University of Alberta
Department of Civil &
Environmental Engineering

Structural Engineering Report No. 52

Decision Table Processing of the Canadian Standards Association Specification S16.1

by

Siu Kwong Fritz Wu

and

D.W. Murray

February, 1976

STRUCTURAL ENGINEERING REPORT NO. 52

DECISION TABLE PROCESSING OF THE CANADIAN STANDARDS ASSOCIATION SPECIFICATION S16.1

by

SIU KWONG FRITZ WU

and

D.W. MURRAY

February 1976

Department of Civil Engineering
University of Alberta
Edmonton, Canada

ACKNOWLEDGEMENTS

This investigation was made possible by funds provided by the National Research Council of Canada, Grant NRC-A5307, and the Department of Civil Engineering, the University of Alberta.

This report was prepared by S.K.F. Wu as an M.Sc. thesis, under the direction of Prof. D.W. Murray.

ABSTRACT

This report presents sections of the Canadian Standards Association Specification S16.1 - Steel Structures for Buildings - Limit States Design, in decision table format. Basic theory of decision tables is discussed. The decision table formulation renders the code requirement checking procedure into an individual module independent from the analysis and member selection procedures.

A specification processing program, in interactive mode and batch mode, is also presented.

A scheme for recursive execution of a decision table within a cycle has been developed leading to the possible saving of overall data items and decision tables.

A number of examples are solved to check the validity of the decision tables compiled.

TABLE OF CONTENTS

CHAPTER			PAGE

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Objective

The purpose of this thesis is to document and implement the 1974 Canadian Standard Association (CSA) Standard S16.1-Steel Structures for Buildings-Limit States Design, in the form of decision tables. The technique of decision logic tables, or simply "decision tables", appears to be the best means available at present for clear definition and presentation of design logic.

It is hoped that this form of presentation will not only help to give a more precise and logical interpretation of the Standard but also facilitate an easier approach in implementing it for computer applications. Implementation of CSA S16.1 in this thesis is accomplished through a general purpose decision table processing program listed in Appendix D and E, which results in the ability to check the adequacy of most types of members against all relevant provisions of the Standard (also referred to as a code or specification).

1.2 The Design Process

The design process can be considered as consisting of the following steps:

(i) Analysis

(ii) Member Selection

(iii) Constraints Checking (i.e. - checking all code provisions)

As the complexity of the code provisions increases, the more desirable a logical format becomes. Design specifications are generally a complex system in which all pertinent combinations of design events and their associated requirements are presented. Presentation of such logically complex systems in the traditional narrative format is difficult due to the necessity of sequential presentation of concepts and the limitations of language with respect to absolutely precise communication. Since design specifications essentially control the quality and safety of a structure, it is important that their requirements be explicit and clearly presented in order to facilitate their use to the best possible extent during the design process.

The common present approach of producing structural design programs based on a particular set of specifications is very inefficient. Specifications are often subjected to revisions. A minor revision often leads to a major modification of the program. In many cases, those responsible for the actual design of a structure do not know and often have difficulty in locating a person who knows exactly what assumptions and interpretations have been included in the program. Hence, developing and revising such programs becomes a very expensive and time consuming task.

Most of the specifications and codes of different authorities at present are not readily adaptable to computer application because they do not display the logic leading to a certain action as clearly as needed for computer application, and the requirements for the same action may be scattered throughout the text of the specification. The design process up to the end of step (ii) involves many considerations other than design constraints whereas step (iii) is entirely dependent on them. Therefore, a more logical approach is to regard the constraints checking as a separate module, independent from analysis and member selection, and then design the programs accordingly. In this way, any member selection programs can be used with programs of different code requirements.

The decision table format of CSA-S16.1 presented in this thesis could therefore, result in a reassessment of the form in which this code is presented and may ultimately result in an improvement to the specification.

CHAPTER II

DECISION TABLE THEORY

2.1 Introduction

Tables are a familiar part of everyone's life. From mathematical tables to the milage tables on road maps. They provide us with an orderly presentation of data.

Traditionally, flow charts and narratives have been used to structure the logic of a problem. These are effective as long as the problem is fairly simple logically. When interaction among a large number of conditions exists, how can one be certain that every possible combination has been considered?

Decision tables are simply a technique for recording "decision-making" processes. They have been used since the early 1950's. Unlike flow charts, which could be as simple or as detailed as the programmer desires, decision tables force one to be thorough and concise. When properly used, they demand that all combinations of conditions be considered and allow irrelevant tests to be deleted. In this chapter, the basic components and theory of decision tables are discussed. For a greater insight into the theory, the reader is referred to references: 25, 10, 14, 18, 20, 26, 22, 6.

2.2 Basic Components

A decision table is simply a tabular display of all elements of a segment of a problem. The table shows all of the conditions affecting the problem and their interrelationship. It further indicates which action or actions are appropriate for each set of conditions.

A decision table consists of sections, which are arranged as shown in Fig. 2.1*. The contents of these sections are discussed below.

2.2.1 The Condition Stub

This area contains all the conditions relevant to the particular segment of the problem. They must all be "logical" conditions, which can have only two possible values: "YES" or "NO".

2.2.2 The Action Stub

This area contains all the possible actions that may be taken resulting from different combinations of conditions. The actions may be printing of messages, computations, or execution of other tables.

2.2.3 The Condition Entries

The questions asked in the condition stub are answered here. This segment lists all the pertinent

---

* All Figures are grouped together preceding the References.

combinations of the logical conditions in a column. Each column specifies a "rule". Responses to conditions are restricted to "Y" for Yes, "N" for No, and "I" for Immaterial.

2.2.4 The Action Entries

The elements of the action entries may either be "Y" or "blank". "Y" signifies the corresponding action in the action stub is to be performed. A "blank" signifies that the corresponding action is not to be taken.

2.2.5 The Ingredients of Conditions and Actions

These two areas list the ingredients of each condition and action respectively. Ingredients are defined in Section 3.4.

2.2.6 Rows and Rules of a Decision Table

Horizontal levels, called "rows", run across the entire table. The condition and action entry portions of the table are further subdivided into vertical columns called "rules".

Decision tables of the type described above are called "limited entry tables" because the condition entries are considered to be limited to "Y", "N" or "blank" and the action entries are limited to "Y" and "blank". Other types of decision tables such as "extended entry tables" and "mixed entry tables" (25, 10, 14, 18, 20, 26), allow for a wider variety of condition and action entries but they are more complex to program.

2.3 Size Of A Decision Table

As explained in Sect. 2.2.1, each logical condition has only two values ("Y" or "N"). It follows that for a table with one condition, there are two rules. Therefore, a table with n conditions has 2^n rules. A table of this type is termed a "Complete Table". However, a "Complete Table" such as that illustrated in Fig. 2.2, rarely occur in practice, for the following reasons:

(a) If the action for rule 2 and 3 is the same, the value of condition 2 for rule 2 and 3 becomes immaterial. The table transforms into the form illustrated in Fig. 2.3. The number of rules has been reduced to 3.

(b) In decision table X.1(1)* of Appendix B, the two conditions are mutually exclusive as discussed in Sect. 3.3. Therefore, there are only two valid combinations of the two conditions, namely, Y N and N Y, rather than $2^2 = 4$ number of rules in a complete table.

* Decision tables in Appendix B are arranged, as closely as possible, in order ofthe number which appears in brackets.

2.4 Conventions

In this section, the notations and conventions used in the formulation of the decision tables presented in Appendix B are discussed. Signs and units of variables are the same as those used in CSA-S16.1.

2.4.1 Conditions

All the conditions in the condition stub are logical variables. There are two types of conditions:

(i) Conditions which are logical data items, such as "SECTION DOUBLY SYMMETRIC?"

(ii) Logical conditions which result from numerical calculations, such as "$b/t \leq 420/\sqrt{F_y}$?" , Which has a value of "YES" or "NO" after the calculations for b/t and $420/\sqrt{F_y}$ are performed.

2.4.2 Actions

Certain actions which are listed in the action stub, are performed according to specific combinations of conditions in the condition stub. There are three types of actions:

(i) An action which directs the execution of the program to another table. For example "Execute Table M".

(ii) An action which outputs appropriate messages. For example, "STRENGTH CRITERION NOT SATISFIED".

(iii) An action which calculates numerical values of variables. For example "$M_r = \phi z F_y$".

2.4.3 Data Requirements

There are in general two types of input data required to process a decision table. The first type is conventional numerical data, such as the value of yield stress F_y = 44 ksi. The second type is logical data, which can only have value of "YES" or "NO", such as "STRUCTURAL STEEL MEMBER?"

The data items necessary for the processing of each table are listed preceding each decision table, as presented in Appendix B, in the format shown in Fig. 2.4. The first column identifies the data item. The second and third columns specify whether this data item has to be externally input or can be obtained from another table.

2.5 An Introductory Example

To demonstrate the use of a decision table, the Decision Table 13.4.A.1 (75) in Appendix B, for section 13.4 of CSA S16.1 Standard, which deals with the calculation of the factored shear resistance V_r, will be used. In this example, assume the problem is to calculate the factored shear resistance V_r of a beam which has been designed using plastic analysis. The values of the five conditions in the condition stub that describe the problem are N Y Y N N. Scanning the condition entry part of the table, rule 4 satisfies the condition requirements, since conditions 4 and 5 of rule 4 are both "immaterial". Following rule 4 to the action entry part of the table, it is noted that there is a Y entry opposite action 4, which means action 4 in the action stub will be performed. It is a numerical calculation which gives a value of $V_r = 0.55\phi w d F_y$.

CHAPTER III

DATA ORGANIZATION

This chapter discusses the nature, properties and sources of values of data elements. In a general processor program, the number of data values required becomes very large even for a small problem. Therefore, it is imperative to have a procedure which can handle large amounts of data efficiently. In other words, it largely becomes a data management problem (13, 19).

3.1 Nature Of Data Elements

There are in general, two types of data elements:

(a) Data which have numerical values in the conventional sense. For example, the length of the steel member, l.

(b) Data which are logical in nature. They can only have values of "YES" or "NO". For example, "Is the section double symmetric?".

3.2 Sources and Presence of Data Elements

There are four sources from which data values can be generated:

(a) Data which are supplied by external input. They can either be logical or numerical.

(b) Data which are generated by actions performed in a decision table. Data generated in this manner can either be logical or numerical.

(c) Data which are generated as a result of calculations performed on other data elements. For example, "$b/t \geq 420/\sqrt{F_y}$?". This type of data element is always logical in nature.

(d) Data to be input from the terminal using Subroutine READIN. This source is only applicable in interactive mode operation which is discussed in Sect. 7.2.

The sources of data discussed above are listed in the heirarchy sequence in which the program searches for data. During processing, when a particular data value is required, the program checks if it has been input externally. If not, it then turns to sources (b), (c) and (d) in that order.

It should be noted that not all the actions of a decision table generate data elements. However, the values of all the data elements generated may be stored in a global array called DATA, together with the values of each data element from the condition and action stubs. The value of each data element may be retrieved from the array DATA by using its number address (i.e. subscript) in the array. This arrangement of global storage enables data to be retrieved easily at every stage during execution of the program.

Due to the multiple sources from which data elements can be generated, it becomes essential to know at any time during processing whether a particular data item has a value or not. This information is provided by an array PRD (Presence of Data), whose subscripts correspond to those of array DATA, and which contains logical flags .TRUE. or .FALSE. . The elements in array PRD are all initialized to .FALSE. at the beginning of execution of the program. Whenever the value of a data element becomes available during execution, the corresponding element in array PRD is set to .TRUE. . The presence of a data element value can then be established at any time during processing by checking with array PRD.

3.3 Table Pointers and Mutually Exclusive Sets

In order to obtain a value of a particular data element by the execution of an appropriate decision table, according to the data source of Sect. 3.2(b), the number of the table which must be executed is specified as a property of that data element. This table number is stored in an array TABD (Table for Data) with subscripts corresponding to those in the array DATA for each data element. A blank in TABD indicates that the data element cannot be retrieved by executing any decision table.

Some data elements have the property that their values are mutually exclusive from one another, forming a mutually exclusive set. Data in a mutually exclusive set are always logical in nature. The implication of a mutually exclusive set is the user need only supply the value of one item in the set. The other items in the same set will automatically be set to "NO". For example, in Decision Table 11.A (6) in Appendix B, the list of data required consists of the following: T-Section, I-Section,]-Section, Rectangular Hollow Section and Box Section. These five data items form a mutually exclusive set since only one can have a value of .TRUE. at any given time. The number of a mutually exclusive set to which a data element belongs is stored in an array ISET. The property of mutually exclusive sets discussed above reduces the number of externally input data considerably.

The subscript (or address) of a data item in the array DATA also serves to retrieve the values in arrays PRD, TABD, and ISET which are applicable to this data item. The interrelationship of these four global arrays is as illustrated in Fig. 3.1.

3.4 Ingredients and Dependents of Data Elements

As indicated in Sect. 3.2(b,c), there are two ways in which data elements can be generated by involving other data elements. When a value of a data element is evaluated as a function of other data elements, the latter are called "ingredients" of the former. Thus, a data element X which may be evaluated as $X = F(u_1, u_2, u_3 \ldots u_n)$, has data elements $u_1, u_2, u_3 \ldots u_n$ as its ingredients. For example, the logical data element "$b/t > 420/\sqrt{F_y}$?" has ingredients b, t, and F_y.

There are instances when the value of one data element may be generated by more than one function, each having a different set of ingredients. For example, the data element M_{ry} in Decision Table 13.6.A.2 (50) in Appendix B, may be generated by three different actions which have different ingredients. Since ingredience is a property of the data element, it is only necessary to associate ingredients with the conditions and actions of decision tables where the ingredients are actually used. The dependent list of each data element is generated internally. Thus, ingredients are associated with rows of a decision table.

The concept of dependence follows directly from that of ingredients. If X has Y as its ingredient, then X is a dependent of Y. In the above example the logical data element "$b/t > 420\sqrt{F_y}$?" is a dependent of b, t and F_y because if any of these change in value, the logical data element may also change in value. The internal procedure of generating dependents imposed by ingredients consists of a loop for the data element which involves a search to see if the data element has any ingredients. If yes, it is placed in the lists of dependents of all the ingredients, unless it is already there. This procedure is illustrated by the flow chart in Fig. 3.2.

Dependence of data elements can also be imposed by the logic of decision tables. Actions performed in the action stub due to a particular combination of conditions are dependents of those conditions, as illustrated in Fig. 3.3, where data elements produced by action B are dependents of conditions A and B. Dependents generated by the logic of decision tables are entered into the lists of dependents of the relevant conditions in the condition stub by a loop in Subroutine SETUP, as flow charted in Fig. 3.4. It should be noted that dependency is a function of the ingredients and the decision logic of the immediate decision table. Dependency beyond this level need not be explicitly established as a data property but is determined at execution time, when necessary for clearing purposes, by the processing program. This procedure is discussed in Sect. 7.4.

CHAPTER IV

PROCESSING OF DECISION TABLES

4.1 Concepts

The approach taken to process the decision tables in Appendix B is to express each decision table as part of the global data, and execute all the tables by a single program. The procedure for processing a complete set of interrelated tables consists of the sequential processing of single tables in the order required by the problem, and checking the availability of data for each of these tables. If a data element is not available, execution of that table is suspended while the data element is being retrieved according to the heirarchy of data sources discussed in Sect. 3.2. This gives rise to the concept of conditional and direct execution which is discussed in Sect. 4.2 and 4.3.

In this section, the method of processing a decision table is introduced. The procedure is flow charted in Fig. 4.1. The data values required are assumed to be all available. The program starts by checking to see if the data values of the conditions match the condition entries of the first rule of the table. Matching is skipped if the condition entry is immaterial. Each rule is tested in turn until a match is found between the data values of the conditions and all condition entries for a rule. The relevant actions in the action stub of the matched rule will then be performed. If no rule is found to match the given conditions, the program will take an error exit route and gives a message to that effect.

In general, not all the data elements need be externally input. If a data value is missing, be it a condition or an action, it can be obtained by the procedure described in Sect. 4.2.

4.2 Conditional Execution

There are three situations which will lead to the suspension of execution of a table:

(a) The value of a condition is missing

(b) The ingredients of a condition are missing

(c) The ingredients of an action are missing.

In all of the above cases, if a table address is given in the array TABD (Fig. 3.1) for the missing data element, the execution of that table will commence immediately while the current table is suspended. The reason of suspension (a, b or c above), the current rule number, the current condition or action number and the current ingredient number of the suspended table are stored for re-execution. Once the missing data has been obtained, control is returned to the suspended table.

If a table address is not given in the array TABD for the missing data element, then the program checks to see if it has ingredients. If the missing data element has ingredients (and they are all present), a "condition subroutine" or "action subroutine" is called to evaluate the element. These subroutines are discussed in Sect. 5.4. If the data element does not have an entry in TABD and also has no ingredients, execution is terminated if in batch mode. In interactive mode, the subroutine READIN is called as discussed in Sect. 7.2.

4.3 Direct Execution

The command for direct execution is "Execute Table X". This is an action specified in the action stub and is referenced by a different code in the action entries. Once execution of Table X is completed, control is returned to the next action of the original table. Unlike conditional execution, direct execution may only occur as a result of an instruction in the action entry of a rule.

CHAPTER V

IMPLEMENTATION OF DECISION TABLE PROCESSOR

The decision table processing program used herein was originally developed by Goel (12) in FORTRAN IV with batch-mode only input facility. The source program listed in Appendix E is similar to the original program but with numerous modifications implemented by the writer. In order to increase the capability of the processor, an interactive-mode of input and execution, and a recursive execution scheme have been introduced which are discussed in Chapter VII.

For specification checking, the program requires two types of data input:

(a) Decision table information which is independent from any other data characteristics. Therefore, decision tables only need to be processed once and stored permanently unless alterations are made to the tables. The storage of this type of input is discussed in Sect. 5.2, and the input procedure is discussed in Appendix C.

(b) Problem orientated data values which are not part of permanent data. For example, "The depth of section d" or "The loadings on the member", as discussed in Sect. 3.2.

As discussed in Chapter III, data elements are stored in global arrays (Fig. 3.1). Therefore each data element can be easily retrieved by its array subscript. For example, DATA (95). However, it is often more convenient and meaningful to identify a data element with a name. This can be achieved by using the FORTRAN EQUIVALENCE STATEMENT. For example, "EQUIVALENCE (DATA (95), $CLAS1)". In this case, data element 95 in array DATA is a logical element $CLAS1= "IS SECTION A CLASS 1 SECTION?". Throughout this thesis, logical data elements are identified by the prefex character "$".

5.1 Computer Coding of Decision Tables

The basic components of a decision table have been discussed in Sect. 2.2. In this section, the coding of each section of a decision table in a form suitable for input to the computer is described.

(a) The condition stub contains logical data elements whose values are stored in the global array DATA. This area contains the addresses (i.e. subscripts) of the conditions in the array DATA.

(b) The action stub contains both numerical and logical data elements. Three types of actions may be performed:

(i) Calculation to obtain the value of a data element by an action subroutine. The action stub contains the address of the data element in array DATA in this case.

(ii) Direct execution of a table. The action stub contains the table number in this case.

(iii) Output of messages by an action subroutine. The action stub contains a "blank" for this type of action.

(c) The condition entry portion of the table has three types of entries. They are "Y", "N" and "I". A "zero" or a "blank" signifies an immaterial entry. A "1" and "2" signifies a YES and NO entry respectively.

(d) The coding in the action entry portion of the tables signifies which type of action discussed in (b) is to be performed. The following coding is used:

A "blank" signifies that no action is to be taken.

A "1" signifies that the action is to be performed by an action subroutine (Sect. 5.4). The action may be calculating a data element or printing out of messages.

A "2" indicates direct execution of another table.

(e) The ingredient list for a condition contains the addresses in array DATA of the ingredients. The maximum number of ingredients per condition which the program can handle is eleven.

(f) The ingredient list for actions is similar to that for conditions.

5.2 Storage Arrays for Decision Tables

Economy in primary storage and efficient access to it are of particular importance in a program which handles large volumes of data information (13, 19). In order to conserve storage, the size of each decision table is defined at the time of input. The size of a table is defined by the number of rules, the number of actions, and the number of conditions. Each of these parameters is stored in the single subscripted arrays L, M, and N respectively. The contents of each decision table are read initially into temporary arrays and then compacted into the permanent one-dimensional arrays LARRY1 to LARRY6, as shown schematically in Fig. 5.1.

There is a pointer array assigned to each permanent array for the purpose of locating data elements efficiently. Fig. 5.2 shows a tabulation of the permanent arrays and their corresponding pointer arrays. As a typical example, the pointer array IPNTRC stores the base address (ie. the reference subscript) of each condition for the purpose of locating ingredients of conditions. The J^{th} ingredient of the I^{th} condition in array LARRY1 can be obtained by:

I1 = IPNTRC(I) + J

IDATA = LARRY5 (I1)

IDATA is the data address in array DATA of the J^{th} ingredient. The number of ingredients which condition I has is obtained from:

IPNTRC(I+1) - IPNTRC(I)

Fig. 5.3 shows schematically the overall array structure and their interrelationship.

5.3 Input Procedure for Decision Tables

For the purpose of input, the decision tables need not be modified in form to accommodate the storage scheme described in Sect. 5.2. The construction of the temporary arrays and the one-dimensional permanent arrays is entirely an internal operation.

The tables are read in one after another but need not be in any particular sequence since each table is identified by a table number. The first card for each table is a table header card which contains the table number (T), the number of rules (L), the number of actions (M) and the number of conditions (N). The header card is followed by a card for each row of the decision table, first the condition rows, then the action rows. Each of these cards contain the data address of the condition or action stub, the condition or action entries code, the data addresses of the condition or action ingredients and a flag "C" if more ingredients are to follow on the next card. A maximum of eleven ingredients per condition or action is permitted. Fig. 5.4 illustrates the coded input form of Decision Table 13.5.A.1 (44) in Appendix B. The input format is discussed in Appendix C: User's Guide.

5.4 Condition and Action Subroutines

These are subroutines associated with the condition or action stub of each decision table. The function of the condition subroutine is to calculate logical values of conditions whereas the action subroutine is used for the calculation of numerical values of data elements or for the output of messages. For example, data sources (b) and (c) discussed in Sect. 3.2 require the use of such subroutines.

Each subroutine is accessed from the main routine by a computed GO TO statement where the integer table number T controls program flow to the proper call statement as shown in the main routine in Appendix D. The presence of the ingredients which are required by each subroutine is checked before executing. The only action which does not require any ingredients is the output of messages. If any ingredients are missing, conditional execution, as discussed in Sect. 4.2, is activated to retrieve the missing ingredients. The actual portion of the subroutine relevant to a particular condition or action is referenced by a computed GO TO statement within the subroutine where the condition or action number controls program flow. A typical example of condition and action subroutines is presented in Fig. 5.5. This figure illustrates the subroutines for Decision Table 13.5.A.1 (44) in Appendix B, which calculates the moment of resistance M_{rx} and the ratio R2. Condition subroutines are designated by CC and action subroutines by AA, followed by the table number T.

CHAPTER VI

DECISION TABLING THE CSA S16.1 STANDARD - STEEL STRUCTURES FOR BUILDINGS - LIMIT STATES DESIGN

6.1 General Description of CSA S16 Standards

The CSA S16 Standard is the principal guide for the design of steel buildings in Canada. The first CSA S16 Standard appeared in 1924. The 1969 edition (15) of the standard was revised in 1974. The new Standard, CSA S16.1-1974 - Steel Structures for Buildings - Limit States Design, provides rules, guidelines and requirements for the design, fabrication and erection of steel buildings where the design is based on limit states (2). This Standard takes its place beside S16-1969, which will continue to provide engineers with a working stress design standard for some time.

The limitations in precisely wording the texts of standards often leads to difficulties in interpretation and possible inconsistencies in their application.

The objective of this thesis is to decision table the CSA S16.1 and incorporate the tables into a general purpose processing program for checking a design against the Standard. The Standard was still in the drafting stage when the tables were being compiled and the decision tables are based on the October 1974 draft.

6.2 Scope Of The Tables Compiled In This Thesis

The decision tables presented in Appendix B cover the sections in Chapters 1 to 13, Chapter 15 and Chapter 19 of the CSA S16.1 Standard, which deal with design decisions and which are not purely descriptive in nature. Regrettably due to the amount of time available, the other chapters of the Standard could not be compiled.

The tables and the Standard, do not provide certain computational procedures such as computing member loadings and forces, or methods of computing section properties. These procedures must be provided by the user and input to the program as external input data. Decision tables are capable of developing all the possible combinations of conditions. However, certain combinations may be invalid or impractical. Therefore, in the development of the decision tables in this thesis, only those combinations which are relevant were incorporated into them.

6.3 Organization Of The Decision Tables

In Appendix A, heirachy charts of the tables compiled in this thesis are presented. These charts show the order of execution of each table when a particular checking task is performed.

Figs. A.1 to A.11 show the general outline of the table organization. A description of the table designation is given as an introduction to the Appendix. Decision Table X.1(1) makes the decision whether checking is to follow the elastic or plastic analysis procedure. The former route is treated in Decision Table X.2(2) and the latter route in Decision Table 8.5 (78).

The elastic analysis procedure treats structural elements according to one of the following.

(a) Structural Steel Member

(b) Girder

(c) Bearing Stiffener

(d) Intermediate Transverse Stiffener

(e) Connection

(f) Built-up Member

(g) Composite Member

(h) Open-Web Steel Joist

Structural steel members are dealt with starting from Decision Table X.3(3), and the chain of execution of the subsequent decision tables are as shown in Figs. A.2 to A.9. Fig. A.2 deals with axial compression or axial tension stresses, shear stresses and bending stresses. Figs. A.3 to A.4 deal with the classification of sections. Figs. A.5 to A.7 deal with pure bending stresses. Finally, Figs. A.8 and A.9 deal with combined stresses.

Fig. A.10 displays the chains of execution for girders, bearing stiffeners and intermediate transverse stiffeners. Fig. A.11 displays the tables required for checking of designs using plastic analysis.

In these heirachy charts, the full lines represent direct execution and the broken lines represent conditional execution. Sections of the Standard dealing with connections, built-up members, composite members, and open-web steel joists have not been decision tabled.

CHAPTER VII

THE SPECIFICATION PROCESSING PROGRAM

This chapter discusses the various operational modes of the processing program used in checking the constraints of the Standard, CSA-S16.1. The program will also be able to check constraints imposed by other standards (or codes), so long as they are presented in decision table form. The original program as developed by GOEL (12), limits execution to batch-mode only. It is also only capable of executing a given decision table once in a particular cycle. Therefore, the versatility of the processor is severely handicapped. Modified batch mode, interactive mode and recursive execution of a table within the same cycle are introduced in this chapter.

The processing program, both in batch mode and interactive mode, has been implemented on the IBM/360/67 at the University of Alberta computing centre which operates under the Michigan Terminal System (MTS).

7.1 Processing Procedure - Batch Mode

This section discusses the batch mode operation. The source program for this procedure is listed in Appendix E. The program in batch mode consists of the following subroutines, whose functions are described below. Their interrelationship and sequence of execution is illustrated schematically in Fig. 7.1

(a) MAIN Routine

(b) SETUP Subroutine

(c) INITIAL Subroutine

(d) INPUT Subroutine

(e) SETS Subroutine

(f) STAK Subroutine

(g) OUTPUT Subroutine

The MAIN Routine does all the constraints checking. The objective when checking design constraints is to identify the applicable rule in each decision table according to a given combination of conditions. This involves matching the condition stub with the corresponding condition entries. Before matching, all required data elements are checked for their availability. In batch mode, if a data element is not available from the data source discussed in Sect. 3.2, the program will take an error exit route aborting the run.

Subroutine SETUP reads the decision table input, the properties of data and creates temporary and compacted permanent arrays. The dependents of data elements are also generated by this subroutine.

Subroutine INITIAL which is called by subroutine SETUP, initialized all the arrays needed by the program.

Subroutine INPUT is used for reading the externally input data and for clearing of dependent data for the second or subsequent cycles. This subroutine is called by the MAIN ROUTINE.

Subroutine SETS is used to evaluate the data in mutually exclusive sets at the time of external input and is called by subroutine INPUT.

Subroutine STAK performs the stacking of decision tables for conditional execution and generates messages to that effect.

Subroutine OUTPUT outputs all the data elements which have a value at the end of each cycle for the purpose of checking and diagnosis.

7.2 Processing Procedure - Interactive Mode

This section discusses the procedure in implementing the interactive mode. In batch mode operation as discussed in the last section, control of the program takes the error exit route, terminating execution whenever a data element is missing and cannot be evaluated internally. This results in a large number of abortive runs which is both uneconomical and time (real time and CPU time) consuming. The interactive mode is implemented with these inefficiences in mind. The main advantage of this mode is that the user can input data items from the on-line terminal, as an alternative to termination of execution.

Instead of taking the error exit route for the reasons discussed above, the subroutine READIN is called by the MAIN ROUTINE to input the missing data element from the terminal. A message is printed on the terminal indicating the nature of the element. The main steps of the procedure are as follows and are also illustrated in Fig. 7.2.

(a) Read in missing data item (its subscript and value). Input a negative interger value for the subscript if the user wishes to terminate execution.

(b) Call subroutine SETS if the data item belongs to a mutually exclusive set.

(c) Clear the data item's dependents if not in the first cycle.

(d) Return to MAIN ROUTINE.

Illustrative worked examples and their computer output are presented in Chapter VIII. A detailed input procedure is also provided in the USER'S GUIDE in Appendix C. The source program listing of interactive mode procedure is presented in Appendix D.

7.3 Recycling Procedure

In a design problem, the checking of a number of design alternatives may be required before a final design is chosen. For this reason, a recycling facility has been incorporated into the processor with the interactive mode procedure. After the completion of cycle one execution, the program awaits the value of the variable INDIC from the terminal. A value of 2 for INDIC indicates there are no further cycles, whereas a value of 1 indicates a further cycle is required. The external data required for this next cycle can be input directly from the terminal. The limit of the number of cycles per run is 999.

In the second and subsequent cycles, a large number of data will be unchanged. Usually only a limited number of data elements will require alteration from the previous cycle. The clearing of dependents of the altered data elements is done internally in subroutine INPUT by setting their "flags" in the array PRD to .FALSE. . This clearing procedure is discussed in Sect. 7.4.

7.4 Clearing Of Dependent Data

The characteristics of ingredients and dependents discussed in Sect. 3.4 lead to the conclusion that when a data element changes its value, its dependents, if any, will change their values as well. Hence it becomes necessary to clear the dependents of a data element once it has an altered value. Clearing of a data element is done simply by setting its presence flag in array PRD to .FALSE. . From the ingredience-dependence relationship discussed, it can be deduced that it is possible for dependents to have their own dependents, down to a number of levels. During the dependent clearing process, these dependents of dependents will also have to be cleared. A data element stacking procedure is provided by the Subroutine STAK with the clearing process which is flow charted in Figs. 7.3 and 7.4.

7.5 Recursive Use Of Tables In A Cycle

There are numerous situations where it would be advantageous to execute a decision table more than once. For example, to calculate the class of several compression elements of a built-up member.

