached the end of Section 1, he is expected to have become familiar with the basic algorithm and also to have learned what he needs to know to effectively use the remainder of the laboratory.
Section 2 is a responsive environment in which the student can select any arbitrary linear programming problem and observe the results of the step-by-step solution by the computer as it applies the simplex algorithm. The student can select any simplex problem ranging in size up to 40 variables and 20 constraints.

There are two options available to the student for entering the problem. The regular method asks for each variable coefficient to be entered as it is called for by the program. A faster method allows the student to enter related groups of data without specifically being asked for each item. This method should be used only after the student has gained some experience using the program.

After the data has been entered, the program displays the data in a condensed table to allow the student to verify the accuracy of the problem. Correction may then be made to any of the data items. The following sample printout from SAL illustrates the appearance of the data.

THE PROBLEM YOU HAVE JUST FINISHED ENTERING
WILL BE CONDENSED INTO A TABLE FOR YOUR EDITING.
NOTE THAT ROW 1 IS THE OBJECTIVE FUNCTION.

******EDIT TABLE ******

```
   COL 1      COL 2      COL 3
ROW 1  -5.0000E+00  -6.0000E+00
ROW 2  2.0000E-01   3.0000E-01
ROW 3  2.0000E-01   1.0000E-01
ROW 4  3.0000E-01   3.0000E-01

CONSTRAINT EQ.  SENSE CODE  RT.HAND SIDE
   1   1   1.8000E+00
   2   1   1.2000E+00
   3   1   2.4000E+00
```
Once the student is satisfied that the problem has been correctly entered, he allows the program to proceed to the first iteration as the simplex algorithm is automatically applied to find the optimal solution to the problem. The first iteration is just a rearrangement of the initial values which are combined with slack and/or surplus variables and presented in a simplified tableau such as:

<table>
<thead>
<tr>
<th>IT NO</th>
<th>BASE</th>
<th>CURRENT VALUES</th>
<th>X 1</th>
<th>X 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X 0</td>
<td>0.</td>
<td>-5.0000E+00</td>
<td>-6.0000E+00</td>
</tr>
<tr>
<td>X41</td>
<td>1.8000E+00</td>
<td>2.0000E-01</td>
<td>3.0000E-01</td>
<td></td>
</tr>
<tr>
<td>X42</td>
<td>1.2000E+00</td>
<td>2.0000E-01</td>
<td>1.0000E-01</td>
<td></td>
</tr>
<tr>
<td>X43</td>
<td>2.4000E+00</td>
<td>3.0000E-01</td>
<td>3.0000E-01</td>
<td></td>
</tr>
</tbody>
</table>

The tableau of the first iteration shows the starting basis for the first feasible solution. If artificial variables are needed to obtain a first feasible solution, the necessary adjustments are made before the tableau is presented. In short, the tableau of the first iteration is the status of the variables before the simplex algorithm is applied.

The student has two options at this point. He may ask for the next iteration or he may ask for only the final iteration. The next iteration is a repeat of the tableau format but shows the results of applying the algorithm to the data in the previous tableau. Requesting only the final iteration will suppress printing of intermediate tableaus and print the final tableau followed with a statement of the type of solution and the optimum value, if applicable, of the objective function. After each
intermediate iteration, the student again has the option of selecting either the next or the final iteration.

The student may select several options after the final iteration has been reached. He may modify or change any number of the original parameters of the current problem and resolve the modified problem. He may modify the final tableau of the current problem and continue from that tableau with further iterations, if applicable. He may proceed to Section 3 for the sensitivity analysis of the current problem. He may enter a new problem and repeat the process. He may proceed to Section 4 for the critique and evaluation. Or, finally, he may stop and discontinue the analysis.

Section 3 is concerned with the sensitivity analysis of the current problem. To be eligible for analysis, the problem must have yielded an optimal solution. There are basically two types of analyses available.

Ranging analysis is available on variables in the objective function and on the constants of the constraint equations. The student may select an objective function variable or constraint constant and the program will give a range through which the item selected may be varied before it will effect the current solution. Ranging analysis uses data from the final tableau but does not destroy that data; therefore, any number of variables or constants may be investigated but only one at a time. Ranging on multiple variations is not allowed.
The other type of analysis is determining the effects of a specific discrete change in a parameter or set of parameters. With this option, called a discrete-change analysis, the student may vary any of the variables or constants, or any multiple combinations of them, of the current problem by a discrete amount and the program will resolve the problem. The student may then compare solutions to determine the effects of the changes. The changes made here alter the original problem setup. Subsequent changes are possible but the original problem parameters should be restored if the original problem is to be used as a basis for comparison. There are three modes available for displaying the results of the discrete-change analysis. The student may elect to have only the value of the objective function printed along with a list of the variables in solution or he may ask for each iteration tableau and/or the final iteration tableau.

After the student elects to terminate the sensitivity analysis, he is again given the options available to him after the final tableau is printed, as explained earlier.

Section 4 is a critique and evaluation section where the student answers questions about his use of SAL. SAL is then deactivated and the student may use the teletype as a typewriter and comment further on SAL. His comments and answers to critique questions are given to the monitor as a tear-off portion of the teletype paper. The information may be used to improve SAL.
In summary, the Simplex Algorithm Laboratory is, in part, a conventional computer-assisted instruction program that refreshes the student's basic knowledge of the simplex algorithm by presenting an abridged treatment of the theory and applications of the algorithm. The program then expands to create an interactive responsive environment wherein the student may experiment with arbitrarily selected problems and variations and observe the application of the algorithm as it solves these problems. The student's freedom in selecting and modifying his own problems and observing the applications of the algorithm to many different situations affords the student an opportunity to better understand the more difficult aspects of the algorithm. This laboratory will provide the student an opportunity to gain a better "feel" for the algorithm. The primary application of the computer in the laboratory is as a computational device rather than as a fancy textbook.
CHAPTER 4

CONCLUSIONS AND RECOMMENDATIONS

The Simplex Algorithm Laboratory is an approach to CAI with the emphasis on the use of the computer in a role which utilizes its highspeed computational capability. The computer plays a vital part in the man-machine interaction. SAL could not function without the assistance of the computer. This is in contrast to most conventional CAI systems where the computer acts, more often than not, as little more than a "page-turner" and where the computer could, and probably should, be replaced by a much less expensive device such as a teaching machine.

SAL uses the computer to create an interactive responsive environment in which the student can, almost unrestricted, experiment with the applications of the simplex algorithm to the solution of linear programming problems. By varying the problems, and parameters within the problems, the student can gain a "feel" for the algorithm that would be difficult or impossible to gain otherwise.

The critique and evaluation section, Section 4, could be used to obtain information for changing, correcting, and improving the system. Cooperation between the users and the monitor could contribute to development of a more efficient program.
The basic control program for SAL could be expanded to also cover the dual simplex algorithm. The inclusion of the dual algorithm would allow the sensitivity analysis portion to be expanded accordingly to include ideas depending on utilization of the duality principle. The larger system would then provide the student with an expanded horizon for the responsive environment.

There are several advantages to a CAI system using the techniques illustrated by SAL. 1) The student can exercise almost unrestricted initiative in his investigation of the subject matter being presented, 2) The interrelationships between elements of a complex system could more easily be visualized and understood by the student, 3) The dynamic behavior of certain types of systems could be better demonstrated, and 4) The use of a computer in such a system can be justified because of the essential contribution of the computers computational characteristics. Inclusion of a visual display device, such as a cathode-ray tube (CRT) could improve the effectiveness of such a system.

Systems like SAL also have some disadvantages. 1) In providing few restrictions on the student's initiative, the program cannot effectively monitor the student's performance and cannot, therefore, provide significant guidance to the student, 2) The subject matter to be presented must be expressable in mathematical or logical terms and
related by well defined combinational rules, and 3) The subject matter must be of sufficient complexity to justify the effort.

When considering any CAI system, one thought must be kept in mind, "... assign to the machine those things it does best..., and reserve for the human the things he does better for himself!" (Zinn 1970, p. 169).
APPENDIX A

USER'S MANUAL
Chapter I: Introduction

This manual is designed to facilitate the use of the Simplex Algorithm Laboratory (SAL). SAL is a computer-assisted instruction (CAI) program coded in FORTRAN IV and uses a remote teletype terminal connected through the CDC INTERCOM 2 time-sharing system to the CDC 6400 computer at The University of Arizona, Tucson, Arizona.

The purpose of SAL is to create an interactive responsive environment wherein the student may experiment with different linear programming problems and variations and observe the results as the simplex algorithm is automatically applied by the computer.

Chapter I explains how to gain access to both the remote terminal facility and to the operational control of the laboratory. Chapter II explains all the student needs to know about the operation of the remote terminal to allow him to place the system into operation and make contact with SAL. Chapter III explains the control and inputs necessary to effectively use SAL. Chapter IV contains a detailed discussion of SAL. Chapter V is devoted to a sufficiently complete presentation of linear programming and the simplex algorithm to prepare the unfamiliar student with an adequate understanding of the concepts and terminology involved.

The user must obtain approval from the University Computer Center (UCC) to use INTERCOM and the remote terminals.
Procedures, telephone number and passwords to be used will be provided by the UCC. The SAL program is available through the Computer Science office. The user must obtain information from the Computer Science office to determine the steps necessary to insure that the program is available to the computer for use by the remote terminal.

Chapter II: Activation and Control of INTERCOM

This chapter explains how to activate and control INTERCOM which is the system that provides the communication link with SAL. It contains all the information needed for the user to operate the console for the purpose of establishing contact with SAL. The following steps must be performed, in order, to activate INTERCOM. The list of steps is followed by a sample of an actual dialogue. The responses by the user have been underlined.

1. Depress the ORIG button located on the lower righthand side of the console face. The button will illuminate and the electric motor inside the typewriter terminal will become operational. (There may or may not be a dial tone from the speaker near the button.)

2. Using the telephone-type dial adjacent to and above the ORIG button, dial the telephone extension provided (see previous chapter). If the system is in service, the typewriter terminal will print a message that will be explained in the next step. A busy tone from the speaker or no response by the typewriter indicates a
33

connection has not been made. In this case, depress the CLR button located to the right of the ORIG button to deactivate the terminal. Steps 1 and 2 may be repeated if necessary.

3. When the terminal has been connected to the computer, the following message will be printed on the paper by the terminal typewriter:

```
CONTROL DATA INTERCOM
DATE 03/18/71
TIME 19.26.25
```

The typewriter then stops and the computer is waiting for the user to send a message. The user should type the following message, including the period:

```
LOGIN.
```

and then depress the RETURN key located on the extreme right of the keyboard. The RETURN key causes the message just typed to be transmitted to the computer.

**NOTE:** There are two methods of correcting errors in a typed message prior to transmitting it. One method is to backspace and retype over the errors. The carriage can not physically be backspaced but the same effect is achieved by typing `< (upper case O).

LOGJN<IN. is equivalent to LOGIN. Another method to correct an error is to void the entire line and start again on a new line where the message is retyped. To void a line, depress the CTRL key located on the lower left portion of the keyboard and while keeping it depressed
strike the letter X. This action will physically shift the carriage to the left margin and advance the paper one line.

4. The computer will type the line:

    TYPE VALID USER NAME-

and stop with the carriage positioned one space to the right of the hyphen (-). The user must type the appropriate name (see previous chapter). The user must always begin typing with the carriage positioned where it has come to rest. The only exception is when a line is voided; in that case the carriage must begin at the left margin.

5. The computer will type the line:

    ################ TYPE VALID PASSWORD-

The user must type a valid password (see previous chapter).

6. If the user name and password are not accepted, the computer will print the following line:

    THE MYSTERY GUEST WILL PLEASE LOGIN

If this happens, the user may return to step 3 and retype LOGIN, or jump to step 7. J. and deactivate the terminal.

7. If the user name and password are accepted, the computer will print:

    03/18/71 LOGGED IN AT 19.27.05 (date, time, and ID will be appropriate)

    COMMAND-

The terminal has now been connected to the computer and is in a time-sharing interactive mode. The user may now request the Simplex Algorithm Laboratory as follows:
A. Type the command

\[ \text{ATTACH (file1, file2)} \] (refer to previous chapter for instructions on obtaining names of file 1 and file 2)

and wait for the computer to respond with

\[ 21.47.03. \text{ATTACH(file1, file2)} \]

COMMAND-

B. Type the command

\[ \text{ATTACH(file3, file4)} \]

and wait for the computer to respond with

\[ 21.47.58. \text{ATTACH(file3, file4)} \]

COMMAND-

NOTE: File1 and file2 may be the same name; file3 and file4 may be the same name but different from file1 and file2. In steps A and B above, if the computer replies with:

\[ ** \text{ILLEGAL P.F. REQ} \]

the computer does not have the referenced files available to it.

C. Type the command

\[ \text{EFL, 070000.} \]

and wait for the computer to respond with

COMMAND-

D. Type the command

\[ \text{ETL, 0020.} \]

and wait for the computer to respond with

COMMAND-
E. Type the command

SETUP.

and wait for the computer to respond with

ON AT 19.28.11. (time will be appropriate)
SYSTEM--FORTRAN
NEW OR OLD FILE--

F. Type the command

OLD/file1 (use same file1 name as in step A above)

and wait for the computer to respond with

READY.

G. Type the command

RUNER.

and wait for the computer to respond with

WAIT DISK (this response may not be present)
PROGRAM TRANSFERRED TO COMPILER (may not be present)

H. The next response by the computer will be under the control of the Simplex Algorithm Laboratory. The user should refer to Chapter III of this manual for instructions on controlling the SAL program. Before leaving this section, the user should read steps I and J below.

I. INTERCOM may be terminated at any time by typing

LOGOUT.

The computer will respond with
The user should then depress the CLR button, located next to the ORIG button, which will disable the terminal.

**NOTE:** Steps I and J (preferably in that order) may be done at any time during use of the terminal. The following sample dialogue illustrates the commands needed for INTERCOM:

```
CONTROL DATA INTERCOM
DATE 04/28/71
TIME 19.15.00.
LOGIN.
TYPE VALID USER NAME-SOME
******** TYPE VALID PASSWORD-

04/28/71 LOGGED IN AT 19.16.01.
WITH USER-ID AX
COMMAND- ATTACH(DBLM1, DRLM1) 19.16.36.
COMMAND- ATTACH(DRLT2, DRLT2) 19.16.53.
COMMAND- EFL,070000.
COMMAND- ETL,0020.
COMMAND- SETUP.
ON AT 19.17.33.
SYSTEM--FORTRAN
NEW OR OLD FILE--OLD/DBLM1

READY.
RUNER
WAIT DISK
PROGRAM TRANSFERRED TO COMPILER

LOGOUT.
CP TIME 22.578
PP TIME 321.479
04/28/71 LOGGED OUT AT 20.18.39.
```
Chapter III: Control of the Simplex Algorithm Laboratory

This chapter contains the necessary information needed by the user to control and use SAL. The procedures given here apply after INTERCOM 2 has been activated and SAL has been attached according to the directions given in the preceding chapter.

The following message will be printed when SAL has been activated.

*****SIMPLEX ALGORITHM LABORATORY**************
THIS IS THE SIMPLEX ALGORITHM LABORATORY (SAL) WHICH IS A COMPUTER-ASSISTED INSTRUCTION (CAI) PROGRAM THAT WILL HELP YOU GAIN A BETTER UNDERSTANDING OF THE SIMPLEX ALGORITHM AS IT IS APPLIED TO THE SOLUTION OF LINEAR PROGRAMMING PROBLEMS.

WHEN YOU ARE READY TO CONTINUE, TYPE GO

SAL is completely automatic in its presentation and will progress from the above introduction paragraph to its last section by following periodic commands from the user. SAL clearly indicates when it expects a command or input from the user and specifies on each occasion the options available for the user to choose from. The command or input that the user must supply is a single word or number that must be typed on the teletype and transmitted to SAL. The commands and inputs follow naturally from the presentation and require no memorization, no special techniques, and no computer programming knowledge or experience by the user.
All commands and inputs must meet certain general requirements that are listed below:

1. All responses (commands or inputs) by the user must be typed on the teletype beginning from wherever SAL has positioned the carriage. There are two possible exceptions to this rule, 1) SAL may direct the user to reposition the carriage prior to an input but this is done only during one phase of the expedite (EXP) mode that will be explained in the next chapter, and 2) the user may elect to correct a typographical error by voiding the line (see Step 3 of the preceding chapter); in this case the carriage will be physically located at the extreme left margin of the paper but SAL thinks the carriage is again located at the original position occupied before the voided message was typed. Except in the EXP mode, SAL will always position the carriage at a standardized location in preparation for a user response. The standardized location is 8 spaces to the right of the left margin of the paper and serves two purposes: first, it is another clue to the user that SAL is waiting for a reply, and second, it enables easy identification of the user responses when the mixed computer-user dialogue is read. Most of the text printed by SAL starts at an indentation of 15 spaces from the left margin.

2. All word responses must be identical with the option selected from the list of alternates. No periods or other punctuation marks are allowed.
3. Number responses are of two types: 1) decimal numbers, and 2) integer numbers. A decimal number must be represented as either a straight decimal number or as a coded scientific notation number. A straight decimal number may be up to 11 characters including the sign and the decimal point. The sign is optional (a number with no sign is assumed to be positive) but the decimal point is required.

Examples of allowable straight decimal numbers are:
7.;+5.;-.00068; 3.0; -17.5769542; 17.5769428

Examples of numbers not allowed as straight decimal numbers are:
7 (no decimal point); +17.57695428 (too many characters)

A coded scientific notation decimal (E format) is illustrated by the following examples of allowable numbers and their values:
+3.0000E+00 = 3.0000; -3.0000E-02 = -.030000;
+1.2345E+03 = 1234.5
+1.0000E+10 = 10000000000.

The E format decimal number must contain 11 characters. A sign must precede the number and a sign must follow the E. The E must be present and must occupy the 8th position. The decimal point must be present and must occupy the 3rd position. The number following the E and its sign must be a two digit integer. Examples of invalid E format numbers are:
1.2345E+00 (no sign before the number)
+1.234E+01 (E out of position)
-1.2345E02 (no sign after the E)
The E format decimal number should be used only when the number to be entered is too large to be represented as a straight decimal number. SAL automatically converts all decimal numbers to the E format before printing them.

An integer number must be represented by two digits without a decimal point. The user must type, for example:

- 01 to represent 1 (one)
- 05 to represent 5 (five)
- 10 to represent 10 (ten)

SAL expects all integers to be represented by two digits. If the user types 3, SAL will interpret it as 30. SAL uses integers only for indexing and bookkeeping purposes and allows no integer larger than 99.

SAL will always accept the command STOP whenever it expects a word reply even though STOP may not be listed as one of the options available at that point. The command STOP will cause SAL to deactivate itself and return control to INTERCOM (see Step 7.I. and 7.J. of the previous chapter for terminating INTERCOM). SAL will not recognize STOP as a valid response when it is expecting a number response.

The user can abort at any time and force SAL into deactivation by depressing the CTRL key and Z simultaneously, then A. This will return control to INTERCOM. This abort technique should be used only for cases where there seems to be a malfunction which would be indicated by SAL responding erratically or not responding at all.