In the original program (12), such checking either had to be performed in separate runs or in different cycles.

Fig. 7.5 illustrates a scheme whereby decision tables can be written in such a way that they can be used recursively in one cycle with the existing processing program. The decision tables associated with section 13.8 of the Code, which checks the constraints for axial compression and bending, have been rewritten as decision tables 56R to 59R as shown in Appendix B, in order to illustrate the application and potential of this scheme. The scheme makes use of counters CHECKI and CHECKN to keep track of the number of times the decision table has been executed.

The steps of checking a Class 1, I-Section for axial compression and bending, as carried out by decision tables 56R to 59R, and as illustrated in Fig. 7.5, are as follows.

(a) Table 56R determines whether the problem is axial compression and bending or axial tension and bending. The value of counter CHECKI is initialized to zero.

(b) Since the section is a class 1, I-section. Rule number 1 applies in Table 57R, and the number of equations to be checked is 3. Hence, the value of CHECKN is set to 3, and Table 58R is to be directly executed.

(c) After the actions in Table 58R are completed, control returns to Table 57R and the next action is to check whether CHECKI > CHECKN. If not, CHECKI is again increased by one and the checking cycle is repeated. If yes, control returns to the MAIN ROUTINE.

In Decision Table 57R, note that the data elements specified in the data statement DATA MCLEAR/ /, and their dependents must be cleared by subroutine CLEAR before the table is recycled. Otherwise these data elements will still be present from the previous cycle and the program will not recognize that the data elements must be recomputed.

From Decision Table 58 and Decision Table 58R, the number of equations needed to be checked for axial compression and bending is reduced from three to two by the scheme discussed above. Thereby, reducing the number of overall data elements. Consequently, the total number of decision tables required should be reduced. This is of advantage since in a standard (or code) of reasonable size, the number of decision tables and data elements will be very large. Reduction in their quantity will lead to shorter processing time and reduced storage requirements.

CHAPTER VIII

APPLICATION EXAMPLES

Three examples are solved in this chapter to illustrate the use of the processor described in chapters V and VII in checking sections under different combinations of stress resultants. The data requirement, a trace of the execution of the decision tables and the relevant computer output (both from the on-line terminal and the line printer) of each example are presented together with supporting manual design calculations to verify the computer results.

8.1 Example 1 - Axially Loaded Column

Consider the column shown in Fig. 8.1 to be part of a frame-shear wall system. The 12 ft. column consists of two $8x6x\frac{5}{8}$ angles of G40.12 - 44W steel, supporting a maximum dead load of 300 kips and a maximum axial live load of 85 kips.

This example illustrates a typical program run in interactive mode. The output from the on-line terminal is presented in Fig. 8.2. The output from the line printer is presented in Figs. 8.3, 8.4, 8.5, 8.6. The bulk of the data information in the output are printed by the line-printer. In Fig. 8.2, statements in lower case letters are those input by the user, whereas statements in upper case letters are those output by the computer. The MTS (Michigan Terminal System) files, input and output units associate with the first statement ($RUN control command) are discussed in Appendix C: Users' Guide.

Since data items 222† and 223† are not available, and are not obtainable by executing any other tables, messages are then output on the terminal indicating the condition or action number of the table to which each data item corresponds. Subroutine READIN is called enabling the missing data items to be input directly from the terminal. Terminal data input can be of the free-format type which is of considerable advantage. The data item numbers (222 and 223) are input first followed by their values (0.0 in both cases).

Upon completion of checking, the design message "Strength Criterion Satisfied" is output on the terminal which indicates that the section is satisfactory for the imposed loading. At this stage, the execution of the program either terminates or re-executes another cycle depending on whether the variable <u>INDIC</u> is given a value of 1 or 2. In this example, a value of 2 is input from the terminal since no further cycles are required.

---

† For data description, see Appendix B.

Fig. 8.3 lists the externally input data items for cycle 1. "KGLOB" is the data number and "DATAK" is the value. These data items are read by Subroutine INPUT. When reading a data item, the subroutine simultaneously checks for whether it belongs to a mutually exclusive set. If so, the values of the other data items in the same set are set to 0.0. Fig. 8.4 lists the externally input data items together with their mutually exclusive set companions, if any, for the purpose of echo checking.

Fig. 8.5 shows a trace of all the decision tables which have been executed in cycle 1. Whenever the execution of a particular table is suspended, the reason and point of suspension are shown on this trace. Once the execution of each table is completed, the applicable rule number is indicated.

Finally, a design message "Strength Criterion Satisfied" is output indicating the section is satisfactory when supporting an axial compressive load of 385 kips. This message is also output on the terminal as shown in Fig. 8.2.

Fig. 8.6 lists the data items which have a value at the end of cycle 1. The correctness of their values is verified by the manual design calculations in Tables 1 and 2.

The significant data item in Fig. 8.6 is data number 150 which represent the ratio $R1 = P_f/P_{rc}$, where P_f is the factored axial compressive force and P_{rc} is the factored compressive resistance. The value of R1 = 0.8944 (< 1.0) indicates the section is satisfactory and not overdesigned. Hence a second cycle to choose a better section will not be necessary.

8.2 Example 2 - Laterally Unsupported Beam

A W12x50 wide - flange section, of G40.12-44W steel, is to be checked for suitability as a beam of length 24 feet. The section is laterally unsupported except at the ends. This beam is required to support a uniformally distributed dead load of 1.2 kip per foot, and a live load of 0.6 kip per foot.

This example is executed in interactive mode using the same procedure employed in Example 1. Figs. 8.7, 8.8, 8.9, 8.10, 8.11, 8.12, 8.13 present the computer output associated with this example. The content and significance of these figures are similar to those of Example 1 which have been discussed in detail with that example.

In Fig. 8.13, data number 151 which represents the ration $R2 = M_{fx}/M_{rx}$, where M_{fx} is the factored moment about the major axis and M_{rx} is the factored moment of resistance about the major axis, has a value of 0.9724 which indicates the section has sufficient moment capacity. The ratio V_f/V_r, where V_f is the factored shear force and V_r is the factored shear resistance, has a value of 0.24 which indicates the section is satisfactory in shear. The correctness of the data values in Fig. 8.13 is verified by the manual design calculations in Tables 3 to 5.

8.3 Example 3 - Axial Compression and Bending

A W10x49 wide-flange section, of G40.12-44W steel, is to be checked for suitability as a column in a building with a 9.5 foot storey height. The sway effects due to wind and other lateral loads are to be resisted by a bracing system. The dead and live load moments, and the axial compressive load on the member are shown in Fig. 8.14.

Figs. 8.15, 8.16, 8.17, 8.18, 8.19, 8.20, 8.21 present the computer output for cycle 1 checking. Upon the completion of this cycle, the section is found to be unsatisfactory both in strength and stability requirements as indicated by the message in Fig. 8.20. From the data values output at the end of cycle 1, as shown in Fig. 8.21, the values of the following three equations are found to be:

(i) $$\frac{M_{fx}}{M_{rx1}} + \frac{M_{fy}}{M_{ry1}} = 1.0017 \; (> 1.0)$$

(ii) $$\frac{P_f}{P_{rc2}} + \frac{0.85\ M_{fx}}{M_{rx1}} + \frac{0.6\ M_{fy}}{M_{ry1}} = 0.9888 \; (< 1.0)$$

(iii) $$\frac{P_f}{P_{rc1}} + \frac{\omega_x\ M_{fx}}{M_{rx2}(1-P_f/c_{ex})} + \frac{\omega_y\ M_{fy}}{M_{ry1}(1-P_f/c_{ey})}$$

$$= 1.0916 \; (> 1.0)$$

These values are verified by the manual design calculations in Tables 6 to 9.

Since the section W10x49 is found to be unsatisfactory in cycle 1, a second cycle to test a heavier section (W10x66) is desired. This can be accomplished without termination of the program run. A value of 1 for the variable INDIC is input from the terminal (Fig. 8.15), indicating cycle 2 is required.

The externally input data items for this cycle are then input next via the terminal using the free-format type of input (Fig. 8.15). The amount of cycle 2 data is considerably less than that for cycle 1 since it only consists of data items whose values have been changed or new data items which the user wishes to add. For example, data number 59 (flange width) changed from 5.0

inch to 5.06 inch since the section has changed to a W10x66. Cycle 2 data items are reproduced on Fig. 8.22 for echo checking.

Once the program receives the data for this cycle, the checking procedure which follows is identical to that for cycle 1. The design message and data values shown in Figs. 8.26 and 8.27 at the end of this cycle indicate the section is satisfactory for the imposed loading. From Fig. 8.27, the values of equations (i), (ii) and (iii) are found to be 0.6999, 0.6984, and 0.8749 respectively. The correctness of these values is verified by the design calculations in Tables 6 to 12. The value of 0.87 for equation (iii) indicates the section is satisfactory but not overdesigned.

DESIGN CALCULATION FOR EXAMPLE 1 AXIAL COMPRESSION MEMBER	SHEET 1 OF 2

LOADING

$P_D = 300$ kips $\quad \gamma = 0.9 \quad \alpha_T = 1.25$

$P_L = 85$ kips $\quad \alpha_D = 1.25 \quad \psi = 1.0$

$P_Q = 0 \quad \alpha_L = 1.5$

$P_T = 0 \quad \alpha_Q = 1.5$

FACTORED COMPRESSIVE FORCE

$$P_f = \gamma[\alpha_D P_D + \psi(\alpha_L P_L + \alpha_Q P_Q + \alpha_T P_T)]$$ (CLAUSE 7.2.2)

$$= 0.9[1.25 \times 300 + 1.0(1.5 \times 85)]$$

$$= 452.25 \text{ kips}$$

COLUMN SECTION $\quad 2 - \angle 8 \times 6 \times 5/8$

G40.12 - 44 W

$L = 12$ ft. $\quad r_x = 2.54$ in $\quad S_x = 19.7$ in^3

$A_g = 16.7$ in^2 $\quad r_y = 2.42$ in

CHECK CLASS OF SECTION

$b/t = 8/.63 = 12.7 \quad \therefore$ CLASS $= 3$ (CLAUSE 11)

SWAY FORCES RESISTED BY SHEAR WALL

$\therefore K_x = K_y = 1.0$ (CLAUSE 9.3.2)

$$\frac{K_x L_x}{r_x} = \frac{1.0 \times 144}{2.54} = 56.69$$

$$\frac{K_y L_y}{r_y} = \frac{1.0 \times 144}{2.42} = 59.5$$ (this Governs)

TABLE 1 DESIGN CALCULATION FOR EXAMPLE 1

DESIGN CALCULATION FOR EXAMPLE 1 AXIAL COMPRESSION MEMBER	SHEET 2 OF 2

$\lambda = \frac{KL}{r}\sqrt{\frac{F_y}{\pi^2 E}} = 59.5\sqrt{\frac{44}{\pi^2 \times 29000}} = 0.7377$ (CLAUSE 13.3.1)

$P_{rc} = \phi A_g F_y \,(1.035 - 0.201\,\lambda - 0.224\,\lambda^2)$

$= 0.9 \times 16.7 \times 44\,(1.035 - 0.201 \times 0.7377 - 0.224 \times 0.7377^2)$

$= 505.79$ Kips

$R1 = P_f / P_{rc}$

$= 452.25 / 505.79$

$= 0.8941 \quad < (1.0)$

∴ STRENGTH CRITERION SATISFIED
SECTION SATISFACTORY

TABLE 2 DESIGN CALCULATION FOR EXAMPLE 1

DESIGN CALCULATION FOR EXAMPLE 2 LATERALLY UNSUPPORTED BEAM	SHEET 1 OF 3

LOADINGS

DEAD LOAD = 1.2 kips/ft.
LIVE LOAD = 0.6 kips/ft.

$$M_{xD} = \frac{1.2 \times 24^2 \times 12}{8} = 1036.8 \text{ in-kip}$$

$$M_{xL} = \frac{0.6 \times 24^2 \times 12}{8} = 518.4 \text{ in-kip}$$

$M_{xQ} = 0$, $M_{xT} = 0$

LOAD FACTORS

$\gamma = 0.9$, $\alpha_D = 1.25$, $\alpha_L = 1.5$, $\alpha_Q = 1.5$, $\alpha_T = 1.25$, $\psi = 1.0$

$$M_{fx} = \gamma[\alpha_D M_{xD} + \psi(\alpha_L M_{xL} + \alpha_Q M_{xQ} + \alpha_T M_{xT})]$$
$$= 0.9[1.25 \times 1036.8 + 1.0(1.5 \times 518.4)]$$
$$= 1866.24 \text{ in-kip}$$

CHECK W12x50 SECTION , G40.12 - 44W STEEL

SECTION PROPERTIES

$F_y = 44$ ksi , $E = 29000$ ksi

$d = 12.19$ in	$w = 0.371$ in	$I_y = 56.4$ in^4
$b = 4.04$ in	$Z_x = 72.5$ in^3	$A_f = b \times t = 5.18$ in^2
$t = 0.641$ in	$S_x = 64.8$ in^3	$A_t = A_f + \frac{1}{6} w(d - 2t) = 5.85$ in^2

$I_t = I_y/2 = 56.4/2 = 28.2$ in^4 (NEGLECTING WEB CONTRIBUTION)

$$r_t = \sqrt{\frac{I_t}{A_t}} = \sqrt{\frac{28.2}{5.85}} = 2.2 \text{ in}$$

TABLE 3 DESIGN CALCULATION FOR EXAMPLE 2

DESIGN CALCULATION FOR EXAMPLE 2 LATERALLY UNSUPPORTED BEAM	SHEET 2 OF 3

CHECK SECTION CLASS

$b/t = 4.04/0.641 = 6.30 < 54/\sqrt{F_y}$ ∴ FLANGE CLASS = 1 (CLAUSE 11)

$h/w = (d-2t)/w = \dfrac{12.19 - 2\times 0.641}{0.371}$

$= 29.4 < 420/\sqrt{F_y}$ ∴ Web CLASS = 1 (CLAUSE 11)

∴ CLASS OF SECTION = 1

CALCULATE M_u

$\sigma_1 = \dfrac{20000}{Ld/A_f} = \dfrac{20000\times 5.18}{24\times 12\times 12.19} = 29.51$

$\sigma_2 = \dfrac{250000}{(L/r_t)^2} = \dfrac{250000\times 2.2^2}{(24\times 12)^2} = 14.59$

$\omega_x = 1.0$ (CLAUSE 13.8.3(a))

$M_u = \dfrac{S_x}{\omega_x}\sqrt{\sigma_1^2 + \sigma_2^2} = \dfrac{64.8}{1.0}\sqrt{29.51^2 + 14.59^2}$ (CLAUSE 13.6.1)

$= 2133.2$ in-kip

$M_p = Z_x F_y = 7.25\times 44 = 3190$ in-kip

∴ $M_u > 2/3\ M_p$

$M_{rx} = 1.15\,\phi M_p\left(1 - \dfrac{0.28\,M_p}{M_u}\right)$ (CLAUSE 13.6.1(a))

$= 1.15\times 0.9\times 3190\left(1 - \dfrac{0.28\times 3190}{2133.2}\right) = 1919.2$ in-kip

∴ $R2 = M_{fx}/M_{rx} = 1866.24/1919.2 = 0.97 < (1.0)$

∴ **STRENGTH CRITERION SATISFIED**

TABLE 4 DESIGN CALCULATION FOR EXAMPLE 2

DESIGN CALCULATION FOR EXAMPLE 2 LATERALLY UNSUPPORTED BEAM	SHEET 3 OF 3

CHECK SHEAR

$$V_D = \frac{1.2 \times 24}{2} = 14.4 \text{ Kips}$$

$$V_L = \frac{0.6 \times 24}{2} = 7.2 \text{ Kips}$$

$V_Q = 0$, $V_T = 0$

$$V_f = 0.9\left[1.25 \times 14.4 + 1.0(1.5 \times 7.2)\right]$$
$$= 25.92 \text{ kips}$$

$$h/w = 29.4 < 167\sqrt{\frac{K_v}{F_y}} = 58.18$$ (CLAUSE 13.4.1)

($K_v = 5.34$ for unstiffened webs)

$\therefore$ $F_s = 0.66 F_y$

$$V_r = \phi A_w F_s = 0.9 \times 4.05 \times 0.66 \times 44$$
$$= 105.85 \text{ Kips}$$

$$\frac{V_f}{V_r} = \frac{25.92}{105.85} = 0.24 \ (<1.0)$$

$\therefore$ SHEAR CRITERION SATISFIED

W12x50
SECTION SATISFACTORY

TABLE 5 DESIGN CALCULATION FOR EXAMPLE 2

DESIGN CALCULATION FOR EXAMPLE 3 AXIAL COMPRESSION AND BENDING	SHEET 1 OF 7

LOADING

$P_D = 90$ kips	$M_{xD} = 0$	$M_{yD} = 0$
$P_L = 21$ kips	$M_{xL} = 1067$ in-kip	$M_{yL} = 297$ in-kip
$P_Q = 0$	$M_{xQ} = 0$	$M_{yQ} = 0$
$P_T = 0$	$M_{xT} = 0$	$M_{yT} = 0$

LOAD FACTORS

$\gamma = 0.9$, $\alpha_D = 1.25$, $\alpha_L = 1.5$, $\alpha_Q = 1.5$ $\alpha_T = 1.25$, $\psi = 1.0$

$P_f = 0.9[1.25 \times 90 + 1.0(1.5 \times 21)] = 129.6$ kips

$M_{fx} = 0.9 \times 1.5 \times 1067 = 1440.45$ in-kips

$M_{fy} = 0.9 \times 1.5 \times 890 = 400.95$ in-kips

CYCLE NUMBER 1

CHECK W10X49 SECTION G40.12 – 44 W STEEL

SECTION PROPERTIES

$d = 10.0$ in	$Z_x = 60.3$ in^3	$A_f = 5.58$ in^2
$b = 5.0$ in	$S_x = 54.6$ in^3	$I_t = I_y/2 = 46.5$ in^4
$t = 0.558$ in	$A_g = 14.4$ in^2	$A_t = A_f + 1/6\, w(d-2t)$
$w = 0.34$ in	$r_x = 4.35$ in	$= 6.08$ in^2
$h = 8.88$ in	$r_y = 2.54$ in	$r_t = \sqrt{I_t/A_t} = 2.77$ in
$F_y = 44$ ksi ,	$E = 29000$ ksi	$S_y = 18.6$ in^3
		$Z_y = 28.2$ in^3

CHECK SECTION CLASS

$P_f/C_y = 129.6/14.4 \times 44 = 0.20 > 0.15$

$b/t = 5.0/0.558 = 8.96 \left(> \frac{54}{\sqrt{F_y}}, < \frac{64}{\sqrt{F_y}}\right)$ $\therefore$ FLANGE CLASS = 2

$h/w = 8.88/0.34 = 26.12 \left(< \frac{420}{\sqrt{F_y}}(1 - 1.4\, P_f/C_y)\right)$ $\therefore$ WEB CLASS = 1

$\therefore$ SECTION CLASS = 2

TABLE 6 DESIGN CALCULATION FOR EXAMPLE 3

DESIGN CALCULATION FOR EXAMPLE 3 AXIAL COMPRESSION AND BENDING	SHEET 2 OF 7

CALCULATION OF RESISTANT FORCES

SWAY FORCES IN BOTH DIRECTIONS RESISTED BY BRACING SYSTEMS

$\therefore \quad K_x = K_y = 1.0$ (CLAUSE 9.3)

$L_x = L_y = 114$ in

$$UKLR = MAX\left(\frac{k_x L_x}{r_x}, \frac{k_y L_y}{r_y}\right)$$

$$= MAX\left(\frac{1.0 \times 114}{4.35}, \frac{1.0 \times 114}{2.54}\right) = 44.88$$

$$\therefore \lambda 1 = UKLR \sqrt{\frac{F_y}{\pi^2 E}}$$ (CLAUSE 13.3.1)

$$= 44.88 \sqrt{\frac{44}{\pi^2 \times 29000}} = 0.557$$

$$P_{rc1} = \phi A_g F_y (1.035 - 0.201\lambda 1 - 0.224 \lambda 1^2)$$ (CLAUSE 13.3.1)

$$= 0.9 \times 14.4 \times 44 (1.035 - 0.201 \times 0.557 - 0.224 \times 0.557^2)$$

$= 486.78$ kips

$P_{rc2} = \phi A_g F_y = 0.9 \times 14.4 \times 44 = 570.24$ Kips (CLAUSE 13.8.2(ii))

$$M_{p(reduced)} = 1.18 M_p (1 - P_f / C_y)$$

$$= 1.18 \times 44 \times 60.3 \left(1 - \frac{129.6}{14.4 \times 44}\right)$$

$= 2490.39$ in-kip

$M_{rx1} = \phi M_{p(reduced)}$

$= 0.9 \times 2490.39$

$= 2241.35$ in-kip

$M_{ry1} = 1116.7$ in-kip (CLAUSE 13.6.1)

TABLE 7 DESIGN CALCULATION FOR EXAMPLE 3

DESIGN CALCULATION FOR EXAMPLE 3 AXIAL COMPRESSION AND BENDING	SHEET 3 OF 7

$$M_u = \frac{S_x}{\omega_x}\sqrt{\sigma_1^2 + \sigma_2^2}$$ (CLAUSE 13.6.1)

$$\sigma_1 = \frac{20000}{Ld/A_f} = \frac{20000 \times 5.58}{114 \times 10} = 97.89$$

$$\sigma_2 = \frac{250000}{(L/r_t)^2} = \frac{250000 \times 2.77^2}{114^2} = 147.6$$

$\omega_x = 1.0$

$\omega_y = 0.85$ (CLAUSE 13.8.3.(a)(ii))

$$M_u = \frac{54.6}{1.0}\sqrt{97.89^2 + 147.6^2} = 9670.3 \text{ in-kip}$$

$M_u > 2/3\ M_p$ (reduced)

$$\therefore M_{rx2} = 1.15\,\phi M_{p(reduced)}\left(1 - \frac{0.28\,M_{p(reduced)}}{M_u}\right)$$ (CLAUSE 13.6.1)

$$= 1.15 \times 0.9 \times 2490.39\left(1 - \frac{0.28 \times 2490.39}{9670.3}\right)$$

$= 2391.69$ in-kip $> \phi M_{p(reduced)}$

$$\therefore M_{rx2} = \phi M_{p(reduced)} = 0.9 \times 2490.39$$
$$= 2241.35 \text{ in-kip}$$

$$C_{ex} = \frac{286000\,A_g}{(K_x L_x/r_x)^2} = 5995.07$$

$$C_{ey} = \frac{286000\,A_g}{(K_y L_y/r_y)^2} = 2044.67$$

TABLE 8 DESIGN CALCULATION FOR EXAMPLE 3

DESIGN CALCULATION FOR EXAMPLE 3
AXIAL COMPRESSION AND BENDING

SHEET 4 OF 7

$$\frac{M_{fx}}{M_{rxL}} + \frac{M_{fy}}{M_{ryL}} = \frac{1440.45}{2241.35} + \frac{400.95}{1116.7} \quad \text{(CLAUSE 13.8.2 (i))}$$

$$= 0.6427 + 0.359 = 1.0017 \ (>1.0)$$

STRENGTH CRITERION NOT SATISFIED

$$\frac{P_f}{P_{rcL}} + \frac{0.85 M_{fx}}{M_{rxL}} + \frac{0.6 M_{fy}}{M_{ryL}}$$

$$= \frac{129.6}{570.24} + \frac{0.85 \times 1440.45}{2241.35} + \frac{0.6 \times 400.95}{1116.7} \quad \text{(CLAUSE 13.8.2 (ii))}$$

$$= 0.2273 + 0.5463 + 0.2153 = 0.9888 \ (<1.0)$$

STRENGTH CRITERION SATISFIED

$$\frac{P_f}{P_{rcL}} + \frac{\omega_x M_{fx}}{M_{rx2}(1 - P_f/C_{ex})} + \frac{\omega_y M_{fy}}{M_{ryL}(1 - P_f/C_{ey})}$$

$$= \frac{129.6}{486.78} + \frac{1.0 \times 1440.45}{2241.35(1 - 129.6/5995.07)} + \frac{0.85 \times 400.95}{1116.7(1 - 129.6/2044.67)}$$

$$= 0.27 + 0.6593 + 0.1623$$

$$= 1.0916 \ (>1.0)$$

STABILITY CRITERION NOT SATISFIED

W10x49 SECTION IS NOT SATISFACTORY

TABLE 9 DESIGN CALCULATION FOR EXAMPLE 3

DESIGN CALCULATION FOR EXAMPLE 3 AXIAL COMPRESSION AND BENDING	SHEET 5 OF 7

CYCLE NUMBER 2

CHECK W10x66 G40.12-44W STEEL SECTION

SECTION PROPERTIES

$d = 10.38$ in	$h = 8.88$ in	$r_x = 4.44$ in
$b = 5.06$ in	$S_x = 73.6$ in^3	$r_y = 2.58$ in
$t = 0.748$ in	$A_g = 19.4$ in^2	$A_f = 7.57$ in^2
$w = 0.457$ in	$Z_x = 82.8$ in^3	$r_b = 2.8$ in

CHECK SECTION CLASS

$P_f/C_y = 129.6/853.6 = 0.1518\ (>0.15)$

$b/t = 5.06/0.748 = 6.76\ (< 54/\sqrt{F_y} = 8.14)$ FLANGE CLASS = 1

$h/w = 8.88/0.457 = 19.43\ \left(< \frac{420}{\sqrt{F_y}}(1-1.4\ P_f/C_y)\right)$ WEB CLASS = 1

$\therefore$ CLASS OF SECTION = 1

CALCULATION OF RESISTANT FORCES

$$UKLR = MAX\left(\frac{k_x L_x}{r_x}, \frac{k_y L_y}{r_y}\right)$$
$$= MAX(25.68, 44.19)$$
$$= 44.19$$

$$\lambda 1 = 44.19\sqrt{\frac{44}{\pi^2 \times 29000}} = 0.548$$

$$P_{rc1} = \phi A_g F_y (1.035 - 0.201\lambda 1 - 0.224\lambda 1^2)$$
$$= 0.9 \times 19.4 \times 44 (1.035 - 0.201 \times 0.548 - 0.224 \times 0.548^2)$$
$$= 658.83 \text{ Kips}$$

$$P_{rc2} = \phi A_g F_y = 0.9 \times 19.4 \times 44$$
$$= 768.24 \text{ kips}$$

TABLE 10 DESIGN CALCULATION FOR EXAMPLE 3

DESIGN CALCULATION FOR EXAMPLE 3 AXIAL COMPRESSION AND BENDING	SHEET 6 OF 7

$$M_{p(reduced)} = 1.18 \times M_p \, (1 - P_f/C_y)$$
$$= 1.18 \times 3643.2\,(1 - 0.1518)$$
$$= 3646.27 \text{ in-Kip}$$

$$M_{rx1} = \phi M_{p(reduced)} = 0.9 \times 3646.27 = 3281.64 \text{ in-Kip}$$

$$\sigma_1 = 20000/(Ld/A_f) = \frac{20000 \times 7.57}{114 \times 10.38} = 127.95$$

$$\sigma_2 = 250000/(L/r_t)^2 = \frac{250000 \times 2.8^2}{114^2} = 150.82$$

$$M_u = \frac{S_x}{\omega_x}\sqrt{\sigma_1^2 + \sigma_2^2} = \frac{73.6}{1.0}\;127.95^2 + 150.82^2 = 14556.3 \text{ in-Kip}$$

$$M_u > 2/3 \; M_{p(reduced)}$$

$$\therefore M_{rx2} = 1.15\phi\, M_{p(reduced)}\left(1 - \frac{0.28\, M_{p(reduced)}}{M_u}\right)$$
$$= 1.15 \times 0.9 \times 3646.27\left[1 - \frac{0.28 \times 3646.27}{14556.3}\right]$$
$$= 3509.19 \text{ in-Kip} > (\phi M_{p(reduced)})$$

$$\therefore M_{rx2} = \phi M_{p(reduced)} = 3281.64 \text{ in-kip}$$

$$C_{ex} = \frac{286000 \times 19.4}{25.68^2} = 8416.3$$

$$C_{ey} = \frac{286000 \times 19.4}{44.19^2} = 2841.3$$

TABLE 11 DESIGN CALCULATION FOR EXAMPLE 3

DESIGN CALCULATION FOR EXAMPLE 3 AXIAL COMPRESSION AND BENDING	SHEET 7 OF 7

$$\frac{M_{fx}}{M_{rx1}} + \frac{M_{fy}}{M_{ry1}} = \frac{1440.45}{3281.64} + \frac{400.95}{1536.48} = 0.4398 + 0.26$$
$$= 0.6999 \quad (<1.0)$$

STRENGTH CRITERION SATISFIED

$$\frac{P_f}{P_{rc1}} + \frac{0.85 M_{fx}}{M_{rx1}} + \frac{0.6 M_{fy}}{M_{ry1}} = \frac{129.6}{768.24} + \frac{0.85 \times 1440.45}{3281.64} + \frac{0.6 \times 400.95}{1536.48}$$
$$= 0.1687 + 0.3731 + 0.1566$$
$$= 0.6984 \quad (<1.0)$$

STRENGTH CRITERION SATISFIED

$$\frac{P_f}{P_{rc1}} + \frac{\omega_x M_{fx}}{M_{rx2}(1 - P_f/C_{ex})} + \frac{\omega_y M_{fy}}{M_{ry1}(1 - P_f/C_{ey})}$$
$$= \frac{129.6}{658.83} + \frac{1.0 \times 1440.45}{3281.64(1 - 129.6/8416.3)} + \frac{0.85 \times 400.95}{1536.48(1 - 129.6/2841.3)}$$
$$= 0.1967 + 0.4458 + 0.2324$$
$$= 0.8749 \quad (<1.0)$$

STABILITY CRITERION SATISFIED

SECTION W10x66 SATISFACTORY

TABLE 12 DESIGN CALCULATION FOR EXAMPLE 3

CHAPTER IX

SUMMARY AND CONCLUSIONS

Currently in engineering, the development of computer programs in analysis is much more advanced than that in design and decision processing. The computer will be an even more valuable tool to engineers if it can be used for implementing decisions to a greater extent.

A large part of the design process in structural engineering is devoted to satisfying specific requirements of the codes or specifications of different authorities. This procedure largely involves checking of logical conditions, arising from numerical calculations.

The method of tabular decision logic used in this thesis is found to be a suitable technique for formulating, displaying, and documenting the decision making procedures required by codes and specifications. Since this technique presents decision making procedures in a logical fashion, it is also useful for implementing these procedures on the computer.

Most computer-aided structural design programs in use at the present time, such as the CISC - Column Selection Program (3), and the CISC - Floor System Selection Program (4), have the code requirements built into the analysis and member selection subroutines.

Since code requirements are often subjected to revisions, revising and updating these programs becomes a tedious task.

The decision table technique used herein to compile sections of the new CSA S16.1 - Steel Structures for Buildings - Limit States Design Standard renders the code requirement checking operation into an individual module entirely independent from the analysis and member selection procedure. In this way, any further code revisions can easily be handled by changing the relevant decision tables and their associated condition and action subroutines (Sect. 5.4). This procedure is schematically illustrated in Fig. 9.1.

In Chapter VIII, a number of example problems are checked against CSA S16.1 using the decision tables compiled in this thesis. The validity of the tables is checked by manual calculations. The design decisions incorporated into the decision tables are completely objective and according to the requirements of the Standard.

Conditional execution has been used extensively in the decision table arrangement. The main reason for this approach is that in conditional execution, not every data item required by a particular decision table need be available before the execution can commence. This is in contrast to the direct execution approach which requires the presence of all the data items appearing in the table regardless of whether they are actually used. In addition, often due to the presence of immaterial conditions in the condition entries of a table, certain conditions may not have to be tested at all in order to locate the governing rule. The precomputation of such data elements is, therefore, wasteful. Hence, this approach is inefficient and results in a large number of redundant data items.

An interactive mode procedure has been developed herein which enables the user to execute program runs and input data items from an on-line terminal remote from the physical location of the computer. This mode also facilitates the input of second or subsequent cycles of data immediately after the result of the previous cycle is known.

A pilot scheme for the recursive execution of tables has been introduced. This leads to the possibility that a considerable saving in the total number of data elements and decision tables may be achieved.

There are a number of areas in which improvements or further developments of the work described in this thesis can be made. Some of these are:

(a) To complete the documentation of CSA S16.1, in decision table format, so that it can be incorporated into analysis and design programs.

(b) To associate the code requirement checking processor with a module which sizes and selects members according to design criteria so that the design process can be more fully automated.

(c) The recursive execution of tables within a cycle increases the complexity of the ingredience-dependence relationship of data elements. A more efficient dynamic concept of ingredience and dependence may be desirable.

(d) The input and output may be improved to be more user orientated. If the output and input of data elements can be referenced by name as well as subscript, it will undoubtly be helpful to the user when interpreting output results. It would also be practical if the user could select whichever data element he desires to be output on the terminal for checking and reviewing.

(e) In a completely integrated scheme, the checking program should have a provision which automatically returns control back to the member selection module if a section does not satisfy the code requirements or is overdesigned, as illustrated in Fig. 9.1. Furthermore, improvements can be made in the interactive mode to provide a flexible system which allows the designer to 'break in' to a cycle at any point he desires.

FIGURES

CONDITION STUB	CONDITION ENTRIES	INGREDIENT LIST OF EACH CONDITION
ACTION STUB	ACTION ENTRIES	INGREDIENT LIST OF EACH ACTION

FIG. 2.1 BASIC COMPONENTS OF A DECISION TABLE

CONDITION 1	Y	N	N	Y
CONDITION 2	N	Y	N	Y
ACTION 1	Y			
ACTION 2		Y	Y	
ACTION 3				Y

FIG. 2.2 A COMPLETE DECISION TABLE

CONDITION 1	Y	N	Y
CONDITION 2	N	I	Y
ACTION 1	Y		
ACTION 2		Y	
ACTION 3			Y

FIG. 2.3 A REDUCED SIZE DECISION TABLE

Definition of data item	External input flag	Table number from which value of data can be obtained

FIG. 2.4 DATA REQUIREMENTS OF A DECISION TABLE

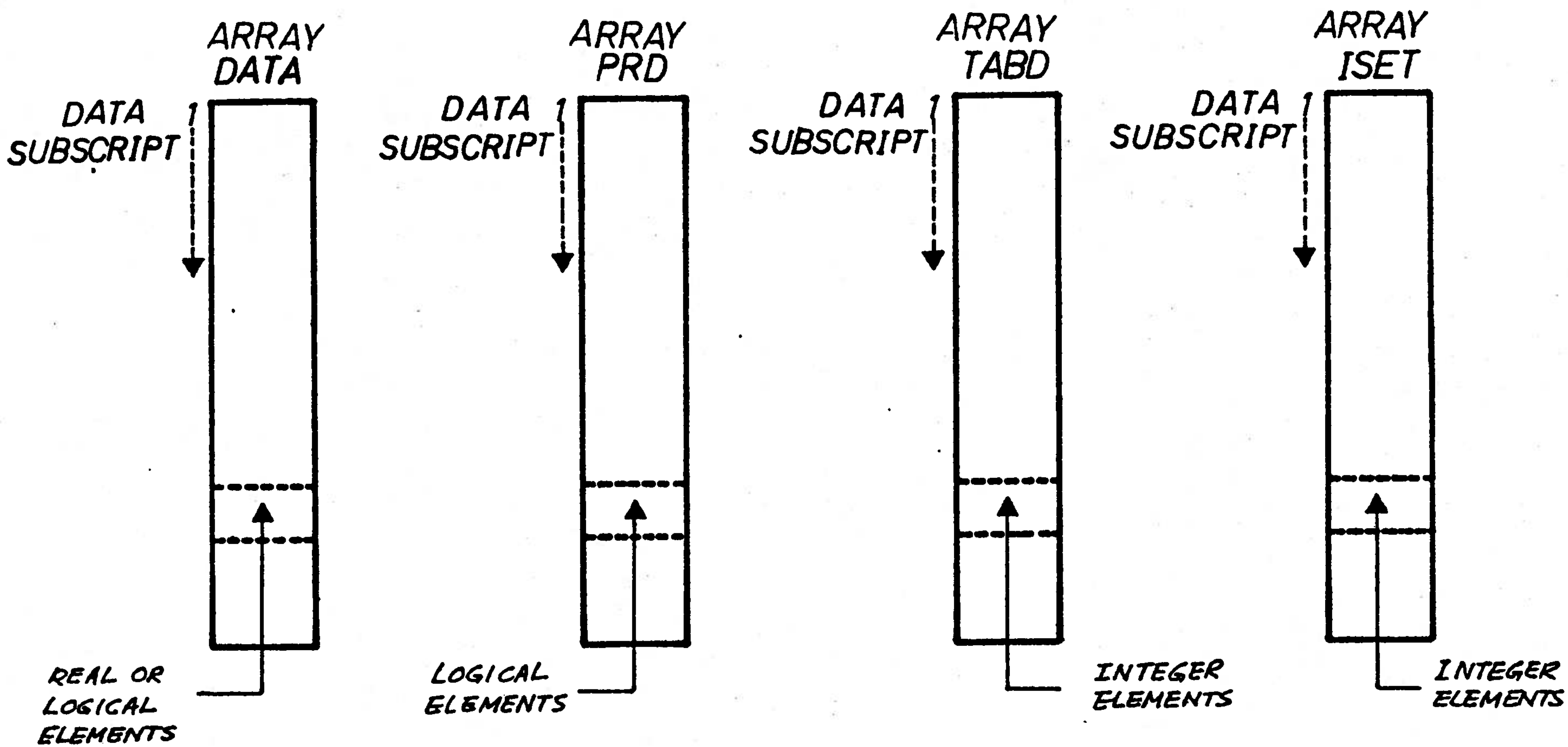


FIG. 3.1 GLOBAL ARRAYS

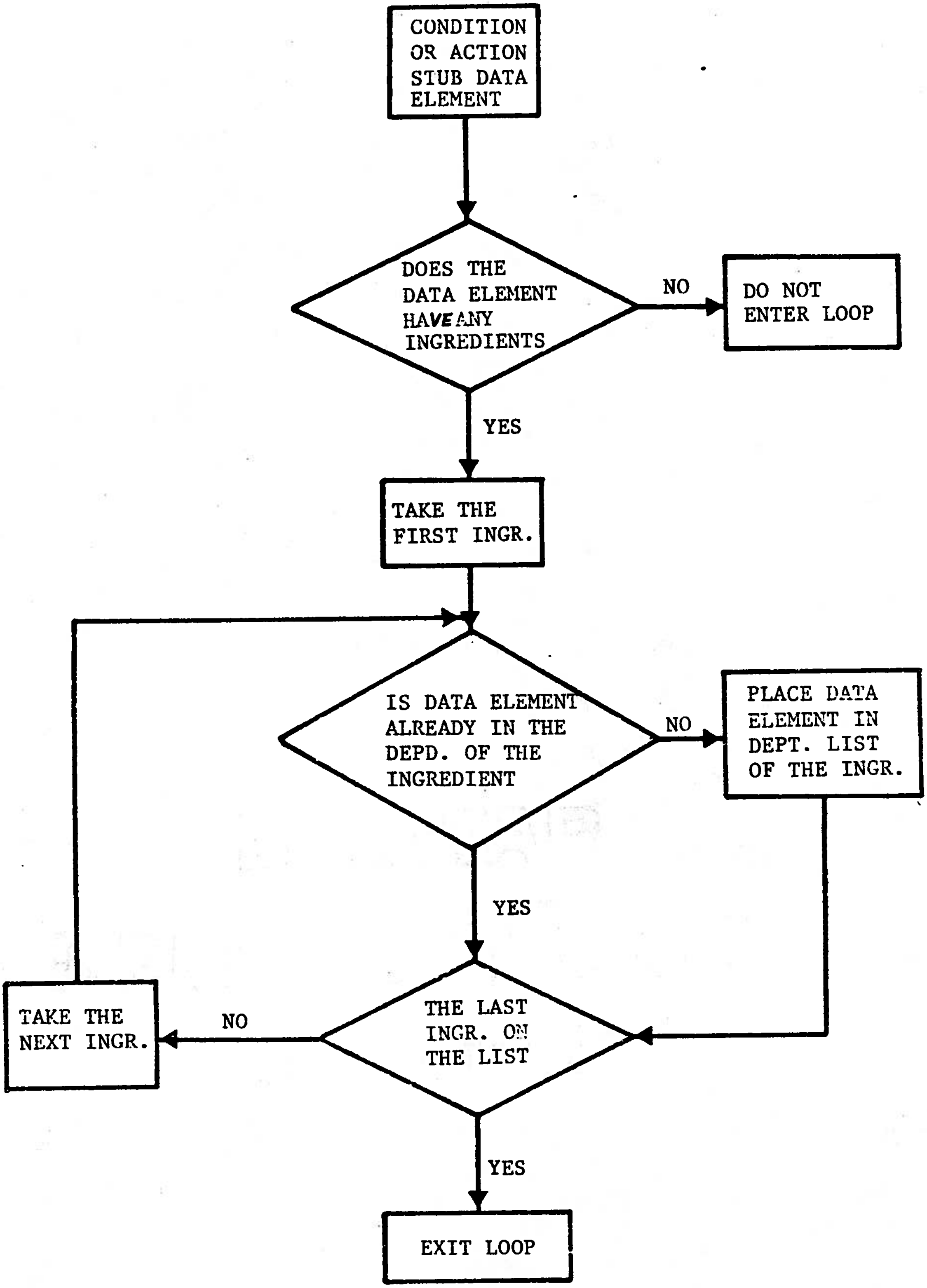


FIG. 3.2 GENERATING DEPENDENTS FROM INGREDIENTS OF A CONDITION OR AN ACTION

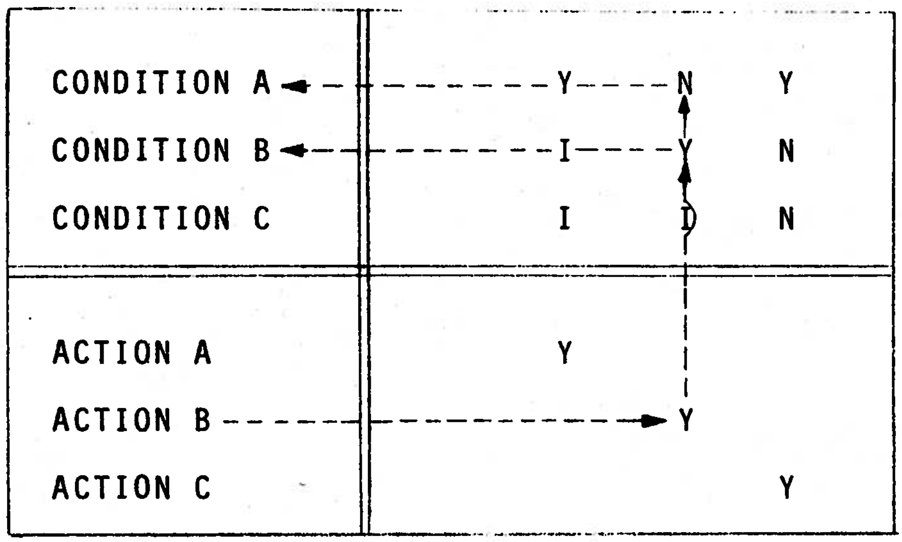


FIG 3.3 ILLUSTRATION OF DEPENDENCE CONCEPT BY THE LOGIC OF DECISION TABLES

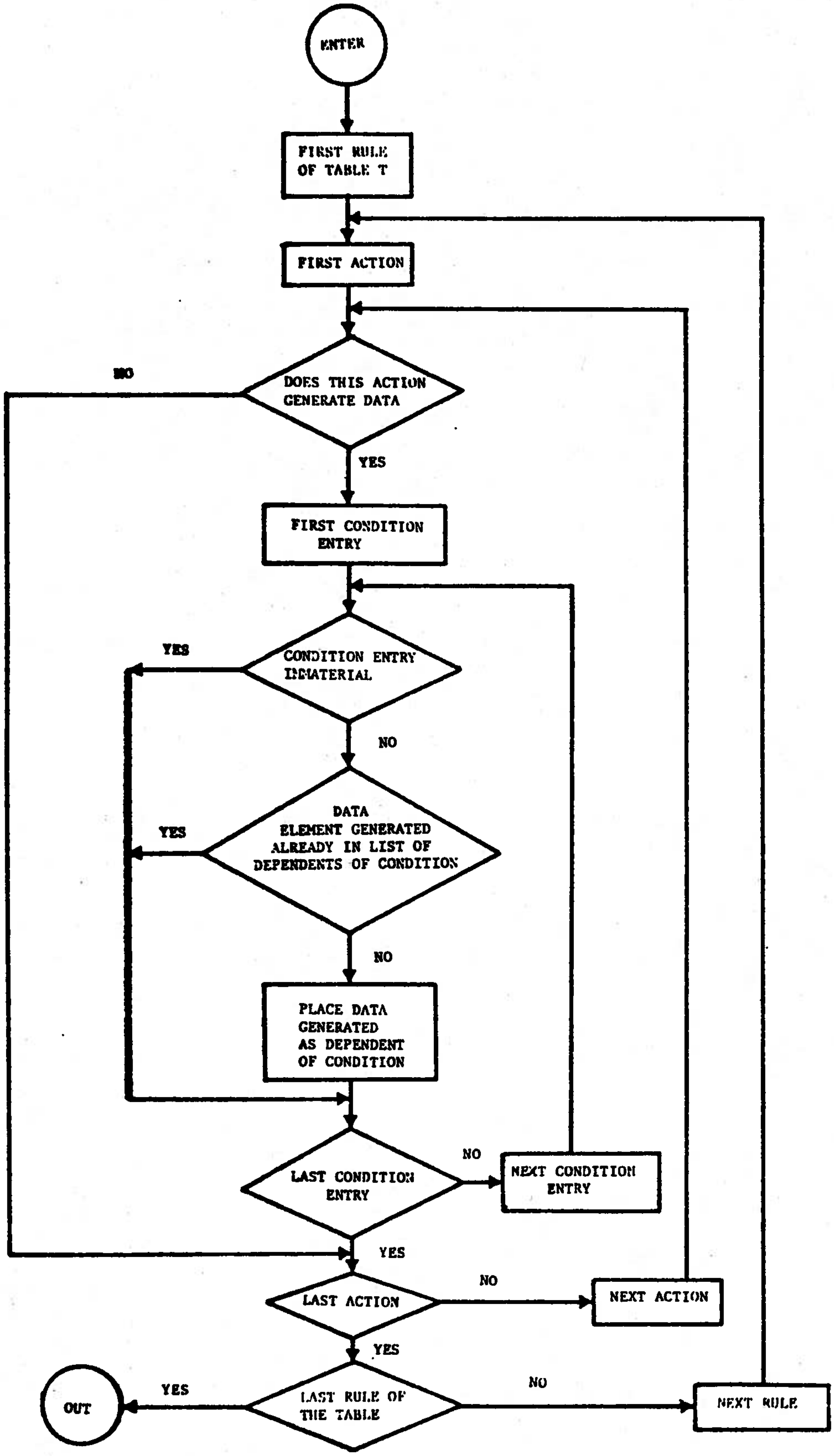


FIG. 3.4 FLOW CHART FOR GENERATING DEPENDENTS BY THE LOGIC OF DECISION TABLES

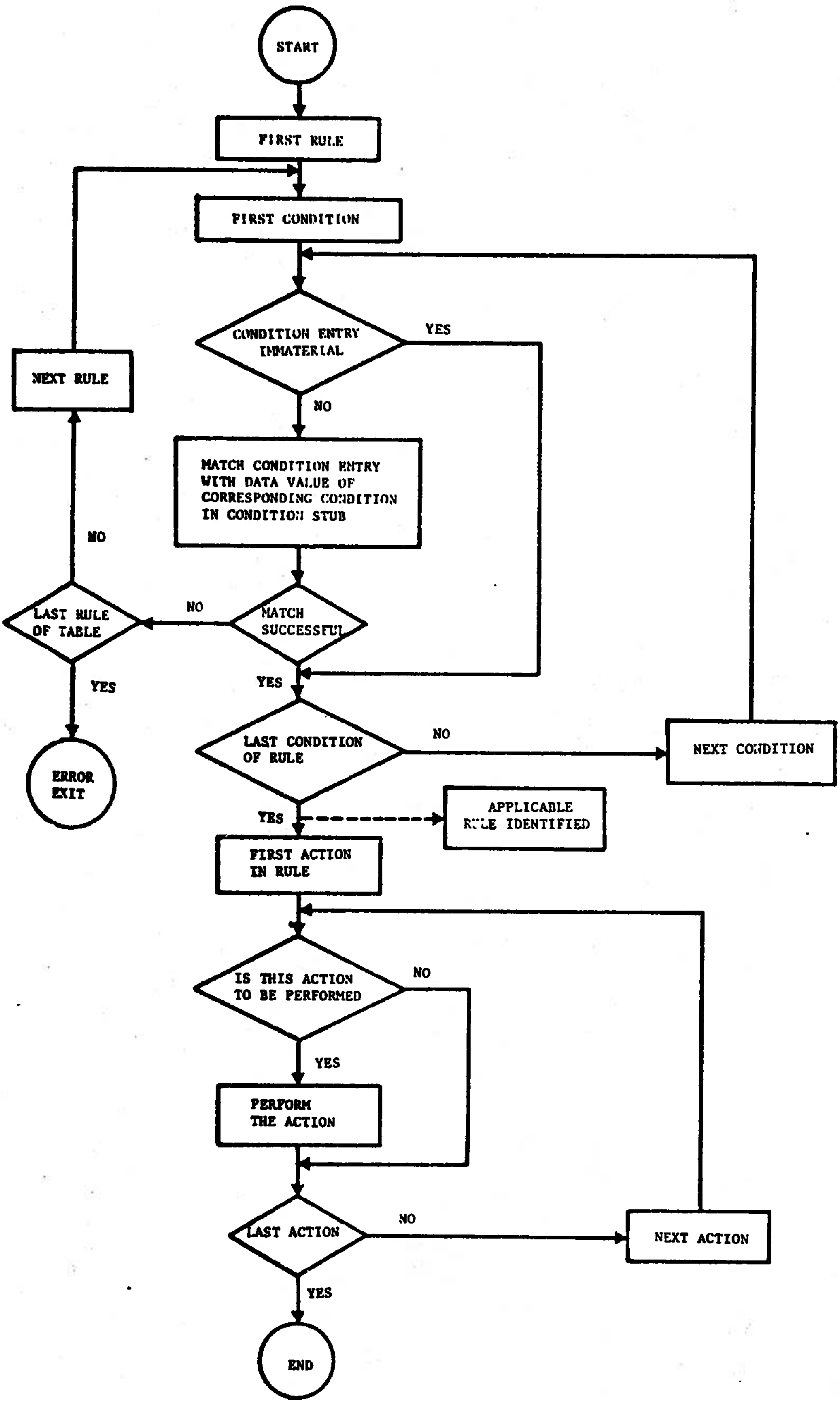


FIG. 4.1 PROCESSING OF A DECISION TABLE

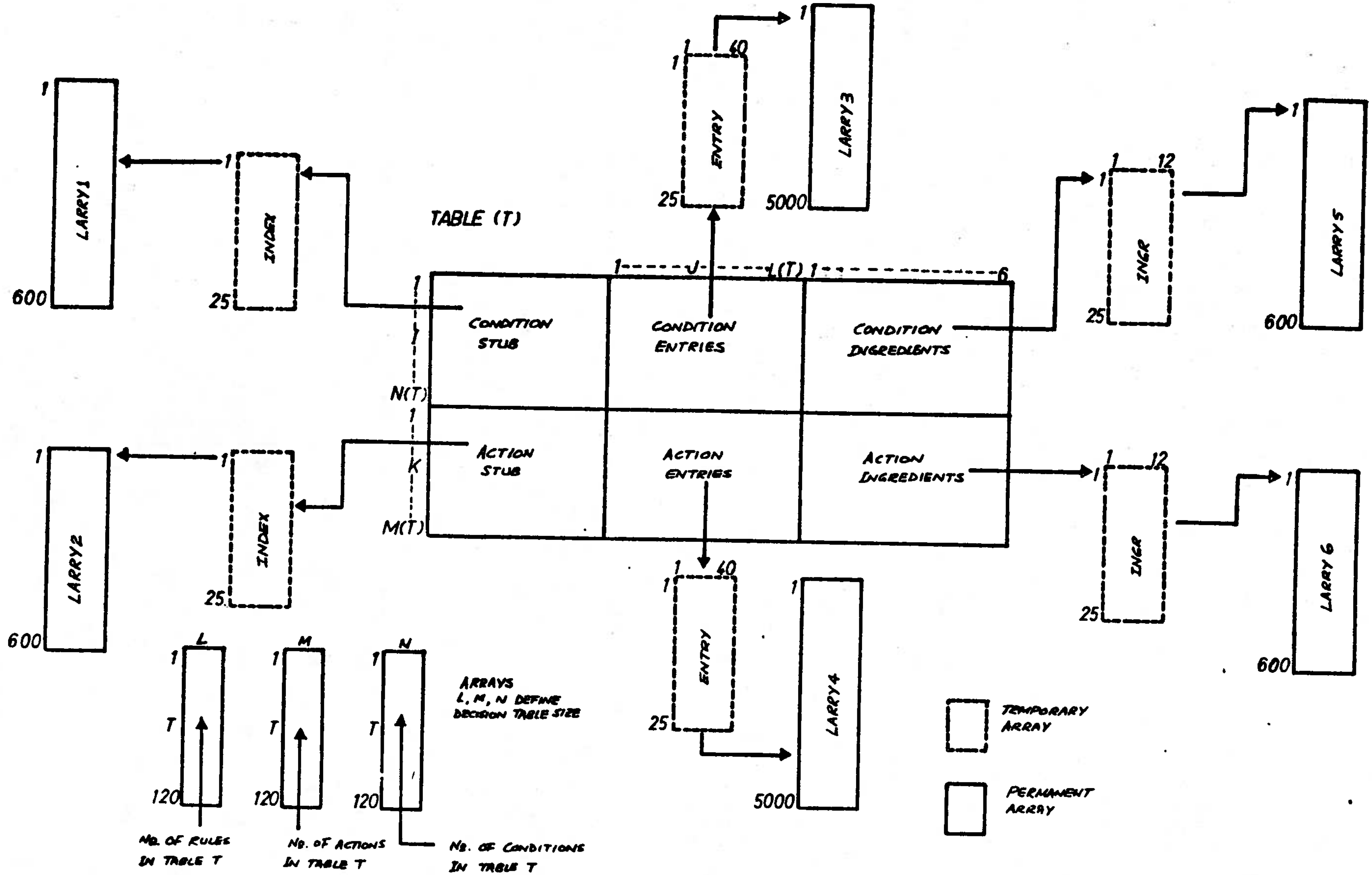


FIG. 5.1 PERMANENT AND TEMPORARY DECISION TABLE STORAGE ARRAYS

CONTENT	TEMPORARY ARRAY	PERMANENT ARRAY	POINTER ARRAY
Condition Stubs	INDEX	LARRY1	IBASE(T,1)
Action Stubs	INDEX	LARRY2	IBASE(T,2)
Condition Entries	ENTRY	LARRY3	IBASE(T,3)
Action Entries	ENTRY	LARRY4	IBASE(T,4)
Condition Ingredients	INGR	LARRY5	IPNTRC
Action Ingredients	INGR	LARRY6	IPNTRA
Mutually Exclusive Sets	TMPSET	MEXSET	MARCA
Dependents	IDEPND	ICLEAR	IARROW

FIG. 5.2 PERMANENT, TEMPORARY ARRAYS AND THEIR POINTER ARRAYS

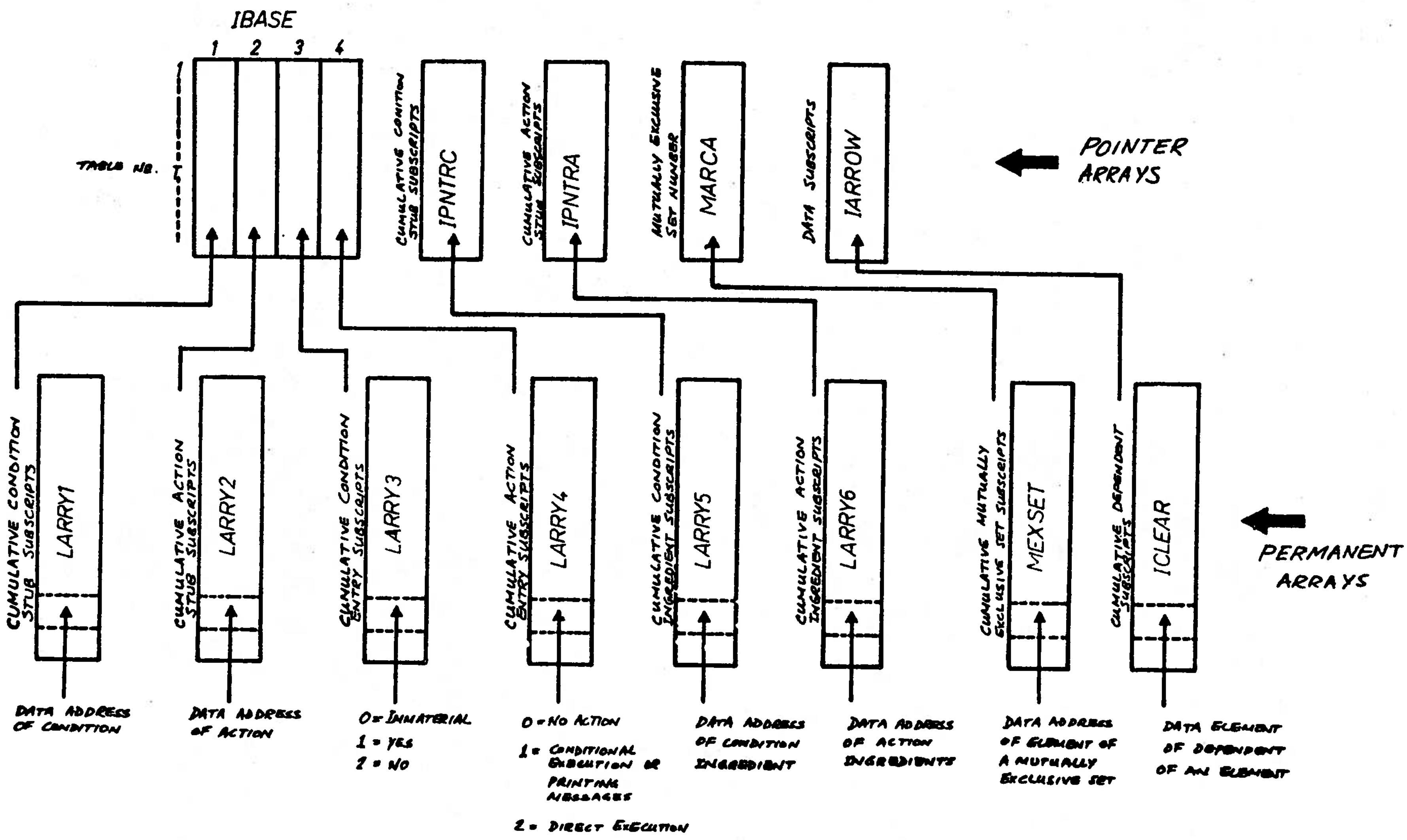


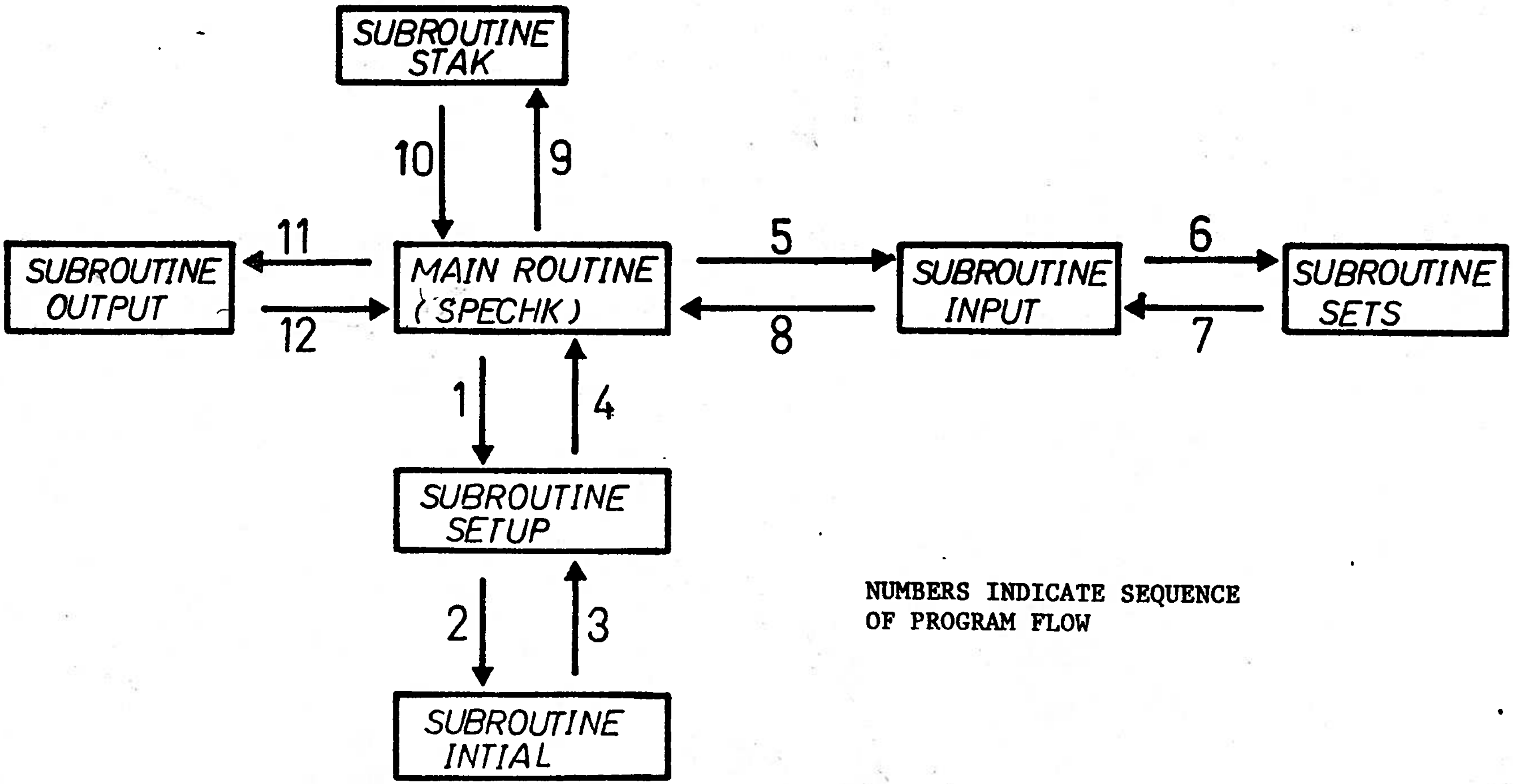
FIG. 5.3 OVERALL ARRAY STRUCTURE AND INTERRELATIONSHIP

```
     44         10          4           10
110    1221222222
111    2122122222
112    2212212222
113    2222221111
 99    2222221222
100    2222222122
101    2222222212
102    1111112221
274    111222                                        63   64   76   75
272    222111                                        63   64   75
134        111111                                   197   80   75   76
134    111                                          197   80   75   76   47   48C
                                                     63   64
                 1
151    111111111                                    132  134
```