If SAL becomes inactive, except when waiting for a user response, for
more than about 90 seconds the user should assume there has been a malfunction. Prior to aborting, the user may elect to try a remedial technique by typing the command GO. If this fails, the user must abort. Once control has been transferred to INTERCOM, SAL may be restarted by repeating the appropriate steps in the preceding chapter.

Normal termination of SAL at the conclusion of the last section is done by the user typing the appropriate commands specified by SAL at that point.

**Chapter IV: Simplex Algorithm Laboratory**

This chapter gives a detailed explanation of what SAL is and how it is used. The approach will be to start at the beginning and discuss the characteristics of SAL as they would be encountered by a user. A sample dialogue appears as Appendix B.

SAL is divided into four sections for explanation purposes. Section 1 contains a brief introduction to SAL. Section 2 is the problem solving section and Section 3 is the sensitivity analysis part. Sections 2 and 3, considered together, make up the responsive environment where the user experiments with the simplex algorithm by selecting problems and variations to these problems and observes the application of the algorithm to the problems. Section 4 is the critique and evaluation section where the user comments on the pros and cons of SAL.
The introductory material is stored within SAL in small blocks of text (such blocks are often called frames). Each frame has a question or a request for a command as the last sentence. SAL prints out one frame and waits for the user to respond with an answer or a command. After the user reads the text material and responds, SAL prints out a new frame of text. The choice of the next frame to be presented is determined by SAL. SAL compares the user's response against predetermined selections and branches accordingly. The current version of SAL does not provide branching to expanded explanations, remedial sections, or repeats of previous material as does most conventional CAI programs. The current version of SAL contains only one main path through the introductory material and provides fixed-order frame by frame presentation that can not be altered. There is, however, always the option of skipping over the remaining introductory material to get to the problem solving section. SAL has adapted this fixed order of presentation for two reasons: first, the introductory material is short and is material the user is basically familiar with, and second, the use of the computer for conventional frame-oriented CAI has questionable merit. SAL presents the introductory material primarily to provide the user with the terminology needed for the responsive environment that follows the introductory material.
After completing section one, SAL prints:

YOU MAY NOW SELECT A L. P. PROBLEM FOR SOLUTION BY ENTERING THE PARAMETERS AS THEY ARE REQUESTED. YOU HAVE TWO OPTIONS FOR ENTERING THE PARAMETERS. THE REGULAR METHOD IS A STEP BY STEP RESPONSE THAT WILL INSURE ACCURACY. THE EXPEDITE METHOD IS FASTER BUT REQUIRES FAMILIARITY WITH THIS PROGRAM TO BE EFFECTIVE. IT IS RECOMMENDED THAT YOU USE THE REGULAR METHOD FOR THE FIRST FEW PROBLEMS.

SELECT THE OPTION YOU WANT. TYPE (REG OR EXP)

SAL has now created an interactive responsive environment and the user may select any linear programming problem (limited in size to up to 40 variables and 20 constraints) and experiment with the simplex algorithm by varying the problems and the parameters of the problems. The user must select the mode he wishes to use for entering the parameters.

Before entering the parameters of the problem, the user must insure that the objective function of the problem is expressed as a maximization (refer to the following chapter for information on changing a minimization problem to a maximization problem). The user must then rewrite the objective function as

\[ x_0 - \text{(objective function)} = 0 \]

For example, if the objective function is

\[ \text{maximize } 4x_1 + 2x_2 \]

rewrite it as

\[ x_0 - 4x_1 - 2x_2 = 0 \]
The objective function that must be entered for SAL to process is then

\[-4x_1 - 2x_2\]

The remainder of the equation is not used by SAL; it serves only to help the user visualize the relationships. In short, the objective function to be entered must be the negative of the maximization function.

As another example, if the problem is to maximize \(3x_1 + 4x_2 + 2x_3\), the user must input this as

\[-3x_1 - 4x_2 - 2x_3\]

The constraints can be entered directly as they occur in the statement of the problem. SAL will accept \(\leq, =, \) or \(\geq\) constraints. Character limitations on the teletype cause SAL to refer to \(\leq\) as \(<=\) and to \(\geq\) as \(>=\). The user must not add surplus, slack or artificial variables to the constraints. This is done automatically by SAL.

After all the parameters have been entered, SAL prints an edit table containing all the parameters that have been entered. The user must check the numbers for accuracy. SAL allows the user to correct any of the numbers. See the sample dialogue in Appendix B for an illustration of the edit table.

After the user is satisfied that the problem has been entered correctly, he asks SAL to solve the problem. SAL takes the problem as it has been defined by the user and adds slack, surplus, or artificial variables, as appropriate, and rearranges the data into a condensed simplex tableau (see the sample dialogue, Appendix B). The original
variables will be labeled $x_1$ to $x_{40}$, the slack/surplus variables will be $x_{41}$ to $x_{60}$, and the artificial variables will be $x_{61}$ to $x_{80}$. The tableau is printed by SAL and is called the first iteration of the algorithm. After the first tableau has been printed, SAL gives the user the option of having the next tableau printed showing the results of the next iteration of the algorithm or having only the final tableau printed showing the optimal solution. The user has the same option after each subsequent iteration.

The algorithm automatically terminates when an optimal solution has been found. At this point SAL provides the user with several options: 1) he can stop, 2) he can repeat the above process by entering a new problem, 3) he can alter the current problem and cause the algorithm to be repeated, 4) he can alter the final iteration and cause the algorithm to be continued, 5) he can apply the sensitivity analysis to the current problem, or 6) he can skip to the critique and evaluation section. Options 2, 3, and 4 will cause SAL to recycle to the appropriate places so the user may make the necessary entries or changes and then cause the basic process to be repeated. Option 1 is obvious (see previous chapter for STOP command) and option 6 will be covered later in this chapter. Option 5, sensitivity analysis, is covered next.

When the sensitivity analysis option has been selected, SAL provides three options for sensitivity analysis: 1) effects of varying the coefficients in the objective function, 2) effects of varying the
constraint constants, and 3) results of selected discrete changes in any variable or constant. Options 1 and 2 of the sensitivity analysis use information from the final tableau and provide ranges over which the test variable or constant may vary before the current solution is no longer optimal. SAL prints the ranges of values. These two options do not alter the final tableau; therefore, any number of the appropriate parameters may be analyzed (only one parameter at a time may be analyzed since multiple ranging analysis is beyond the capability of SAL). Option 3 of the sensitivity analysis causes the initial problem to be altered and a new solution obtained. The user compares the new solution against the old solution. If option 3 is exercised more than once, the original problem must be restored before subsequent analysis can be made. Option 3 is basically the process of resolving the original problem that has been modified but has an additional feature of allowing the user to have only the optimal value of the objective function and a list of the basic variables printed to show the results of the change. Otherwise, the user can select the regular methods of tableau presentation. The user may continue the sensitivity analysis for as long as he desires.

When finished with the sensitivity analysis, the user can stop or enter a new problem or modify the current problem and repeat the process or skip to the critique and evaluation section.
When SAL enters the critique and evaluation section, it will print

*************** TEAR HERE ***************

YOU ARE NOW IN THE CRITIQUE AND EVALUATION SECTION. PLEASE ANSWER THE FOLLOWING QUESTIONS ABOUT YOUR USE OF SAL. AFTER YOU HAVE ANSWERED THE QUESTIONS, YOU WILL BE INSTRUCTED TO DEACTIVATE SAL. THE TELETYPING WILL THEN BE AUTOMATICALLY DISCONNECTED FROM THE COMPUTER AND WILL BE JUST LIKE AN ELECTRIC TYPEWRITER. USING THE TELETYPING AS A TYPEWRITER, PLEASE CONTINUE TYPING WITH ANY COMMENTS YOU MAY HAVE. WHEN FINISHED, TEAR THE PAPER WHERE INDICATED ABOVE AND PROCESS THIS PORTION OF THE PAPER AS DIRECTED BY THE INDIVIDUAL MONITORING THE PROGRAM. WHEN READY TO CONTINUE, TYPE GO.

The evaluation and comments by the users may be used by the monitor to modify and improve SAL.

In summary, SAL is a CAI program that presents a small amount of introductory material in the form of conventional frames of text to prepare the user for the responsive environment that follows. The responsive environment allows the user to observe the application of the simplex algorithm to problems he selects and varies. The user is almost unrestricted in the selection of the problems and in making variations to them. Rapid application of the algorithm to many problems provides the user with a "feel" for the techniques of the algorithm that would be difficult to obtain otherwise.
Chapter V: Linear Programming and the Simplex Algorithm

This chapter is intended to give the reader who is not familiar with the simplex algorithm and linear programming enough background information to enable him to effectively use the Simplex Algorithm Laboratory. The material in this chapter was heavily influenced by Gass (1964) and Wagner (1969).

Programming problems, including linear programming problems, deal with the efficient use or allocation of limited resources to meet specified objectives. These problems have a large number of solutions that satisfy the conditions of each problem. The selection of a particular solution to a problem depends on the specified objective desired. A solution that satisfies both the conditions of the problem and the specified objective is termed an "optimum solution." The optimum solution may be either a maximum solution or a minimum solution and depends on the stated objective. A simplified problem that provides a good example is that of a manufacturer who must determine what allocation of his available resources will permit him to manufacture his products in a way that meets his production schedule and maximize his profits. The problem has as its conditions the limitations of the resources and the requirements of the production schedule, and as its objective the desire to maximize the gain.
Linear Programming

Linear programming problems are a special subclass of programming problems. A linear programming problem can be described by a mathematical model using relationships which are linear. Mathematically, these relationships are of the form

\[ a_1x_1 + a_2x_2 + \ldots + a_jx_j + \ldots + a_nx_n = a_0 \]

where the \( a_j \) are known coefficients and the \( x_j \) are unknown variables.

Geometrically, these relationships are equivalent to straight lines, planes, and hyperplanes. The complete mathematical statement of the problem includes a set of simultaneous linear equations which represent the conditions of the problem and a linear function which expresses the objective of the problem.

The solution of a linear programming problem is dependent upon the solution or solutions to the associated set of linear equations. The following simple example will illustrate the relationship. The set of two equations in two unknowns

\[ 2x_1 + 3x_2 = 8 \]
\[ x_1 + 2x_2 = 5 \]

has the unique solution \( x_1 = 1 \) and \( x_2 = 2 \), while the single equation

\[ x_1 + 2x_2 = 8 \]

has an infinite number of solutions. For every \( x_2 \) (or \( x_1 \)) there is a corresponding value of \( x_1 \) (or \( x_2 \)). If \( x_1 \) and \( x_2 \) are restricted to be nonnegative, the range of \( x_1 \) and \( x_2 \) is limited, since
\[ x_1 + 2x_2 = 8, \quad x_1 \geq 0, \quad x_2 \geq 0 \]

then

\[ x_1 = 8 - 2x_2 \geq 0 \text{ implies } 0 \leq x_2 \leq 4 \]

and

\[ x_2 = 4 - 0.5x_1 \geq 0 \text{ implies } 0 \leq x_1 \leq 8 \]

There are still an infinite number of solutions but the addition of restrictions or constraints resulted in less freedom for the variables. The nonnegativity of the variables is a requirement of linear programming problems and will be discussed later. Systems of simultaneous equations in which there are more variables than equations are called underdetermined. In general, underdetermined systems of linear equations have either no solution or an infinite number of solutions.

One method of solving an underdetermined system of equations is to reduce the system to a set containing the same number of variables as equations. Such a set is called a determined set. Reduction to a determined set is done by setting the appropriate number of variables equal to zero. For example, the underdetermined system

\[
\begin{align*}
2x_1 + 3x_2 + x_3 &= 8 \\
-x_1 + 2x_2 + 2x_3 &= 5
\end{align*}
\]

has three solutions

\[
\begin{align*}
x_1 &= 0, \quad x_2 = 2.750, \quad x_3 = -0.250 \\
x_1 &= 3.667, \quad x_2 = 0, \quad x_3 = 0.667 \\
x_1 &= 1, \quad x_2 = 2, \quad x_3 = 0
\end{align*}
\]
resulting from setting each of the variables, in turn, to zero.

Linear programming is concerned with only the nonnegative solutions to underdetermined systems of linear equations. If the example just cited represents the conditions of a linear programming problem, then only two of the three solutions are possible candidates for the solution to the linear programming problem. The negative value for \( x_3 \) in one solution excludes that solution from consideration. The linear function representing the specified objective of the problem is then used to select the best solution from the possible candidates. This linear function, called the objective function, must be optimized by the selected solution. If, for example, the goal was to maximize the objective function \( x_1 + x_2 + x_3 \), then \( x_1 = 3.667, x_2 = 0, \) and \( x_3 = 0.667 \) is the optimal solution that yields a value of 4.333 for the objective function. If the goal was to minimize the objective function \( x_1 - x_2 \), then the solution \( x_1 = 1, x_2 = 2, x_3 = 0 \) would be optimum with a value of -1. The optimum solution always minimizes or maximizes the objective function. All future references to optimization in this thesis will be to maximize the objective function. No generality is lost because the minimum of an objective function is equal to minus the maximum of the negative of the objective function, that is

\[
\text{minimum of } f(x) = -\text{maximum of } [-f(x)].
\]

Combining the linear constraints with the optimization of an objective function will, in general, transform an underdetermined
system of linear equations with many possible solutions to a system that can be solved for a solution that yields the unique optimum value of the objective function. The simplex algorithm (Dantzig 1953) is one computational technique that will yield this unique optimum solution. The latter portion of this chapter will be devoted to explaining the simplex algorithm.

There are several ways to mathematically express linear programming problems. It can be shown (but it will not be done here) that the different forms are equivalent. The one that will be used here is

maximize the objective function

\[ \sum_{j=1}^{n} c_j x_j \]

subject to the constraints

\[ \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, \ldots, m \]

\[ x_j \geq 0 \text{ for all } j \]

where the \( c_j \) for \( j = 1, 2, \ldots, n \); the \( b_i \) for \( i = 1, 2, \ldots, m \); and \( a_{ij} \) are all constants, and \( m \leq n \).
The procedures and techniques involved in analyzing a real-life problem and reducing it to its equivalent mathematical statement in the form of a linear programming problem is beyond the scope of this thesis. However, one simple problem will be reduced to its equivalent linear programming problem to provide the reader with some insight into the relationships involved.

The following simplified problem (Wagner 1969, pp. 15-17) illustrates how a problem might be formulated. Assume a food company produces potatoes for French fries, hash browns, and flakes. The raw potatoes are stored by length and quality before being allocated to the separate production lines. The company buys its potatoes from two sources. Potatoes from source 1 have a 20% yield of hash browns, a 20% yield of French fries, and a 30% yield of flakes, and the remaining 30% is waste. Source 2 potatoes have yields of 10%, 30%, and 30% for the same categories, respectively, and 30% waste. How many pounds of potatoes should the company buy from each source? The answer depends on the relative profit contributions of the sources. These relative figures are calculated by adding the sales revenues associated with the yields for the separate products, and subtracting the costs of buying the potatoes. Other expenses such as sales and distribution costs are ignored since they do not affect the purchase allocation decision. Assume that the calculations show that the relative profit contribution is 5 for source 1 and 6 for source 2.
Two other factors affect the purchase decision: the maximum amount of each product that the company can sell, and the maximum amount that the company can manufacture. Assume that these two factors, in concert, implies that the total production cannot exceed 1.8 units for French fries, 1.2 for hash browns, and 2.4 for flakes.

Let \( x_1 \) be the amount (in weight) of potatoes that will be purchased from source 1, and \( x_2 \) the amount from source 2. The problem may now be expressed by the following linear programming problem:

\[
\text{maximize } 5x_1 + 6x_2 \\
\text{subject to the constraints}
\]

\[
0.2x_1 + 0.3x_2 \leq 1.8 \\
0.2x_1 + 0.1x_2 \leq 1.2 \\
0.3x_1 + 0.3x_2 \leq 2.4 \\
x_1 \geq 0, \quad x_2 \geq 0
\]

The need for the nonnegativity restrictions on the variables in linear programming problems is now apparent. For example, a value of \( x_1 = -3 \) has no meaning in this problem.

The potato problem will be given a geometric interpretation to provide an insight into the relationships of the constraints and the objective function. Figure 1 is a graphical representation of the problem. The inequality constraints have been replaced by equations (straight lines) with an arrow indicating the permissible values of the
Fig. 1. Optimal Solution.
variables. The variables are also bounded by the coordinate axis because of the nonnegativity restriction. The parallel straight lines represent various values of the objective function, with the arrow indicating the direction of increasing value. The polygon \( oabc \) represents the region of values for \( x_1 \) and \( x_2 \) that satisfy all constraints. Note that the third constraint inequality has no effect on this problem. The polygon is called the solution set and the vertices \( o, a, b, \) and \( c \) are called the extreme points. The optimal solution occurs at the extreme point \( b \) where \( x_1 = 4.5, x_2 = 3 \) and the objective function value is 40.5. This is intuitively obvious from Figure 1.

If the objective function of the potato problem was changed to \( 4x_1 + 6x_2 \), then Figure 2 would represent the new problem. The objective function is shown as parallel lines and it is again intuitively obvious that the optimum solution occurs at extreme point \( b \) with \( 4x_1 + 6x_2 = 36 \). Extreme point \( a \) is also an optimal solution with \( x_1 = 0, x_2 = 6 \), and \( 4x_1 + 6x_2 = 36 \). In fact there are an infinite number of optimum solutions along the line segment \( ab \). In this case there are "alternative optimal solutions."

The problem defined by

\[
\begin{align*}
\text{maximize} & \quad -2x_1 + 6x_2 \\
\text{subject to} & \\
-1x_1 - 1x_2 & \leq -2 \\
-1x_1 + 1x_2 & \leq 1 \\
x_1 \geq 0 \text{ and } x_2 \geq 0
\end{align*}
\]
Fig. 2. Alternative Optimal Solution.
is graphed in Figure 3 and illustrates a problem that has an "unbounded" optimal solution. The objective function can be made arbitrarily large while satisfying the constraint inequalities. The solution set is unbounded.

The problem defined by

\[
\begin{align*}
\text{maximize} & \quad 1x_1 + 1x_2 \\
\text{subject to} & \quad -1x_1 + 1x_2 \leq -1 \\
& \quad 1x_1 - 1x_2 \leq -1 \\
& \quad x_1 \geq 0 \text{ and } x_2 \geq 0
\end{align*}
\]

is graphed in Figure 4 and illustrates a problem that has no feasible solution. Note that the solution set is empty.

The following conclusions may be obtained from the geometric interpretations just presented. If the solution set is nonempty, the optimal value of the objective function may be finite or may be unbounded. If finite, then the optimal solution exists at an extreme point (a problem with alternative optimum solutions has an extreme point included among the infinite number of solution points). These conclusions have not been proved but do provide the basic concepts needed to understand the discussion of the simplex algorithm that follows.
Fig. 3. Unbounded Optimal Solution.