FIG. 5.4 CODED INPUT FORM OF DECISION TABLE 13.5.A.1 (44)

```
      SUBROUTINE CC44(I)
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      COMMON/MICA/DATA(700),PRD(700)
      EQUIVALENCE (DATA(63),SH),(DATA(64),SW),(DATA(76),FY),
     *(DATA(272),$HW13),(DATA(273),$HW14),(DATA(75),FCR),
     *(DATA(274),$HW15)
      GO TO (9999,9999,9999,9999,9999,9999,9999,9999,90,100),I
   90 IF(SH/SW.GT.12000.0/FY) GO TO 102
      IF(SH/SW.LE.690.0/SQRT(FCR)) GO TO 102
      $HW15=1.0
      GO TO 104
  102 $HW15=0.0
  104 PRD(274)=.TRUE.
      RETURN
  100 IF(SH/SW.LE.690.0/SQRT(FCR)) GO TO 106
      $HW13=0.0
      GO TO 108
  106 $HW13=1.0
  108 PRD(272)=.TRUE.
      RETURN
 9999 WRITE(6,110)
  110 FORMAT(1X,'NO SUBROUTINE NECESSARY FOR THIS CONDITION')
      RETURN
      END
      SUBROUTINE AA44(K)
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      COMMON/MICA/DATA(700),PRD(700)
      COMMON/MINCE/ICYCLE
      EQUIVALENCE (DATA(134),FMRX),(DATA(197),FHAI),(DATA(80),S),
     *(DATA(75),FCR),(DATA(47),AW),(DATA(48),AF),(DATA(63),SH),
     *(DATA(64),SW),(DATA(151),R2),(DATA(132),FMFX),(DATA(76),FY)
      GO TO (10,20,30,40),K
   10 FMRX=AMIN1(FHAI*S*FCR,FHAI*S*FY)
      PRD(134)=.TRUE.
      RETURN
   20 FMRX=AMIN1(FHAI*S*FY,FHAI*S*FCR)*(1.0-0.0005*AW/AF*(SH/SW-690.0/
     *SQRT(FCR)))
      PRD(134)=.TRUE.
      RETURN
   30 WRITE(6,100)
      WRITE(7,100)
  100 FORMAT(1X,'***** FMRX=DATA(134), TO BE DETERMINED BY CLAUSE 12,',
     *' CSA S136. *****')
      WRITE(6,102)
      WRITE(7,102)
  102 FORMAT(1H1,2X,'******CALL OUTPUT,THEN TERMINATE PROGRAM',
     *' BECAUSE OF THE ABOVE MESSAGE FROM AA44******')
      CALL OUTPUT(ICYCLE)
      STOP
   40 R2=FMFX/FMRX
      PRD(151)=.TRUE.
      RETURN
      END
      SUBROUTINE CC45(I)
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      COMMON/MICA/DATA(700),PRD(700)
      EQUIVALENCE (DATA(59),SB),(DATA(58),ST),(DATA(191),SKB),
     *(DATA(76),FY),(DATA(258),$BT18)
      GO TO (10),I
   10 IF (SB/ST.LE.201.0*SQRT(SKB/FY)) GO TO 100
      $BT18=0.0
      GO TO 102
  100 $BT18=1.0
  102 PRD(258)=.TRUE.
      RETURN
      END
```

FIG. 5.5 CONDITION AND ACTION SUBROUTINES FOR DECISION TABLE 13.5.A.1 (44)



NUMBERS INDICATE SEQUENCE OF PROGRAM FLOW

FIG. 7.1 SUBROUTINE STRUCTURE FOR BATCH MODE PROCESSING

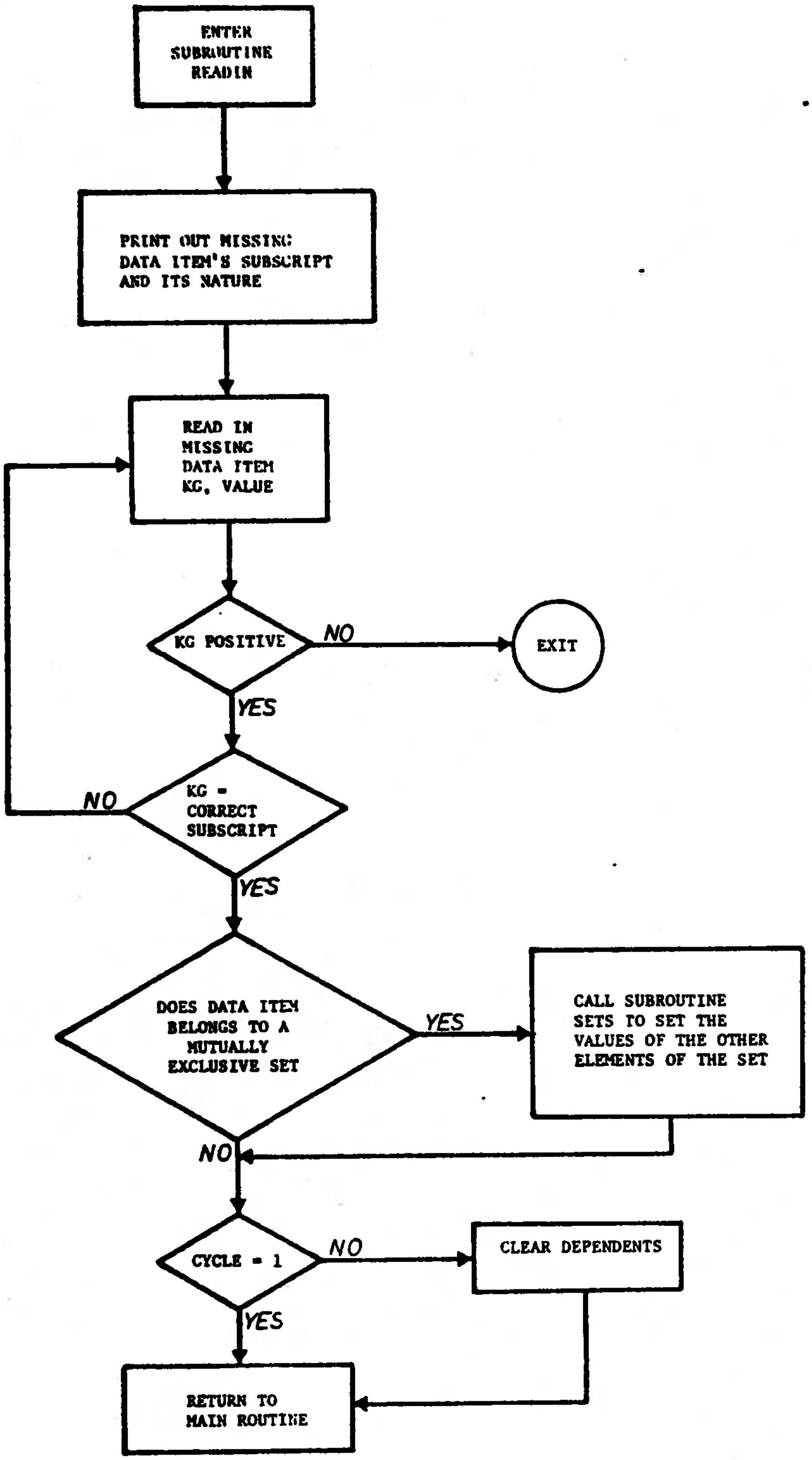


FIG. 7.2 MISSING DATA INPUT PROCEDURE IN INTERACTIVE MODE

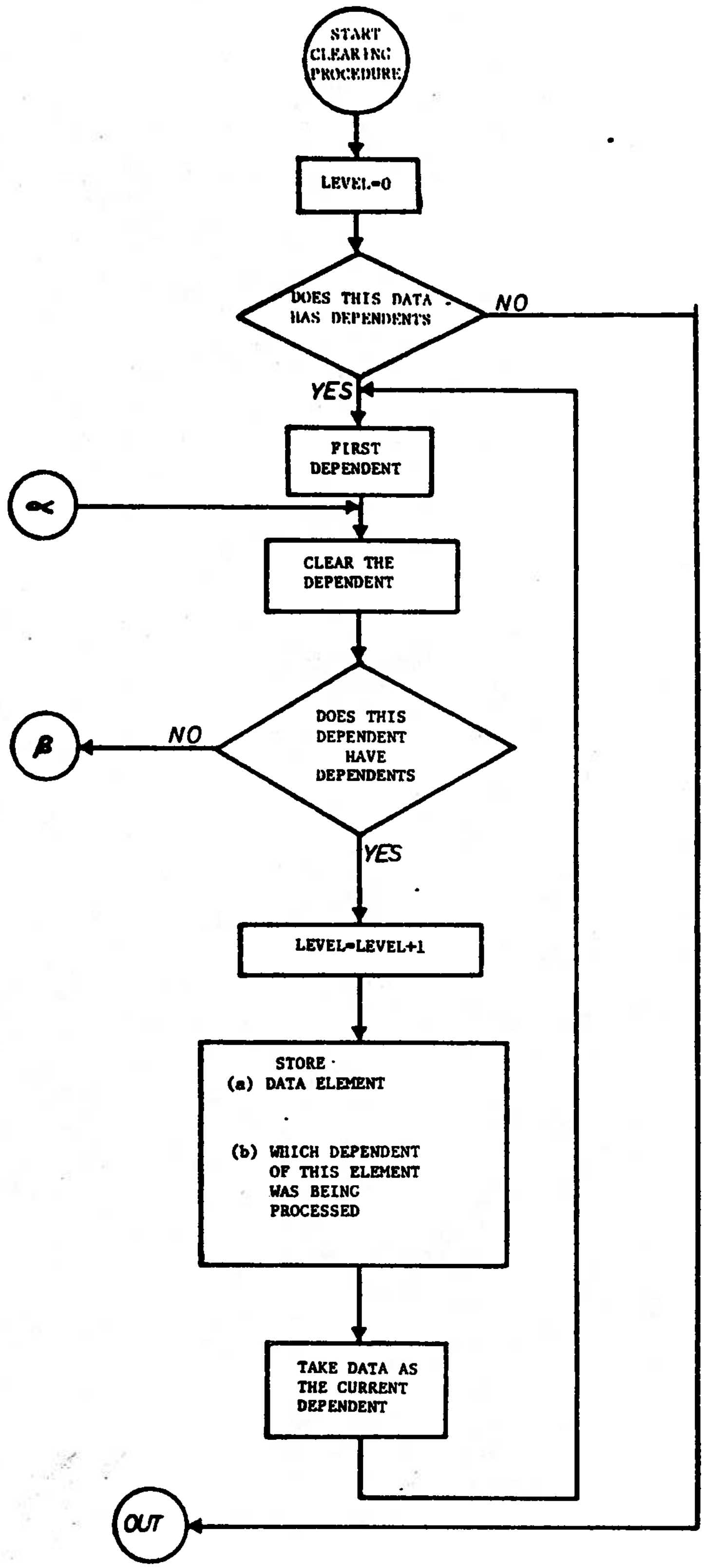


FIGURE 7.3 PROCEDURE FOR CLEARING DEPENDENT DATA ELEMENTS

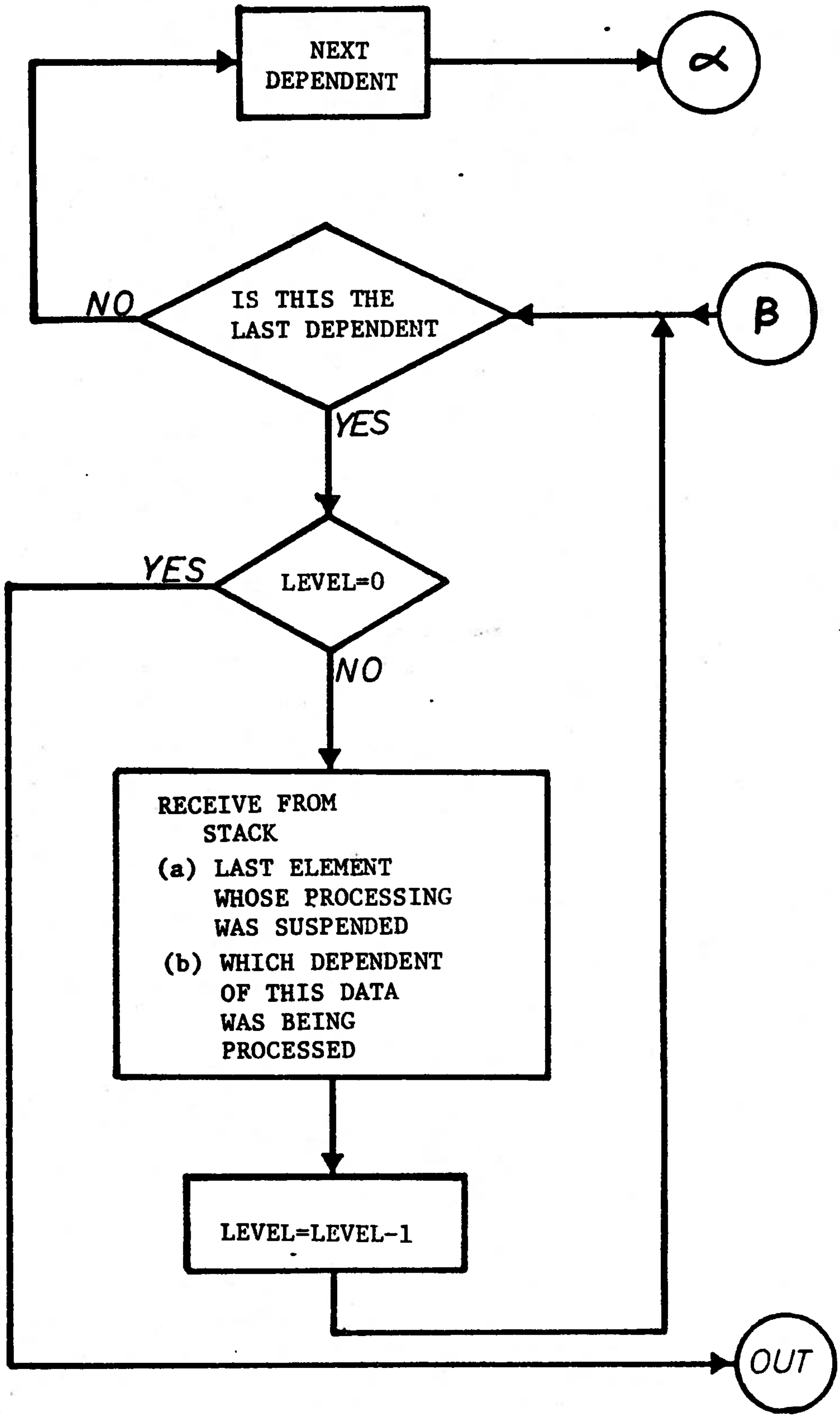


FIG. 7.4 PROCEDURE FOR CLEARING DEPENDENT DATA ELEMENTS

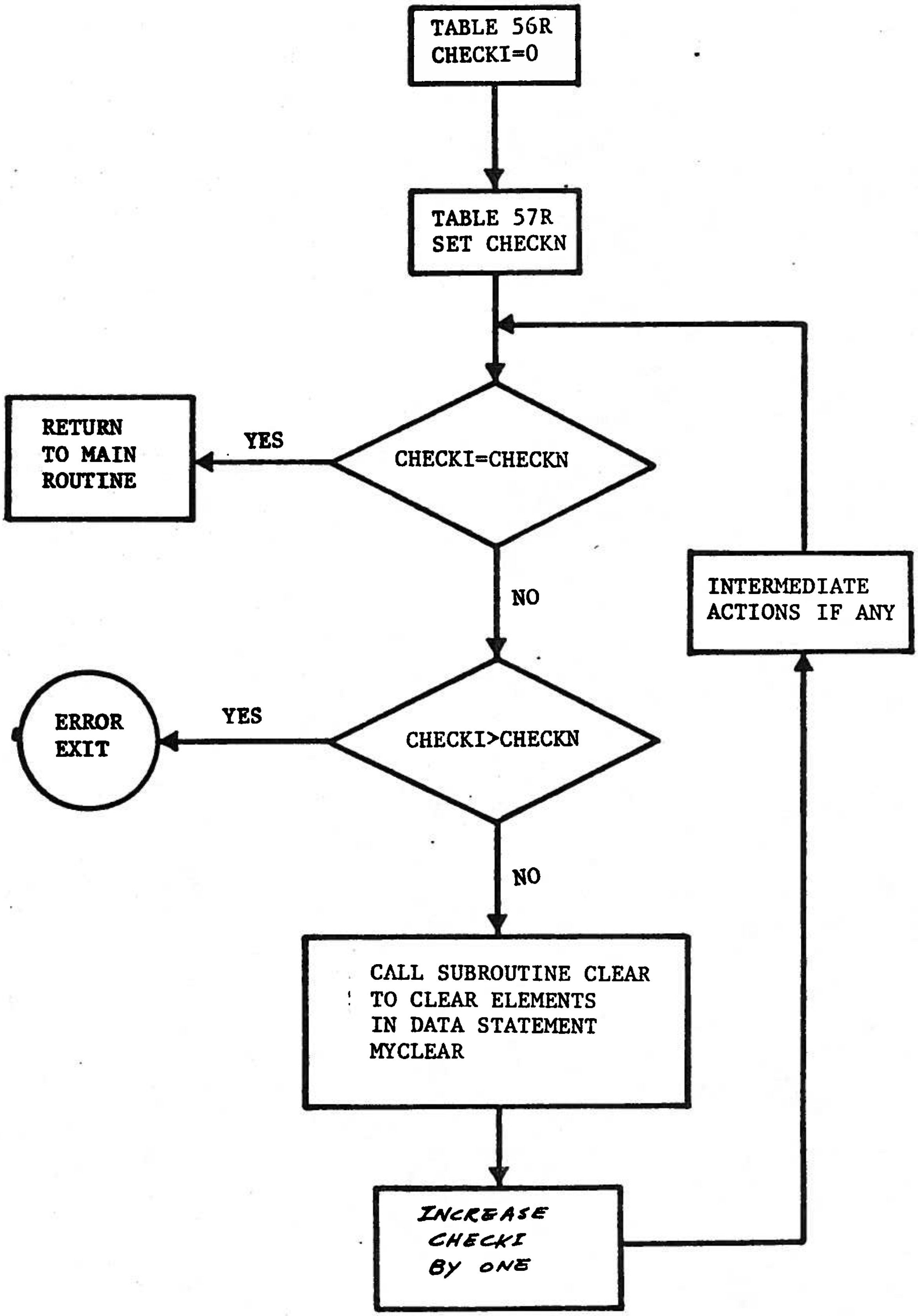


FIG. 7.5 RECURSIVE OPERATION OF A DECISION TABLE

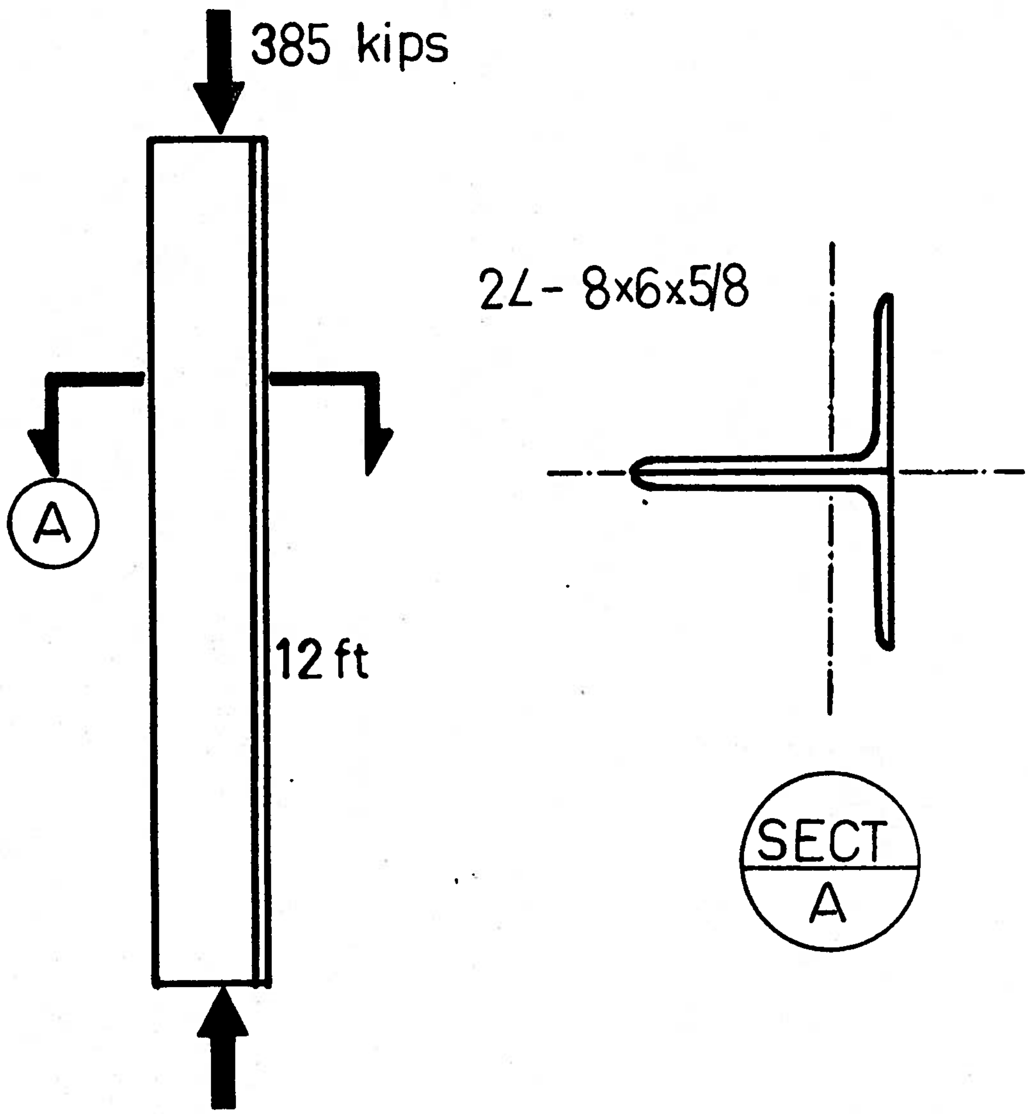


FIG. 8.1 LOADING CONDITION FOR EXAMPLE 1

```
$run obcombine 5=axcom 9=decidatal 8=csasl6 2=mapnsave 6=savel 4=*source* 7=*sink*
#20:18.42

          CONDITION NUMBER  5OF TABLE   4IS NOT AVAILABLE. THIS CORRESPONDS TO DATA NUMBER 222
          SUBROUTINE READIN IS CALLED TO INPUT THIS DATA ITEM
*****AWAITING INPUT FOR DATA ITEM WITH SUBSCRIPT= 222
THIS DATA ITEM IS A CONDITION.
222,0.0,

          CONDITION NUMBER  6OF TABLE   4IS NOT AVAILABLE. THIS CORRESPONDS TO DATA NUMBER 223
          SUBROUTINE READIN IS CALLED TO INPUT THIS DATA ITEM
*****AWAITING INPUT FOR DATA ITEM WITH SUBSCRIPT= 223
THIS DATA ITEM IS A CONDITION.
223,0.0,



 STRENGTH CRITERION SATISFIED



          PLEASE INPUT A VALUE OF 1 OR 2 FOR INDIC
         1 INDICATES THERE ARE FURTHER CYCLES
         2 INDICATES NO FURTHER CYCLES
2,
EXECUTION OF PROGRAM IS COMPLETED.
COLLECT YOUR OUTPUT FROM THE COMPUTING CENTER.COME BACK SOON
#20:20.23 4.156 RC=0
#
```

FIG. 8.2 TERMINAL OUTPUT FOR EXAMPLE 1

THE FOLLOWING NUMERICAL DATA HAS BEEN SUPPLIED FOR CYCLE NUMBER 1

KGLOB	DATAK
1	1.0000
10	1.0000
23	1.0000
25	0.0
46	16.7000
58	0.6300
59	8.0000
76	44.0000
78	29000.0000
119	1.0000
121	0.0
122	1.0000
123	1.0000
124	1.0000
125	1.0000
166	1.0000
179	1.0000
186	144.0000
188	2.4200
197	0.9000
208	2.5400
216	144.0000
400	0.9000
401	1.2500
402	1.5000
403	1.5000
404	1.2500
405	1.0000
406	300.0000
407	85.0000
408	0.0
409	0.0
220	1.0000
221	1.0000

FIG. 8.3 COMPUTER OUTPUT FOR EXAMPLE 1

DATA PRINTED AGAIN FOR CHECKING. ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
23	1.0000	T
24	0.0	T
25	0.0	T
46	16.7000	T
58	0.6300	T
59	8.0000	T
76	44.0000	T
78	29000.0000	T
119	1.0000	T
121	0.0	T
122	1.0000	T
123	1.0000	T
124	1.0000	T
125	1.0000	T
126	0.0	T
165	0.0	T
166	1.0000	T
167	0.0	T
168	0.0	T
169	0.0	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
179	1.0000	T
186	144.0000	T
188	2.4200	T
197	0.9000	T
208	2.5400	T
216	144.0000	T
220	1.0000	T
221	1.0000	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
406	300.0000	T
407	85.0000	T
408	0.0	T
409	0.0	T

FIG. 8.4 COMPUTER OUTPUT FOR EXAMPLE 1

```
CYCLE NUMBER 1     ***    START EXECUTION WITH TABLE   1   ***

     SCANNING OF TABLE   1 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   1 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   2 FOR DIRECT EXECUTION

     SCANNING OF TABLE   2 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   2 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   3 FOR DIRECT EXECUTION

     SCANNING OF TABLE   3 IS COMPLETE. RULE NO.  7 APPLIES

SUSPENDED EXECUTION OF TABLE   3 AT ACTION  5 OF RULE  7
 STARTED EXECUTION OF TABLE  27 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  27 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 150
 STARTED EXECUTION OF TABLE  28

     SCANNING OF TABLE  28 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  28 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  30 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  30 AT CONDITION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   5 TO OBTAIN VALUE OF DATA NUMBER  95

     SCANNING OF TABLE   5 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   5 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  21 FOR DIRECT EXECUTION

     SCANNING OF TABLE  21 IS COMPLETE. RULE NO. 11 APPLIES

     RESTART EXECUTION OF TABLE   5   AT ACTION   1   OF RULE  1

     RESTART EXECUTION OF TABLE  30   AT CONDITION  1   OF RULE  1

     SCANNING OF TABLE  30 IS COMPLETE. RULE NO.  3 APPLIES

SUSPENDED EXECUTION OF TABLE  30 AT ACTION  1 OF RULE  3
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 185
 STARTED EXECUTION OF TABLE   4

     SCANNING OF TABLE   4 IS COMPLETE. RULE NO.  4 APPLIES

     RESTART EXECUTION OF TABLE  30   AT ACTION   1   OF RULE  3
SUSPENDED EXECUTION OF TABLE  30 AT ACTION  2 OF RULE  3
 STARTED EXECUTION OF TABLE  31 FOR DIRECT EXECUTION

     SCANNING OF TABLE  31 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  31 AT ACTION  5 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 120
 STARTED EXECUTION OF TABLE  93

     SCANNING OF TABLE  93 IS COMPLETE. RULE NO.  7 APPLIES

     RESTART EXECUTION OF TABLE  31   AT ACTION   5   OF RULE  1

     RESTART EXECUTION OF TABLE  30   AT ACTION   2   OF RULE  3

     RESTART EXECUTION OF TABLE  28   AT ACTION   1   OF RULE  1

     RESTART EXECUTION OF TABLE  27   AT CONDITION  1   OF RULE  1

     SCANNING OF TABLE  27 IS COMPLETE. RULE NO.  1 APPLIES

STRENGTH CRITERION SATISFIED

              RESTART EXECUTION OF TABLE   3   AT ACTION  5   OF RULE  7

              RESTART EXECUTION OF TABLE   2   AT ACTION  1   OF RULE  1

              RESTART EXECUTION OF TABLE   1   AT ACTION  1   OF RULE  1
```

FIG. 8.5 COMPUTER OUTPUT FOR EXAMPLE 1

DATA VALUES AT THE END OF CYCLE NO. 1
ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
23	1.0000	T
24	0.0	T
25	0.0	T
46	16.7000	T
58	0.6300	T
59	8.0000	T
76	44.0000	T
78	29000.0000	T
95	0.0	T
96	0.0	T
97	1.0000	T
98	0.0	T
119	1.0000	T
120	452.2498	T
121	0.0	T
122	1.0000	T
123	1.0000	T
124	1.0000	T
125	1.0000	T
126	0.0	T
145	505.6328	T
150	0.8944	T
151	0.0	T
152	0.0	T
153	0.0	T
154	1.0000	T
165	0.0	T
166	1.0000	T
167	0.0	T
168	0.0	T
169	0.0	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
179	1.0000	T
185	1.0000	T
186	144.0000	T
188	2.4200	T
197	0.9000	T
208	2.5400	T
214	1.0000	T
215	59.5041	T
216	144.0000	T
217	59.5041	T
219	56.6929	T
220	1.0000	T
221	1.0000	T
222	0.0	T
223	0.0	T
230	0.7382	T
231	1.0000	T
232	0.0	T
233	0.0	T
234	0.0	T
254	0.0	T
255	0.0	T
256	1.0000	T
257	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
406	300.0000	T
407	85.0000	T
408	0.0	T
409	0.0	T

EXECUTION OF PROGRAM IS COMPLETED

FIG. 8.6 COMPUTER OUTPUT FOR EXAMPLE 1

```
$run obcombine 5=lub 9=decidata1 8=csas16 2=mapnsave 6=save2 4=*source* 7=*sink*
#20:22.14

          ERROR MESSAGE; DATA NUMBER    63
 IS NOT AVAILABLE.THIS IS AN INGREDIENT OF A CONDITION
 *****AWAITING INPUT FOR DATA ITEM WITH SUBSCRIPT=  63
 THIS DATA ITEM IS A MISSING INGREDIENT OF A CONDITION
 63,10.91,



  STRENGTH CRITERION SATISFIED
 *****SHEAR CRITERION SATISFIED*****



          PLEASE INPUT A VALUE OF 1 OR 2 FOR INDIC
         1 INDICATES THERE ARE FURTHER CYCLES
         2 INDICATES NO FURTHER CYCLES
 2,
 EXECUTION OF PROGRAM IS COMPLETED.
 COLLECT YOUR OUTPUT FROM THE COMPUTING CENTER.COME BACK SOON
#20:23.35 5.243 RC=0
#
```

FIG. 8.7 TERMINAL OUTPUT FOR EXAMPLE 2

THE FOLLOWING NUMERICAL DATA HAS BEEN SUPPLIED FOR CYCLE NUMBER 1

KGLOB	DATAK
1	1.0000
10	1.0000
22	1.0000
26	1.0000
29	1.0000
40	1.0000
47	4.0500
48	5.1800
58	0.6410
59	4.0400
64	0.3710
65	12.1900
76	44.0000
78	29000.0000
79	72.5000
80	64.8000
119	1.0000
121	1.0000
122	0.0
123	1.0000
124	0.0
127	0.0
128	1.0000
129	0.0
130	1.0000
138	3190.0000
169	1.0000
176	0.0
177	1.0000
180	0.0
187	288.0000
189	2.2000
197	0.9000
225	1.0000
320	1.0000
342	1.0000
334	0.0
400	0.9000
401	1.2500
402	1.5000
403	1.5000
404	1.2500
405	1.0000
410	1036.8001
411	518.3999
412	0.0
413	0.0
418	14.4000
419	7.2000
420	0.0
421	0.0
125	0.0

FIG. 8.8 COMPUTER OUTPUT FOR EXAMPLE 2

DATA PRINTED AGAIN FOR CHECKING. ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
21	0.0	T
22	1.0000	T
26	1.0000	T
27	0.0	T
28	0.0	T
29	1.0000	T
30	0.0	T
40	1.0000	T
41	0.0	T
47	4.0500	T
48	5.1800	T
58	0.6410	T
59	4.0400	T
64	0.3710	T
65	12.1900	T
76	44.0000	T
78	29000.0000	T
79	72.5000	T
80	64.8000	T
119	1.0000	T
121	1.0000	T
122	0.0	T
123	1.0000	T
124	0.0	T
125	0.0	T
126	0.0	T
127	0.0	T
128	1.0000	T
129	0.0	T
130	1.0000	T
138	3190.0000	T
165	0.0	T
166	0.0	T
167	0.0	T
168	0.0	T
169	1.0000	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
176	0.0	T
177	1.0000	T
180	0.0	T
187	288.0000	T
189	2.2000	T
197	0.9000	T
224	0.0	T
225	1.0000	T
226	0.0	T
320	1.0000	T
321	0.0	T
322	0.0	T
334	0.0	T
342	1.0000	T
343	0.0	T
344	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
410	1036.8001	T
411	518.3999	T
412	0.0	T
413	0.0	T
418	14.4000	T
419	7.2000	T
420	0.0	T
421	0.0	T

FIG. 8.9 COMPUTER OUTPUT FOR EXAMPLE 2

```
CYCLE NUMBER  1      ***   START EXECUTION WITH TABLE   1   ***

     SCANNING OF TABLE   1 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   1 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   2 FOR DIRECT EXECUTION

     SCANNING OF TABLE   2 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   2 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   3 FOR DIRECT EXECUTION

     SCANNING OF TABLE   3 IS COMPLETE. RULE NO.  5 APPLIES

SUSPENDED EXECUTION OF TABLE   3 AT ACTION  5 OF RULE  5
 STARTED EXECUTION OF TABLE  27 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  27 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 151
 STARTED EXECUTION OF TABLE  42

     SCANNING OF TABLE  42 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE  42 AT ACTION  3 OF RULE  2
 STARTED EXECUTION OF TABLE  46 FOR DIRECT EXECUTION

     SCANNING OF TABLE  46 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  46 AT ACTION  2 OF RULE  1
 STARTED EXECUTION OF TABLE  47 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  47 AT CONDITION  3 OF RULE  1
 STARTED EXECUTION OF TABLE   5 TO OBTAIN VALUE OF DATA NUMBER  95

SUSPENDED EXECUTION OF TABLE   5 AT CONDITION  7 OF RULE  7
 STARTED EXECUTION OF TABLE   6 TO OBTAIN VALUE OF DATA NUMBER  99

     SCANNING OF TABLE   6 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE   6 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  14 FOR DIRECT EXECUTION

     SCANNING OF TABLE  14 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE  14 AT ACTION  2 OF RULE  2
 STARTED EXECUTION OF TABLE  16 FOR DIRECT EXECUTION

     SCANNING OF TABLE  16 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  14    AT ACTION   2    OF RULE  2


     RESTART EXECUTION OF TABLE   6    AT ACTION   1    OF RULE  2

     RESTART EXECUTION OF TABLE   5    AT CONDITION   7    OF RULE  7

SUSPENDED EXECUTION OF TABLE   5 AT CONDITION 11 OF RULE  7
 STARTED EXECUTION OF TABLE  20 TO OBTAIN VALUE OF DATA NUMBER 110

     SCANNING OF TABLE  20 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE  20 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  21 FOR DIRECT EXECUTION

     SCANNING OF TABLE  21 IS COMPLETE. RULE NO.  5 APPLIES
```

FIG. 8.10 COMPUTER OUTPUT FOR EXAMPLE 2

```
      RESTART EXECUTION OF TABLE  20   AT ACTION   1   OF RULE  2

      RESTART EXECUTION OF TABLE   5   AT CONDITION 11   OF RULE  7

      SCANNING OF TABLE   5 IS COMPLETE. RULE NO.  7 APPLIES

 SUSPENDED EXECUTION OF TABLE   5 AT ACTION  7 OF RULE  7
  STARTED EXECUTION OF TABLE   7 FOR DIRECT EXECUTION

      SCANNING OF TABLE   7 IS COMPLETE. RULE NO.  4 APPLIES

      RESTART EXECUTION OF TABLE   5   AT ACTION   7   OF RULE  7

      RESTART EXECUTION OF TABLE  47   AT CONDITION  3   OF RULE  1

      SCANNING OF TABLE  47 IS COMPLETE. RULE NO.  1 APPLIES

 SUSPENDED EXECUTION OF TABLE  47 AT ACTION  1 OF RULE  1
  STARTED EXECUTION OF TABLE  48 FOR DIRECT EXECUTION

 SUSPENDED EXECUTION OF TABLE  48 AT CONDITION  1 OF RULE  1
  REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 137
  STARTED EXECUTION OF TABLE  51

      SCANNING OF TABLE  51 IS COMPLETE. RULE NO.  1 APPLIES

 SUSPENDED EXECUTION OF TABLE  51 AT ACTION  2 OF RULE  1
  REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 194
  STARTED EXECUTION OF TABLE  52

      SCANNING OF TABLE  52 IS COMPLETE. RULE NO.  1 APPLIES

 SUSPENDED EXECUTION OF TABLE  52 AT ACTION  1 OF RULE  1
  STARTED EXECUTION OF TABLE  53 FOR DIRECT EXECUTION

      SCANNING OF TABLE  53 IS COMPLETE. RULE NO.  2 APPLIES


      RESTART EXECUTION OF TABLE  52   AT ACTION   1   OF RULE  1

      RESTART EXECUTION OF TABLE  51   AT ACTION   2   OF RULE  1

 SUSPENDED EXECUTION OF TABLE  51 AT ACTION  2 OF RULE  1
  REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 195
  STARTED EXECUTION OF TABLE  55

      SCANNING OF TABLE  55 IS COMPLETE. RULE NO.  1 APPLIES

      RESTART EXECUTION OF TABLE  51   AT ACTION   2   OF RULE  1

      RESTART EXECUTION OF TABLE  48   AT CONDITION  1   OF RULE  1

      SCANNING OF TABLE  48 IS COMPLETE. RULE NO.  1 APPLIES

 SUSPENDED EXECUTION OF TABLE  48 AT ACTION  3 OF RULE  1
  REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 132
  STARTED EXECUTION OF TABLE  93

      SCANNING OF TABLE  93 IS COMPLETE. RULE NO.  5 APPLIES

      RESTART EXECUTION OF TABLE  48   AT ACTION   3   OF RULE  1

      RESTART EXECUTION OF TABLE  47   AT ACTION   1   OF RULE  1

      RESTART EXECUTION OF TABLE  46   AT ACTION   2   OF RULE  1

      RESTART EXECUTION OF TABLE  42   AT ACTION   3   OF RULE  2

      RESTART EXECUTION OF TABLE  27   AT CONDITION  1   OF RULE  1

      SCANNING OF TABLE  27 IS COMPLETE. RULE NO.  1 APPLIES
```

FIG. 8.11 COMPUTER OUTPUT FOR EXAMPLE 2

```
STRENGTH CRITERION SATISFIED

              RESTART EXECUTION OF TABLE    3   AT ACTION   5   OF RULE  5

          SUSPENDED EXECUTION OF TABLE   3 AT ACTION  7 OF RULE  5
           STARTED EXECUTION OF TABLE  74 FOR DIRECT EXECUTION

          SUSPENDED EXECUTION OF TABLE  74 AT CONDITION  1 OF RULE  1
           REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 326
           STARTED EXECUTION OF TABLE  75

              SCANNING OF TABLE  75 IS COMPLETE. RULE NO.  1 APPLIES

          SUSPENDED EXECUTION OF TABLE  75 AT ACTION  1 OF RULE  1
           REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 327
           STARTED EXECUTION OF TABLE  76

          SUSPENDED EXECUTION OF TABLE  76 AT CONDITION  1 OF RULE  1
           REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 328
           STARTED EXECUTION OF TABLE  77

              SCANNING OF TABLE  77 IS COMPLETE. RULE NO.  1 APPLIES

              RESTART EXECUTION OF TABLE  76   AT CONDITION  1   OF RULE  1

              SCANNING OF TABLE  76 IS COMPLETE. RULE NO.  1 APPLIES

              RESTART EXECUTION OF TABLE  75   AT ACTION  1   OF RULE  1

              RESTART EXECUTION OF TABLE  74   AT CONDITION  1   OF RULE  1

              SCANNING OF TABLE  74 IS COMPLETE. RULE NO.  1 APPLIES
*****SHEAR CRITERION SATISFIED*****

              RESTART EXECUTION OF TABLE   3   AT ACTION  7   OF RULE  5

              RESTART EXECUTION OF TABLE   2   AT ACTION  1   OF RULE  1

              RESTART EXECUTION OF TABLE   1   AT ACTION  1   OF RULE  1
```

FIG. 8.12 COMPUTER OUTPUT FOR EXAMPLE 2

DATA VALUES AT THE END OF CYCLE NO. 1
ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
21	0.0	T
22	1.0000	T
26	1.0000	T
27	0.0	T
28	0.0	T
29	1.0000	T
30	0.0	T
40	1.0000	T
41	0.0	T
47	4.0500	T
48	5.1800	T
58	0.6410	T
59	4.0400	T
63	10.9100	T
64	0.3710	T
65	12.1900	T
76	44.0000	T
78	29000.0000	T
79	72.5000	T
80	64.8000	T
95	1.0000	T
96	0.0	T
97	0.0	T
98	0.0	T
99	1.0000	T
100	0.0	T
101	0.0	T
102	0.0	T
110	1.0000	T
111	0.0	T
112	0.0	T
113	0.0	T
119	1.0000	T
121	1.0000	T
122	0.0	T
123	1.0000	T
124	0.0	T
125	0.0	T
126	0.0	T
127	0.0	T
128	1.0000	T
129	0.0	T
130	1.0000	T
132	1866.2398	T
134	1919.1516	T
137	2133.1218	T
138	3190.0000	T
139	1.0000	T
150	0.0	T
151	0.9724	T
152	0.0	T
154	1.0000	T
155	1.0000	T
165	0.0	T
166	0.0	T
167	0.0	T
168	0.0	T
169	1.0000	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
176	0.0	T
177	1.0000	T
180	0.0	T
187	288.0000	T
189	2.2000	T
194	1.0000	T
195	29.5096	T
196	14.5881	T
197	0.9000	T
224	0.0	T
225	1.0000	T
226	0.0	T
254	1.0000	T
255	0.0	T
256	0.0	T
257	0.0	T
262	1.0000	T
263	0.0	T
264	0.0	T
265	0.0	T
275	0.0	T
284	1.0000	T
285	0.0	T
286	0.0	T
287	0.0	T
320	1.0000	T
321	0.0	T
322	0.0	T
325	25.9200	T
326	105.8508	T
327	29.0400	T
328	5.3400	T
329	1.0000	T
330	0.0	T
334	0.0	T
342	1.0000	T
343	0.0	T
344	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
410	1036.8001	T
411	518.3999	T
412	0.0	T
413	0.0	T
418	14.4000	T
419	7.2000	T
420	0.0	T
421	0.0	T

EXECUTION OF PROGRAM IS COMPLETED

FIG. 8.13 COMPUTER OUTPUT FOR EXAMPLE 2

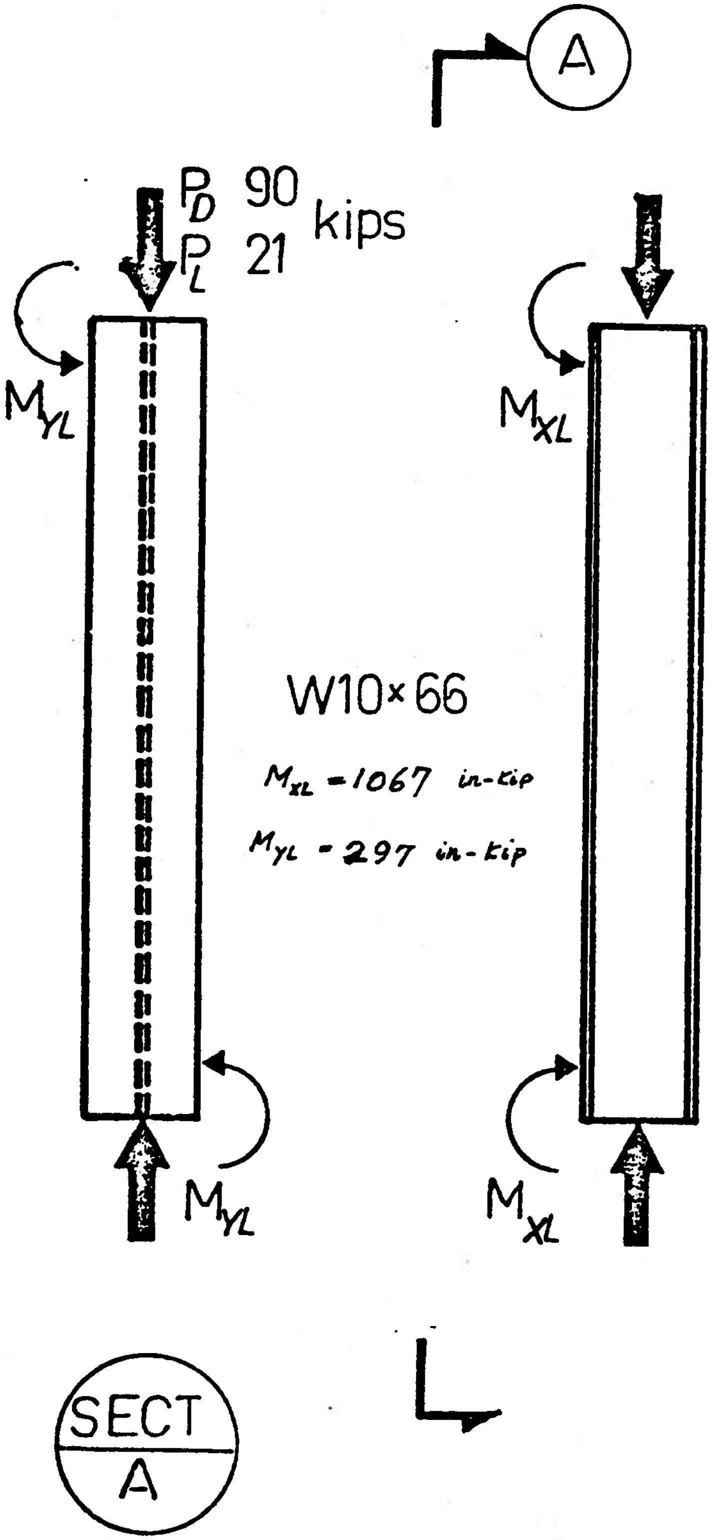


FIG. 8.14 LOADING CONDITION FOR EXAMPLE 3

```
$run obcombine 5=comben 9=decidatal 8=csas16 2=mapnsave 6=save 4=*source* 7=*sink*
#21:49:35


 ***** BOTH STRENGTH AND STABILITY CRITERIA NOT SATISFIED *****


          PLEASE INPUT A VALUE OF 1 OR 2 FOR INDIC
         1 INDICATES THERE ARE FURTHER CYCLES
         2 INDICATES NO FURTHER CYCLES
1,
         AWAITING NEXT DATA ITEM
65,10.38,
         AWAITING NEXT DATA ITEM
59,5.06,
         AWAITING NEXT DATA ITEM
58,0.748,
         AWAITING NEXT DATA ITEM
64,0.457,
         AWAITING NEXT DATA ITEM
63,8.88,
         AWAITING NEXT DATA ITEM
79,82.8,
         AWAITING NEXT DATA ITEM
80,73.6,
         AWAITING NEXT DATA ITEM
46,19.4,
         AWAITING NEXT DATA ITEM
48,7.57,
         AWAITING NEXT DATA ITEM
188,2.58,
         AWAITING NEXT DATA ITEM
208,4.44,
         AWAITING NEXT DATA ITEM
189,2.8,
         AWAITING NEXT DATA ITEM
138,3646.27,
         AWAITING NEXT DATA ITEM
190,853.5,
         AWAITING NEXT DATA ITEM
440,25.5,
         AWAITING NEXT DATA ITEM
441,38.8,
         AWAITING NEXT DATA ITEM
0,


 ***** STRENGTH AND STABILITY CRITERIA SATISFIED.*****


          PLEASE INPUT A VALUE OF 1 OR 2 FOR INDIC
         1 INDICATES THERE ARE FURTHER CYCLES
         2 INDICATES NO FURTHER CYCLES
2,
EXECUTION OF PROGRAM IS COMPLETED.
COLLECT YOUR OUTPUT FROM THE COMPUTING CENTER.COME BACK SOON
#21:54:04 1.048 RC=0
#
```

FIG. 8.15   TERMINAL OUTPUT FOR EXAMPLE 3

THE FOLLOWING NUMERICAL DATA HAS BEEN SUPPLIED FOR CYCLE NUMBER 1

KGLOB	DATAK		
1	1.0000	404	1.2500
10	1.0000	405	1.0000
23	1.0000	406	90.0000
25	0.0	407	21.0000
26	1.0000	408	0.0
28	1.0000	409	0.0
40	1.0000	410	0.0
46	14.4000	411	1067.0000
48	5.5800	412	0.0
58	0.5580	413	0.0
59	5.0000	414	0.0
63	8.8800	415	297.0000
64	0.3400	416	0.0
76	44.0000	417	0.0
78	29000.0000	186	114.0000
79	60.3000	216	114.0000
80	54.6000	180	0.0
119	1.0000	130	1.0000
121	0.0	178	1.0000
122	0.0	222	0.0
123	0.0	223	0.0
124	1.0000	176	0.0
125	1.0000	65	10.0000
127	1.0000	323	0.0
128	1.0000	324	1.0000
129	1.0000	440	18.6000
138	2490.3899	441	28.2000
169	1.0000		
177	1.0000		
187	114.0000		
188	2.5400		
189	2.7700		
190	633.6001		
197	0.9000		
208	4.3500		
220	1.0000		
221	1.0000		
226	1.0000		
290	1.0000		
320	1.0000		
342	1.0000		
400	0.9000		
401	1.2500		
402	1.5000		
403	1.5000		

FIG. 8.16 COMPUTER OUTPUT FOR EXAMPLE 3

DATA PRINTED AGAIN FOR CHECKING. ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
23	1.0000	T
24	0.0	T
25	0.0	T
26	1.0000	T
27	0.0	T
28	1.0000	T
29	0.0	T
30	0.0	T
40	1.0000	T
41	0.0	T
46	14.4000	T
48	5.5800	T
58	0.5580	T
59	5.0000	T
63	8.8800	T
64	0.3400	T
65	10.0000	T
76	44.0000	T
78	29000.0000	T
79	60.3000	T
80	54.6000	T
119	1.0000	T
121	0.0	T
122	0.0	T
123	0.0	T
124	1.0000	T
125	1.0000	T
126	0.0	T
127	1.0000	T
128	1.0000	T
129	1.0000	T
130	1.0000	T
138	2490.3899	T
165	0.0	T
166	0.0	T
167	0.0	T
168	0.0	T
169	1.0000	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
176	0.0	T
177	1.0000	T
178	1.0000	T
180	0.0	T
186	114.0000	T
187	114.0000	T
188	2.5400	T
189	2.7700	T
190	633.6001	T
197	0.9000	T
208	4.3500	T
216	114.0000	T
220	1.0000	T
221	1.0000	T
222	0.0	T
223	0.0	T
224	0.0	T
225	0.0	T
226	1.0000	T
290	1.0000	T
291	0.0	T
320	1.0000	T
321	0.0	T
322	0.0	T
323	0.0	T
324	1.0000	T
342	1.0000	T
343	0.0	T
344	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
406	90.0000	T
407	21.0000	T
408	0.0	T
409	0.0	T
410	0.0	T
411	1067.0000	T
412	0.0	T
413	0.0	T
414	0.0	T
415	297.0000	T
416	0.0	T
417	0.0	T
440	18.6000	T
441	28.2000	T

FIG. 8.17 COMPUTER OUTPUT FOR EXAMPLE 3

```
CYCLE NUMBER  1     ***    START EXECUTION WITH TABLE   1   ***

     SCANNING OF TABLE   1 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   1 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   2 FOR DIRECT EXECUTION

     SCANNING OF TABLE   2 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   2 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   3 FOR DIRECT EXECUTION

     SCANNING OF TABLE   3 IS COMPLETE. RULE NO. 13 APPLIES

SUSPENDED EXECUTION OF TABLE   3 AT ACTION  6 OF RULE 13
 STARTED EXECUTION OF TABLE  56 FOR DIRECT EXECUTION

     SCANNING OF TABLE  56 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  56 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  57 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  57 AT CONDITION  3 OF RULE  1
 STARTED EXECUTION OF TABLE   5 TO OBTAIN VALUE OF DATA NUMBER  95

SUSPENDED EXECUTION OF TABLE   5 AT CONDITION  7 OF RULE  7
 STARTED EXECUTION OF TABLE   6 TO OBTAIN VALUE OF DATA NUMBER  99

     SCANNING OF TABLE   6 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE   6 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  14 FOR DIRECT EXECUTION

     SCANNING OF TABLE  14 IS COMPLETE. RULE NO.  3 APPLIES

SUSPENDED EXECUTION OF TABLE  14 AT ACTION  3 OF RULE  3
 STARTED EXECUTION OF TABLE  17 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  17 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 120
 STARTED EXECUTION OF TABLE  93

     SCANNING OF TABLE  93 IS COMPLETE. RULE NO. 13 APPLIES

     RESTART EXECUTION OF TABLE  17   AT CONDITION  1   OF RULE  1

     SCANNING OF TABLE  17 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  17 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  18 FOR DIRECT EXECUTION

     SCANNING OF TABLE  18 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  17   AT ACTION  1   OF RULE  1

     RESTART EXECUTION OF TABLE  14   AT ACTION  3   OF RULE  3

     RESTART EXECUTION OF TABLE   6   AT ACTION  1   OF RULE  2

     RESTART EXECUTION OF TABLE   5   AT CONDITION  7   OF RULE  7

SUSPENDED EXECUTION OF TABLE   5 AT CONDITION 11 OF RULE  7
 STARTED EXECUTION OF TABLE  20 TO OBTAIN VALUE OF DATA NUMBER 110

     SCANNING OF TABLE  20 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE  20 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  21 FOR DIRECT EXECUTION

     SCANNING OF TABLE  21 IS COMPLETE. RULE NO.  6 APPLIES

     RESTART EXECUTION OF TABLE  20   AT ACTION  1   OF RULE  2

     RESTART EXECUTION OF TABLE   5   AT CONDITION 11   OF RULE  7

     SCANNING OF TABLE   5 IS COMPLETE. RULE NO.  8 APPLIES
```

FIG. 8.18 COMPUTER OUTPUT FOR EXAMPLE 3

```
SUSPENDED EXECUTION OF TABLE   5 AT ACTION  8 OF RULE  8
 STARTED EXECUTION OF TABLE   8 FOR DIRECT EXECUTION

     SCANNING OF TABLE   8 IS COMPLETE. RULE NO.  5 APPLIES

     RESTART EXECUTION OF TABLE   5    AT ACTION  8    OF RULE  8

     RESTART EXECUTION OF TABLE  57    AT CONDITION  3    OF RULE  1

     SCANNING OF TABLE  57 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE  57 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  58 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 295
 STARTED EXECUTION OF TABLE  60

     SCANNING OF TABLE  60 IS COMPLETE. RULE NO.  2 APPLIES

     RESTART EXECUTION OF TABLE  58    AT CONDITION  1    OF RULE  1

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  2 OF RULE  4
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 298

 STARTED EXECUTION OF TABLE  67

     SCANNING OF TABLE  67 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  58    AT CONDITION  2    OF RULE  4

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  5
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 297
 STARTED EXECUTION OF TABLE  65

SUSPENDED EXECUTION OF TABLE  65 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 313
 STARTED EXECUTION OF TABLE  66

     SCANNING OF TABLE  66 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  66 AT ACTION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 215
 STARTED EXECUTION OF TABLE   4

     SCANNING OF TABLE   4 IS COMPLETE. RULE NO.  4 APPLIES

     RESTART EXECUTION OF TABLE  66    AT ACTION  1    OF RULE  1

     RESTART EXECUTION OF TABLE  65    AT CONDITION  1    OF RULE  1

     SCANNING OF TABLE  65 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  58    AT CONDITION  3    OF RULE  5

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  5
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 194
 STARTED EXECUTION OF TABLE  52

     SCANNING OF TABLE  52 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  52 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  53 FOR DIRECT EXECUTION

     SCANNING OF TABLE  53 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  53 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  54 FOR DIRECT EXECUTION

     SCANNING OF TABLE  54 IS COMPLETE. RULE NO.  3 APPLIES

     RESTART EXECUTION OF TABLE  53    AT ACTION  1    OF RULE  1

     RESTART EXECUTION OF TABLE  52    AT ACTION  1    OF RULE  1
```

FIG. 8.19 COMPUTER OUTPUT FOR EXAMPLE 3

```
        RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  5

    SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  5
     REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 299
     STARTED EXECUTION OF TABLE  62

        SCANNING OF TABLE  62 IS COMPLETE. RULE NO.  2 APPLIES

    SUSPENDED EXECUTION OF TABLE  62 AT ACTION  1 OF RULE  2
     STARTED EXECUTION OF TABLE  63 FOR DIRECT EXECUTION

    SUSPENDED EXECUTION OF TABLE  63 AT CONDITION  1 OF RULE  1
     REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 137
     STARTED EXECUTION OF TABLE  51

        SCANNING OF TABLE  51 IS COMPLETE. RULE NO.  1 APPLIES

    SUSPENDED EXECUTION OF TABLE  51 AT ACTION  2 OF RULE  1
     REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 195
     STARTED EXECUTION OF TABLE  55

        SCANNING OF TABLE  55 IS COMPLETE. RULE NO.  1 APPLIES

        RESTART EXECUTION OF TABLE  51   AT ACTION  2   OF RULE  1

        RESTART EXECUTION OF TABLE  63   AT CONDITION  1   OF RULE  1

        SCANNING OF TABLE  63 IS COMPLETE. RULE NO.  1 APPLIES

        RESTART EXECUTION OF TABLE  62   AT ACTION  1   OF RULE  2

        RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  5

    SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  5
     REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 311
     STARTED EXECUTION OF TABLE  68

        SCANNING OF TABLE  68 IS COMPLETE. RULE NO.  1 APPLIES

        RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  5

    SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  5
     REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 310
     STARTED EXECUTION OF TABLE  69

        SCANNING OF TABLE  69 IS COMPLETE. RULE NO.  1 APPLIES

    SUSPENDED EXECUTION OF TABLE  69 AT ACTION  1 OF RULE  1
     STARTED EXECUTION OF TABLE  70 FOR DIRECT EXECUTION

        SCANNING OF TABLE  70 IS COMPLETE. RULE NO.  1 APPLIES

    SUSPENDED EXECUTION OF TABLE  70 AT ACTION  1 OF RULE  1
     STARTED EXECUTION OF TABLE  71 FOR DIRECT EXECUTION

        SCANNING OF TABLE  71 IS COMPLETE. RULE NO.  4 APPLIES

        RESTART EXECUTION OF TABLE  70   AT ACTION  1   OF RULE  1

        RESTART EXECUTION OF TABLE  69   AT ACTION  1   OF RULE  1

        RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  5

        SCANNING OF TABLE  58 IS COMPLETE. RULE NO.  6 APPLIES

***** BOTH STRENGTH AND STABILITY CRITERIA NOT SATISFIED *****

        RESTART EXECUTION OF TABLE  57   AT ACTION  1   OF RULE  2

        RESTART EXECUTION OF TABLE  56   AT ACTION  1   OF RULE  1

        RESTART EXECUTION OF TABLE   3   AT ACTION  6   OF RULE 13

        RESTART EXECUTION OF TABLE   2   AT ACTION  1   OF RULE  1

        RESTART EXECUTION OF TABLE   1   AT ACTION  1   OF RULE  1
```