Fig. 4. No Optimal Solution.
**Simplex Algorithm**

The purpose of the following discussion is to provide the reader with a basic understanding and an intuitive feeling of the simplex algorithm and how it is applied. The discussion is not mathematically rigorous, no proofs will be given, and some of the finer points will be omitted. Examples will be used to illustrate the techniques and concepts. The simplex algorithm is a mathematically rigorous computational technique that always yields an exact optimal solution if one exists.

The simplex algorithm will be used to solve the following problem:

\[
\begin{align*}
\text{maximize} & \quad 4x_1 + 5x_2 + 9x_3 + 11x_4 \\
\text{subject to} & \quad 1x_1 + 1x_2 + 1x_3 + 1x_4 \leq 15 \\
& \quad 7x_1 + 5x_2 + 3x_3 + 2x_4 \leq 120 \\
& \quad 3x_1 + 5x_2 + 10x_3 + 15x_4 \leq 100 \\
& \quad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0
\end{align*}
\]

Let \( x_0 \) be the value of the objective function and rewrite the inequality constraints as equalities by introducing the variables \( x_5, x_6, \) and \( x_7 \). These new variables are called "slack" variables ("surplus" variable is the appropriate term when the associated inequality is \( \geq \)) and are also restricted to be nonnegative. The system is now written as
\[ \begin{align*}
1x_0 - 4x_1 - 5x_2 - 9x_3 - 11x_4 &= 0 \quad \text{Row 1} \\
1x_1 + 1x_2 + 1x_3 + 1x_4 + 1x_5 &= 15 \quad \text{Row 2} \\
7x_1 + 5x_2 + 3x_3 + 2x_4 + 1x_6 &= 120 \quad \text{Row 3} \\
3x_1 + 5x_2 + 10x_3 + 15x_4 + 1x_7 &= 100 \quad \text{Row 4}
\end{align*} \]

The row indicators at the extreme right are for reference only and play no part in the algorithm. The numbering scheme of the rows in this thesis does not conform to the scheme used by Wagner (1969) but has been selected to be compatible with the Simplex Algorithm Laboratory. Note that the value of the objective function \(x_0\) has been set equal to zero and that the variables have been transferred to the left side of the equation. This implies that \(x_1, x_2, x_3,\) and \(x_4\) are equal to zero. Rows 1, 2, 3, and 4 can be considered as an underdetermined system of equations which has been reduced to a determined system by setting \(x_1, x_2, x_3\) and \(x_4\) equal to zero. The system now has a unique solution which is called the "initial basic solution" and \(x_0, x_5, x_6, x_7\) are called the "basic variables" or "the basis." The remaining variables are called "nonbasic." The current solution yields an objective function value \(x_0\) of zero. Hopefully, another solution exists which will provide a larger value for \(x_0\).

Other solutions may be obtained by selecting different combinations of variables to be in the basis. The algorithm specifies that the most likely candidate be selected from the nonbasic variables. In other words, select from the nonbasic variables in Row 1 the variable with
the most negative coefficient and enter that variable into the basis.

Each coefficient in Row 1 represents the increase (for negative coefficients) or decrease (for positive coefficients) in \( x_0 \) with a unit increase of the associated nonbasic variable.

Simplex Criterion I of the simplex algorithm states that if there are nonbasic variables having a negative coefficient in Row 1, select one such variable with the most negative coefficient. If all nonbasic variables have positive or zero coefficients in Row 1, an optimal solution has been reached.

Simplex Criterion I specifies that \( x_4 \) should be entered into the basis. It is also implied that \( x_4 \) be made as large as possible. One of the basic variables must be removed to allow the entry of \( x_4 \). As \( x_4 \) is increased, every basic variable must be decreased in any row where \( x_4 \) has a positive coefficient. For example, if \( x_4 \) is increased to 1 then \( x_6 \) in Row 3 must be decreased by 2 so that the constraint equality remains satisfied. As \( x_4 \) is increased, one of the basic variables will be decreased to zero, which is the lower limit, and will be removed from the basis. This variable is determined by applying Simplex Criterion II which is defined below.

Criterion II states that (a) For each row, take the ratios of the current right-hand side to the coefficients of the entering variable (ignore ratios with zero or negative numbers in the denominator). (b) Select the minimum ratio and exit the corresponding basic variable by setting it equal to zero.
The calculations specified by Criterion II are shown below:

<table>
<thead>
<tr>
<th>ROW NO.</th>
<th>BASIC VARIABLE</th>
<th>CURRENT SOLUTION</th>
<th>COEFF OF X₄</th>
<th>RATIO</th>
<th>MIN.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x₀</td>
<td>0</td>
<td>-11</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>x₅</td>
<td>15</td>
<td>1</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>x₆</td>
<td>120</td>
<td>2</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>x₇</td>
<td>100</td>
<td>15</td>
<td>6.66</td>
<td>6.66</td>
</tr>
</tbody>
</table>

The next solution will contain the variables x₀, x₅, x₆, x₄ in the basis and x₁, x₂, x₃, x₇ will be the nonbasic variables. The set of equations must now be modified such that x₄ has a coefficient of 1 in Row 4 and vanishes from the other Rows. This is done by a "change-of-basis" calculation or a "pivot operation." First Row 4 is divided by 15, the coefficient of x₄, to create a coefficient of 1 for x₄. Then the coefficients of x₄ in Rows 1, 2, and 3 are reduced to zero as follows:

Row 1: multiply Row 4 by 11 and add to Row 1,
Row 2: multiply Row 4 by -1 and add to Row 2,
Row 3: multiply Row 4 by -2 and add to Row 3.

The result of the pivot operation is

\[ \begin{align*}
1x_0 - 1.80x_1 - 1.33x_2 - 1.67x_3 & + 0.733x_7 = 73.3 \\
0.800x_1 + 0.667x_2 + 0.333x_3 & + 1x_5 - 0.067x_7 = 8.33 \\
6.60x_1 + 4.33x_2 + 1.67x_3 & + 1x_6 - 0.133x_7 = 107 \\
0.200x_1 + 0.333x_2 + 0.667x_3 + 1x_4 & + 0.067x_7 = 6.67
\end{align*} \]

The resulting system of equations shown above may be interpreted as follows:
Row 1 shows that $x_1$, $x_2$, $x_3$, and $x_7$ (all equal to zero) are the nonbasic variables and that the objective function value $x_0$ equals 73.3.

Row 2 shows that $x_5$ is a basic variable equal to 8.33,

Row 3 shows that $x_6$ is a basic variable equal to 107, and

Row 4 shows that $x_4$ is a basic variable equal to 6.67.

Stated equivalently, the current solution is $x_1 = 0$, $x_2 = 0$, $x_3 = 0$, $x_4 = 6.67$ with an objective function value of 73.3. The information contained in this system of equations can more conveniently be presented in a "condensed simplex tableau" or simply a "tableau."

The following tableau is equivalent to the set of equations immediately above.

<table>
<thead>
<tr>
<th>ITERATION NUMBER</th>
<th>BASIS</th>
<th>CURRENT VALUES</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_7$</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$x_0$</td>
<td>73.3</td>
<td>-1.80</td>
<td>-1.33</td>
<td>-1.67</td>
<td>.733</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$x_5$</td>
<td>8.33</td>
<td>.800</td>
<td>.667</td>
<td>.333</td>
<td>-.067</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>$x_6$</td>
<td>107</td>
<td>6.60</td>
<td>4.33</td>
<td>1.67</td>
<td>-.133</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>$x_4$</td>
<td>6.67</td>
<td>.200</td>
<td>.333</td>
<td>.667</td>
<td>.067</td>
<td>4</td>
</tr>
</tbody>
</table>

The nonbasic variables are identified in the heading row and the basic variables are listed in a column under BASIS with the current values to their immediate right. This tableau shows the results of the first iteration of the problem but the iteration number has a value of 2. The convention used in this thesis is that iteration number 1 is the process of organizing the data into the necessary form to begin the algorithm.
Criterion I is again applied to determine if the optimal solution has been obtained, and if not, to determine the next variable to enter into the basis. Inspection of the tableau for iteration 2 shows that Row 1 contains negative valued coefficients for some of the nonbasic variables; therefore, the current solution is not optimal and $x_1$ should be entered into the basis since $x_1$ has the most negative coefficient.

Applying Criterion II reveals that the minimum ratio (current value $\frac{1}{\text{coefficient of } x_1}$) is 10.4, approximately, for Row 2. Therefore $x_5$ should be removed from the basis and $x_1$ entered into the basis. The value .800, the coefficient of $x_1$ in Row 2, becomes the "pivotal element" for the change-of-basis calculations that follow:

Row 2: divide Row 2 by .800 to create a new Row 2 with a coefficient of 1 for $x_1$,

Row 1: multiply Row 2 by 1.80 and add to Row 1,

Row 3: multiply Row 2 by -6.60 and add to Row 3, and

Row 4: multiply Row 2 by -.200 and add to Row 4.

The pivot operation produces the results of the third iteration shown in the following tableau.

<table>
<thead>
<tr>
<th>ITERATION NUMBER</th>
<th>BASIS</th>
<th>CURRENT VALUES</th>
<th>$x_5$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_7$</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>$x_0$</td>
<td>92.1</td>
<td>2.25</td>
<td>.167</td>
<td>-.917</td>
<td>.583</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>$x_1$</td>
<td>10.4</td>
<td>1.25</td>
<td>.833</td>
<td>.417</td>
<td>-.083</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>$x_6$</td>
<td>37.9</td>
<td>-8.25</td>
<td>-1.16</td>
<td>-1.08</td>
<td>.416</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>$x_4$</td>
<td>4.58</td>
<td>-.250</td>
<td>.167</td>
<td>.583</td>
<td>.083</td>
<td>4</td>
</tr>
</tbody>
</table>
In iteration 3, \( x_1 \) (\( x_1 = 10.4 \)) has replaced \( x_5 \) in the basis and the objective function \( x_0 \) has increased from 73.3 to 92.1.

Repeating the process reveals that the current solution is still not optimal and that \( x_3 \) should be entered into solution. Applying Criterion II shows that the pivotal element is .583, the coefficient of \( x_3 \) in Row 4. Performing the change-of-basis calculations results in the iteration 4 tableau below.

<table>
<thead>
<tr>
<th>ITERATION NUMBER</th>
<th>BASIS</th>
<th>CURRENT VALUES</th>
<th>( x_5 )</th>
<th>( x_2 )</th>
<th>( x_4 )</th>
<th>( x_7 )</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>( x_0 )</td>
<td>99.3</td>
<td>1.86</td>
<td>.429</td>
<td>1.57</td>
<td>.714</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>( x_1 )</td>
<td>7.14</td>
<td>1.43</td>
<td>.714</td>
<td>-.714</td>
<td>-.143</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>( x_6 )</td>
<td>46.4</td>
<td>-8.71</td>
<td>-.857</td>
<td>1.86</td>
<td>.571</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>( x_3 )</td>
<td>7.85</td>
<td>-.429</td>
<td>.286</td>
<td>1.71</td>
<td>.143</td>
<td>4</td>
</tr>
</tbody>
</table>

The current solution is \( x_1 = 7.14 \), \( x_2 = 0 \), \( x_3 = 7.85 \), and \( x_4 = 0 \) to give an objective function value of 99.3 (\( 4 \times 7.14 + 5 \times 0 + 9 \times 7.85 + 11 \times 0 \approx 99.2 \)). Roundoff error accounts for the discrepancy between 99.3 and 99.2.

Row 1 of iteration number 4 has no negative coefficients; therefore, the current solution is optimal. The algorithm has found the desired optimal solution and has stopped. The following brief discussion is not part of the algorithm but will illustrate that the optimal solution has, in fact, been obtained. Row 1 of the final iteration may be rewritten as

\[
x_0 = 99.3 - 1.86x_5 - .429x_2 - 1.57x_4 - .714x_7.
\]
From this equation it is apparent that if any of the nonbasic variables \( (x_5, x_2, x_4, \text{ or } x_7) \) have any value greater than their present value of zero, then the value of \( x_0 \) will be decreased. The coefficients of the original variables that appear in Row 1 of the final iteration are called "relative" or "shadow costs" and represent the decrease in the optimal value of the objective function resulting from a unit increase in the nonbasic variable, assuming the final basis remains feasible.

The simplex algorithm may be summarized in the following four steps:

Step 1. Select an initial basis.

Step 2. Apply Criterion I. If the current trial solution is not optimal, proceed to Step 3. Otherwise, stop.

Step 3. Apply Criterion II.

Step 4. Make a change-of-basis and return to Step 2.

Certain ambiguities may arise during application of the simplex algorithm as it is stated above. The following rules and interpretations are provided to resolve these possible difficulties. These ideas are straightforward and will not be expanded beyond the short treatment here.

An "alternative optimal solution" exists if there is a zero valued coefficient for any nonbasic variable in Row 1 of the final iteration. If there is such a coefficient, then the variable of that coefficient may be entered into the basis without affecting the value of
the objective function. Thus, there are two basic solutions yielding the same optimal value for $x_0$. Actually there are an infinite number of optimal solutions that may be obtained by taking any positive-weighted average of these two basic solutions. The details will not be discussed here.

A "degenerate" basis occurs when two or more variables in the current basis go to zero upon entering the new variable during Criterion II calculations. Cycling difficulties can be avoided by following a consistent rule to determine which variable to exit and which variable to leave in the basis at zero level. One such rule could be to always exit the variable with the lowest subscript.

An "unbounded optimal solution" exists if there is no positive coefficient in any row for the entering variable when applying Criterion II. If this happens, the entering variable can be made arbitrarily large (resulting in $x_0$ growing without bound) and the current basic variables remain nonnegative.

The selection of an initial basis to begin the simplex algorithm may not be obvious. If each constraint $i$ is of the form

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad \text{where } b_i \geq 0$$

then adding a slack variable to each constraint will provide an all-slack initial basis. This was the case for the example problem used above to illustrate the algorithm. However, in general, the constraints are of the form
\[ \sum_{j=1}^{n} a_{ij}x_j = b_i \text{ for } i = 1, 2, \ldots, m \text{ where } b_i \geq 0 \]

Various techniques are available for getting an initial starting basis but the one used in the Simplex Algorithm Laboratory is to add an artificial variable to the constraints that require them. This gives

\[ \sum_{j=1}^{n} a_{ij}x_j + y_i = b_i \text{ for } i = 1, 2, \ldots, m \text{ where } b_i \geq 0 \text{ and } y_i \geq 0. \]

Then the \( y_i \) (artificial variables) are used as part of the initial basis. The optimal solution must not contain any of the \( y_i \) in the basis. The presence of any \( y_i \) in the basis at the final iteration indicates that a feasible solution does not exist for the problem. The steps involved in getting the starting basis for this technique will not be given here.

One final comment of a general nature before proceeding on to the sensitivity analysis. The simplex algorithm will always yield the exact optimal solution (ignoring roundoff error) in a finite number of iterations. The number of iterations required will be approximately 1.5 to 3 times the number of constraints involved.

**Sensitivity Analysis**

Sensitivity analysis, also called postoptimality analysis, uses the information available at the final iteration of the algorithm to answer such questions as: Does the current solution remain optimal if the profit contribution of a particular basic activity decreases?
What happens if a new activity is added? What happens to the current solution if resource availability is decreased?

The coverage in this thesis will be restricted to two areas: 1) Effects on the current optimal solution of varying the coefficients in the original objective function, and 2) Effects on the current optimal basis of varying the constants on the right-hand side of the constraint equations. The effects of altering the coefficients of the variables in the constraint equations are not covered; this analysis depends on concepts obtained from the duality principle which is beyond the scope of this presentation.

The final iteration Row 1 coefficients of the nonbasic variables represent the largest positive increments to the original objective function coefficients that leave the current solution optimal. For example on page 67 the final iteration Row 1 coefficient for \( x_4 \) is 1.57. This means that \( x_4 \) in the original objective function could be increased from 11 to 12.57 before the current basis would no longer be optimal. A larger increment than 1.57 would cause Criterion I to enter \( x_4 \) into the basis.

Analysis of the allowable ranges of the basic variables involves simultaneous consideration of Row 1 coefficients and increments of corresponding coefficients in the appropriate rows. The details of determining the allowable range of a basic variable will not be given. The idea is illustrated by considering \( x_1 \) from the problem represented
on page 67. The necessary calculations show that the incremental changes on \( x_1 \) must not be less than -0.600 and must not be greater than 2.20. This means that if \( x_1 \) was changed to less than 3.4 or changed to greater than 6.2 in the original objective function, then the current basis would no longer be optimal.

The technique of determining the allowable variations of the right-hand side constants will be shown by using a specific example. How much can the constant 15 on the right-hand side of constraint equation 1 (Row 2) in the original problem be altered before the current solution is no longer feasible? The first iteration on page 62 shows that slack variable \( x_5 \) was added to the constraint inequality to provide the equality. The slack variable \( x_5 \) in the final iteration on page 67 will assist in answering the question. The coefficient of \( x_5 \) in each Row of the final iteration is also the coefficient of the incremental change in the right-hand side value. The right-hand side value must remain nonnegative for each row. Rewriting the final iteration right-hand side values with the incremental change in constraint equation constant yields

\[
\begin{align*}
\text{Row 1:} & \quad 99.3 + 1.865 \geq 0 \\
\text{Row 2:} & \quad 7.14 + 1.436 \geq 0 \\
\text{Row 3:} & \quad 46.4 - 8.716 \geq 0 \\
\text{Row 4:} & \quad 7.86 - .429 \geq 0 
\end{align*}
\]
where $\delta$ is the allowable increment to the constant 15. Note that the coefficients of $\delta$ are the same as the coefficients of $x_5$ in the final iteration (page 67). Solving the above set of inequalities gives

$$-5 \leq \delta \leq 5.33.$$ 

This means the constant 15 may be reduced to 10 or increased to 20.33 before the current basis is no longer optimal.

The reader desiring a more thorough treatment of linear programming and the simplex algorithm should refer to Wagner (1969), Gass (1964), or any suitable textbook on the subject.
APPENDIX B

SAMPLE DIALOGUE
YOU MAY NOW SELECT A L.P. PROBLEM FOR SOLUTION
BY ENTERING THE PARAMETERS AS THEY ARE REQUESTED.
YOU HAVE TWO OPTIONS FOR ENTERING THE
PARAMETERS.  THE REGULAR METHOD IS A STEP
BY STEP RESPONSE THAT WILL INSURE ACCURACY.
THE EXPEDITED METHOD IS FASTER BUT REQUIRES
FAMILIARITY WITH THIS PROGRAM TO BE EFFECTIVE.
IT IS RECOMMENDED THAT YOU USE THE REGULAR METHOD FOR
THE FIRST FEW PROBLEMS.
SELECT THE OPTION YOU WANT.  TYPE (REG OR EXP)

REG

ENTER THE COEFFICIENTS FOR THE OBJECTIVE FUNCTION
VARIABLES AS THEY ARE REQUESTED BY TYPING A DECIMAL
NUMBER SUCH AS 12. OR +1.2000E+01

X 1= -4.0
X 2= -5.0
X 3= -9.0
X 4= -11.0
X 5= 0.0

MORE VARIABLES FOR OBJ. FUNCTION? TYPE (YES OR NO)

NO

TYPE THE COEFFICIENTS FOR THE VARIABLES IN
CONSTRAINT EQUATION 1.  USE A DECIMAL NUMBER.