FIG. 8.20 COMPUTER OUTPUT FOR EXAMPLE 3

DATA VALUES AT THE END OF CYCLE NO. 1
ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
23	1.0000	T
24	0.0	T
25	0.0	T
26	1.0000	T
27	0.0	T
28	1.0000	T
29	0.0	T
30	0.0	T
40	1.0000	T
41	0.0	T
46	14.4000	T
48	5.5800	T
58	0.5580	T
59	5.0000	T
63	8.8800	T
64	0.3400	T
65	10.0000	T
76	44.0000	T
78	29000.0000	T
79	60.3000	T
80	54.6000	T
95	0.0	T
96	1.0000	T
97	0.0	T
98	0.0	T
99	1.0000	T
100	0.0	T
101	0.0	T
102	0.0	T
110	0.0	T
111	1.0000	T
112	0.0	T
113	0.0	T
119	1.0000	T
120	129.6000	T
121	0.0	T
122	0.0	T
123	0.0	T
124	1.0000	T
125	1.0000	T
126	0.0	T
127	1.0000	T
128	1.0000	T
129	1.0000	T
130	1.0000	T
132	1440.4500	T
133	400.9500	T
136	1.0000	T
137	9670.4375	T
138	2490.3899	T
139	1.0000	T
153	0.0	T
165	0.0	T
166	0.0	T
167	0.0	T
168	0.0	T
169	1.0000	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
176	0.0	T
177	1.0000	T
178	1.0000	T
180	0.0	T
185	1.0000	T
186	114.0000	T
187	114.0000	T
188	2.5400	T
189	2.7700	T
190	633.6001	T
194	1.0000	T
195	97.8947	T
196	147.6012	T
197	0.9000	T
208	4.3500	T
214	1.0000	T
215	44.8819	T
216	114.0000	T
217	44.8819	T
219	26.2069	T
220	1.0000	T
221	1.0000	T
222	0.0	T
223	0.0	T
224	0.0	T
225	0.0	T
226	1.0000	T
254	0.0	T
255	1.0000	T
256	0.0	T
257	0.0	T
266	1.0000	T
267	0.0	T
268	0.0	T
275	0.0	T
290	1.0000	T
291	0.0	T
295	2241.3508	T
296	1116.7195	T
297	486.7878	T
298	570.2397	T
299	2241.3508	T
310	0.8500	T
311	5996.4922	T
312	2044.4961	T
313	0.5568	T
314	1.0000	T
315	0.0	T
316	0.0	T
317	0.0	T
320	1.0000	T
321	0.0	T
322	0.0	T
323	0.0	T
324	1.0000	T
335	0.0	T
336	1.0000	T
337	0.0	T
342	1.0000	T
343	0.0	T
344	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
406	90.0000	T
407	21.0000	T
408	0.0	T
409	0.0	T
410	0.0	T
411	1067.0000	T
412	0.0	T
413	0.0	T
414	0.0	T
415	297.0000	T
416	0.0	T
417	0.0	T
440	18.6000	T
441	28.2000	T

FIG. 8.21 COMPUTER OUTPUT FOR EXAMPLE 3

THE FOLLOWING NUMERICAL DATA HAS BEEN SUPPLIED FOR CYCLE NUMBER 2

KGLOB	DATAK
65	10.3800
59	5.0600
58	0.7480
64	0.4570
63	8.8800
79	82.8000
80	73.6000
46	19.4000
48	7.5700
188	2.5800
208	4.4400
189	2.8000
138	3646.2700
190	853.5000
440	25.5000
441	38.8000

DATA PRINTED AGAIN FOR CHECKING. ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
23	1.0000	T
24	0.0	T
25	0.0	T
26	1.0000	T
27	0.0	T
28	1.0000	T
29	0.0	T
30	0.0	T
40	1.0000	T
41	0.0	T
46	19.4000	T
48	7.5700	T
58	0.7480	T
59	5.0600	T
63	8.8800	T
64	0.4570	T
65	10.3800	T
76	44.0000	T
78	29000.0000	T
79	82.8000	T
80	73.6000	T
95	0.0	T
96	1.0000	T
101	0.0	T
119	1.0000	T
120	129.6000	T
121	0.0	T
122	0.0	T
123	0.0	T
124	1.0000	T
125	1.0000	T
126	0.0	T
127	1.0000	T
128	1.0000	T
129	1.0000	T

FIG. 8.22 COMPUTER OUTPUT FOR EXAMPLE 3

130	1.0000	T
132	1440.4500	T
133	400.9500	T
138	3646.2700	T
153	0.0	T
165	0.0	T
166	0.0	T
167	0.0	T
168	0.0	T
169	1.0000	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
176	0.0	T
177	1.0000	T
178	1.0000	T
180	0.0	T
185	1.0000	T
186	114.0000	T
187	114.0000	T
188	2.5800	T
189	2.8000	T
190	853.5000	T
194	1.0000	T
197	0.9000	T
208	4.4400	T
214	1.0000	T
216	114.0000	T
220	1.0000	T
221	1.0000	T
222	0.0	T
223	0.0	T
224	0.0	T
225	0.0	T
226	1.0000	T
290	1.0000	T
291	0.0	T
310	0.8500	T
320	1.0000	T
321	0.0	T
322	0.0	T
323	0.0	T
324	1.0000	T
342	1.0000	T
343	0.0	T
344	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
406	90.0000	T
407	21.0000	T
408	0.0	T
409	0.0	T
410	0.0	T
411	1067.0000	T
412	0.0	T
413	0.0	T
414	0.0	T
415	297.0000	T
416	0.0	T
417	0.0	T
440	25.5000	T
441	38.8000	T

FIG. 8.23 COMPUTER OUTPUT FOR EXAMPLE 3

```
CYCLE NUMBER  2     ***   START EXECUTION WITH TABLE   1   ***

     SCANNING OF TABLE   1 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   1 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   2 FOR DIRECT EXECUTION

     SCANNING OF TABLE   2 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE   2 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE   3 FOR DIRECT EXECUTION

     SCANNING OF TABLE   3 IS COMPLETE. RULE NO. 13 APPLIES

SUSPENDED EXECUTION OF TABLE   3 AT ACTION  6 OF RULE 13
 STARTED EXECUTION OF TABLE  56 FOR DIRECT EXECUTION

     SCANNING OF TABLE  56 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  56 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  57 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  57 AT CONDITION  5 OF RULE  2
 STARTED EXECUTION OF TABLE   5 TO OBTAIN VALUE OF DATA NUMBER  97

SUSPENDED EXECUTION OF TABLE   5 AT CONDITION  7 OF RULE  7
 STARTED EXECUTION OF TABLE   6 TO OBTAIN VALUE OF DATA NUMBER  99

     SCANNING OF TABLE   6 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE   6 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  14 FOR DIRECT EXECUTION

     SCANNING OF TABLE  14 IS COMPLETE. RULE NO.  3 APPLIES

SUSPENDED EXECUTION OF TABLE  14 AT ACTION  3 OF RULE  3
 STARTED EXECUTION OF TABLE  17 FOR DIRECT EXECUTION

     SCANNING OF TABLE  17 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  17 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  18 FOR DIRECT EXECUTION

     SCANNING OF TABLE  18 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  17   AT ACTION  1   OF RULE  1

     RESTART EXECUTION OF TABLE  14   AT ACTION  3   OF RULE  3

     RESTART EXECUTION OF TABLE   6   AT ACTION  1   OF RULE  2

     RESTART EXECUTION OF TABLE   5   AT CONDITION  7   OF RULE  7

SUSPENDED EXECUTION OF TABLE   5 AT CONDITION 11 OF RULE  7
 STARTED EXECUTION OF TABLE  20 TO OBTAIN VALUE OF DATA NUMBER 110

     SCANNING OF TABLE  20 IS COMPLETE. RULE NO.  2 APPLIES

SUSPENDED EXECUTION OF TABLE  20 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  21 FOR DIRECT EXECUTION

     SCANNING OF TABLE  21 IS COMPLETE. RULE NO.  5 APPLIES

     RESTART EXECUTION OF TABLE  20   AT ACTION  1   OF RULE  2

     RESTART EXECUTION OF TABLE   5   AT CONDITION 11   OF RULE  7

     SCANNING OF TABLE   5 IS COMPLETE. RULE NO.  7 APPLIES

SUSPENDED EXECUTION OF TABLE   5 AT ACTION  7 OF RULE  7
 STARTED EXECUTION OF TABLE   7 FOR DIRECT EXECUTION

     SCANNING OF TABLE   7 IS COMPLETE. RULE NO.  7 APPLIES

     RESTART EXECUTION OF TABLE   5   AT ACTION  7   OF RULE  7

     RESTART EXECUTION OF TABLE  57   AT CONDITION  5   OF RULE  2

     SCANNING OF TABLE  57 IS COMPLETE. RULE NO.  2 APPLIES
```

FIG. 8.24 COMPUTER OUTPUT FOR EXAMPLE 3

```
 STARTED EXECUTION OF TABLE  65

SUSPENDED EXECUTION OF TABLE  65 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 313
 STARTED EXECUTION OF TABLE  66

    SCANNING OF TABLE  66 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  66 AT ACTION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 215
 STARTED EXECUTION OF TABLE   4

    SCANNING OF TABLE   4 IS COMPLETE. RULE NO.  4 APPLIES

    RESTART EXECUTION OF TABLE  66   AT ACTION  1   OF RULE  1

    RESTART EXECUTION OF TABLE  65   AT CONDITION  1   OF RULE  1

    SCANNING OF TABLE  65 IS COMPLETE. RULE NO.  1 APPLIES

    RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  2

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  2
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 299
 STARTED EXECUTION OF TABLE  62

    SCANNING OF TABLE  62 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  62 AT ACTION  1 OF RULE  1
 STARTED EXECUTION OF TABLE  63 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  63 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 137
 STARTED EXECUTION OF TABLE  51

    SCANNING OF TABLE  51 IS COMPLETE. RULE NO.  1 APPLIES

SUSPENDED EXECUTION OF TABLE  51 AT ACTION  2 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 195
 STARTED EXECUTION OF TABLE  55

    SCANNING OF TABLE  55 IS COMPLETE. RULE NO.  1 APPLIES

    RESTART EXECUTION OF TABLE  51   AT ACTION  2   OF RULE  1

    RESTART EXECUTION OF TABLE  63   AT CONDITION  1   OF RULE  1

    SCANNING OF TABLE  63 IS COMPLETE. RULE NO.  1 APPLIES

    RESTART EXECUTION OF TABLE  62   AT ACTION  1   OF RULE  1
```

FIG. 8.25 COMPUTER OUTPUT FOR EXAMPLE 3

```
SUSPENDED EXECUTION OF TABLE  57 AT ACTION  1 OF RULE  2
 STARTED EXECUTION OF TABLE  58 FOR DIRECT EXECUTION

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  1 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 295
 STARTED EXECUTION OF TABLE  60

     SCANNING OF TABLE  60 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  58   AT CONDITION  1   OF RULE  1

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  2 OF RULE  1
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 298
 STARTED EXECUTION OF TABLE  67

     SCANNING OF TABLE  67 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  58   AT CONDITION  2   OF RULE  1

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  2
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 297




     RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  2

SUSPENDED EXECUTION OF TABLE  58 AT CONDITION  3 OF RULE  2
 REASON: MISSING INGREDIENT CORRESPONDING TO DATA NUMBER 311
 STARTED EXECUTION OF TABLE  68

     SCANNING OF TABLE  68 IS COMPLETE. RULE NO.  1 APPLIES

     RESTART EXECUTION OF TABLE  58   AT CONDITION  3   OF RULE  2

     SCANNING OF TABLE  58 IS COMPLETE. RULE NO.  3 APPLIES




***** STRENGTH AND STABILITY CRITERIA SATISFIED.*****

     RESTART EXECUTION OF TABLE  57   AT ACTION  1   OF RULE  2

     RESTART EXECUTION OF TABLE  56   AT ACTION  1   OF RULE  1

     RESTART EXECUTION OF TABLE   3   AT ACTION  6   OF RULE 13

     RESTART EXECUTION OF TABLE   2   AT ACTION  1   OF RULE  1

     RESTART EXECUTION OF TABLE   1   AT ACTION  1   OF RULE  1
```

FIG. 8.26 COMPUTER OUTPUT FOR EXAMPLE 3

DATA VALUES AT THE END OF CYCLE NO. 2
ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE

KGLOB	DATAK	PRD
1	1.0000	T
2	0.0	T
10	1.0000	T
11	0.0	T
12	0.0	T
13	0.0	T
14	0.0	T
15	0.0	T
16	0.0	T
17	0.0	T
23	1.0000	T
24	0.0	T
25	0.0	T
26	1.0000	T
27	0.0	T
28	1.0000	T
29	0.0	T
30	0.0	T
40	1.0000	T
41	0.0	T
46	19.4000	T
48	7.5700	T
58	0.7480	T
59	5.0600	T
63	8.8800	T
64	0.4570	T
65	10.3800	T
76	44.0000	T
78	29000.0000	T
79	82.8000	T
80	73.6000	T
95	1.0000	T
96	0.0	T
97	0.0	T
98	0.0	T
99	1.0000	T
100	0.0	T
101	0.0	T
102	0.0	T
110	1.0000	T
111	0.0	T
112	0.0	T
113	0.0	T
119	1.0000	T
120	129.6000	T
121	0.0	T
122	0.0	T
123	0.0	T
124	1.0000	T
125	1.0000	T
126	0.0	T
127	1.0000	T
128	1.0000	T
129	1.0000	T
130	1.0000	T
132	1440.4500	T
133	400.9500	T
136	1.0000	T
137	14556.3047	T
138	3646.2700	T
139	1.0000	T
153	0.0	T
165	0.0	T
166	0.0	T
167	0.0	T
168	0.0	T
169	1.0000	T
170	0.0	T
171	0.0	T
172	0.0	T
173	0.0	T
174	0.0	T
175	0.0	T
176	0.0	T
177	1.0000	T
178	1.0000	T
180	0.0	T
185	1.0000	T
186	114.0000	T
187	114.0000	T
188	2.5800	T
189	2.8000	T
190	853.5000	T
194	1.0000	T
195	127.9451	T
196	150.8156	T
197	0.9000	T
208	4.4400	T
214	1.0000	T
215	44.1860	T
216	114.0000	T
217	44.1860	T
219	25.6757	T
220	1.0000	T
221	1.0000	T
222	0.0	T
223	0.0	T
224	0.0	T
225	0.0	T
226	1.0000	T
254	1.0000	T
255	0.0	T
256	0.0	T
257	0.0	T
266	1.0000	T
267	0.0	T
268	0.0	T
275	0.0	T
290	1.0000	T
291	0.0	T
295	3281.6428	T
296	1536.4797	T
297	658.7854	T
298	768.2395	T
299	3281.6428	T
310	0.8500	T
311	8416.3516	T
312	2841.8264	T
313	0.5481	T
314	1.0000	T
315	0.0	T
316	0.0	T
317	0.0	T
320	1.0000	T
321	0.0	T
322	0.0	T
323	0.0	T
324	1.0000	T
335	1.0000	T
336	1.0000	T
337	1.0000	T
342	1.0000	T
343	0.0	T
344	0.0	T
400	0.9000	T
401	1.2500	T
402	1.5000	T
403	1.5000	T
404	1.2500	T
405	1.0000	T
406	90.0000	T
407	21.0000	T
408	0.0	T
409	0.0	T
410	0.0	T
411	1067.0000	T
412	0.0	T
413	0.0	T
414	0.0	T
415	297.0000	T
416	0.0	T
417	0.0	T
440	25.5000	T
441	38.8000	T

EXECUTION OF PROGRAM IS COMPLETED
$SIGNOFF

FIG. 8.27 COMPUTER OUTPUT FOR EXAMPLE 3

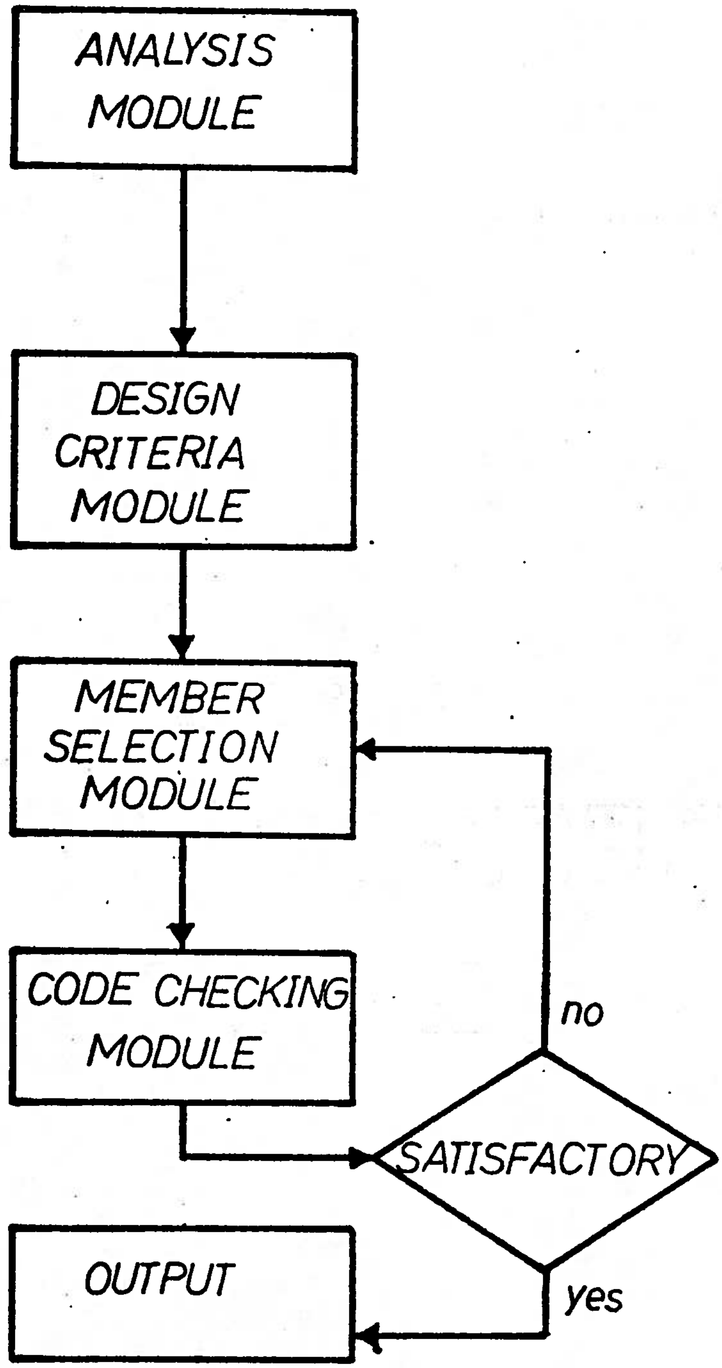


FIG. 9.1 THE COMPLETE DESIGN PROCESS

REFERENCES

1. ACI Committee 118. Decision Logic Tables For Building Code Requirements for Reinforced Concrete (ACI 318-71), ACI Journal, December 1973.

2. Allen, D.E., Limit States Design: A Probabilistic Study. Canadian Journal of Civil Engineering, Vol. 2, March, 1975, pp. 36-49.

3. Canadian Institute of Steel Construction. Column Selection Program (2), User's Manual.

4. Canadian Institute of Steel Construction, Floor System Selection Program (2), User's Manual.

5. Carlson, W.M., Engineering Approach to Information Processing, Journal of the Structural Division, ASCE, ST4, August, 1963.

6. Fenves, S.J., Tabular Decision Logic for Structural Design, Journal of the Structural Division, ASCE, ST6, December, 1966.

7. Fenves, S.J., Representation of the Computer-Aided Design Process by a Network of Decision Tables. Computer and Structures, Vol. 3, No. 5, September, 1973.

8. Fenves, S.J., Scenario for a Third Computer Revolution in Structural Engineering. Journal of the Structural Division, ASCE, ST1, January, 1971.

9. Fenves, S.J., Needs and Prospects for Computer-Aided Structural Design, Proceedings, 1966 Illinois Structural Engineering Conference.

10. Gilersleeve, T.R., Decision Tables and Their Practical Application in Data Processing, Prentice-Hall Inc., 1970.

11. Gilmor, M.I. and Selby, K.A., Structural Steel Design Using STRUDL, Canadian Structral Engineering Conference, 1970.

12. Goel, S.K., Computer-Aided Processing of Structural Design Specifications, Ph.D. Thesis, University of Illinois, 1970.

13. Hatfield, F.J. and Fenves, S.J., The Information Organizer: A System for Symbolic Data Manipulation, Computer and Structures, Vol. 1, 1973.

14. Hughes, M.L., Shank, R.M. and Stein, E.S., Decision Tables, Management Development Institute Publications, Division of Information Industries Inc., Wayne, Penn. U.S.A.

15. Krentz, H.A., CSA Standard S16-1969, Steel Structures for Buildings, Canadian Structural Engineering Conference, 1970.

16. Logcher, R.D. and Sturman, G.M., STRUDL - A Computer System for Structural Design, Journal of the Structural Division, ASCE, ST6, December, 1966.

17. Logcher, R.D., Mozzotta, S.G. and Teague, L.C., Languages for User Defined Member Design Processes, Journal of the Structural Division, ASCE, June, 1967.

18. London, K.R., Decision Tables, Auerbach Publishers, 1972.

19. Lopez, L.A., FILES: Automated Engineering Data Management System, Journal of the Structural Division, ASCE, April, 1975.

20. McDaniel, H., Introduction to Decision Logic Tables, J. Wiley and Sons, 1968.

21. Miller, C.L., Man-Machine Communications in Civil Engineering, Journal of the Structural Division, ASCE, ST4, August, 1963.

22. Noland, J. and Feng, C.C., Formulation of Decision Logic Tables, Journal of the Structural Division ASCE, ST1, January, 1971.

23. Noland, J. and Feng, C.C., ACI Building Code in Decision Table Format, Journal of the Structural Division, ASCE, ST4, April, 1975.

24. Nyman, D.J. and Fenves, S.J., Organizational Model for Design Specifications, Journal of the Structural Division, ASCE, ST4, April, 1975.

25. Pollack, S.L., Hicks, H.T. and Harrison, W.J., Decision Tables: Theory and Practice, J. Wiley, 1971.

26. Shaw, C., Decision Tables, System Development Corp., Santa Monica, California 1965.

27. Springfield, J., Application of Computers in Structural Engineering Design, Canadian Structural Engineering Conference, 1970.

28. Vahl, T., Present and Future Possibilities Within Computer Aided Design, Computer and Structures, Vol. 4, 1974.

29. Wright, R.M., Boyle, L.T. and Melin, J.W., Constraint Processing in Design, Journal of the Structural Division, ASCE, ST1, January, 1971.

APPENDIX A

DECISION TABLE HEIRACHY CHARTS

AND

DECISION TABLE INDEX

Figs. A.1 to A.11 present the heirachy charts of the decision tables in Appendix B. These charts illustrate the order of execution of decision tables when a particular task is performed. The full lines in the charts represent direct execution and the broken lines represent conditional execution.

Fig. A.12 provides an explanation of the decision table designation used in Figs. A.1 to A.11.

A decision table index is provided in Fig. A.13 where the table number for computer input is tabulated with the corresponding code designation. In referring to a decison table in the text, both systems of designation are used. For example, Decison Table X.2 of Fig. A.1 is referred to as X.2(2) where the number in brackets indicates the computer input number for this table.