X 1= 1.0
X 2= 1.0
X 3= 1.0
X 4= 1.0
X 5= 0.0

MORE VARIABLES FOR EQ. 1? TYPE (YES OR NO)

NO

ENTER CONSTRAINT EQ. 1 SENSE CODE.
TYPE (1 FOR <=, 2 FOR =, 3 FOR >=)

1

TYPE RIGHTHAND CONSTRAINT VALUE FOR CONSTRAINT
EQ. 1 USING DECIMAL NO. SUCH AS 13. OR +1.3000E+01

15.0

YES

MORE CONSTRAINT EQUATIONS? TYPE (YES OR NO)

YES

TYPE THE COEFFICIENTS FOR THE VARIABLES IN
CONSTRAINT EQUATION 2.  USE A DECIMAL NUMBER.

X 1= 7.0
X 2= 5.0
X 3= 3.0
X 4= 2.0
X 5= 0.0

MORE VARIABLES FOR EQ. 2? TYPE (YES OR NO)

NO

ENTER CONSTRAINT EQ. 2 SENSE CODE.
TYPE (1 FOR <=, 2 FOR =, 3 FOR >=)

1

TYPE RIGHTHAND CONSTRAINT VALUE FOR CONSTRAINT
EQ. 2 USING DECIMAL NO. SUCH AS 13. OR +1.3000E+01

120.0

YES

MORE CONSTRAINT EQUATIONS? TYPE (YES OR NO)

YES

TYPE THE COEFFICIENTS FOR THE VARIABLES IN
CONSTRAINT EQUATION 3.  USE A DECIMAL NUMBER.

X 1= 3.0
X 2= 5.0
X 3= 10.0
X 4= 15.0
X 5= 0.0

MORE VARIABLES FOR EQ. 3? TYPE (YES OR NO)

NO
ENTER CONSTRAINT EQ. 3 SENSE CODE.
TYPE (1 FOR <=, 2 FOR =, 3 FOR >=)

TYPE RIGHHTHAND CONSTRAINT VALUE FOR CONSTRAINT
EQ. 3 USING DECIMAL NO. SUCH AS 13. OR +1.3000E+01.

MORE CONSTRAINT EQUATIONS? TYPE (YES OR NO)

THE PROBLEM YOU HAVE JUST FINISHED ENTERING
WILL BE CONDENSED INTO A TABLE FOR YOUR EDITING.
NOTE THAT ROW 1 IS THE OBJECTIVE FUNCTION.

**************EDIT TABLE**************

<table>
<thead>
<tr>
<th>ROW 1</th>
<th>COL 1</th>
<th>COL 2</th>
<th>COL 3</th>
<th>COL 4</th>
<th>COL 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROW 1</td>
<td>-4.0000E+00</td>
<td>-5.0000E+00</td>
<td>-9.0000E+00</td>
<td>-1.1000E+01</td>
<td></td>
</tr>
<tr>
<td>ROW 2</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td></td>
</tr>
<tr>
<td>ROW 3</td>
<td>7.0000E+00</td>
<td>5.0000E+00</td>
<td>3.0000E+00</td>
<td>2.0000E+00</td>
<td></td>
</tr>
<tr>
<td>ROW 4</td>
<td>3.0000E+00</td>
<td>5.0000E+00</td>
<td>1.0000E+01</td>
<td>1.5000E+01</td>
<td></td>
</tr>
</tbody>
</table>

CONSTRAINT EQ. SENSE CODE RT.HAND SIDE
1     1     1     5000E+01
2     1     1     2000E+02
3     1     1     0000E+02

************** EDIT TABLE **************

ANY CORRECTIONS TO ABOVE NUMBERS? TYPE (YES OR NO)

IT NO BASE CURRENT IT NO BASE CURRENT
1 X 0 0.  X 1  X 2  X 3  X 4
X41 1.5000E+01 1.0000E+00 1.0000E+00 1.0000E+00 1.0000E+00
X42 1.2000E+02 7.0000E+00 5.0000E+00 3.0000E+00 2.0000E+00
X43 1.0000E+02 3.0000E+00 5.0000E+00 1.0000E+01 1.5000E+01

DO YOU WANT THE NEXT OR FINAL ITERATION?
TYPE (NEXT OR FINAL)

IT NO BASE CURRENT IT NO BASE CURRENT
2 X 0 7.3333E+01  X 1  X 2  X 3  X 4
X41 8.3333E+00 8.0000E-01 6.6667E-01 3.3333E-01 6.6667E-02
X42 1.0667E+02 6.6000E+00 4.3333E+00 1.6667E+00 1.3333E+01
X 4 6.6667E+00 2.0000E-01 3.3333E-01 6.6667E-01 6.6667E-02

DO YOU WANT THE NEXT OR FINAL ITERATION?
TYPE (NEXT OR FINAL)
<table>
<thead>
<tr>
<th>IT NO</th>
<th>CURRENT BASE VALUES</th>
<th>X41</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
<td>9.2083E+01</td>
<td>2.2500E+00</td>
<td>1.6667E-01</td>
<td>-9.1667E-01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0417E+01</td>
<td>1.2500E+00</td>
<td>6.3333E-01</td>
<td>4.1667E-01</td>
</tr>
<tr>
<td>X42</td>
<td></td>
<td>3.7917E+00</td>
<td>-8.2500E+00</td>
<td>-1.1667E+00</td>
<td>-1.0833E+00</td>
</tr>
<tr>
<td>X4</td>
<td></td>
<td>4.5833E+00</td>
<td>-2.5000E-01</td>
<td>1.6667E-01</td>
<td>5.8333E-01</td>
</tr>
</tbody>
</table>

DO YOU WANT THE NEXT OR FINAL ITERATION?  
TYPE (NEXT OR FINAL)

<table>
<thead>
<tr>
<th>IT NO</th>
<th>CURRENT BASE VALUES</th>
<th>X41</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td>9.9286E+01</td>
<td>1.8571E+00</td>
<td>4.2857E-01</td>
<td>1.5714E+00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7.1429E+00</td>
<td>1.4286E+00</td>
<td>7.1429E-01</td>
<td>-7.1429E-01</td>
</tr>
<tr>
<td>X42</td>
<td></td>
<td>4.6429E+00</td>
<td>-8.7143E+00</td>
<td>-8.5714E+00</td>
<td>1.8571E+00</td>
</tr>
<tr>
<td>X4</td>
<td></td>
<td>7.8571E+00</td>
<td>-4.2857E-01</td>
<td>2.8571E-01</td>
<td>1.7143E+00</td>
</tr>
</tbody>
</table>

DO YOU WANT THE NEXT OR FINAL ITERATION?  
TYPE (NEXT OR FINAL)

OPTIMAL SOLUTION---OBJECTIVE FUNCTION= 9.9286E+01

IF YOU WANT TO STOP, TYPE STOP  
IF YOU WANT TO CONTINUE WITH A NEW PROBLEM, TYPE GO  
IF YOU WANT TO ALTER THE ORIGINAL PROBLEM, TYPE CUR  
IF YOU WANT TO ALTER THE FINAL ITERATION, TYPE ITER  
IF YOU WANT TO CHECK SENSITIVITY, TYPE SEN

YOU MAY SELECT THREE CASES FOR SENSITIVITY ANALYSIS:  
1. EFFECTS OF VARYING THE COEFFICIENTS IN THE OBJECTIVE FUNCTION.  
2. EFFECTS OF VARYING THE CONSTRAINT CONSTANTS (RIGHTHAND SIDE VALUE).  
3. RESULTS OF SELECTED DISCRETE CHANGES IN ANY OF THE VARIABLES AND/OR CONSTANTS, IN ANY COMBINATION.

SELECT THE SENSITIVITY ANALYSIS CASE YOU WANT.  
TYPE (1, 2, OR 3)

<table>
<thead>
<tr>
<th>VAR=</th>
<th>01</th>
</tr>
</thead>
</table>

YOU SELECTED X 1 FOR ANALYSIS.  
IS THIS WHAT YOU WANTED? TYPE YES OR NO

YES

ANALYSIS----BASIC VARIABLE X 1 MAY RANGE FROM -6.2000E+00 TO -3.4000E+00 IN THE ORIGINAL OBJECTIVE FUNCTION AND STILL HAVE THE CURRENT BASIS OPTIMAL. HOWEVER, THE OBJECTIVE FUNCTION VALUE WILL RANGE FROM 9.5000E+01 TO 1.1500E+02 AS X 1 VARIES.

DO YOU WANT TO TRY ANOTHER OBJECTIVE FUNCTION VARIABLE?  
TYPE YES OR NO

YES
SELECT THE OBJECTIVE FUNCTION VARIABLE YOU WANT TO INVESTIGATE. TYPE A TWO DIGIT INTEGER SUCH AS 03 TO IDENTIFY THE VARIABLE.

VAR= 02

YOU SELECTED X 2 FOR ANALYSIS. IS THIS WHAT YOU WANTED? TYPE YES OR NO

YES

ANALYSIS-----NONBASIC VARIABLE X 2 MAY RANGE FROM ITS ORIGINAL OBJECTIVE FUNCTION VALUE OF -5.0000E+00 TO -5.4286E+00 AND STILL HAVE THE CURRENT SOLUTION OPTIMAL AT 9.9286E+01.

DO YOU WANT TO TRY ANOTHER OBJECTIVE FUNCTION VARIABLE? TYPE YES OR NO

NO

DO YOU WANT TO CONTINUE WITH SENSITIVITY ANALYSIS? TYPE YES OR NO

YES

SELECT THE SENSITIVITY ANALYSIS CASE YOU WANT. TYPE (1, 2, 3)

EQ NO= 02

ANALYSIS-----THE RIGHTHAND SIDE CONSTRAINT OF EQUATION 2 (ORIGINAL VALUE 1.2000E+02) MAY RANGE FROM 6.5000E+01 TO 1.7000E+02 AND STILL HAVE THE CURRENT BASE OPTIMAL, HOWEVER, THE OBJECTIVE FUNCTION VALUE WILL RANGE FROM 6.0000E+01 TO 1.3500E+02 AS THE RIGHTHAND SIDE VARIES.

DO YOU WANT TO SELECT ANOTHER EQUATION FOR ANALYSIS OF THE RIGHTHAND SIDE CONSTRAINT? TYPE YES OR NO

NO

DO YOU WANT TO CONTINUE WITH SENSITIVITY ANALYSIS? TYPE YES OR NO

YES

SELECT THE SENSITIVITY ANALYSIS CASE YOU WANT. TYPE (1, 2, 3)

3

YOU MAY SELECT ANY VARIABLES OR CONSTANTS (OR ANY COMBINATION THEREOF) FROM THE CURRENT PROBLEM AND DETERMINE THE EFFECTS ON THE SOLUTION OF A SPECIFIC DISCRETE CHANGE IN EACH. FOR EXAMPLE, YOU COULD CHANGE X2 IN CONSTRAINT EQ. 1, X3 IN CONSTRAINT EQ. 4 AND THE RIGHTHAND SIDE OF CONSTRAINT EQ. 2, ETC., AND DETERMINE HOW THIS EFFECTS THE SOLUTION.

TO CHANGE VARIABLE COEFFICIENTS, TYPE COEF TO CHANGE RIGHTHAND CONSTRAINT CONSTANTS, TYPE CONST

COEF

TYPE TWO DIGIT INTEGERS (03 ETC.) TO LOCATE THE COEFFICIENT IN THE EDIT TABLE ABOVE.

ROW= 01

COL= 03

NEW= -3.

YOU WILL CHANGE ROW 1, COL 3 TO -3.0000E+00 IS THIS WHAT YOU WANT? TYPE (YES OR NO)

YES

ANY MORE COEFFICIENT CHANGES? TYPE (YES OR NO)

NO

ANY RIGHTHAND SIDE CONSTANT CHANGES? TYPE (YES OR NO)

NO
YOU MAY SELECT THREE DISPLAY MODES FOR THE RESULTS
OF THE CHANGES YOU JUST MADE.
1. FULL TABLEAU FOR ALL ITERATIONS.
2. FULL TABLEAU FOR ONLY FINAL ITERATION.
3. VALUE OF OBJECTIVE FUNCTION AND LIST OF BASIS.

SELECT THE MODE YOU WANT. TYPE (1, 2, OR 3)

DETERMINED SOLUTION

IT NO BASE CURRENT VALUES X1 X2 X3 X4
1 X 0 0 -4.0000E+00 -5.0000E+00 -3.0000E+00 -1.1000E+01
X41 1.5000E+01 -1.0000E+00 1.0000E+00 1.0000E+00 1.0000E+00
X42 1.2000E+02 7.0000E+00 5.0000E+00 3.0000E+00 2.0000E+00
X43 1.0000E+02 3.0000E+00 5.0000E+00 1.0000E+01 1.5000E+01

DO YOU WANT THE NEXT OR FINAL ITERATION?
TYPE (NEXT OR FINAL)

NEXT

IT NO BASE CURRENT VALUES X1 X2 X3 X4
2 X 0 7.3333E+01 -1.8000E+00 -1.3333E+00 4.3333E+00 7.3333E-01
X41 8.3333E+00 8.0000E-01 6.6667E-01 3.3333E+00 -6.6667E+02
X42 1.0667E+02 6.6000E+00 4.3333E+00 1.6667E+00 -1.3333E+01
X 4 6.6667E+00 2.0000E-01 3.3333E-01 6.6667E-01 6.6667E-02

DO YOU WANT THE NEXT OR FINAL ITERATION?
TYPE (NEXT OR FINAL)

NEXT

IT NO BASE CURRENT VALUES X1 X2 X3 X4
3 X 0 9.2083E+01 2.2500E+00 1.6667E-01 5.0833E+00 5.8333E-01
X 1 1.0417E+01 1.2500E+00 8.3333E-01 4.1667E-01 -6.3333E+02
X42 3.7917E+01 -8.2500E+00 -1.1667E+00 -1.0833E+00 4.1667E-01
X 4 4.5833E+00 -2.5000E+00 1.6667E-01 5.8333E-01 8.3333E-02

DO YOU WANT THE NEXT OR FINAL ITERATION?
TYPE (NEXT OR FINAL)

NEXT

OPTIMAL SOLUTION---OBJECTIVE FUNCTION= 9.2083E+01

IF YOU WANT TO STOP, TYPE STOP
IF YOU WANT TO CONTINUE WITH A NEW PROBLEM, TYPE GO
IF YOU WANT TO ALTER THE ORIGINAL PROBLEM, TYPE CUR
IF YOU WANT TO ALTER THE FINAL ITERATION, TYPE ITER
IF YOU WANT TO CHECK SENSITIVITY, TYPE SEN

YOU MAY SELECT THREE CASES FOR SENSITIVITY ANALYSIS:
1. EFFECTS OF VARYING THE COEFFICIENTS IN THE
OBJECTIVE FUNCTION.
2. EFFECTS OF VARYING THE CONSTRAINT CONSTANTS
(RIGHTHAND SIDE VALUE).
3. RESULTS OF SELECTED DISCRETE CHANGES IN ANY OF
THE VARIABLES AND/OR CONSTANTS, IN ANY COMBINATION.

SELECT THE SENSITIVITY ANALYSIS CASE YOU WANT.
TYPE (1, 2, OR 3)
3

TO CHANGE VARIABLE COEFFICIENTS, TYPE COEF
TO CHANGE RIGHHAND CONSTRAINT CONSTANTS, TYPE CONST

CP-OF

TYPE TWO DIGIT INTEGERS (03 ETC.) TO LOCATE THE
COEFFICIENT IN THE EDIT TABLE ABOVE.

ROW= 01   COL= 03

NEW= -9.

TYPE A DECIMAL NO. (12. OR +1.2000E+01) FOR NEW VALUE.

YOU WILL CHANGE ROW 1, COL 3 TO -9.0000E+00
IS THIS WHAT YOU WANT? TYPE (YES OR NO)

YES

ANY MORE COEFFICIENT CHANGES? TYPE (YES OR NO)

NO

ANY RIGHHAND SIDE CONSTANT CHANGES? TYPE (YES OR NO)

YES

TYPE TWO DIGIT INTEGER (03 ETC.) FOR CONSTRAINT
EQUATION NUMBER FROM EDIT TABLE

EQ. NO. = 02

TYPE DECIMAL NUMBER (12. OR +1.2000E+01) FOR NEW VALUE
FOR RT. HAND SIDE OF CONSTRAINT EQ. 2.

NEW= 30.

YOU WILL CHANGE CONSTRAINT EQ. NUMBER 2
TO 3.0000E+01. IS THIS WHAT YOU WANT? TYPE (YES OR NO)

YES

ANY MORE CONSTRAINT CONSTANT CHANGES? TYPE (YES OR NO)

NO

ANY COEFFICIENT CHANGES? TYPE (YES OR NO)

NO

YOU MAY SELECT THREE DISPLAY MODES FOR THE RESULTS
OF THE CHANGES YOU JUST MADE:
1. FULL TABLEAU FOR ALL ITERATIONS.
2. FULL TABLEAU FOR ONLY FINAL ITERATION.
3. VALUE OF OBJECTIVE FUNCTION AND LIST OF BASIS.

SELECT THE MODE YOU WANT. TYPE (1, 2, OR 3)

2

IT  NO  BASE  CURRENT  X42  X 2  X 1  X43
    VALUES

  4   X 0  9.0000E+01  1.0000E+00  3.0000E+00  4.8000E+00  6.0000E-01
  X41  5.0000E+00 -2.0000E-01 -2.0000E-01 -5.2000E-01 -4.0000E-02
  X 3  1.0000E+01  6.0000E-01  2.6000E+00  3.9600E+00  6.0000E-02
  X 4  5.6843E-14  4.0000E-01  1.4000E+00  2.4400E+00  1.2000E-01

OPTIMAL SOLUTION—OBJECTIVE FUNCTION=  9.0000E+01

IF YOU WANT TO STOP, TYPE STOP
IF YOU WANT TO CONTINUE WITH A NEW PROBLEM, TYPE GO
IF YOU WANT TO ALTER THE ORIGINAL PROBLEM, TYPE CUR
IF YOU WANT TO ALTER THE FINAL ITERATION, TYPE ITER
IF YOU WANT TO CHECK SENSITIVITY, TYPE SEN
YOU MAY SELECT THREE CASES FOR SENSITIVITY ANALYSIS:
1. EFFECTS OF VARYING THE COEFFICIENTS IN THE OBJECTIVE FUNCTION.
2. EFFECTS OF VARYING THE CONSTRAINT CONSTANTS (RIGHHAND SIDE VALUE).
3. RESULTS OF SELECTED DISCRETE CHANGES IN ANY OF THE VARIABLES AND/OR CONSTANTS, IN ANY COMBINATION.

SELECT THE SENSITIVITY ANALYSIS CASE YOU WANT.
TYPE (1, 2, OR 3)

TO CHANGE VARIABLE COEFFICIENTS, TYPE COEF.
TO CHANGE RIGHHAND CONSTRAINT CONSTANTS, TYPE CONST.

TYPE TWO DIGIT INTEGER (03 ETC.) FOR CONSTRAINT EQUATION NUMBER FROM EDIT TABLE

EQ. NO. = 02
NEW = 120.

YOU WILL CHANGE CONSTRAINT EQ. NUMBER 2 TO 1.2000E+02. IS THIS WHAT YOU WANT? TYPE (YES OR NO)

ANY MORE CONSTRAINT CONSTANT CHANGES? TYPE (YES OR NO)

ANY COEFFICIENT CHANGES? TYPE (YES OR NO)

TYPE TWO DIGIT INTEGERS (03 ETC.) TO LOCATE THE COEFFICIENT IN THE EDIT TABLE ABOVE.

ROW = 03
COL = 02
NEW = 9.

YOU WILL CHANGE ROW 3, COL 2 TO 9.0000E+00.

IS THIS WHAT YOU WANT? TYPE (YES OR NO)

ANY MORE COEFFICIENT CHANGES? TYPE (YES OR NO)

ANY RIGHHAND SIDE CONSTANT CHANGES? TYPE (YES OR NO)

YOU MAY SELECT THREE DISPLAY MODES FOR THE RESULTS OF THE CHANGES YOU JUST MADE:
1. FULL TABLEAU FOR ALL ITERATIONS.
2. FULL TABLEAU FOR ONLY FINAL ITERATION.
3. VALUE OF OBJECTIVE FUNCTION AND LIST OF BASIS.

SELECT THE MODE YOU WANT. TYPE (1, 2, OR 3)

THE OPTIMAL SOLUTION = 9.9286E+01.
BASE IS X 1 42 3

IF YOU WANT TO STOP, TYPE STOP.
IF YOU WANT TO CONTINUE WITH A NEW PROBLEM, TYPE GO.
IF YOU WANT TO ALTER THE ORIGINAL PROBLEM, TYPE CUR.
IF YOU WANT TO ALTER THE FINAL ITERATION, TYPE ITER.
IF YOU WANT TO CHECK SENSITIVITY, TYPE SEN.

STOP
19.23.42.STOP
APPENDIX C

CONTROL PROGRAM LISTING
CCCCCCCCCCCC
C THIS IS THE MAIN PROGRAM FOR THE SIMPLEX ALGORITHM LABORATORY.
C TAPE1 IS INPUT FROM THE TERMINAL, TAPE2 IS OUTPUT TO THE TERMINAL,
C AND TAPE3 IS FILE DBLM2 CONTAINING THE TEXT DATA FOR INTRODUCTORY
C MATERIAL IN SECTION 1.
CCCCCCCCCCCC
PROGRAM DBLM(INPUT,OUTPUT,DBLM2,TAPE1=INPUT,TAPE2=OUTPUT,TAPE3=DBLM
IT2)
EXTERNAL SUB1
EXTERNAL SUB2
CCCCCCCCCCCC
C ARRAYS ARE AS FOLLOWS:
C X FOR STORING PROBLEM AS ENTERED
C REDUCE FOR STORING CONDENSED PROBLEM
C NONBASE FOR STORING NONBASIC VARIABLES
C BASE FOR STORING BASIC VARIABLES
C TEXT FOR STORING MATERIAL FOR SECTION 1.
CCCCCCCCCCCC
COMMON/COM1/X(21,81),REDUCE(21,61),NONBASE(60)
INTEGER CONSEN(21),SLACK(21),SURPLUS(21),ARTIF(21),BASE(21)
1 SLA,SLR,ART,PCONT,HLT
INTEGER TEXT(200,5),ANSWER(200),LINE(200,4),ANS
COMMON/COM2/CONSEN,SLACK,SURPLUS,ARTIF,BASE,SLA,SLR,ART,PCONT,HLT
COMMON/NORMS,NOCOLS,FINAL,ITNO
COMMON/COM4/NONB,ARTIF,HLCOMMON/COM5/ARTIF,KFLG
COMMON/S1/KS1,KS2,KEXP,IFOUR
DIMENSION FM1(18),FM11(14),FM12(16),FM13(17),NS1(6),NS2(6)
DATA NS1,0,0,0,0,0/
DATA NS2,0,0,0,0,0/
CCCCCCCCCCCC
C NEXT 15 STATEMENTS READS TEXT IN AND PRINTS IT OUT IN FRAMES.
DO 6105 I=1,173
6105 READ(3,6107) (TEXT(I,J),J=1,5),ANSWER(I),(LINE(I,K),K=1,4)
6107 FORMAT (6A10,4I5)
6109 ISTART=1
6110 FORMAT (61,6112) ((TEXT(I,J),J=1,5),I=ISTART,ISTOP)
6112 FORMAT (15X,5A10)
6114 FORMAT (/)
6116 NAME=(1,6116) ANS
6116 FORMAT (A10)
IF (ANS,GT,4HSKIP,OR,ISTOP,GE,173) GO TO 6000
IF (ANS,LT,4HSTOP) GO TO 405
6118 ISTART=ISTOP+1
6120 IF (IMAGE,I=ISTOP) GO TO 6010
6000 NS1(5) = LOCF(SUB1);
6000 NS2(5) = LOCF(SUB2);
6000 CALL SYSTEMC(78,NS1)
6000 CALL SYSTEMC(79,NS2)
6000 FORMAT(15X,*YOU HAVE TWO OPTIONS FOR ENTERING THE*,/
115X,*PARAMETERS. THE REGULAR METHOD IS A STEP*,/
115X,*BY STEP RESPONSE THAT WILL INSURE ACCURACY*,/
115X,*THE EXPEDITE METHOD IS FASTER BUT REQUIREFS*,/
115X,*FAMILIARITY WITH THIS PROGRAM TO BE EFFECTIVE*,/
115X,*IT IS RECOMMENDED THAT YOU USE THE REGULAR METHOD FOR*,/
115X,*THE FIRST FEW PROBLEMS,*
CCCCCCCCCCCC
C INITIALIZE
310 DO 100 I=1,3042
100 X(I)=0
DO 101 I=1,110
101 CONSEN(I)=0
ISHORT=0
IFOUR=0
KFLG=0
IPROB=0
KFLG=0
NROWS=0
NOCOLS=0
IFINAL=0
ITNO=0
WRITE(2,201)
WRITE (2,600)
IF (IPROB .GT. 0) WRITE (2,602)
602 FORMAT(15X,*YOU MAY NOW ENTER A NEW PROBLEM*)
603 WRITE (2,604)
604 FORMAT(15X,*SELECT THE OPTION YOU WANT. TYPE (REG OR EXP)*,/)CCC
C READ IN MODE FOR ENTERING DATA:
READ (1,219) IEXP
IF (IEXP.EQ. 3HEXP) GO TO 630
IF (IEXP .EQ. 3HREG) GO TO 320
IF (IEXP .EQ. 4HSTOP) GO TO 405
WRITE (2,608)
608 FORMAT(15X,*I DONT UNDERSTAND, TRY AGAIN*)
GO TO 603
320 WRITE(2,204)
324 N=0
TEND=0
CCC
C ASSIGN XXX TO KS1 AND KS2 ARE RETURN POINTS AFTER ERROR ROUTINE IF
C USER ENTERS INVALID FORMAT TYPES.
ASSIGN 326 TO KS1
ASSIGN 326 TO KS2
CCC
C NEXT 50 STATEMENTS ASKS USER TO ENTER PROBLEM PARAMETERS AND
C STORES THEM.
325 N=N*1
326 WRITE(2,205) N
READ(1,206) X(1,N)
IF(N .NE. 5 .AND. N .NE. 10 .AND. N .NE. 15 .AND. N .NE. 20 .AND.)
 1 N .NE. 25 .AND. N .NE. 30 .AND. N .NE. 35 .AND. N .NE. 40) GO TO
  1 325
IF(IPROB .EQ. 0 .AND. N .LT. 6) GO TO 210
WRITE(2,209)
GO TO 230
210 WRITE (2,211)
230 READ (1,202) IEND
IF (IEND .EQ. 2HNO .OR. N .EQ. 40) GO TO 330
GO TO 325
330 NOCONS = N
JEND = 0
335 NOCONS = NOCONS + 1
NOROWS = NOCONS + 1
WRITE (2,208) NOCONS
NN=0
IEND=0
ASSIGN 341 TO KS1
ASSIGN 341 TO KS2

340 NN=NN+1
341 WRITE (2,205) NN
READ (1*266) (X(NOROWS,NN))
IF (X(1,NN),EQ,5 AND NN,GE,10 AND NN,GE,15 AND NN,GE,20 AND NN,GE,1
AND NN,GE,25 AND NN,GE,30 AND NN,GE,35 AND NN,GE,40) GO TO 340
WRITE (2,209) NN
GO TO 343
342 WRITE (2,198) NOCONS
343 READ (1*202) IEND
IF (IEND,NE,200 OR IEND,NE,400) GO TO 345
GO TO 340
344 IF (IProb,NE,0) GO TO 347
WRITE (2,346) NOCONS
GO TO 348
345 ASSIGN 345 TO KS1
ASSIGN 345 TO KS2
IF (IProb,NE,0) GO TO 347
WRITE (2,349) NOCONS
GO TO 351
346 WRITE (2,212) NOCONS
347 WRITE (2,213) CONSENS(NOROWS)
ASSIGN 344 TO KS1
ASSIGN 344 TO KS2
348 IF (IProb,NE,0) GO TO 350
WRITE (2,349) NOCONS
GO TO 351
349 WRITE (2,214) NOCONS
350 READ (1*206) X(NOROWS,81)
WRITE (2,215)
351 CONTINUE
352 READ (1*202) IEND
IF (IEND,NE,200 OR IEND,NE,400) GO TO 6
IF (IEND,NE,400 AND STOP.) GO TO 405
GO TO 335

C CCCCCCCCCCCCCCCCC
C NEXT 10 STATEMENTS PRINTS OUT EDIT TABLE FOR USER TO CHECK
C PARAMETERS HE HAS ENTERED.
6 WRITE (2,250)
250 FORMAT (1X,15X,*THE PROBLEM YOU HAVE JUST FINISHED ENTERING*,115X,*WILL BE CONDENSED INTO A TABLE FOR YOUR EDITING*,115X,*NOTE THAT ROW 1 IS THE OBJECTIVE FUNCTION*,14(1X,6H*********),*EDIT TABLE*,4(1X,6H*********)/,)
DO 251 I=1,N
ILST=(N*1)-1
IF (X(I,ILST),EQ,0) GO TO 251
DO 259 I=1,NOROWS
WRITE (2,261) I,(X(I,J),J=KST,IJ)
259 CONTINUE
251 CONTINUE
253 KST=I
KON=5
254 WRITE (2,257) (J,J=KST,KON)
257 FORMAT (1I2X,*COL*,12,8(2X,*COL*,12))
IJ=ILST
IF (ILST,GT,KON) IJ=KON
DO 259 I=1,NOROWS
WRITE (2,261) I,(X(I,J),J=KST,IJ)
261 FORMAT(1X,*ROW*,12,5(2X,11E11,4))
259 CONTINUE
IF (ILST,LE,KON) GO TO 260
KST=KST+5
KON=KON+5
GO TO 254
WRITE (2,264)  
FORMAT (//,3X, 'CONSTRAINT EQophonc ',2X, 'SENSE CODE*2X,ART,HAND SIDE*'),)
DO 265 I=2,NOROWS
ILES=I-1
WRITE (2,267) ILES, CONSEN(I), X(I,81)
267 FORMAT (8X,12,13X,12,5X,E11,4)
265 CONTINUE
WRITE (2,1010)
1010 FORMAT (//, 8(1X,6H**」**))
WRITE (2,263)
263 FORMAT (///, 8X, 'ANY CORRECTIONS TO ABOVE NUMBERS? TYPE (YES OR NO)*',1,/) READ (1,202) ICOR
IF (ICOR, EQ, 3HYES) GO TO 500
IF (ICOR, EQ, 4HSTOP) GO TO 405
C CCCCCCCCCCCCCCCCCCCCCC
C NEXT 110 STATEMENTS ADDS SLACK, SURPLUS, AND ARTIFICIAL VARIABLES
C AND REARRANGES DATA INTO FORM FOR PRINTING AND USE BY SIMPLEX
C ALGORITHM*
270 XMAX=X(1,1)
PCONT = 0
DO 10 I=1,NOROWS
DO 10 J=1,40.
IF (XMAX,LT, X(I,J)) XMAX=X(I,J)
10 CONTINUE
BIGM=10.*XMAX
DO 30 I=2,NOROWS
IF (CONSEN(I), EQ, 2) I5,15,20,25
15 SLA=39+I
SLACK(I)=SLA
X(I,SLA)=1.
GO TO 30
25 SUR=39+I
SURPLUS(I)=SUR
X(I,SUR)=1.
20 ART=59+I
ARTIF(I)=ART
X(I,ART)=1.
X(I,ART)=BIGM
30 CONTINUE
DO 35 I=2,NOROWS
IF (ARTIF(I) *EQ. 0) GO TO 35
DO 34 J=1,81
34 X(I,J)=X(I,J) - BIGM*X(I,J)
35 CONTINUE
DO 40 I=2,NOROWS
IF (SLACK(I) *NE. 0) GO TO 36
BASE(I)=ARTIF(I)
GO TO 40
36 BASE(I)=SLACK(I)
40 CONTINUE
JNONB=0
DO 45 J=1,60
IF (X(I,J) *EQ. 0) GO TO 45
JNONB=JNONB + 1
NONBASE(JNONB)=J
45 CONTINUE
NOCOLS=JNONB
DO 50 K1=1,NOROWS
REDUCE(K1,11)=X(K1,81)
DO 50 K2=1,JNONB
   J=NONBASE(K2)
   REDUCE(K1,K2+1)=X(K1,J)
   CONTINUE
   ITNO=9
   HLT=0
   ENCODE(57,1404,FM10)
   ENCODE(133,1104,FM11) NOCONS
   ENCODE(57,1404,FM12)
   ENCODE(116,1004,FM13(7)) NOCONS
   ENCODE(98,1204,FM12(7)) NOCONS
   ENCODE(107,1304,FM13(7)) NOCONS
55   ITNO=ITNO+1
   IF (KFLG,GT,1) GO TO 3549
   IF (JNONB,GT,4) GO TO 62
   GO TO (65,64,63,62),JNONB
56   MORE = JNONB-4
   JSTART = 5
54   IF (MORE,GT,4) GO TO 60
   GO TO (57,58,59,60),MORE
57   INC=JSTART
   IN1 = INC + 1
   WRITE(2,FM11) (NONBASE(J),J=JSTART,INC),ITNO,BASE(I),REDUCE(1,1),
(REDUCE(I,J),J=JSTART,IN1), (BASE(I),REDUCE(I,J)), (REDUCE(I,J),J=JSTART,IN1),
1 I=2,NOROWS)
   GO TO 67
58   INC=JSTART+1
   IN1 = INC + 1
   WRITE(2,FM12) (NONBASE(J),J=JSTART,INC),ITNO,BASE(I),REDUCE(1,1),
(REDUCE(I,J),J=JSTART,IN1), (BASE(I),REDUCE(I,J)), (REDUCE(I,J),J=JSTART,IN1),
1 I=JSTAR1,IN1), I=2,NOROWS)
   GO TO 67
59   INC=JSTART+2
   IN1 = INC + 1
   WRITE(2,FM13) (NONBASE(J),J=JSTART,INC),ITNO,BASE(I),REDUCE(1,1),
(REDUCE(I,J),J=JSTART,IN1), (BASE(I),REDUCE(I,J)), (REDUCE(I,J),J=JSTART,IN1),
1 I=JSTAR1,IN1), I=2,NOROWS)
   IF (MORE,LE,4) GO TO 67
   MORE = MORE-4
   JSTART = JSTART+4
   JSTART = JSTART+1
   GO TO 54
62   WRITE(2,FM10) (NONBASE(J),J=1,4), ITNO,BASE(I),
(REDUCE(I,J),J=1,5), (BASE(I),REDUCE(I,J),J=1,5),
1 I=2,NOROWS)
   IF (JNONB,GT,4) GO TO 56
   GO TO 67
63   WRITE(2,FM13) (NONBASE(J),J=1,3), ITNO,BASE(I),
(REDUCE(I,J),J=1,4), (BASE(I),REDUCE(I,J),J=1,4),
1 I=2,NOROWS)
   GO TO 67
64   WRITE(2,FM12) (NONBASE(J),J=1,2), ITNO,BASE(I),
I (REDUCE(I,J), J=1,3), (BASE(I), REDUCE(I,J), J=1,3),
1 3) 1 i=2, NOROWS)
GO TO 67
65 WRITE(2,FM11) NONBASE(I), ITNO,BASE(I),
1 (REDUCE(I,J), J=1,2), (BASE(I), REDUCE(I,J), J=1,2),
1 2) i=2, NOROWS)
67 IF (PCONT NE 0) GO TO 360
68 WRITE (2,216)
READ (1,217) TDESIRE
IF (IDESIRE EQ 4HNEXT ) GO TO 355
IF (IDESIRE NE 4HSTOP ) GO TO 405
3549 IFINAL = 1
355 CALL SIMPLEX
CCC CCCC CCCC CCCC CCCC CCCC CCCC CCCC CCCC CCCC
C NEXT 12 STATEMENTS CONTROLS PRINTING OF TABLEAUS.
IF (KFLG ,EQ. 3) GO TO 360
IF (PCONT ,EQ. 0 ,OR. IFINAL ,EQ. 1) GO TO 53
360 GO TO (370,380,390,391), PCONT
370 IF (KFLG ,EQ. 3) GO TO 9001
WRITE (2,1005) REDUCE(1,1)
GO TO 400
380 WRITE (2,1006)
GO TO 400
390 WRITE (2,1007)
GO TO 400
391 IF (KFLG ,EQ. 3) GO TO 9004
WRITE (2,1008) REDUCE(1,1)
400 WRITE (2,218)
READ (1,219) NOW
IPROB = -IPROB = 1
ISHORT=1
IF (NOW ,EQ. 2HGO) GO TO 310
IF (NOW ,EQ. 3HCUR) GO TO 500
IF (NOW ,EQ. 4HITER) GO TO 900
IF (NOW ,EQ. 4HSTOP) GO TO 405
IF(NOW =EQ. 3HSEN .AND. (PCONT =EQ. 1 OR. PCONT =EQ. 4)) GO TO 40
11
IF(NOW =EQ. 3HSEN .AND. (PCONT =EQ. 2 OR. PCONT =EQ. 3)) GO TO 402
IF (NOW =EQ. 4HCRIT) GO TO 404
WRITE(2,410)
410 FORMAT (15X,*I DONT UNDERSTAND2*)
GO TO 400
402 FORMAT (2,403)
403 FORMAT('15X,*THE SENSITIVITY ANALYSIS DOES NOT APPLY TO THE CURRENT
15X,*SOLUTION SINCE IT IS NOT AN OPTIMAL SOLUTION,*')
GO TO 400
401 CALL SENSIT
IF (KFLG =GT. 0) GO TO 270
GO TO 400
404 CALL CRITIQ
405 STOP
CC CCCC CCCC CCCC CCCC CCCC
C NEXT 100 STATEMENTS ALLOWS USER TO CHANGE THE DATA IN THE EDIT TABLE.
500 WRITE (2,502)
502 FORMAT (15X,*TO CORRECT A VARIABLE COEFFICIENT, TYPE COEF*/,
115X,*TO CORRECT A CONSTRAINT EQUATION SENSE CODE, TYPE SENSE*/,
115X,*TO CORRECT A CONSTRAINT RIGHTHAND SIDE VALUE, TYPE RIGHT*/)
503 READ (1,219) IEDIT
IF (IEDIT =EQ. 4HCOEF) GO TO 508
IF (IEDIT =EQ. 5HSENSE) GO TO 540
IF (IEOT, EQ. 5HRIGHT) GO TO 560
IF (IEOT, EQ. 3HALL) GO TO 568
IF (IEOT, EQ. 4HSTOP) GO TO 405
WRITE (2,504)
504 FORMAT (15X,*I DONT UNDERSTAND>>/> TYPE COEF OR SENSE OR RIGHT*,/)
GO TO 503
508 IF (IEXP, EQ. 3HEXP) GO TO 516
WRITE (2,510)
510 FORMAT (15X,*TO CORRECT A VARIABLE COEFFICIENT YOU MUST LOCATE*,/)
115X,*THE NUMBER IN THE ABOVE TABLE BY ROW AND COLUMN. NOTE*,/)
115X,*THAT ROW 1 IS FOR THE OBJECTIVE FUNCTION, ROW 2 FOR*,/)
115X,*THE 1ST CONSTRAINT EQUATION, ETC. AND COL 1 IS FOR*,/)
115X,*VARIABLE 1, COL 2 FOR VARIABLE 2, ETC.*/)
512 WRITE (2,514)
514 FORMAT (15X,*TYPE THE ROW AND COLUMN NUMBERS AS REQUESTED*,/)
115X,*USE TWO DIGIT INTEGERS ONLY SUCH AS 03 OR 12 ETC./)
ASSIGN 516 TO KS1
ASSIGN 516 TO KS2
516 WRITE (2,518)
518 FORMAT (* ROW=*)
READ (1,520) KROW
520 FORMAT (12)
ASSIGN 5201 TO KS1
ASSIGN 5201 TO K21
5201 WRITE (2,522)
522 FORMAT (* COL=*)
READ (1,520) KCOL
IF (IEXP, EQ. 3HEXP) GO TO 525
WRITE (2,524) KROW, KCOL, X(KROW,KCOL)
524 FORMAT (15X,*YOU JUST IDENTIFIED ROW *12,* AND COL *12,*)
1* FOR *E11,4/*
115X,*IS THIS THE NUMBER YOU WANT TO CORRECT? TYPE YES OR NO*/)
READ (1,219) ICONFR
IF (ICONFR, EQ. 2HNO) GO TO 516
IF (ICONFR, EQ. 4HSTOP) GO TO 405
ASSIGN 525 TO KS1
ASSIGN 525 TO KS2
525 WRITE (2,526)
526 FORMAT (15X,*TYPE THE NEW NUMBER YOU WANT*/*) NEW=*)
READ (1,206) XNEW
WRITE (2,528) XNEW, KROW, KCOL
528 FORMAT (15X,*YOU HAVE ASKED FOR *E11,4,* TO BE ENTERED*/)
115X,*AT ROW *12,* AND COL *12,/)
115X,*IS THIS WHAT YOU WANT? TYPE YES OR NO*/)
READ (1,219) ICONFR
IF (ICONFR, EQ. 3HYES) GO TO 530
IF (ICONFR, EQ. 4HSTOP) GO TO 405
GO TO 525
530 X(KROW,KCOL)=XNEW
WRITE (2,532)
532 FORMAT (15X,*ANY MORE CORRECTIONS TO COEFFICIENTS? TYPE YES OR NO*
1*,/)
READ (1,219) ICONFR
IF (ICONFR, EQ. 3HYES) GO TO 516
IF (ICONFR, EQ. 4HSTOP) GO TO 405
WRITE (2,534)
534 FORMAT (15X,*TO CORRECT CONSTRAINT EQ. SENSE, TYPE SENSE*/)
115X,*TO CORRECT RIGHHAND SIDE VALUES, TYPE RIGHT*/)
115X,*IF NO MORE CORRECTIONS, TYPE ALL*/)
GO TO 503
540  WRITE (2,542)
ASSIGN 540 TO KS1
ASSIGN 540 TO KS2
542  FORMAT (15X, 'TYPE THE CONSTRAINT EQ. NUMBER AS TWO DIGIT INTEGER',
1/15X, 'SUCH AS 03 OR 15', 'EQ. = ',)
READ (1,520) KEQ
ASSIGN 5421 TO KS1
ASSIGN 5421 TO KS2
5421 WRITE (2,544)
544  FORMAT (15X, 'TYPE THE CORRECT CONSTRAINT SENSE CODE AS 1 OR 2 OR 3',
15X, 'CODE = ',)
READ (1,521) KCODE
WRITE (2,546) KEO, KCODE
546  FORMAT (15X, 'YOU HAVE REQUESTED CONSTRAINT EQ. TO HAVE CODE',
15X, 'IS THIS WHAT YOU WANTED? TYPE YES OR NO',)
READ (1,521) ICONFR
IF (ICONFR .EQ. 3) GO TO 548
IF (ICONFR .EQ. 4) GO TO 405
GO TO 540
548  CONSEN(KEO+1)=KCODE
WRITE (2,551)
551  FORMAT (15X, 'ANY MORE CONSTRAINT SENSE CORRECTIONS? TYPE YES OR NO',)
READ (1,521) ICONFR
IF (ICONFR .EQ. 3) GO TO 540
IF (ICONFR .EQ. 4) GO TO 405
WRITE (2,550)
550  FORMAT (15X, 'TO CORRECT RIGHHAND SIDE VALUES, TYPE RIGHT',
15X, 'IS THERE ARE NO MORE CORRECTIONS, TYPE ALL',)
GO TO 503
560  WRITE (2,542)
ASSIGN 560 TO KS1
ASSIGN 560 TO KS2
READ (1,520) KEQ
ASSIGN 5602 TO KS1
ASSIGN 5602 TO KS2
5602 WRITE (2,562)
562  FORMAT (15X, 'TYPE THE CORRECTED RIGHHAND VALUE',
15X, 'VAL = ',)
READ (1,526) XRT
WRITE (2,564) XRT, KEQ
564  FORMAT (15X, 'YOU HAVE ASKED FOR ' E11, 4, TO RE THE ',
15X, 'RIGHHAND VALUE FOR EQ. ',
15X, 'IS THIS WHAT YOU WANT? TYPE YES OR NO',)
READ (1,521) ICONFR
IF (ICONFR .EQ. 3) GO TO 565
IF (ICONFR .EQ. 4) GO TO 405
GO TO 566
565  X(KEQ+1)=XRT
WRITE (2,566)
566  FORMAT (15X, 'ANY MORE RIGHHAND SIDE CORRECTIONS? TYPE YES OR NO',
2/)
READ (1,521) ICONFR
IF (ICONFR .EQ. 3) GO TO 560
IF (ICONFR .EQ. 4) GO TO 405
WRITE (2,568)
568  FORMAT (15X, 'ANY OTHER CORRECTIONS? TYPE YES OR NO',
2/)
READ (1,521) ICONFR
IF (ICONFR .EQ. 3) GO TO 500
IF (ICONFR.EQ.4HSTOP) GO TO 405
WRITE (2,1012)
1012 FORMAT(15X,*>DO YOU WANT A REVISED TABLE PRINTED*,/)
115X,*>SHOWING THE CORRECTIONS* TYPE(YES OR NO)*,/
READ (1,219) ITAB
IF (ITAB.EQ.3HYES) GO TO 6
IF (ITAB.EQ.4HSTOP) GO TO 405
WRITE (2,1014)
1014 FORMAT(15X,*>YOU SHOULD PENCIL IN YOUR CORRECTIONS*,/
115X,*>ON THE ABOVE TABLE*)
GO TO 270