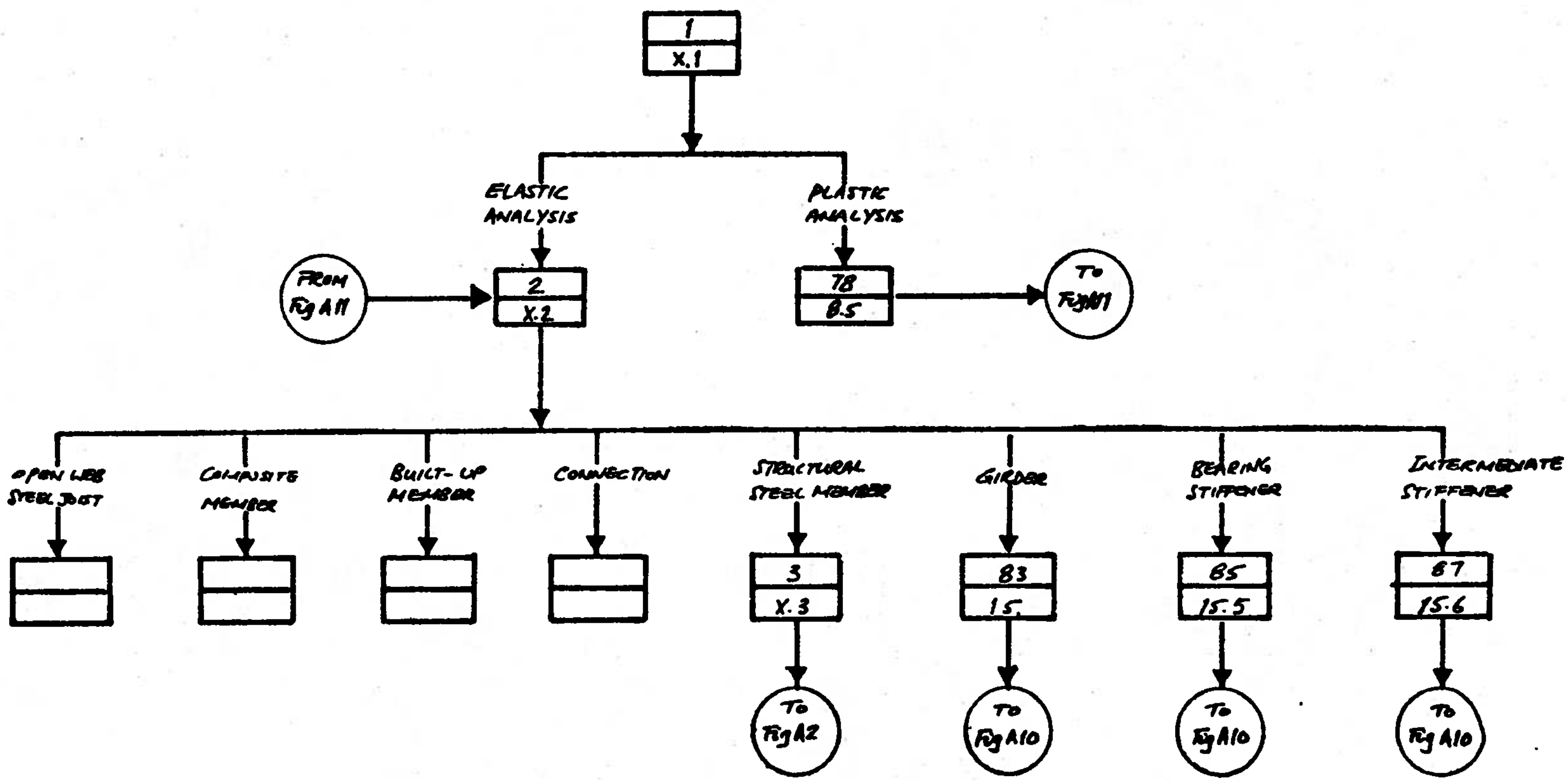


FIG. A.1 DECISION TABLE HEIRACHY CHART 1

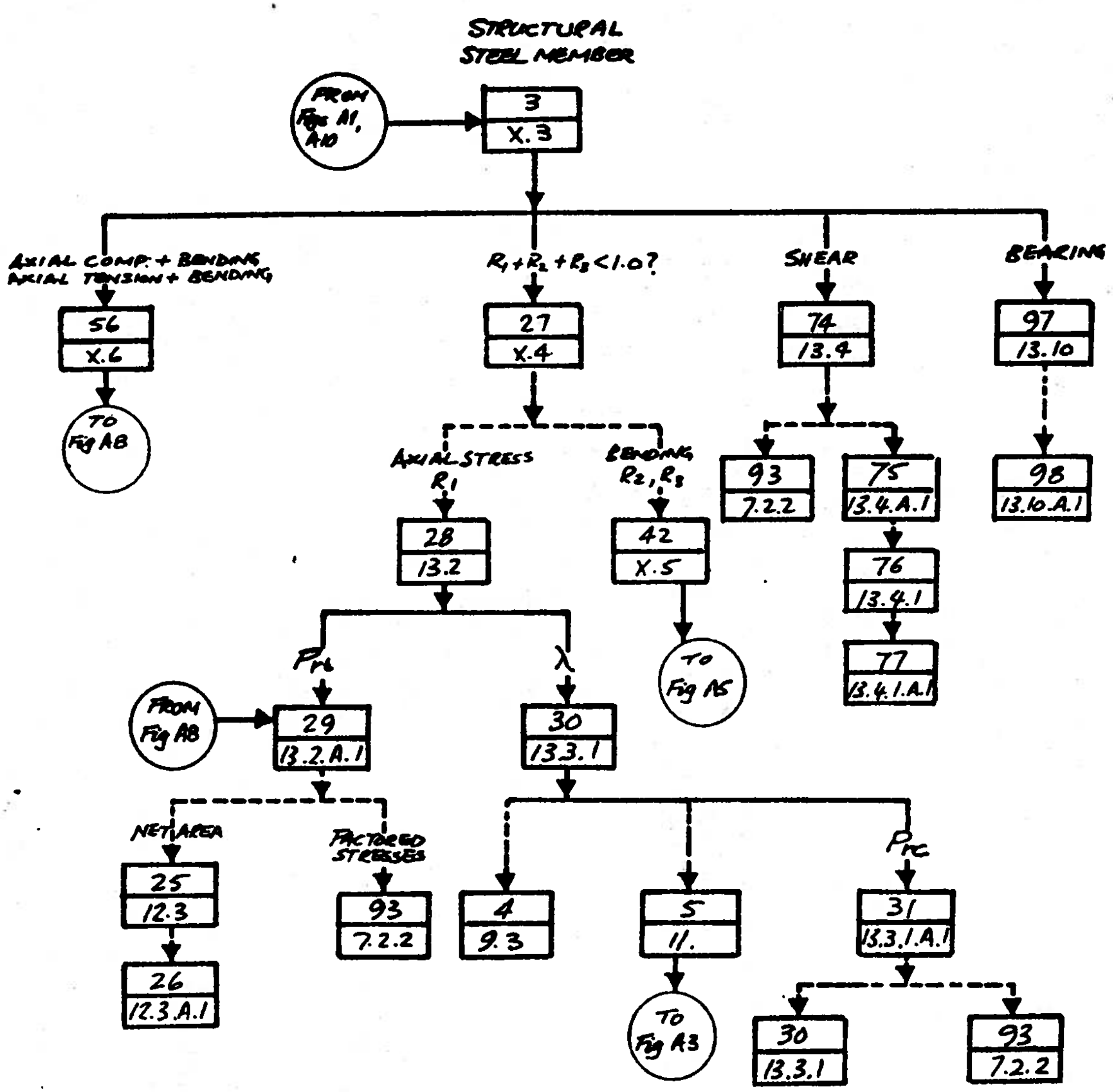


FIG. A.2 DECISION TABLE HEIRACHY CHART 2

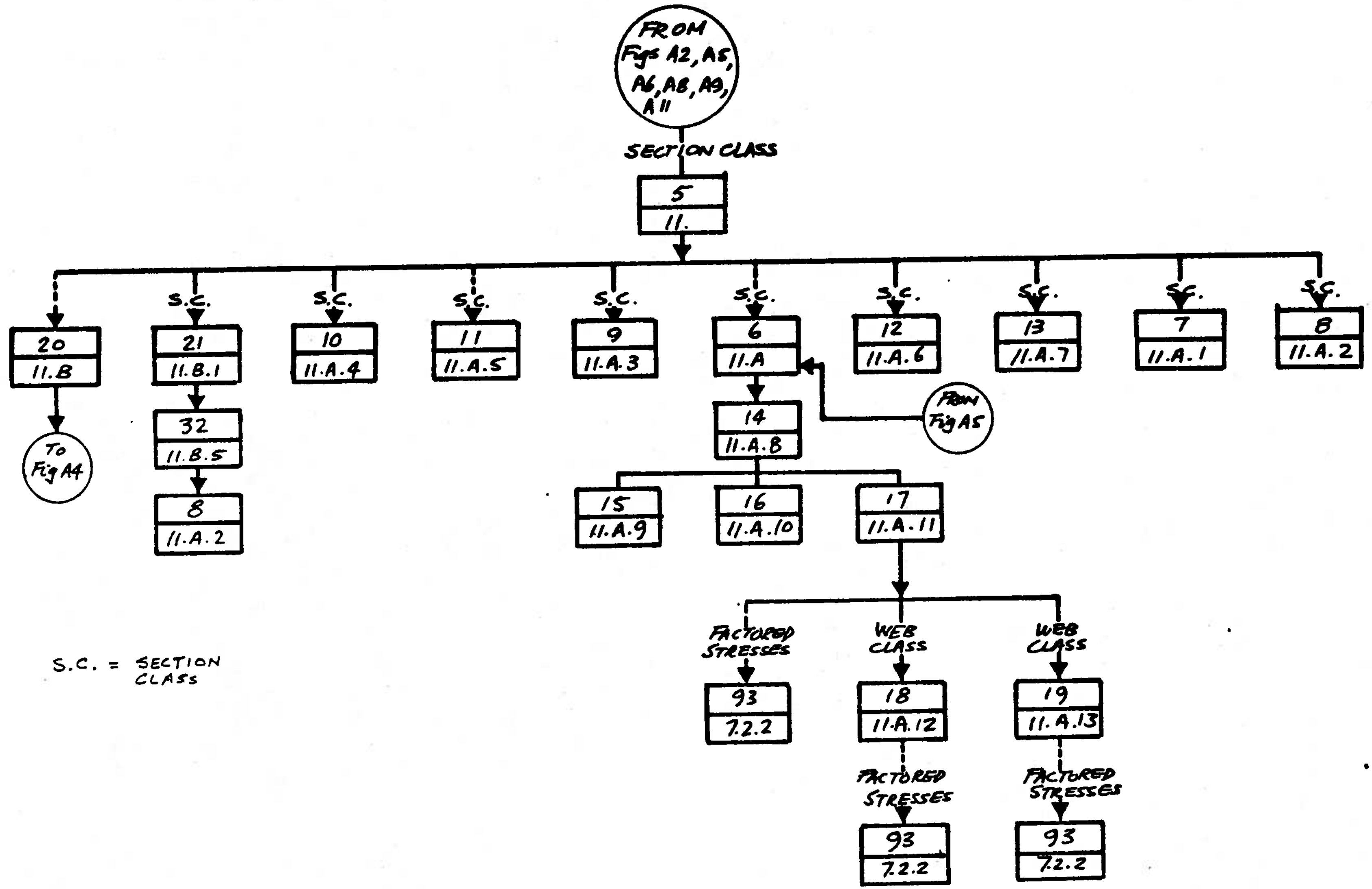


FIG. A.3 DECISION TABLE HEIRACHY CHART 3

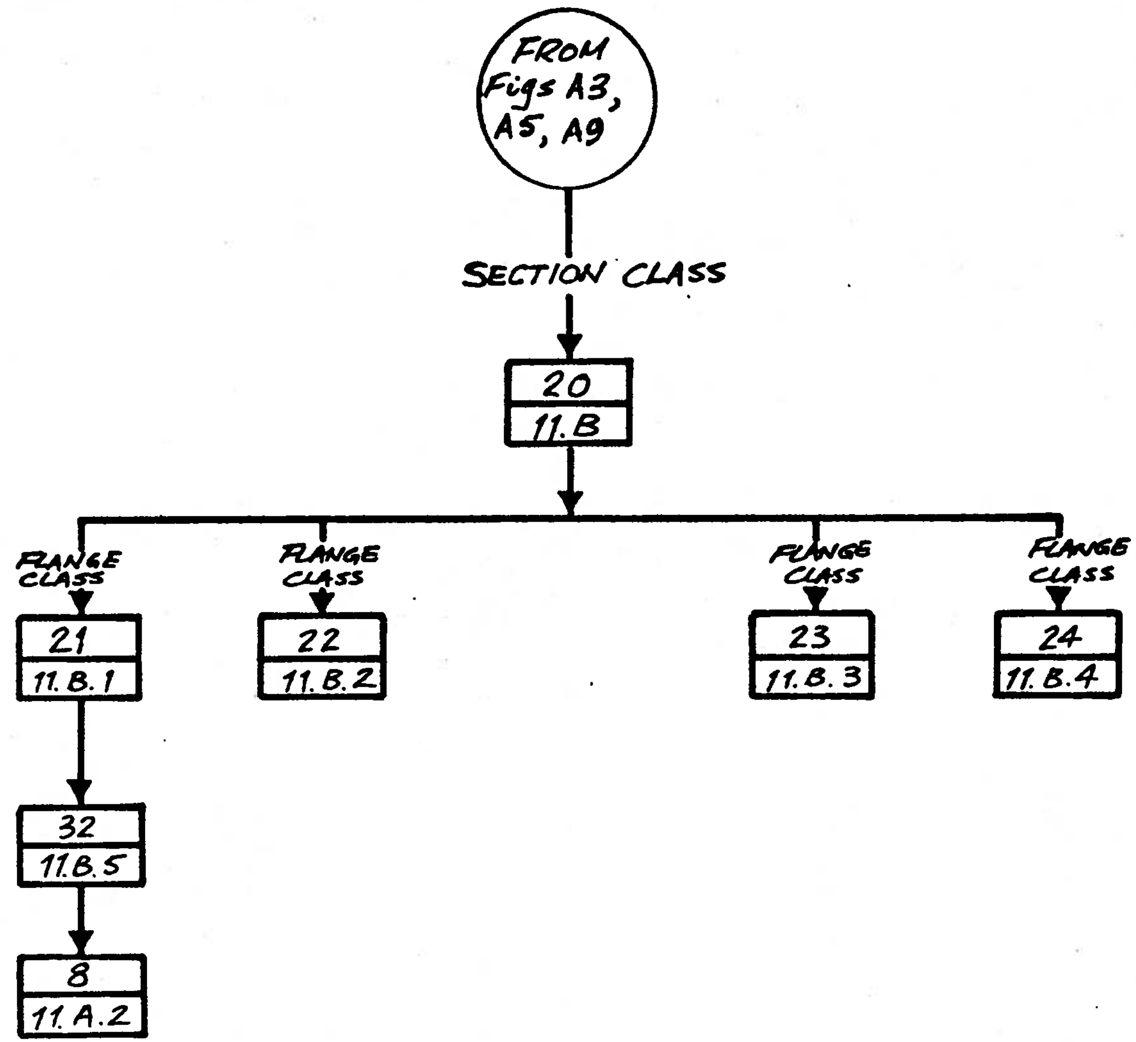


FIG. A.4 DECISION TABLE HEIRACHY CHART 4

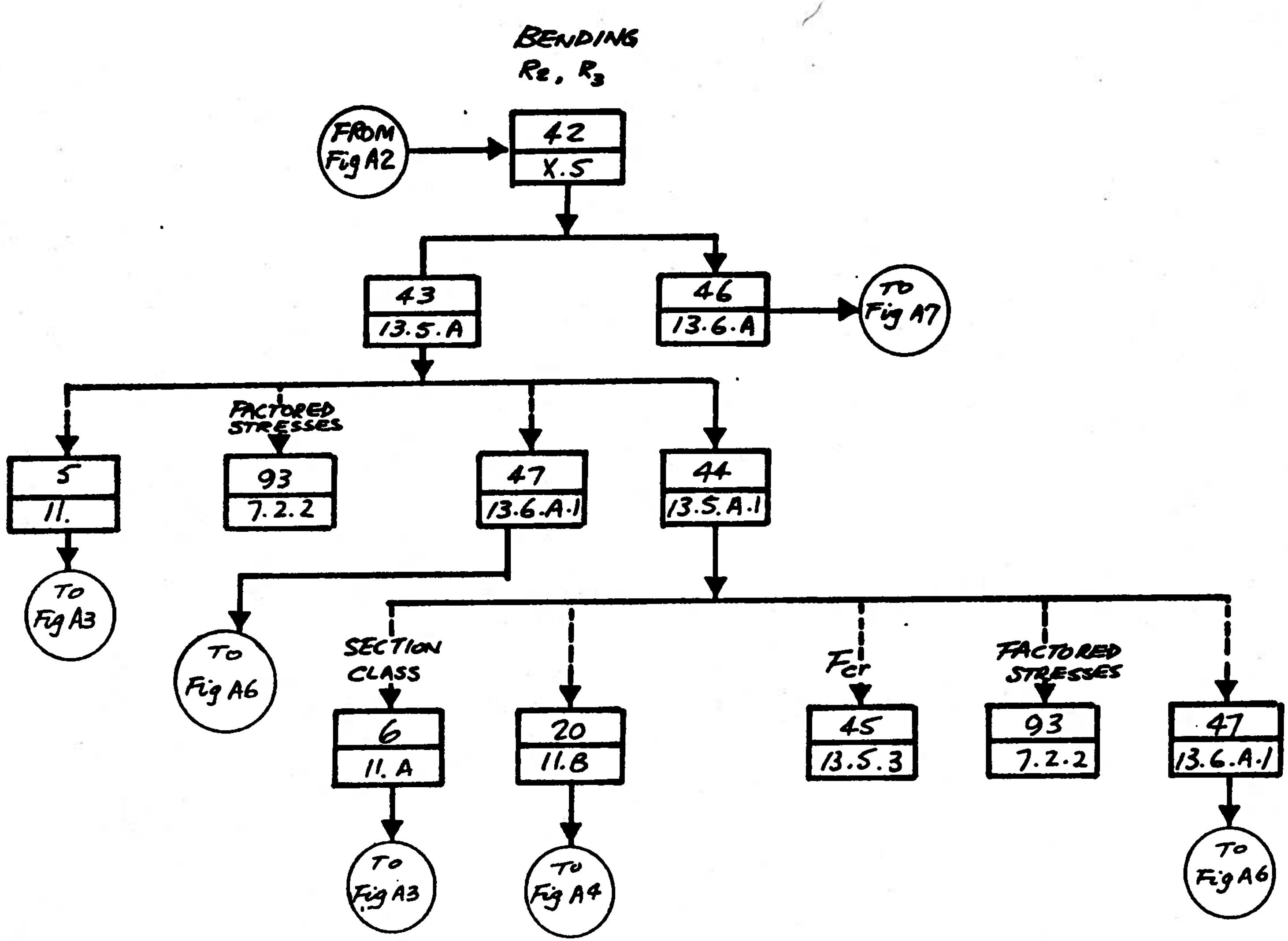


FIG. A.5 DECISION TABLE HEIRACHY CHART 5

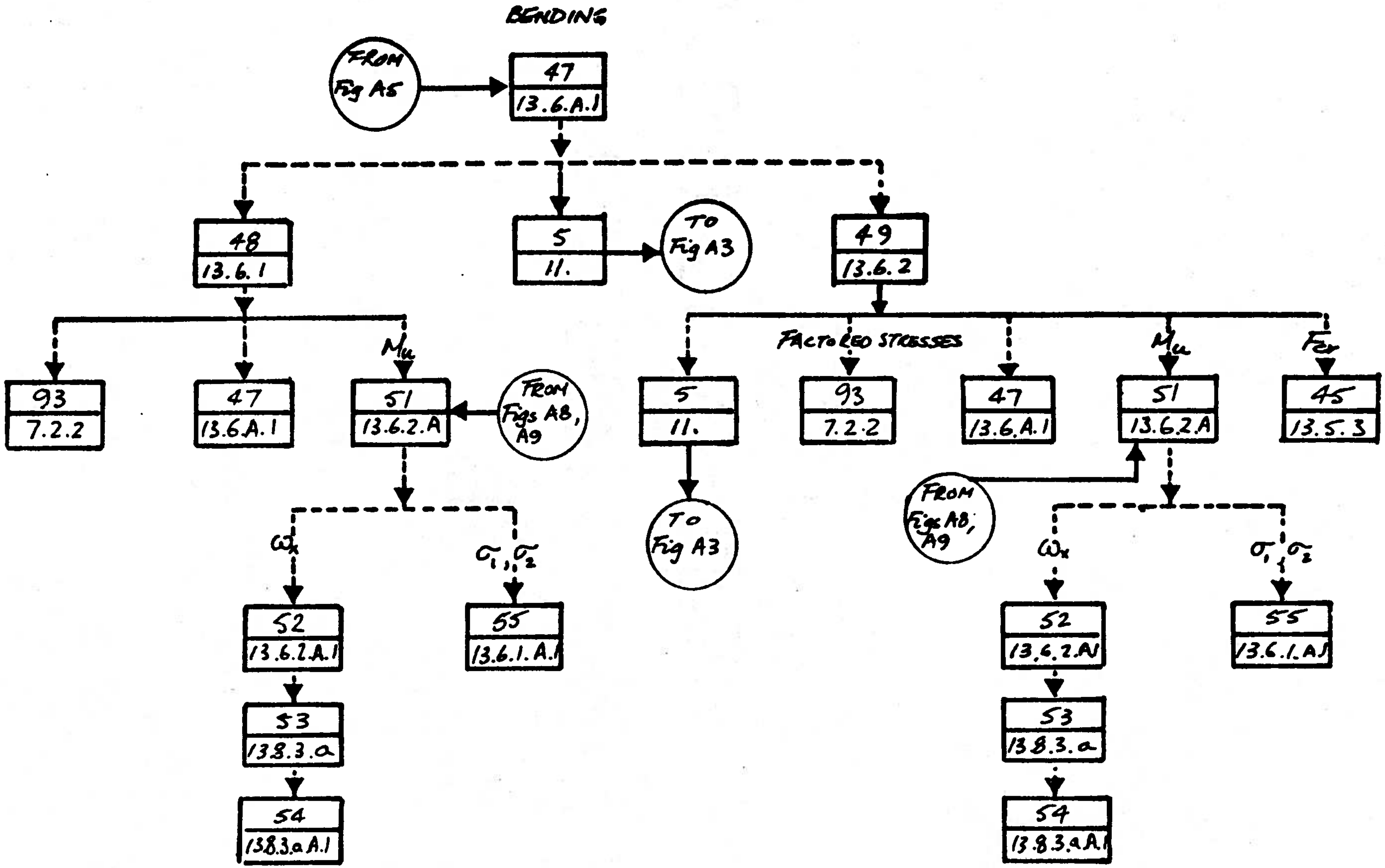


FIG. A.6 DECISION TABLE HEIRACHY CHART 6

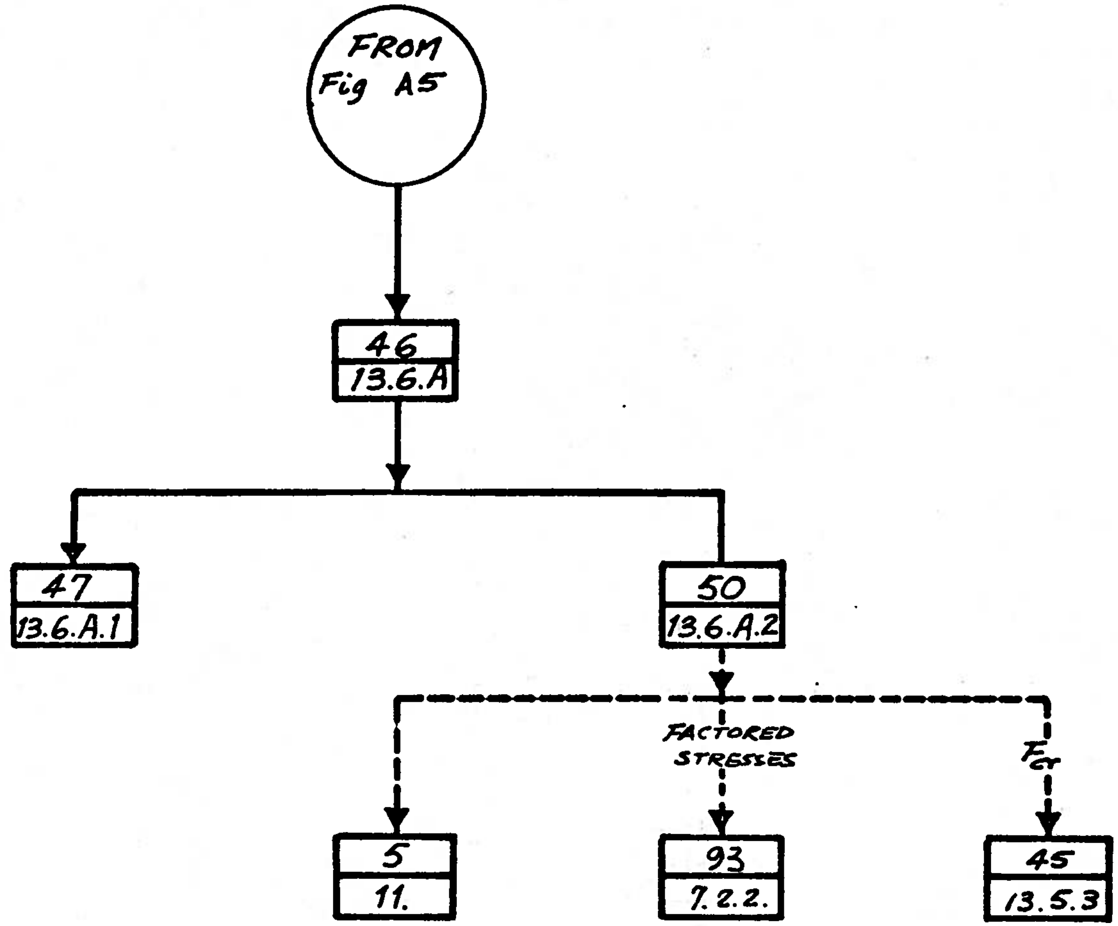


FIG. A.7 DECISION TABLE HEIRACHY CHART 7

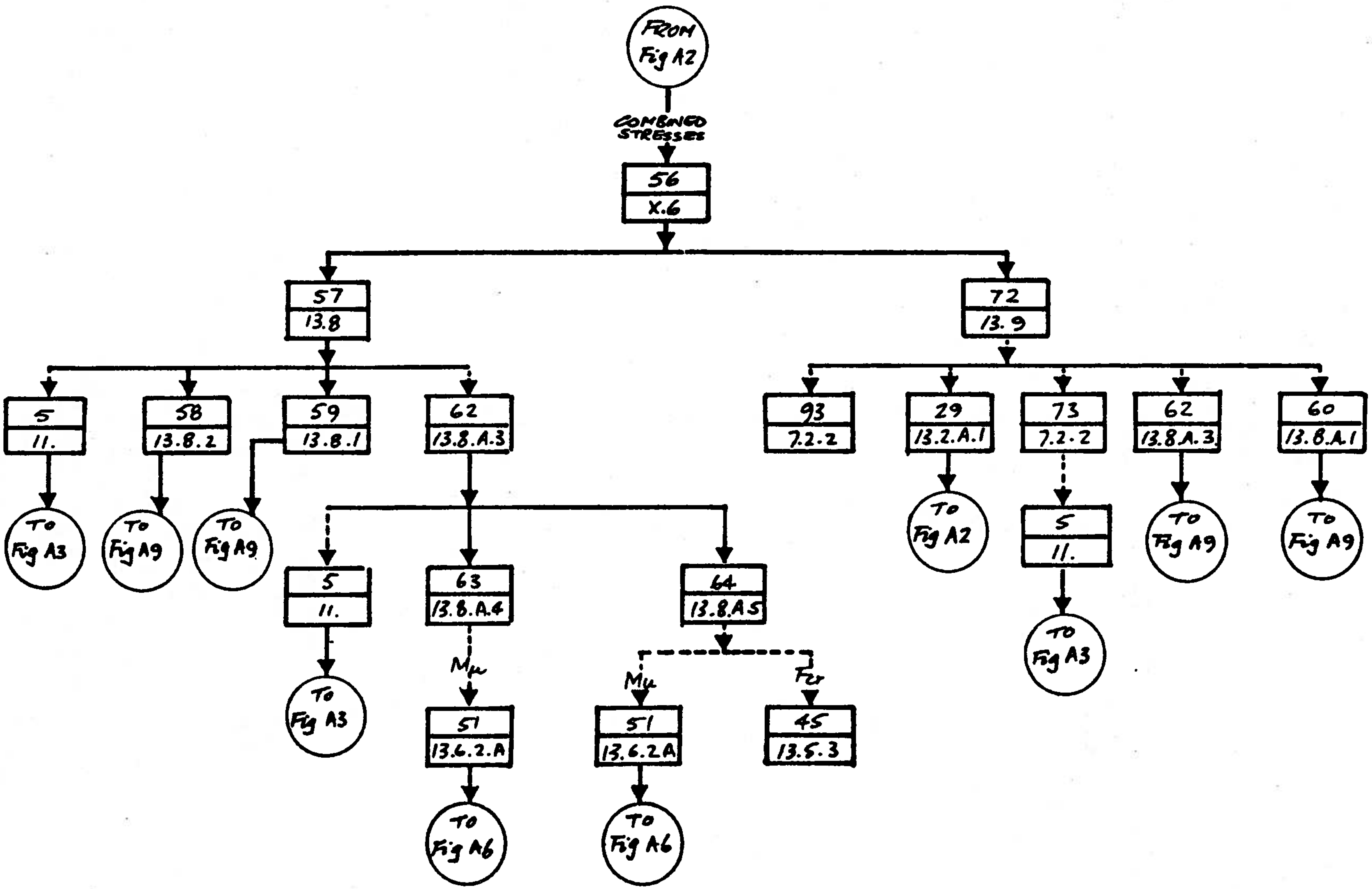


FIG. A.8 DECISION TABLE HEIRACHY CHART 8

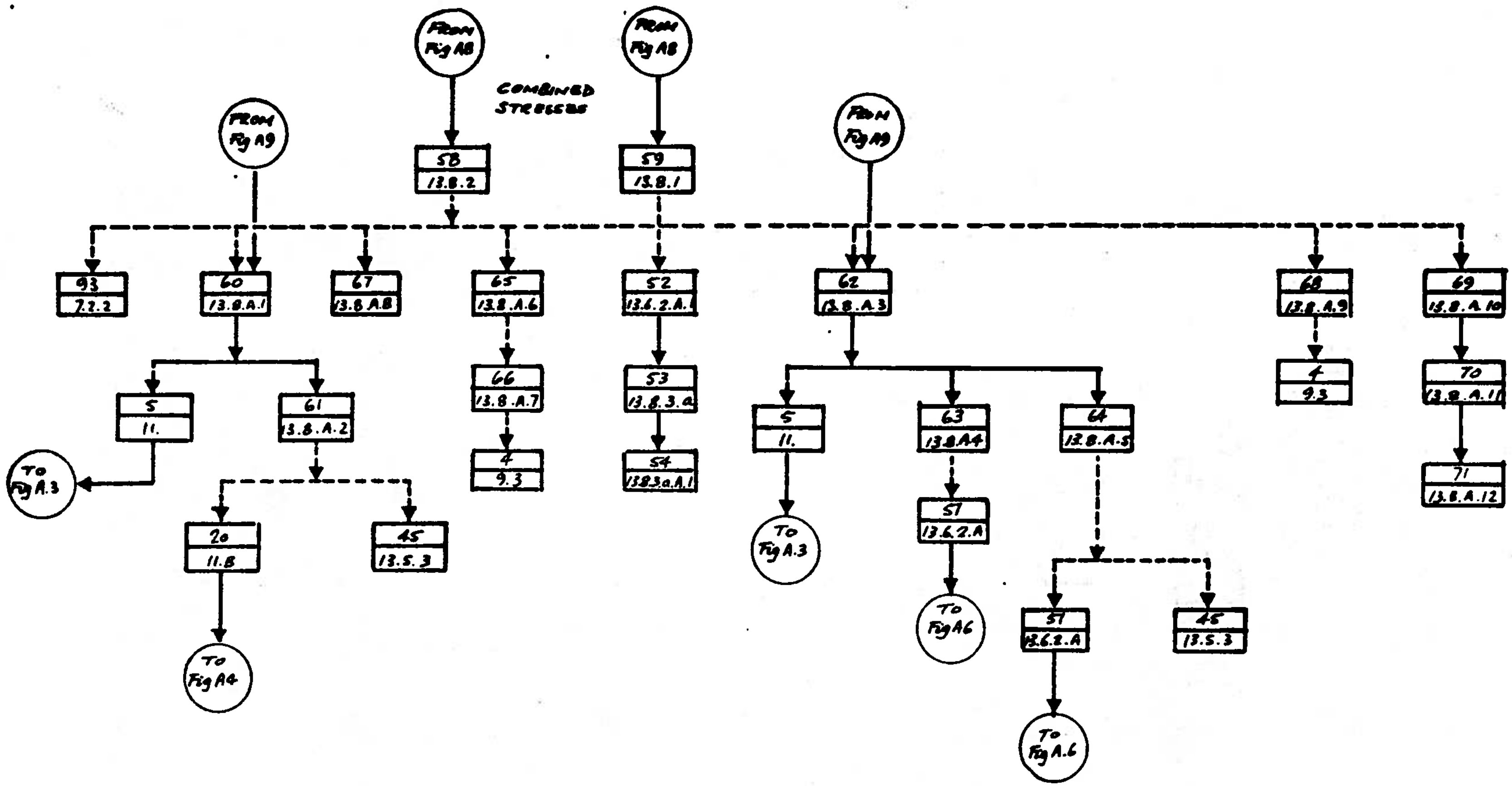


FIG. A.9 DECISION TABLE HEIRACHY CHART 9

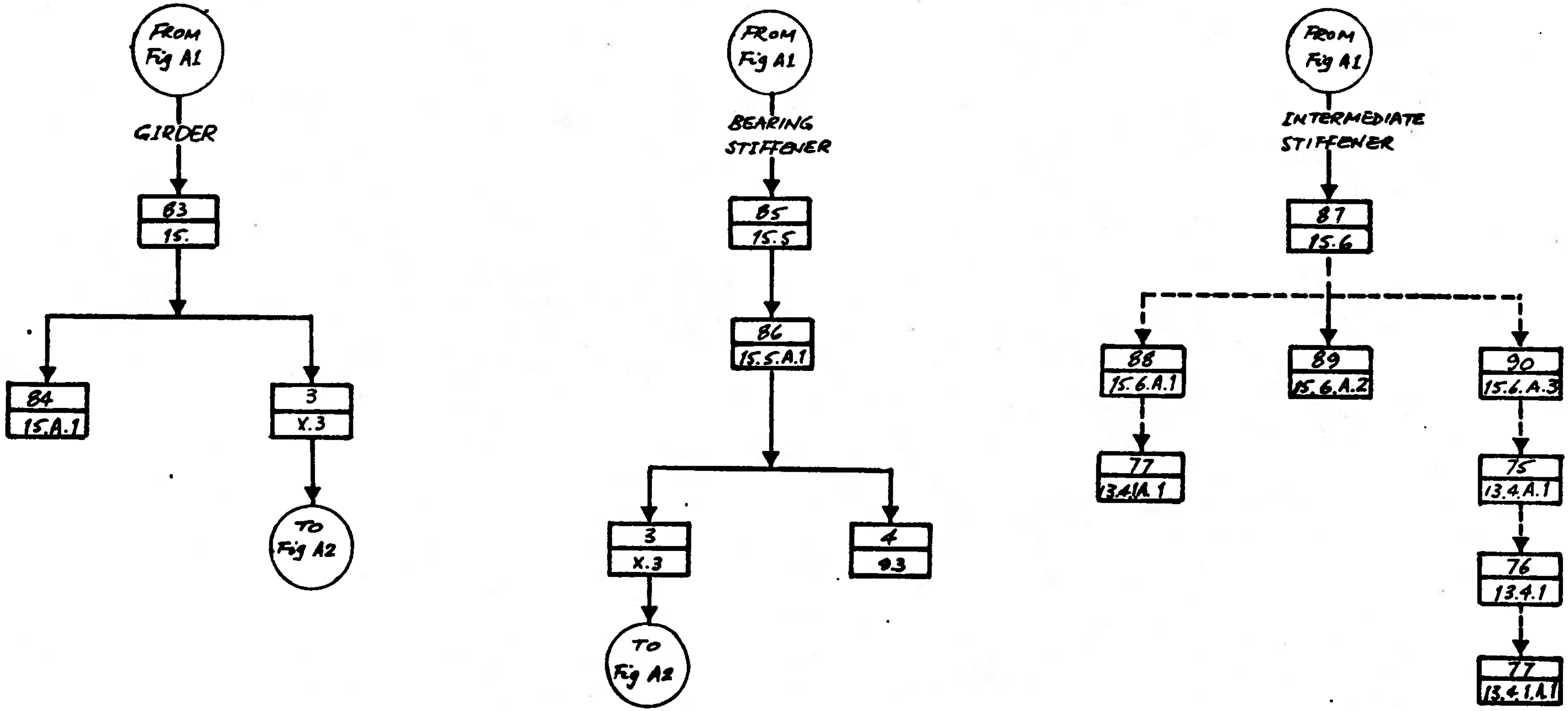


FIG. A.10 DECISION TABLE HEIRACHY CHART 10

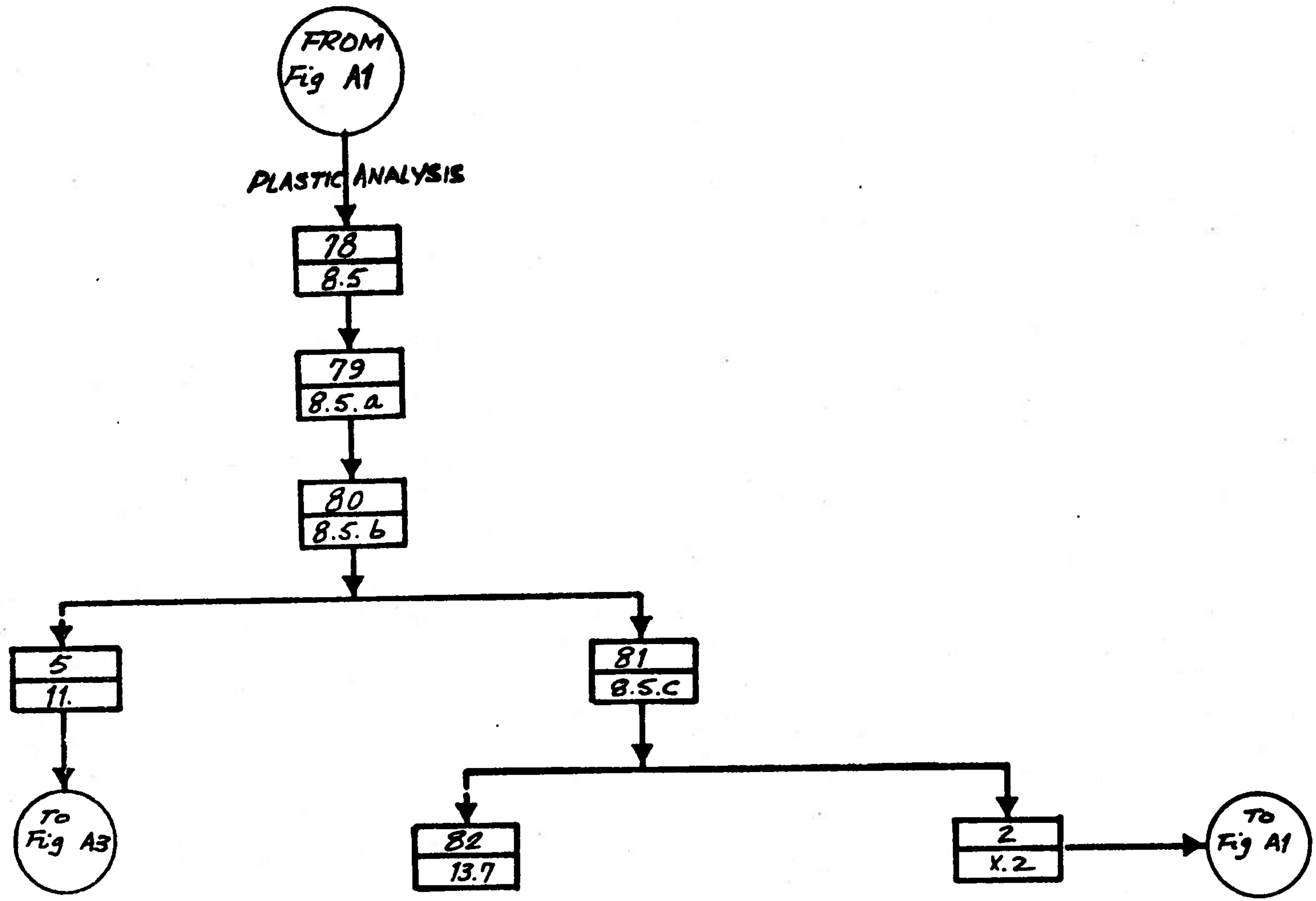


FIG. A.11 DECISION TABLE HEIRACHY CHART 11

T
X. m

T
Sect. θ. n

T = Table Number of Computer Input. (T is an integer number)

X = Indicates a "Switching Table" which does not correspond to a Code section.

m = The "Switching Table" number (m is an integer number)

Sect = Section or clause of the Code.