C CCCCCCCCCCCCCCCCCC
C NEXT 26 STATEMENTS ARE THE EXPEDITE ROUTINE FOR ENTERING DATA*
630 WRITE (2,632)
ASSIGN 630 TO KS1
ASSIGN 630 TO KS2
632 FORMAT(15X,*>TYPE NO. OF VARIABLES IN OBJ. FUNCTION (03 ETC)*,/
READ (1,520) N
WRITE (2,634)
634 FORMAT(15X,*>TYPE ALL COEFFICIENTS OF OBJ. FUNCTION IN ORDER*,/
115X,*>DEPRESS RETURN KEY BEFORE STARTING*)
ASSIGN 6341 TO KS1
ASSIGN 6341 TO KS2.
6341 READ (1,206) (X(I,J), J=1,N)
6342 WRITE (2,638)
638 FORMAT(15X,*>TYPE NO. OF CONSTRAINT EQUATIONS (03 ETC)*,/
ASSIGN 6342 TO KS1
ASSIGN 6342 TO KS2
READ (1,520) NOCONS
-NOROWS=-NOCONS-1
DO 650 I=2,NOROWS
ICONS=I-1
WRITE (2,640) ICONS
640 FORMAT(15X,*>TYPE COEF., SENSE CODE(1 IS <=, 2 IS =, 3 IS >=),*,/
115X,*>AND RT. HAND SIDE, IN ORDER, FOR CONST. EQ. *,/2*,/
115X,*>DEPRESS RETURN KEY BEFORE STARTING*)
ASSIGN 6401 TO KS1
ASSIGN 6401 TO KS2
6401 READ (1,206) (X(I,J), J=1,N)
ASSIGN 6402 TO KS1
ASSIGN 6402 TO KS2
6402 READ (1,213) CONSEN(I)
ASSIGN 6403 TO KS1
ASSIGN 6403 TO KS2
6403 READ (1,206) X(I,81)
650 CONTINUE
GO TO 6

C CCCCCCCCCCCCCCCCCC
C NEXT 20 STATEMENTS ARE ROUTINE FOR ALTERING FINAL ITERATION
C TABLEAU.
900 IF (IEXP.EQ.3HEXP) GO TO 904
WRITE (2,902)
902 FORMAT(15X,*>YOU MAY ALTER THE NUMBERS IN THE FINAL*,/
115X,*>ITERATION BY IDENTIFYING THE APPROPRIATE ROW AND*,/
115X,*>COLUMN. YOU MUST DETERMINE THE ROW AND COLUMN*,/
115X,*>NUMBERS BY COUNTING SEQUENTIALLY*)
904 WRITE (2,905)
905 FORMAT(15X,*>TYPE TWO DIGIT INTEGERS (03 ETC.) FOR ROW AND COL*,/
115X,*>ROW=*)
ASSIGN 904 TO KS1
ASSIGN 904 TO KS2
READ (1,520) ITROW
WRITE (2,906)
906 FORMAT (* COL=*
ASSIGN 9051 TO KS1
ASSIGN 9051 TO KS2
READ (1,520) ITCOL
WRITE (2,908) ITROW,ITCOL,REDUCE(ITROW,ITCOL)
908 FORMAT(15X,*ROW * ,12,*,AND COL * ,12,*,LOCATES * ,E11.4,*),/
115X,*IS THIS WHAT YOU WANT TO ALTER? TYPE YES OR NO*),/
READ (1,219) ICONFR
IF (ICONFR *EQ. 3HYES) GO TO 910
IF (ICONFR *EQ. 4HSTOP) GO TO 405
GO TO 904

910 WRITE (2,912)
912 FORMAT(15X,*TYPE NEW VALUE YOU WANT TO ENTER*),/
1* NEW=*)
ASSIGN 910 TO KS1
ASSIGN 910 TO KS2
READ (1,206) REDUCE(ITROW,ITCOL)
WRITE (2,914)
914 FORMAT(15X,*ANY OTHER ALTERATIONS? TYPE YES OR NO*),/
READ (1,219) ICONFR
IF (ICONFR *EQ. 3HYES) GO TO 904
GO TO 53

CCCCCCCCCCCCCCCCCCCC
C NEXT 4 STATEMENTS CONTROL PRINTING OF CONDENSED DATA DURING
C SENSITIVITY PRINTOUT.
9001 WRITE(2,9003)REDUCE(1,1),(BASE(I),I=2,NOROWS)
9003 FORMAT(15X,*THE OPTIMAL SOLUTION = * ,E11.4,*,*,/,
115X,*BASE IS X * ,2013)
GO TO 400

9004 WRITE(2,9006)REDUCE(1,1),(BASE(I),I=2,NOROWS)
9006 FORMAT(15X,*AN ALTERNATIVE OPTIMAL SOLUTION = * ,E11.4,*,*,/,
115X,*BASE IS X * ,2013)
GO TO 400

198 FORMAT(15X,*MORE VARIABLES FOR EQ. * ,12,*, TYPE(YES OR NO*) ),/
201 FORMAT(15X,*YOU MAY NOW SELECT A LP PROBLEM FOR SOLUTION*),/
115X,*BY ENTERING THE PARAMETERS AS THEY ARE REQUESTED.*)
202 FORMAT(A10)
204 FORMAT(15X,*ENTER THE COEFFICIENTS FOR THE OBJECTIVE FUNCTION*),/
115X,*VARIABLES AS THEY ARE REQUESTED BY TYPING A DECIMAL*),/
115X,*NUMBER SUCH AS 12. OR +1.2000E+01* )
205 FORMAT(* X12.=*)
206 FORMAT (E11.4)
208 FORMAT(15X,*TYPE THE COEFFICIENTS FOR THE VARIABLES IN*),/
115X,*CONSTRAINT EQUATION * ,12,*, USE A DECIMAL NUMBER*,*)
209 FORMAT(15X,*MORE VARIABLES? TYPE (YES OR NO*) ),/
211 FORMAT(15X,*MORE VARIABLES FOR OBJ. FUNCTION? TYPE (YES OR NO*) ),/
212 FORMAT(15X,*ENTER CONSTRAINT EQ. * ,12,*, SENSE CODE * * ),/
115X,*TYPE (1 FOR <=, 2 FOR =, 3 FOR >=) * * )
213 FORMAT (I1)
214 FORMAT(15X,*TYPE RIGHHTHAND CONSTRAINT VALUE FOR CONSTRAINT*,/
115X,*EQ. * ,12,*, USING DECIMAL NO. SUCH AS 13. OR +1.2000E+01,*)
215 FORMAT(15X,*MORE CONSTRAINT EQUATIONS? TYPE (YES OR NO*) ),/
216 FORMAT(/,15X,*DO YOU WANT THE NEXT OR FINAL ITERATION?*),/
115X,*TYPE (NEXT OR FINAL) * * )
217 FORMAT (AS)
218 FORMAT(/,15X,*IF YOU WANT TO STOP, TYPE STOP*),/
115X,*IF YOU WANT TO CONTINUE WITH A NEW PROBLEM, TYPE GO*,
115X * IF YOU WANT TO ALTER THE ORIGINAL PROBLEM, TYPE CUR*, /*
115X * IF YOU WANT TO ALTER THE FINAL ITERATION, TYPE ITER*, */
115X * IF YOU WANT TO CHECK SENSITIVITY, TYPE SFR*, */
115X * IF YOU WANT CRITIQUE AND EVAL. SECTION, TYPE CRIT*, */
219 FORMAT (10X)
346 FORMAT(15X * TYPE EQ. *, I2, *, SENSE CODE (1 IS <=, 2 IS =, 3 IS >=)*, */
349 FORMAT (15X * TYPE DECIMAL NO. FOR RT. HAND VALUE OF EQ. *, I2, */
1004 FORMAT (55H 18X * X*, I2, 3 (1X*X, I2), /, 1X, I2, 2X, x*, I2, 5 (2X, E11.4), I2, */
1005 FORMAT (/, 15X * OPTIMAL SOLUTION — OBJECTIVE FUNCTION = *, E11.4)
1006 FORMAT (/, 15X * NO FEASIBLE SOLUTION*, */
1007 FORMAT (/, 15X * UNBOUND OPTIMAL SOLUTION*, */
1008 FORMAT (/, 15X * ALTERNATIVE OPTIMAL SOLUTION — OBJECTIVE FUNCTION = *, */
E11.4)
1204 FORMAT (55H 18X * X*, I2, 1 (1X*X, I2), /, 1X, I2, 2X, x*, I2, 2 (2X, E11.4), I2, */
132H (5X, X*, I2, 2X, E11.4, 2X, E11.4, 2X, E11.4))
END
CCCC CCCCC CCCCC
C THIS SUBROUTINE APPLIES SIMPLEX ALGORITHM TO PROBLEM ENTERED
C BY USER AND FINDS SOLUTION,
C IF FINAL USED AS FLAG TO CONTROL PRINTING OF EACH ITERATION OR
C ONLY FINAL ITERATION.
C SUBROUTINE SIMPLEX
COMMON/COM1/X (21, 21), REDUCE (21, 60), NONBASE (60)
INTEGER CONSEN (21), SLACK (21), SURPLUS (21), ARTIF (21), BASE (21), */
SLA, SUR, ART, PCONT, HLT
INTEGER NEG (41)
REAL MINRAT
COMMON/COM2/CONSEN, SLACK, SURPLUS, ARTIF, BASE, SLA, SUR, ART, PCONT, HLT
COMMON/COM3/NOROWS, NOCOLS, IFINAL, ITNO
COMMON/COM4/JNONS, KFLG
68 DO 63 I=2, NOCOLS
63 NEG(I)=0
66 ROWMIN = REDUCE (1, 2)
JCOLMN=2
64 DO 70 J=2, NOCOLS
IF (NEG(J)) EQ. -1) GO TO 70
IF (ROWMIN =. LE. REDUCE (1, J)) GO TO 70
ROWMIN = REDUCE (1, J)
JCOLMN= J
70 CONTINUE
IF (ROWMIN .EQ. 0) GO TO 100
INROWMIN = REDUCE (1, J)
J COLMN= J
70 CONTINUE
IF (REDUCE (1, JCOLMN) .GE. 0. AND REDUCE (I, 1), GE. 0) GO TO 74
72 CONTINUE
GO TO 50
74 MINRAT = REDUCE (IPOS, 1) / REDUCE (IPOS, JCOLMN)
IROH=IPOS
IPOS=IPOS+1
DO 75 I=IPOS,NOROWS
IF (REDUCE(I,JCOLMN) .LE. 0 .OR. REDUCE(I,1) .LT. 0) GO TO 75
RATIO=REDUCE(I,1)/REDUCE(I,JCOLMN)
IF (MINRAT .LE. RATIO) GO TO 75
MINRAT=RATIO
IROW=I
CONTINUE
75 CONTINUE
69 PIVIT = REDUCE(IROW,JCOLMN)
DO 76 J=1,NOCOLS
REDUCE(IROW,J)=REDUCE(IROW,J)/PIVIT
DO 80 I=1,NOROWS
IF (I.EQ.IROW) GO TO 80
DO 79 J=1,NOCOLS
IF (J.EQ.JCOLMN) GO TO 79
REDUCE(I,J)=REDUCE(I,J)-REDUCE(IROW,J)*REDUCE(I,JCOLMN)
CONTINUE
79 CONTINUE
REDUCE(I,JCOLMN) = - REDUCE(I,JCOLMN) / PIVIT
CONTINUE
80 CONTINUE
REDUCE(IROW,JCOLMN) = 1 / PIVIT
JTEMP=NONBASE(JCOLMN-1)
NONBASE(JCOLMN-1)=BASE(IROW)
BASE(IROW) = JTEMP
ITNO = ITNO + 1
IF (IFINAL.EQ.0) RETURN
GO TO 68
71 DO 90 I=2,NOROWS
90 IF (BASE(I) .GT. 60) GO TO 91
PCONT = 1
RETURN
91 PCONT = 2
RETURN
73 PCONT = 3
RETURN
100 PCONT=4
RETURN
50 IF (REDUCE(IPOS,1) .LT. 0) GO TO 52
GO TO 73
52 NEG(JCOLMN) = -1
DO 54 II=2,NOCOLS
IF (NEG(II) .EQ. -1) GO TO 54
ROWMIN=REDUCE(II,II)
JCOLMN=II
GO TO 64
54 CONTINUE
GO TO 91
END

CCCCCCCCCCCCCCCCCCCCCC
C THIS SUBROUTINE PROVIDES THE SENSITIVITY ANALYSIS FOR PROBLEM C USER HAS ENTERED AND PROGRAM HAS SOLVED.
SUBROUTINE SENSIT
EXTERNAL SUB1
EXTERNAL SUB2
COMMON/COM1/X(21,31),REDUCE(21,61),NONBASE(60)
INTEGER CONSEN(21),SLACK(21),SURPLUS(21),ARTIF(21),BASE(21)
1 SLA,ST,ART,PCONT,HLT
COMMON/COM2/CONSEN,SLACK,SURPLUS,ARTIF,BASE,SLA,ST,ART,PCONT,HLT
COMMON/COM3/NOROWS,NOCOLS,IFINAL,ITNO
COMMON/COM4/JNONB,KFLG
COMMON/Sl,KS1,KS2,IESP,IFOUR
DIMENSION FM1(18),FM11(14),FM12(16),FM13(17),NSI(6),NS2(6)
DATA NSI('0',0,0,0,0,0/)
DATA NS2('0',0,0,0,0,0/)
NSI(5) = LORF(SUB1)
NS2(5) = LORF(SUB2)
CALL SYSTEMC(78,NSI)
CALL SYSTEMC(79,NS2)
KFLG=0
WRITE(2,720)
720 FORMAT(15X,'YOU MAY SELECT THREE CASES FOR SENSITIVITY ANALYSIS,*
1/115X,*1. EFFECTS OF VARYING THE COEFFICIENTS IN THE*,/
115X,*2. EFFECTS OF VARYING THE CONSTRAINT CONSTANTS*/*
115X,*3. RESULTS OF SELECTED DISCRETE CHANGES IN ANY OF*,/*
115X,*THE VARIABLES AND/OR CONSTANTS, IN ANY COMBINATION*/*)
WRITE(2,721)
721 FORMAT(15X,'SELECT THE SENSITIVITY ANALYSIS CASE YOU WANT,*/*),
115X,*TYPE (1, 2, OR 3)*/*)
READ(1,724) KASE
724 FORMAT(A10)
IF (KASE .EQ. '1H1') GO TO 728
IF (KASE .EQ. '1H2') GO TO 770
IF (KASE .EQ. '1H3') GO TO 900
IF (KASE .EQ. '4HSTOP') RETURN
WRITE(2,726)
726 FORMAT(15X,'I D O  N T  UNDERSTAND?')
GO TO 721
CCCCCCCCCCCCCCCCCCC
C NEXT 60 STATEMENTS ARE OBJECTIVE FUNCTION ANALYSIS.
728 WRITE(2,730)
730 FORMAT(15X,*SELECT THE OBJECTIVE FUNCTION VARIABLE*/*),
115X,*YOU WANT TO INVESTIGATE. TYPE A TWO DIGIT INTEGER*/*,
115X,*SUCH AS 03 TO IDENTIFY THE VARIABLE*/*,
115X,*VAR=*/*)
IF(FOUR=0)
ASSIGN 728 TO KS1
ASSIGN 728 TO KS2
READ(1,732) KVAR
732 FORMAT(I2)
WRITE(2,734) KVAR
734 FORMAT(15X,*YOU SELECTED X*/*IX*/ FOR ANALYSIS*/*),
115X,*IS THIS WHAT YOU WANTED? TYPE YES OR NO*/*)
READ(1,724) KONFR
IF (KONFR .EQ. '3HYES') GO TO 736
IF (KONFR .EQ. '4HSTOP') RETURN
GO TO 728
736 DO 740 I=1,JNONB
KNON = I
IF (NONBASE(KNON) .EQ. KVAR) GO TO 748
740 CONTINUE
DO 742 I=2,NORWS
KHSE = I
IF (BASE(KHSE) .EQ. KVAR) GO TO 760
742 CONTINUE
WRITE(2,744) KVAR
744 FORMAT(15X,*X*/IX*/, HAS NO MEANING FOR THIS PROBLEM*/*),
115X,*DO YOU WANT TO TRY ANOTHER VARIABLE? TYPE YES OR NO*/*)
745 READ(1,724) KONFR
IF (KONFR .EQ. 3*YES) GO TO 728
IF (KONFR .EQ. 4*HSTOP) RETURN

7451 WRITE (2,746)
746 FORMAT (15X,'DO YOU WANT TO CONTINUE WITH SENSITIVITY ANALYSIS?/

115X,'TYPE YES OR NO?/

747 READ (1,724) KONFR
IF (KONFR .EQ. 3*YES) GO TO 721
IF (KONFR .EQ. 4*HSTOP) RETURN
KFLG = 1
RETURN

748 IF (NONBASE(KNON) .LE. 40) GO TO 752
WRITE (2,750) NONBASE(KNON)
GO TO 756

752 OMAX = X(1,KVAR) - REDUCE(1,KNON+1)
WRITE (2,754) NONBASE(KNON), X(1,KVAR), OMAX, REDUCE(1,1)

754 FORMAT (15X,'ANALYSIS----NONBASIC VARIABLE X* IS MAY RANGE /

115X, FROM ITS ORIGINAL OBJECTIVE FUNCTION VALUES /

115X, OF *E11,4* TO *E11,4* AND STILL /

115X, HAVE THE CURRENT SOLUTION OPTIMAL AT *E11,4*

756 WRITE (2,758)
758 FORMAT (15X,'DO YOU WANT TO TRY ANOTHER OBJECTIVE FUNCTION VARIAR /

1LE?/

115X,'TYPE YES OR NO?/

GO TO 745

760 IF (BASE(KBSE) .LE. 40) GO TO 761
WRITE (2,750) BASE(KBSE)
GO TO 756

761 TESTN = 1.E+99
TESTP = =1.E-99
DO 764 I=2,NOCOLS
IF (REDUCE(KBSE,I) .EQ. 0) GO TO 764
IF (REDUCE(1,I) .EQ. 0) GO TO 763
TEST = -REDUCE(1,I) / REDUCE(KBSE,I)
IF (TEST .LT. 0) GO TO 762
IF (TEST .GE. TESTN) GO TO 764
TESTN = TEST
JNTST = I
GO TO 764

762 IF (TEST .LE. TESTP) GO TO 764
TESTP = TEST
JPTST = I
GO TO 764

763 TESTP = 0.
JPTST = I

764 CONTINUE
TESTL = X(1,KVAR) - TESTN
TESTH = X(1,KVAR) - TESTP
OBJL = REDUCE(1,1) + REDUCE(KBSE,1) * TESTP
OBJH = REDUCE(1,1) + REDUCE(KBSE,1) * TESTN
WRITE (2,766) BASE(KBSE), TESTL, TESTH, OBJL, OBJH, BASE(KBSE)

766 FORMAT (15X,'ANALYSIS----BASIC VARIABLE X* MAY RANGE /

115X, FROM *E11,4* TO *E11,4* AS X* VARIES/

767 C NEXT 60 STATEMENTS ARE FOR RT HAND CONSTANT ANALYSIS.
770 WRITE (2,772)
FORM(15X.*,SELECT THE CONSTRAINT YOU WANT TO INVESTIGATE.*,/, 115X.*,REFER BACK TO THE EDIT TABLE TO IDENTIFY THE.*,/, 115X.*,CONSTRAINT EQUATION NUMBER AND THE RIGHHAND SIDE VALUE.*,/, 115X.*,TYPE THE EQ. NUMBER AS A TWO DIGIT INTEGER SUCH AS 04.*,/, 1* EQ NO=*)
IF(FOUR=)
   ASSIGN 776 TO KS1
   ASSIGN 776 TO KS2
   READ (1,732) KEQ
   IF (KEQ .LE. NOROWS-1) GO TO 778
   WRITE (2,774) KEQ
FORM(15X.*,THERE IS NO CONSTRAINT EQ.*I2,* FOR THIS PROBLEM.*,)
FORM(15X.*,DO YOU WANT TO SELECT ANOTHER EQUATION FOR ANALYSIS*  
1.*/15X.*,OF THE RIGHHAND SIDE CONSTRAINT? TYPE YES OR NO.*,)
READ (1,724) KONFR
   IF (KONFR .EQ. 3*YES) GO TO 770
   IF (KONFR .EQ. 4*STOP) RETURN
   WRITE (2,746)
   GO TO 747
   WRITE (2,776)
778 DO 782 I=I,J,NOROWS
   KCOLB = I + 1
   IF (NONBASE(I) .EQ. (40 + KEQ)) GO TO 790
   IF (NONBASE(I) .EQ. (60 + KEQ)) GO TO 790
   CONTINUE
DO 786 I=2,NOROWS
   KROWB = I
   IF (BASE(I) .EQ. (39 + KEQ)) GO TO 800
   IF (BASE(I) .EQ. (59 + KEQ)) GO TO 806
   CONTINUE
790 RTESTP = J,E*99
   RTESTN = -1.*E+99
   IF (REDUCE(I+1) .EQ. 0.) GO TO 793
   IF (REDUCE(I*KCOLB) .EQ. 0.) GO TO 794
   DO 794 I=2,NOROWS
   RTEST = -REDUCE(I+1) / REDUCE(I*KCOLB)
   IF (RTEST .GT. 0.) GO TO 792
   IF (RTEST .GE. RTESTP) GO TO 794
   RTESTP = RTEST
   RJPTST = I
   GO TO 794
792 IF (RTEST .LE. RTESTN) GO TO 794
   RTESTN = RTEST
   RJNTST = I
   GO TO 794
793 RTESTN = 0.
   RJNTST = I
   CONTINUE
794 RTESTL = X(KEQ+1,81) + RTESTN
   RTESTH = X(KEQ+1,91) + RTESTP
   ROBJL = REDUCE(I+1) + REDUCE(I*KCOLB) * RTESTN
   ROBJH = REDUCE(I+1) + REDUCE(I*KCOLB) * RTESTP
   WRITE(2,798) KEQ,X(KEQ+1,81),RTESTL,RTESTH,ROBJL,ROBJH
FORM(15X.*,ANALYSIS-----THE RIGHHAND SIDE CONSTRAINT OF.*,/)  
115X.*,EQUATION *I2,* (ORIGINAL VALUE *E11,4,*) R/1/,  
115X.*,MAY RANGE FROM *E11,4,0 TO *E11,4,9 AND STILL R/1/,  
115X.*,HAVE THE CURRENT BASE OPTIMAL, HOWEVER,*R/1/,  
115X.*,THE OBJECTIVE FUNCTION VALUE WILL RANGE FROM*,/)  
115X.,*E11,4,0 TO *E11,4,9 AS THE RIGHHAND SIDE VARIES.*)
GO TO 775
RTESTN = REDUCE(KROW,1)
RTESTL = X(KEQ+1,81) * RTESTN
WRITE (2,802) KEQ,X(KEQ+1,81),RTESTL

802 FORMAT(15X,ANALYSIS---THE RIGHHAND SIDE CONSTRAINT OF /, 115X,EQUATION *12,9 (ORIGINAL VALUE *111.4,5)*/, 115X,MAY RANGE FROM *111.4,5 TO AN UNLIMITED UPPER BOUND*)
GO TO 775

806 WRITE (2,808)
808 FORMAT(15X,THE CURRENT SOLUTION YOU ARE USING IS NOT A*/,
115X,FEASIBLE SOLUTION, THEREFORE ANY ANALYSIS IS NOT VALID*)
GO TO 775

CCCCCCCCCCCCCCCCCCCCC
C NEXT 70 STATEMENTS FOR DISCRETE CHANGE ANALYSIS.
900 IF (IFOUR .GT. 0) GO TO 904
WRITE (2,902)

902 FORMAT(15X,YOU MAY SELECT ANY VARIABLES OR CONSTANTS (OR ANY*/,
115X,COMBINATION THEREOF) FROM THE CURRENT PROBLEM*/,
115X,AND DETERMINE THE EFFECTS ON THE SOLUTION OF A SPECIFIC*/,
115X,DISCRETE CHANGE IN EACH. FOR EXAMPLE, YOU COULD CHANGE X2*/,
115X,IN CONSTRAINT EQ. 1, X3 IN CONSTRAINT EQ. 4 AND*/,
115X,THE RIGHHAND SIDE OF CONSTRAINT EQ. 2, ETC., AND*/,
115X,DETERMINE HOW THIS EFFECTS THE SOLUTION*/)
IFOUR=1
IF (IFOUR .GT. 0) GO TO 904
WRITE (2,906)

906 FORMAT(15X,TO CHANGE VARIABLE COEFFICIENTS, TYPE COEF*/,
115X,TO CHANGE RIGHHAND CONSTRAINT CONSTANTS, TYPE CONST*/)
READ (1,724) KONFR
IF (KONFR .EQ. 4HCOEF) GO TO 910
IF (KONFR .EQ. 5HCONST) GO TO 938
IF (.NOT.KONFR .EQ. 4HSTOP) RETURN
WRITE (2,908)

908 FORMAT(15X,I DONT UNDERSTAND###)
GO TO 904

910 WRITE (2,912)

912 FORMAT(15X,TYPE TWO DIGIT INTEGERS (03 ETC.) TO LOCATE THE*/,
115X,COEFFICIENT IN THE EDIT TABLE ABOVE.*)
WRITE (2,916)

914 FORMAT (15X, ROW=9)
ASSIGN 914 TO KS1
ASSIGN 914 TO KS2
READ (1,732) KROW
WRITE (2,920)

920 FORMAT (15X, COL=9)
ASSIGN 918 TO KS1
ASSIGN 918 TO KS2
READ (1,732) KCOL
WRITE (2,924)

924 FORMAT(15X,TYPE A DECIMAL NO. (12. OR *1.20000E+01) FOR NEW VALUE.
1*9,9,9 NEW=9)
ASSIGN 922 TO KS1
ASSIGN 922 TO KS2
READ (1,926) XNEW

926 FORMAT (15X,9)
WRITE (2,928) KROW,KCOL,XNEW

928 FORMAT(15X,YOU WILL CHANGE ROW *12,9, COL *12,9 TO *111.4,5*/,
115X,THIS WHAT YOU WANT2 TYPE (YES OR NO)*/)
READ (1,724) KONFR
IF (KONFR .EQ. 3HYES) GO TO 930
IF (KONFR .EQ. 4HSTOP) RETURN
GO TO 914
930 X(KROW,KCOL)=XNEW
931 WRITE(2,932)
932 FORMAT(15X,*ANY MORE COEFFICIENT CHANGES? TYPE (YES OR NO)*, /)
933 READ (1,724) KONFR
934 IF (KONFR .EQ. 3HYES) GO TO 914
935 IF (KONFR .EQ. 2HNO) GO TO 934
936 IF (KONFR .EQ. 4HSTOP) RETURN
937 WRITE (2,908)
938 GO TO 931
939 WRITE (2,936)
940 FORMAT(15X,*ANY RIGHAND SIDE CONSTANT CHANGES? TYPE (YES OR NO)*, /)
941 READ (1,724) KONFR
942 IF (KONFR .EQ. 3HYES) GO TO 938
943 IF (KONFR .EQ. 2HNO) GO TO 956
944 IF (KONFR .EQ. 4HSTOP) RETURN
945 WRITE (2,940)
946 FORMAT(15X,*TYPE TWO DIGIT INTEGER (03 ETC.) FOR CONSTRAINT* , /)
947 ASSIGN 938 TO KS1
948 ASSIGN 938 TO KS2
949 READ (1,732) KEQ
950 WRITE (2,944) KEQ
951 FORMAT(15X,*TYPE DECIMAL NUMBER (12, OR +1.2000E+0I) FOR NEW VALUE
952 */15X,*FOR RIGHAND SIDE OF CONSTRAINT EQ. *.,12,*,/,* NEW*=
953 ASSIGN 942 TO KS1
954 ASSIGN 942 TO KS2
955 READ (1,926) XNEW
956 WRITE (2,946) KEQ,XNEW
957 FORMAT(15X,*YOU WILL CHANGE CONSTRAINT EQ. NUMBER *.,12,*,/)
958 ASSIGN 942 TO KS1
959 ASSIGN 942 TO KS2
960 READ (1,926) XNEW
961 WRITE (2,946) KEQ,XNEW
962 FORMAT(15X,*YOU WILL CHANGE CONSTRAINT EQ. NUMBER *.,12,*,/)
963 ASSIGN 942 TO KS1
964 ASSIGN 942 TO KS2
965 READ (1,926) XNEW
966 WRITE (2,946) KEQ,XNEW
967 FORMAT(15X,*ANY MORE CONSTRAINT CONSTANT CHANGES? TYPE (YES OR NO)*, /)
968 READ (1,724) KONFR
969 IF (KONFR .EQ. 3HYES) GO TO 914
970 IF (KONFR .EQ. 2HNO) GO TO 952
971 IF (KONFR .EQ. 4HSTOP) RETURN
972 WRITE (2,908)
973 GO TO 949
974 X(KEQ+1,R1)=XNEW
975 WRITE (2,950)
976 FORMAT(15X,*ANY MORE CONSTRAINT CONSTANT CHANGES? TYPE (YES OR NO)* ,/)
977 READ (1,724) KONFR
978 IF (KONFR .EQ. 3HYES) GO TO 914
979 IF (KONFR .EQ. 2HNO) GO TO 952
980 IF (KONFR .EQ. 4HSTOP) RETURN
981 WRITE (2,908)
982 GO TO 949
983 WRITE (2,954)
984 FORMAT(15X,*ANY COEFFICIENT CHANGES? TYPE (YES OR NO)*, /)
985 READ (1,724) KONFR
986 IF (KONFR .EQ. 3HYES) GO TO 910
987 IF (KONFR .EQ. 2HNO) GO TO 956
988 IF (KONFR .EQ. 4HSTOP) RETURN
989 WRITE (2,908)
990 GO TO 952
991 WRITE (2,958)
992 FORMAT(15X,*YOU MAY SELECT THREE DISPLAY MODES FOR THE RESULTS* ,/)
993
THE CHANGES YOU JUST MADE,
FULL TABLEAU FOR ALL ITERATIONS,
FULL TABLEAU FOR ONLY FINAL ITERATION,
VALUE OF OBJECTIVE FUNCTION AND LIST OF BASIS

ITNO=0

WRITE (2,960)
WRITE (2,908) RETURN ' 

KFLG=1 RETURN . . . . . 
KFLG=2 RETURN 
KFLG=3 ' 

THIS SUBROUTINE CONTROLS THE CRITIQUE AND EVALUATION SECTION
COMMON/COM3/IANS(4)

WRITE (2,498)
WRITE (2,499)

FORMAT (15X,50H+***************+ TEAR HERE +***************+),
1/X 15X*YOU ARE NOW IN THE CRITIQUE AND EVALUATION*
15X*SECTION. PLEASE ANSWER THE FOLLOWING QUESTIONS*
15X*ABOUT YOUR USE OF SAL. AFTER YOU HAVE ANSWERED*
15X*THE QUESTIONS, YOU WILL BE INSTRUCTED TO*
15X*DEACTIVATE SAL. THE TELETYPE WILL THEN RE*
15X*AUTOMATICALLY DISCONNECTED FROM THE COMPUTER AND*
15X*WILL BE JUST LIKE AN ELECTRIC TYPEWRITER. USING*
15X*THE TELETYPE AS A TYPEWRITER, PLEASE CONTINUE*
15X*FINISHED, TEAR THE PAPER WHERE INDICATED ABOVE*
15X*AND PROCESS THIS PORTION OF THE PAPER AS DIRECTED*
15X*WHEN READY TO CONTINUE, TYPE GO,*

READ (1,500) IANS(1)
WRITE (2,500) IANS(1)
WRITE (2,501) IANS(1)
WRITE (2,502) IANS(1)
WRITE (2,503) IANS(1)

FORMAT (15X,*), IS THE MATERIAL IN SECTION 1*
120X,*A. TOO LONG*
120X,*B. TOO SHORT*
120X,*C. ABOUT RIGHT*

READ (1,500) IANS(1)

FORMAT (A10)

READ (1,500) IANS(1)

FORMAT (15X,*2), DO YOU THINK THE MATERIAL IN SECTION 1 IS*
120X,*A. EQUATEAS A REVIEW OF THE SIMPLEX ALGORITHM*
120X,*B. YES*
120X,*C. NO*

READ (1,500) IANS(1)

FORMAT (15X,*3), DO YOU THINK THERE SHOULD BE*
120X,*A. MORE ILLUSTRATIVE EXAMPLES, OR*
120X,*B. FEWER ILLUSTRATIVE EXAMPLES*
115X,*IN SECTION 1*

READ (1,500) IANS(1)
101

WRITE (2,504)
FORMAT (15X,*, 'FOR THE PROBLEM SOLVING AND ANALYSIS SECTIONS',*)
115X,A 'DO YOU THINK THE ENTERING OF THE DATA WAS'
120X,A 'TOO DIFFICULT'
120X,A 'TOO SIMPLE AND COULD BE SPEEDED UP'
120X,A 'ABOUT RIGHT'
READ (1,500) IANS(1)
WRITE (2,505)

505 FORMAT (15X,A5, 'IS THE EXPLANATION AND USE OF DECIMAL NUMBERS'
115X,A,SATISFACTORY'
120X,A, YES'
120X,A, NO'
READ (1,500) IANS(1)
WRITE (2,506)

506 FORMAT (15X,A6, 'IS THE USE OF TWO DIGIT INTEGERS DIFFICULT'
115X,A, OR CONFUSING'
120X,A, YES'
120X,A, NO'
READ (1,500) IANS(1)
WRITE (2,507)

507 FORMAT (15X,A7, 'IS THE PRESENTATION OF THE DATA IN TABLEAU'
115X,A, FORM'
120X,A, EASY TO USE AND UNDERSTAND, OR'
120X,A, CONFUSING'
READ (1,500) IANS(1)
WRITE (2,508)

508 FORMAT (15X,A8, 'DID SAL HELP YOU GAIN A BETTER FEEL FOR THE'
115X,A, SIMPLEX ALGORITHM'
120X,A, YES'
120X,A, NO'
120X,A, QUESTIONABLE'
READ (1,500) IANS(1)
WRITE (2,509)

509 FORMAT (15X,A9, 'THANK YOU FOR THE ANSWERS. SAL WILL NOW'
115X,A, DEACTIVATE ITSELF. AFTER SAL HAS STOPPED'
115X,A, TERMINATE INTERCOM BY LOGGING OUT WITH COMMAND'
115X,A, LOGOUT. AFTER THE TERMINAL HAS STOPPED PRINTING'
115X,A, ITS LOGGED-OUT MESSAGE, DEPRESS THE CLR BUTTON TO'
115X,A, DEACTIVATE THE TERMINAL. YOU MAY THEN DEPRESS THE'
115X,A, LCL BUTTON AND USE THE TERMINAL AS AN ELECTRIC'
115X,A, TYPEWRITER AND CONTINUE WITH ANY COMMENTS OR'
115X,A, SUGGESTIONS ON ANY ASPECTS OF SAL. THE'
115X,A, INFORMATION YOU PROVIDE COULD BE HELPFUL IN'
115X,A, IMPROVING SAL'
115X,A, THANK YOU FOR YOUR COOPERATION'
WRITE (2,509)

STOP

CCCCCCCCCCCCCC
C SUB1 AND SUB2 ARE ERROR ROUTINES TO CATCH ERRORS IN USER-INPUT
C NUMBERS WITH BAD OR ILLEGAL FORMAT, PREVENTS A FATAL ERROR
C FROM TERMINATING THE PROGRAM. IE, IF USER INPUT 12,36 INSTEAD
C OF 1236 THE ERROR ROUTINES WILL ALLOW USER TO CORRECT THE ERROR
C AND CONTINUE EXECUTING,
SUBROUTINE SUB1
COMMON/S1/KS1,KS2,IEXP
IF (IEXP .EQ. 3HEXP) GO TO 5
WRITE (2,1)
1 FORMAT(15X,A, 'YOU HAVE MADE A MISTAKE IN THE ABOVE NUMBER',*)
115X,A,TRY AGAIN'
GO TO KS1
5 WRITE (2,7)
7 FORMAT(15X, 'YOU HAVE MADE A MISTAKE IN ONE OF * ',/ 
115X, 'THE NUMBERS IN THE ABOVE LIST * ',/ 
115X, 'RETYPE THE ENTIRE LIST * ',/ 
115X, 'FIRST DEPRESS THE RETURN KEY * ')
GO TO KS1
3 CONTINUE
RETURN
END
SUBROUTINE SUB?
COMMON /S 1/KS1, KS2, IEXP
IF (IEXP .EQ. 0) GO TO 6
WRITE (2,2)
2 FORMAT(15X, 'YOU HAVE MADE A MISTAKE IN THE ABOVE NUMBER * '/ 
115X, 'THE MAGNITUDE IS TOO BIG FOR THE COMPUTER * '/ 
115X, 'CHECK THE POSITION OF THE E AND TRY AGAIN * ')
GO TO KS2
6 WRITE (2,8)
8 FORMAT(15X, 'ONE OF THE NUMBERS IN THE ABOVE STRING * '/ 
115X, 'IS TOO BIG FOR THE COMPUTER, CHECK THE POSITION * '/ 
115X, 'OF THE E AND RETYPE THE ENTIRE LIST * '/ 
115X, 'FIRST DEPRESS THE RETURN KEY * ')
GO TO KS2
4 CONTINUE
RETURN
END
THIS IS THE SIMPLEX ALGORITHM LABORATORY (SAL).
SAL IS A COMPUTER-ASSISTED INSTRUCTION (CAI)
PROGRAM THAT WILL HELP YOU GAIN A BETTER
UNDERSTANDING OF THE SIMPLEX ALGORITHM AS IT IS
APPLIED TO THE SOLUTION OF LINEAR PROGRAMMING
PROBLEMS. IT IS ASSUMED THAT YOU ALREADY HAVE A
BASIC UNDERSTANDING OF LINEAR PROGRAMMING AND THE
SIMPLEX ALGORITHM. SAL IS DIVIDED INTO FOUR
SECTIONS. SECTION 1 IS INTRODUCTORY MATERIAL TO
GIVE YOU A QUICK REVIEW OF THE ALGORITHM AND TO
FAMILIARIZE YOU WITH THE CONTROL TECHNIQUES NEEDED
TO USE SAL. AT PERIODIC INTERVALS YOU MUST ELECT
TO EITHER CONTINUE WITH SECTION 1 OR SKIP TO
SECTION 2. THE TWO COMMANDS ARE GO AND SKIP.
TYPE (GO OR SKIP).

SECTION 2 AND SECTION 3 ARE THE PROBLEM SOLVING
AND SENSITIVITY ANALYSIS SECTIONS WHERE YOU MAY
ENTER LINEAR PROGRAMMING PROBLEMS AND OBSERVE
THE RESULTS AS THE COMPUTER AUTOMATICALLY APPLIES
THE SIMPLEX ALGORITHM TO OBTAIN SOLUTIONS. YOU
MAY ENTER ANY LINEAR PROGRAMMING PROBLEM UP TO
A MAXIMUM SIZE OF 40 VARIABLES AND 20 CONSTRAINTS.
SECTION 4 IS THE CRITIQUE AND EVALUATION SECTION
WHERE YOU WILL EVALUATE SAL BY ANSWERING A FEW
SPECIFIC QUESTIONS ON YOUR USE OF SAL AND GIVE ANY
GENERAL COMMENTS YOU MAY WISH TO MAKE.
TYPE (GO OR SKIP).

THE FOLLOWING GENERAL COMMENTS APPLY TO YOUR USE
OF SAL.
1. ALWAYS BEGIN TYPING YOUR RESPONSE FROM THE
POSITION WHERE THE CARRIAGE HAS STOPPED.
2. TO CORRECT AN ERROR YOU CAN BACK SPACE USING
UPPER CASE 0 AND THEN RETYPE OVER THE ERROR
OR YOU CAN VOID AN ENTIRE LINE BY DEPRESSING THE
CTRL KEY AND THE X SIMULTANEOUSLY. THIS WILL MOVE
THE CARRIAGE TO THE LEFT MARGIN OF A NEW LINE.
YOU WILL BEGIN TYPING AT THAT POSITION.
3. YOU MUST TYPE ALL RESPONSES EXACTLY AS SHOWN
WHEN SELECTING A REPLY FROM A LIST OF OPTIONS.
TYPE (GO OR SKIP).
4. YOU WILL BE USING TWO TYPES OF NUMBERS
(INTEGER AND DECIMAL) DURING SECTIONS 2 AND 3.
ALL INTEGERS MUST BE TWO DIGITS (03 MEANS THREE)
AND CAN NOT HAVE A DECIMAL POINT. DECIMAL NUMBERS
CAN BE EITHER STRAIGHT DECIMAL OR E FORMAT.
FOR EXAMPLE +136.470 IS EQUIVALENT TO +1.3647E+02.
THE STRAIGHT DECIMAL NUMBER MUST CONTAIN A DECIMAL
POINT AND MAY CONTAIN UP TO 11 CHARACTERS
COUNTING THE . AND THE SIGN (IF SIGN OMITTED,
ASSUMED TO BE POSITIVE). THE E FORMAT DECIMAL
NUMBERS MUST HAVE EXACTLY THE FORMAT SHOWN ABOVE.
THE SIGN MUST BE PRESENT.
TYPE (GO OR SKIP).
5. AT ANY TIME DURING PROBLEM SOLVING AND
SENSITIVITY ANALYSIS, YOU CAN TERMINATE SAL BY
TYPING STOP WHENEVER SAL IS EXPECTING A WORD
COMMAND (SAL WILL NOT RECOGNIZE STOP IF IT IS
EXPECTING A NUMBER).
6. SAL CAN BE ABORTED ANY TIME BY DEPRESSING THE
CTRL KEY AND THE Z SIMULTANEOUSLY AND THEN THE A.
7. IF SAL IS INACTIVE FOR MORE THAN ABOUT 90 SEC.
(EXCEPT WHEN WAITING FOR YOUR RESPONSE) IT MAY
BE MALFUNCTIONING. YOU MAY ABORT IF YOU DESIRE.
BUT PRIOR TO ABORTING, YOU CAN TRY ONCE TO CORRECT
THE PROBLEM BY TYPING GO.
TYPE (GO OR SKIP).
TO PROVIDE A QUICK REVIEW OF THE SIMPLEX
ALGORITHM, A SIMPLE PROBLEM WILL BE SOLVED USING
THE ALGORITHM.
PROBLEM: MAXIMIZE 5X1 + 6X2
SUBJECT TO 2X1 + 3X2 <= 1.8
2X1 + 1X2 <= 1.2
3X1 + 3X2 <= 2.4

TYPE (GO OR SKIP).
INTRODUCE SLACK (OR SURPLUS) VARIABLES AND LET
OBJECTIVE FUNCTION BE X0. THEN REWRITE AS
X0 - 5X1 - 6X2 = 0.
2X1 + 3X2 + 1X41 = 1.8
2X1 + 1X2 + 1X42 = 1.2
3X1 + 3X2 + 1X43 = 2.4

NOTE THAT OBJECTIVE FUNCTION (X0) IS SET EQUAL 0
AND X1 AND X2 TRANSPOSED TO LEFT OF = SIGN WITH
THEIR SIGN REVERSED. INITIALLY LET X1 = X2 = 0.
TYPE (GO OR SKIP).

THE ABOVE DATA CAN BE CONDENSED INTO A TABLEAU

<p>| IT | CURRNET |</p>
<table>
<thead>
<tr>
<th>NO BASE</th>
<th>VALUES</th>
<th>X1</th>
<th>X2</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X0</td>
<td>0</td>
<td>-5</td>
<td>-6</td>
</tr>
<tr>
<td></td>
<td>X41</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>X42</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X43</td>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

X0, X41, X42, X43 ARE IN SOLUTION (THEY ARE CALLED
THE BASIC VARIABLES OR BASIS) WITH VALUES SHOWN.

X1 AND X2 ARE THE NONBASIC VARIABLES AND ARE = 0.

TYPE (GO OR SKIP).

THE ALGORITHM SAYS TO CHECK ROW 1 VARIABLES FOR
NEGATIVE COEFFICIENTS AND SELECT THE MOST NEGATIVE
VARIABLE TO ENTER THE BASIS. IF ALL COEFFICIENTS
ARE POSITIVE OR ZERO, THE OPTIMUM SOLUTION HAS
BEEN REACHED. TO DETERMINE THE VARIABLE TO EXIT,
CALCULATE FOR EACH ROW THE RATIO OF THE CURRENT
VALUES TO THE COEFFICIENTS OF THE ENTERING
VARIABLE AND SELECT THE MINIMUM RATIO (IGNORE
NEGATIVE NUMBERS OR ZEROS IN THE DENOMINATOR).
APPLYING THIS CRITERION TO ABOVE PROBLEM, X2 WILL
BE ENTERED INTO AND X41 WILL EXIT FROM THE BASIS.

TYPE (GO OR SKIP).

A CHANGE-OF-BASIS CALCULATION IS PERFORMED TO
DETERMINE THE VALUE FOR X2 AND ASSIGN A VALUE OF
ZERO TO X41. THE RESULTS OF THE STEPS (CALLED ONE
ITERATION OF THE ALGORITHM) ARE SHOWN IN TABLEAU
FORM AS FOLLOWS.

<p>| IT | CURRNET |</p>
<table>
<thead>
<tr>
<th>NO BASE</th>
<th>VALUES</th>
<th>X1</th>
<th>X41</th>
<th>ROW</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NOTE THAT FOR THIS ITERATION NUMBER (IT NO) X0, 
THE VALUE OF THE OBJECTIVE FUNCTION HAS INCREASED 
TO 36.0. CHECKING ROW 1 SHOWS THAT THE OPTIMAL 
SOLUTION HAS NOT BEEN REACHED (X1 HAS A NEGATIVE 
COEFFICIENT). APPLYING THE ALGORITHM AGAIN WILL 
BRING X1 INTO THE BASIS AND EXIT X42. YOU SHOULD 
CONVINCE YOURSELF THAT THE MINIMUM RATIO IS 
(.60 DIVIDED BY .13).

SAL WILL AUTOMATICALLY APPLY ALL NECESSARY CHECKS 
AND PERFORM ALL CALCULATIONS TO PROCEED FROM ONE 
ITERATION TO THE NEXT. PRIOR TO ASKING SAL FOR 
THE NEXT ITERATION, YOU SHOULD TRY AND DETERMINE 
WHICH VARIABLES WILL ENTER AND EXIT FROM THE 
BASIS. THE SIMPLEX ALGORITHM WILL NOT BE 
DISCUSSED ANY MORE HERE. IF YOU ARE UNSURE OF 
THE ALGORITHM YOU MAY STOP AT THIS POINT BY TYPING 
STOP.

TO PREPARE YOUR PROBLEM FOR ENTRY INTO SAL FOR 
SOLUTION, YOU MUST EXPRESS THE OBJECTIVE FUNCTION 
AS A MAXIMIZE PROBLEM. THEN REVERSE THE SIGN OF 
ALL THE COEFFICIENTS IN THE OBJECTIVE FUNCTION. 
FOR EXAMPLE, IF YOUR PROBLEM IS TO 
MAXIMIZE 3X1 + 6X2 + 8X3 - 2X4 
YOU WILL ENTER IT INTO SAL AS: -3, FOR X1 
-6, FOR X2, -8, FOR X3 AND +2, (OR 2.) FOR X4.

THE CONSTRAINT EQUATIONS ARE ENTERED AS THEY 
APPEAR IN YOUR PROBLEM. FOR EXAMPLE, IF YOU HAD 
THE FOLLOWING AS ONE OF THE CONSTRAINTS:
3X1 + 7X2 - 2X3 + 5X4 <= 14.5
YOU WILL ENTER 3. (MAY OMIT + SIGN, SEE COMMENT 4 
ABOVE) FOR X1, 7. FOR X2, -2. FOR X3, 5. FOR X4, 
A CODE 1 FOR <= (THIS WILL BE EXPLAINED IN THE 
PROBLEM SECTION THAT FOLLOWS) AND FINALLY, YOU 
WILL ENTER 14.5 FOR THE RIGHThAND SIDE CONSTANT.

YOU MUST NOT ENTER SLACK, SURPLUS, OR ARTIFICIAL 
VARIABLES WITH THE CONSTRAINT. SAL WILL ADD 
THESE AUTOMATICALLY AS NEEDED. SAL ASSIGNs X41 
THROUGH X60 AS SLACK AND/OR SURPLUS VARIABLES AND 
X61 THROUGH X80 AS ARTIFICIAL VARIABLES. YOU 
WILL ENTER YOUR PROBLEM VARIABLES AS X1 THROUGH 
X40. THIS CONCLUDES THE INTRODUCTORY MATERIAL. 
WHEN YOU CONTINUE YOU WILL BE IN THE PROBLEM 
SOLVING SECTION. 

TYPE GO WHEN READY.

7/8/9
6/7/8/9
REFERENCES