θ = Level 1 subtable designation (θ is either A or B)

n = Level 2 subtable designation (n is an integer number).

Fig. A.12 Explanation of Table Designation

S16.1 SECTION NUMBER	DECISION TABLE NUMBER	S16.1 SECTION NUMBER	DECISION TABLE NUMBER	S16.1 SECTION NUMBER	DECISION TABLE NUMBER
X.1	1		39	13.4.1.A.1	77
X.2	2		40	8.5	78
X.3	3		41	8.5.a.	79
9.3	4	X.5	42	8.5.b.	80
11.	5	13.5.A	43	8.5.c.	81
11.A.	6	13.5.A.1	44	13.7	82
11.A.1	7	13.5.3	45	15.	83
11.A.2	8	13.6.A	46	15.A.1	84
11.A.3	9	13.6.A.1	47	15.5	85
11.A.4	10	13.6.1	48	15.5.A.1	86
11.A.5	11	13.6.2	49	15.6	87
11.A.6	12	13.6.A.2	50	15.6.A.1	88
11.A.7	13	13.6.2.A	51	15.6.A.2	89
11.A.8	14	13.6.2.A.1	52	15.6.A.3	90
11.A.9	15	13.8.3.a	53	13.9	91
11.A.10	16	13.8.3.a.A.1	54	13.9.A.1	92
11.A.11	17	13.6.1.A.1	55	7.2.2	93
11.A.12	18	X.6	56	7.2.3	94
11.A.13	19	13.8	57	7.2.4	95
11.B	20	13.8.2	58	7.2.5	96
11.B.1	21	13.8.1	59	13.10	97
11.B.2	22	13.8.A.1	60	13.10.A.1	98
11.B.3	23	13.8.A.2	61		99
11.B.4	24	13.8.A.3	62		100
12.3	25	13.8.A.4	63		101
12.3.A.1	26	13.8.A.5	64		102
X.4	27	13.8.A.6	65		103
13.2	28	13.8.A.7	66		104
13.2.A.1	29	13.8.A.8	67		105
13.3.1	30	13.8.A.9	68		106
13.3.1.A.1	31	13.8.A.10	69		107
11.B.5	32	13.8.A.11	70		108
	33	13.8.A.12	71		109
	34	13.9	72		110
	35	13.9.A.1	73		111
	36	13.4	74		112
	37	13.4.A.1	75		113
	38	13.4.1	76		114

FIG. A13 Decision Table Index

APPENDIX B

GLOBAL DATA SHEETS, DATA DEPENDENT LISTS

AND

DECISION TABLES OF CSA S16.1

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
1	$ELAN		1	Elastic Analysis		*
2	$PLAN		1	Plastic Analysis		*
10	$SSM		2	Structural Steel Member		
11	$OWSJ		2	Open-Web Steel Joists		
12	$COM		2	Composite Construction Member		
13	$BUM		2	Built-Up Member		
14	$CON		2	Connection		
15	$GIRD		2	Girder		
16	$BSTI		2	Bearing Stiffener		
17	$INST		2	Intermediate Transverse Stiffener		
19	$TEN			Tension Member Without Holes		*
20	$MPCON			Member Using Pin-Connections		*
21	$LSM		4	Continuously Laterally Supported Member		*
22	$LUSM		4	Laterally Unsupported Member		*
23	$CPM		6	Compression Member		*
24	$TENM		6	Tension Member		*
25	$CTRUS			Compression Member in Truss		*
26	$BBELS		8	Beam with Both Ends With Effective Lateral Support To The Compression Flange		*
27	$BOELS		8	Beam With Only One End With Effective Lateral Support For The Compression Flange		*
28	$LOAD1		10	Member Not Subjected To Transverse Loads In The Minor Axis Plane Between Supports		*
29	$LOAD2		10	Member Subjected To Distributed Load Or Series Of Point Loads In The Minor Axis Plane Between Supports		*
30	$LOAD3		10	Member Subjected To A Concentrated Load Or Moment In The Minor Axis Plane Between Supports		*
40	$CURV1		12	Member Bent In Single Curvature About The Major Axis		*
41	$CURV2		12	Member Bent In Double Curvature About The Major Axis		*
45	ANET	12.3		Net Area	A_n	*
46	AGRS			Gross Area/Nominal Area of Pin	A_g	*
47	AW			Web Area	A_w	*
48	AF			Flange Area	A_f	*
55	W			Width of Element (In A Plate)		

*See lists of dependents

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
56	W1			Width Of One Angle Leg		
57	W2			Width Of The Other Leg Of Angle		
58	ST			Flange Thickness	t	*
59	SB			Flange Width	b	*
60	D1			Outside Dia. Of Circular Section		*
61	D2			Dia. of Rivet Or Bolt/Overall Depth Of A Section	d	*
62	FNU			Number Of Holes In A Tension Member	n	*
63	SH			Depth Of Web	h	*
64	SW			Thickness Of Web	w	*
65	SD			Depth of Section	d	*
66	TP			Thickness Of Plate	t_p	*
67	TWEB			Thickness Of Web Of Girder	t_{web}	*
75	FCR	13.5.3		Critical Plate Buckling Stress	F_{cr}	*
76	FY			Specified Minimum Yield Stress, Or Yield Strength	F_y	*
77	FU			Specified Minimum Tensile Strength	F_u	*
78	E			Elastic Modulus Of Steel (29000 ksi Assumed)	E	*
79	z			Plastic Modulus Of A Steel Section	z	*
80	S			Elastic Modulus Of A Steel Section	s	*
81	G			Shear Modulus Of Steel (11200 ksi Assumed)	G	*
82	VJ			St. Venant's Torsional Constant	J	*
83	CW			Warping Torsional Constant	C_w	*
95	$CLAS1	11	14	Set Of Section = 1		*
96	$CLAS2	11	14	Class Of Section = 2		*
97	$CLAS3	11	14	Class Of Section = 3		*
98	$CLAS4	11	14	Class Of Section = 4		*
99	$WBCL1	11.A	16	Web Class = 1		*
100	$WBCL2	11.A	16	Web Class = 2		*
101	$WBCL3	11.A	16	Web Class = 3		*
102	$WBCL4	11.A	16	Web Class = 4		*
110	$FLCL1	11.B	18	Flange Class = 1		*
111	$FLCL2	11.B	18	Flange Class = 2		*
112	$FLCL3	11.B	18	Flange Class = 3		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
113	$FLCL4	11.8	18	Flange Class = 4		*
119	$BF			Zero Bearing Force	$B_f=0$	*
120	FPF	7.2.2		Factored Axial Force	P_f	*
121	$PFO			Zero Axial Force	$P_f=0$	*
122	$MFXO			Zero Moment About Major A.is	$M_{fx}=0$	*
123	$MFYO			Zero Moment About Minor Axis	$M_{fy}=0$	*
124	$VO			Zero Shear Force	$V_f=0$	*
125	$PFLTO		20	Axial Compression	$P_f>0$	*
126	$PFSTO		20	Axial Tension	$P_f<0$	
127	$MYGTO			Moment Applied About The Minor Axis	$M_{fy}>0$	*
128	$MXGTO			Moment Applied About The Major Axis	$M_{fx}>0$	*
129	$BMGTO			M_{fx}.and.$M_{fy}>0$		*
130	$MGTO			M_{fx}.and.or $M_{fy}>0$		*
131	FPRT	13.2.A.1		Factored Axial Tensile Resistance	P_{rt}	*
132	FMFX	7.2.2		Factored Moment About Major Axis	M_{fx}	*
133	FMFY	7.2.2		Factored Moment About Minor Axis	M_{fy}	*
134	FMRX	13.6.A.1		Factored Moment Of Resistance About Major Axis	M_{rx}	*
135	FMRY			Factored Moment Of Resistance About Minor Axis	M_{ry}	*
136	$PC015			$P_f/C_y > 0.15$		
137	FMU	13.6.2.A		Moment of Resistance Of a Member Subjected To Lateral Buckling	M_u	*
138	FMP			Plastic Moment	M_p	*
139	$MU23			$M_u > 2/3\ M_p$		*
140	FMF1			Smaller Moment At The Ends Of The Unbraced Length About The Major Axis	M_{f1}	*
141	FMF2			Larger Moment At The Ends Of The Unbraced Length About The Major Axis	M_{f2}	*
142	FMY			Yield Moment	M_y	*
143	FMF1Y			Smaller Moment At The Ends Of The Unbraced Length About The Minor Axis	M_{f1y}	*
144	FMF2Y			Larger Moment At The Ends Of The Unbraced Length About The Minor Axis	M_{f2y}	*
145	FPRC	13.3.1.A.1		Factored Axial Compressive Resistance	P_{rc}	*
146	$MU23y			$M_u > 2/3\ M_y$		*
150	R1	13.2		P_f/P_r Ratio		*
151	R2	X.5		M_{fx}/M_{rx} Ratio		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
152	R3	X.5		M_{fy}/M_{ry} Ratio		*
153	R4			V_f/V_r Ratio		
154	$R123			$R_1 + R_2 + R_3 \leq 1.0$		
155	$R4			$V_f/V_r \leq 1.0$		
165	$SANG		22	Single Angle		*
166	$DANG		22	Double Angle		*
167	$CHOLW		22	Circular Hollow Section		*
168	$T		22	T Section		*
169	$I		22	I Section		*
170	$CHAL		22	Channel Section		*
171	$REHOL		22	Rectangular Hollow Section		*
172	$BOX		22	Box Section		*
173	$FCOV		22	Flange Cover Plate/Diaphram Plate		*
174	$PCOV		22	Perforated Cover Plate		*
175	$ISTIF		22	Plate Girder Transverse Intermediate Stiffener/Bearing Stiffener		*
176	$CHPT			Channel Prevented From Twisting		
177	$DOSY			Doubly Symmetric Section		*
178	$ASYLO			Sections Axis Of Sym. In Plane Of Loading		*
179	$CONT			Is The Double Angle Continuously Connected By Adequate Mechanical Fasteners Or Welds?		*
180	$DHOL			Doubly Symmetric Hollow Section (Square Hollow Section Or Circular Hollow Section))		*
185	UKX	9.3		Effective Length Factor About The Major Axis	K_x	*
186	ELX			Length Of Compression Or Tension Member Along The Major Axis	L_x	*
187	UFL			Unsupported Length Of Compression Flange	L	*
188	SRY			Radius Of Gyration Of Member About Its Minor Axis	r_y	*
189	SRT			Radius Of Gyration About Its Axis Of Symmetry Of A Tee Section Comprising The Compression Flange and 1/6 Of Web	r_t	*
190	CY			Axial Compressive Load At Yield Stress	C_y	*
191	SKB			Plate Buckling Coefficient	K_b	*
192	FIY			Moment Of Inertia About The Minor Axis	I_y	*
193	FIX			Moment Of Inertia About The Major Axis	I_x	

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
194	OMEGAX	13.6.2.A.1		Coefficient Used To Determine Equivalent Uniform Bending Effect In Beam Columns Or Coefficient In Column Equation About The Major Axis	ω_x	*
195	SIGMA1	13.6.1.A.1		$\sigma_1 = 20000/(Ld/A_f)$	σ_1	*
196	SIGMA2	13.6.1.A.1		$\sigma_2 = 250000/(L/r_t)^2$	σ_2	*
197	FHAI			Performance Factor	ϕ	*
198	S(1)			Pitch Between Successive Fastener Holes Along Critical Section		*
199	S(2)			Pitch Between Successive Fastener Holes Along Critical Section		*
200	S(3)			Pitch Between Successive Fastener Holes Along Critical Section		*
201	S(4)			Pitch Between Successive Fastener Holes Along Critical Section		*
202	G(1)			Gauge Distance Between Fasteners Along Critical Section		*
203	G(2)			Gauge Distance Between Fasteners Along Critical Section		*
204	G(3)			Gauge Distance Between Fasteners Along Critical Section		*
205	G(4)			Gauge Distance Between Fasteners Along Critical Section		*
208	SRX			Fadius Of Gyration Of Member About Its Major Axis	r_x	*
211	TSLRAX	13.2		Tension Member Slenderness Ratio About The Major Axis		*
212	TSLRAY	13.2		Tension Member Slenderness Ratio About The Minor Axis		*
213	TLR	13.2		Maximum of (TSLRAX, TSLRAY)		*
214	UKY	9.3		Effective Length Factor About The Minor Axis	K_y	*
215	UKLR	9.3		Maximum of (SLRAX, SLRAY) In Compression Member		*
216	ELY			Length Of Compression Or Tension Member Along The Minor Axis	L_y	*
217	SLRAY	9.3		Compression Member Slenderness Ratio About The Minor Axis		*
218	REDUTN	12.3.A.1		$\sum_K S_K^2/4g_k$		*
219	SLRAX	9.3		Compression Member Slenderness Ratio About The Major Axis		*
220	$SWYX			Sway Effects About The Major Axis Included In Analysis/ Sway Effects About The Major Axis Resisted By Bracing Or Shear Wall		*
221	$SWYY			Sway Effects About The Minor Axis Included In Analysis/ Sway Effects About The Minor Axis Resisted By Bracing Or Shear Wall		*
222	$ROTX			Rotational Restraint At The End Of The Unbraced Lengths Shows K Can Be < 1.0 Along The Major Axis		*
223	$ROTY			Rotational Restraint At The End Of The Unbraced Lengths Shows K Can Be < 1.0 Along The Minor Axis		
224	$WBACM		26	Web In Axial Compression		
225	$WBFLX		26	Web In Flexural Compression		

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
226	$WBCOB		26	Web In Combine Axial And Flexural Compression		
230	SNMGA	13.3.1		Non-Dimensional Slenderness Ratio In Column Formula	λ	*
231	$NMGA1		28	$0 \leq \lambda \leq 1.0$		*
232	$NMGA2		28	$1.0 < \lambda \leq 2.0$		*
233	$NMGA3		28	$2.0 < \lambda \leq 3.6$		*
234	$NMGA4		28	$3.6 < \lambda$		*
236	$DT1		30	$D1/t \leq 260/\sqrt{F_y}$		*
237	$DT2		30	$260/\sqrt{F_y} < D1/t \leq 365/\sqrt{F_y}$		*
238	$DT3		30	$365/\sqrt{F_y} < D1/t \leq 3300/F_y$		*
239	$DT4		30	$D1/t > 3300/F_y$		*
241	$BT2		32	$b/t \leq 75/\sqrt{F_y}$		*
242	$BT2		32	$b/t > 75/\sqrt{F_y}$		*
243	$BT3		34	$b/t \leq 100/\sqrt{F_y}$		*
244	$BT4		34	$b/t > 100/\sqrt{F_y}$		*
245	$BT5		36	$b/t \leq 200/\sqrt{F_y}$		*
246	$BT6		36	$200/\sqrt{F_y} < b/t \leq 255/\sqrt{F_y}$		*
247	$BT7		36	$b/t > 255/\sqrt{F_y}$		*
248	$BT8		38	$b/t \leq 320/\sqrt{F_y}$		*
249	$BT9		38	$b/t > 320/\sqrt{F_y}$		*
250	$BT10		40	$b/t \leq 160/\sqrt{F_y}$		*
251	$BT11		40	$160/\sqrt{F_y} < b/t \leq 200/\sqrt{F_y}$		*
252	$BT12		40	$200/\sqrt{F_y} < b/t \leq 255/\sqrt{F_y}$		*
253	$BT13		40	$b/t > 255/\sqrt{F_y}$		*
254	$BT14		42	$b/t \leq 54/\sqrt{F_y}$		*
255	$BT15		42	$54/\sqrt{F_y} < b/t \leq 64/\sqrt{F_y}$		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
256	$BT16		42	$64/\sqrt{F_y} < b/t \leq 100/\sqrt{F_y}$		*
257	$BT17		42	$b/t > 100/\sqrt{F_y}$		*
258	$BT18			$b/t \leq 201/\sqrt{K_b/F_y}$		*
260	$hW1		44	$h/w \leq 255/\sqrt{F_y}$		*
261	$hW2		44	$h/w < 255/\sqrt{F_y}$		*
262	$hW3		46	$h/w \leq 420/\sqrt{F_y}$		*
263	$hW4.		46	$420/\sqrt{F_y} < h/w \leq 520/\sqrt{F_y}$		*
264	$hW5		46	$520/\sqrt{F_y} < h/w \leq 690/\sqrt{F_y}$		*
265	$hW6		46	$h/w > 690/\sqrt{F_y}$		*
266	$hW7		48	$h/w \leq 420/\sqrt{F_y}(1-1.4 \frac{P_f}{C_y})$		*
267	$hW8		48	$420/\sqrt{F_y}(1-1.4 \frac{P_f}{C_y}) < h/w \leq 450/\sqrt{F_y} (1-0.43 \frac{P_f}{C_y})$		*
268	$hW9		48	$h/w > 450/\sqrt{F_y} (1-0.43 \frac{P_f}{C_y})$		*
269	$hW10		50	$420/\sqrt{F_y}(1-1.4 \frac{P_f}{C_y}) < h/w \leq 520/\sqrt{F_y} (1-1.28 \frac{P_f}{C_y})$		*
270	$hW11		50	$520/\sqrt{F_y} (1-1.28 \frac{P_f}{C_y}) < h/w \leq 690/\sqrt{F_y} (1-2.6 \frac{P_f}{C_y})$		*
271	$hW12		50	$h/w > 690/\sqrt{F_y} (1-2.6 \frac{P_f}{C_y})$		
272	$hW13		52	$h/w \leq 690/\sqrt{F_y}$		*
273	$hW14		52	$690/\sqrt{F_y} < h/w \leq 690/\sqrt{\phi s F_{cr}/\phi s}$		*
274	$hW15		52	$690/\sqrt{\phi s F_{cr}/\phi s} < h/w \leq 12000/F_y$		*
275	$hW16			$h/w > 12000/F_y$		*
276	$hW17		50	$h/w \leq 420/\sqrt{F_y} (1-1.4 P_f/C_y)$		*
280	$FYFU1		54	$F_y/F_u \leq 0.75$		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
281	$FYFU2		54	$0.75 < F_y/F_u \leq 0.85$		*
282	$FYFU3		54	$0.85 < F_y/F_u$		*
283	$ANG1			$A_n/A_g \geq F_y/F_u$		*
284	$HW17			$h/w \leq 167\sqrt{K_v/F_y}$		*
285	$HW18			$167\sqrt{K_v/F_y} < h/w \leq 190\sqrt{K_v/F_y}$		*
286	$HW19			$190\sqrt{K_v/F_y} < h/w \leq 239\sqrt{K_v/F_y}$		*
287	$HW20			$239\sqrt{K_v/F_y} < h/w$		*
290	$CNB		56	Axial Compression And Bending		
291	$TNB		56	Axial Tension And Bending		
295	FMRX1	13.8.A.1		Factored Moment of Resistance About The Major Axis Calculated By Clause 13.5	M_{rx1}	*
296	FMRY1	13.8.A.1		Factored Moment Of Resistance About The Minor Axis Calculated By Clause 13.5	M_{ry1}	*
297	FPRC1	13.8.A.6		Factored Axial Compressive Resistance Used In Data 337 and Data 339	P_{rc1}	*
298	FPRC2	13.8.A.8		Factored Axial Compressive Resistance Used In Data 336 and Data 338	P_{rc2}	*
299	FMRX2	13.8.A.3		Factored Moment Of Resistance About Major Axis Defined In Clause 13.6	M_{rx2}	*
300	FMR1	13.9.A.1		Factored Moment Of Resistance About Both Axis In Axial Tension And Bending	M_{r1}	*
310	OMEGAY	13.8.A.10		Coefficient Used To Determine Equivalent Uniform Bending Effect In Beam Columns Or Coefficient In Column Equation About The Minor Axis	ω_y	*
311	CEX	13.8.A.9		Euler Buckling Load About The Major Axis	C_{ex}	*
312	CEY	13.8.A.9		Euler Buckling Load About The Minor Axis	C_{ey}	*
313	SNMGA1	13.8.A.7		Non Dimensional Slenderness Ratio In Column Formula For Beam Column	$\lambda 1$	*
314	$NMGA5		58	$0 \leq \lambda 1 \leq 1.0$		*
315	$NMGA6		58	$1.0 < \lambda 1 \leq 2.0$		*
316	$NMGA7		58	$2.0 < \lambda 1 \leq 3.6$		*
317	$NMGA8		58	$3.6 < \lambda 1$		*
320	$LOAD4		60	Member Not Subjected To Transverse Loads In The Major Axis Plane Between Supports		*
321	$LOAD5		60	Member Subjected To Distributed Load Or Series Of Point Loads In The Major Axis Plane Between Supports		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
322	$LOAD6		60	Member Subjected To A Concentrated Load Or Moment In The Major Axis Plane Between Supports		*
323	$CURV3		62	Member Bent In Single Curvature About The Minor Axis		*
324	$CURV4		62	Member Bent In Double Curvature About The Minor Axis		*
325	VF	7.2.2		Shear Force In A Member Or Component Under Factored Load	V_f	*
326	VR	13.4.A.1		Factored Shear Resistance In A Member Or Component	V_r	*
327	FS	13.4.1		Ultimate Shear Strength	F_s	*
328	SKV	13.4.1.A.1		Shear Buckling Coefficient	K_v	*
329	T	13.4.1.A.1		Non-Dimensional Coefficient In F_s Formula	T	*
330	SNAGA	13.4.1.A.1		Non-Dimensional Coefficient In F_s Formula	η	*
331	SA			Distance Between Stiffeners	a	*
332	$AH1		64	a/h < 1.0		
333	$AH2		64	a/h $\geq$ 1.0		*
334	$STW			Stiffened Webs		*
335	$R5			$\frac{M_{fx}}{M_{rx1}} + \frac{M_{fy}}{M_{ry1}} \leq 1.0$		
336	$R6			$\frac{P_f}{P_{rc2}} + \frac{0.85\ M_{fx}}{M_{rx1}} + \frac{0.6\ M_{fy}}{M_{ry1}} \leq 1.0$		
337	$R7			$\frac{P_f}{P_{rc1}} + \frac{\omega_x M_{fx}}{M_{rx2}(1-\frac{P_f}{C_{ey}})} + \frac{\omega_y M_{fy}}{M_{ry1}(1-\frac{P_f}{C_{ey}})} \leq 1.0$		
338	$R8			$\frac{P_f}{P_{rc2}} + \frac{M_{fx}}{M_{rx1}} + \frac{M_{fy}}{M_{ry1}} \leq 1.0$		
339	$R9			$\frac{P_f}{P_{rc1}} + \frac{\omega_x M_{fx}}{M_{rx2}(1-\frac{P_f}{C_{ex}})} + \frac{\omega_y M_{fy}}{M_{ry1}(1-\frac{P_f}{C_{ey}})} \leq 1.0$		

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
340	$R10			$\frac{P_f}{P_{rt}} + \frac{M_{fx}}{M_{r1}} + \frac{M_{fy}}{M_{r1}} \leq 1.0$		
341	$R11			$\frac{M_{fx}}{M_{rx2}} + \frac{M_{fy}}{M_{ry1}} - \frac{P_f}{P_{rt}} \leq 1.0$		
342	$FLSH		66	Flexural Member Subjected To Shear		*
343	$GUSP		66	Gusset Plates		*
344	$PINS		66	Pins		*
350	$PLA1			Web Stiffener Are Supplied On The Member At A Point Of Load Application Where A Plastic Hinge Would Form		
351	$PLA2			Splices In The Member Are Designed To Transmit 1.1 Times The Max. Computed Moment Under Factor Loads At The Splice Location Or 0.25 M_p Whichever is Greater		
352	$PLA3			Member Is Not Subjected To Repeated Impact Or Fatigue		
353	$PLA4			The Influence Of Inelastic Deformation On The Strength Of The Structure Shall Be Taken Into Account		
354	$F08			$F_y \leq 0.8\ F_u$		
355	ELP			Laterally Unbraced Length Between Plastic Hinges	L_{ph}	*
356	ELCR	13.7		Max. Unbraced Length Adjacent To A Plastic Hinge	L_{cr}	*
357	$L1			ELP $\leq$ ELCR		
358	$MF1		68	$M_{f1}/M_{f2} > 0.5$		*
359	$MF2		68	$M_{f1}/M_{f2} \leq 0.5$		*
360	$COV			Cover Plate Used		
361	$BGIR		70	Bolted Girder		
362	$WGIR		70	Welded Girder		
363	ACOV			Area Of Cover Plate	A_{cov}	*
364	FPCOV			Force In Cover Plate	P_{cov}	*
365	FMFC			Moment Due To Factored Load At Point Of Theoretical Cut Off	M_{fc}	*
366	Y			Distance From Centroid Of Cover Plate To Neutral Axis Of Cover Plated Section	y	
367	FICOV			Moment of Inertia Of Cover Plated Section	I_{cov}	*
368	$A			$A_{cov} \leq 70\%\ A_f$		

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
369	$PCOV			$P_{cov} \geq (A_{cov} \cdot M_{fc} \cdot Y)/I_{cov}$		
370	$IBST		72	Interior Bearing Stiffener		*
371	$EBST	15.5	72	End Bearing Stiffener		
372	AEWEB			Equivalent Web Area To B · Included In Area Of Stiffener	A_{eweb}	*
373	ABST			Area Of Bearing Stiffener	A_{bst}	*
374	FIYWB			Moment Of Inertia Of Intermediate Stiffener Or Pair Of Stiffeners About An Axis In The Plane Of The Web	FI_{ywb}	*
375	AIST			Area Of Intermediate Stiffener	A_{ist}	*
376	C	15.6.A.1		$C=1-(45000K_v)/F_y(h/w)^2$	C	*
377	YRAT	15.6.A.1		Ratio Of Specified Min. Yield Strength Of Web Steel To Specified Min. Yield Strength Of Flange Steel	YRAT	*
378	SFACT	15.6.A.2		Stiffener Factor	SFACT	*
379	REFAC	15.6.A.3		Reduction Factor V_f/V_r		*
380	VC			Shear Capacity Of Weld Connection In (k/in) Between The Web And The Stiffener	V_c	*
381	TLOAD			Total Load Required To Be Transmitted To The Web Through The Stiffener	TLOAD	*
382	$IST1			$I_{ywb} \geq ((h/50)^4$		
383	$IST2			$A_{ist} \geq aw/2 \left[1- \frac{a/h}{\sqrt{1+(a/h)^2}}\right] C \times YRAT \times SFACT \times REFAC$		
384	$IST3			$V_c \geq \max\ (0.0026\ hF_y^{3/2} \times REFAC,\ TLOAD/h)$		
385	FYSTIF			F_y Of Stiffener Steel	F_{ystif}	*
386	$IST4		74	Intermediate Stiffener Furnished in Pairs		*
387	$IST5		74	Intermediate Single Angle Stiffener		*
388	$IST6		74	Intermediate Single Plate Stiffener		*
389	VFADJ			Largest Factored Shear In An Adjacent Panel	V_{fadj}	*
390	$IST7			$V_{fadj}/V_r < 1.0$		*
391	BR1	13.9.A.1		Bearing Resistance Of Stiffener	B_{r1}	*
392	SLOAD			Sum Of Loads Not Supported By A Bearing Stiffener On The Compressive Edge Of Web Plate	SLOAD	*
393	$BL1			$B_{r1} \geq SLOAD$		
394	$FLRO		76	Flange Not Restrainted Against Rotation		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
395	$FLNRO		76	Flange Restrained Against Rotation		*
396	ATW			A In Clause 15.9	ATW	*
397	$BST1			$K_x L_x > 3/4 L_x$		
398	$BST2			$K_y L_y > 3/4\ L_y$		
400	GAMA	7.2.5		Importance Factor	γ	*
401	ALFAD	7.2.3		Load Factor For Dead Load	α_D	*
402	ALFAL	7.2.3		Load Factor For Live Load	α_L	*
403	ALFAQ	7.2.3		Load Factor For Earthquake Load	α_Q	*
404	ALFAT	7.2.3		Load Factor For Temperature Effect	α_T	*
405	SIYE	7.2.4		Load Combination Factor	ψ	*
406	FPD			Axial Load Due To Dead Load	P_D	*
407	FPL			Axial Load Due To Live Load	P_L	*
408	FPQ			Axial Load Due To Earthquake	P_Q	*
409	FPT			Axial Load Due To Temperature Effect	P_T	*
410	FMXD			Moment About Major Axis Due To Dead Load	M_{XD}	*
411	FMXL			Moment About Major Axis Due To Live Load	M_{XL}	*
412	FMXQ			Moment About Major Axis Due To Earthquake Load	M_{XQ}	*
413	FMXT			Moment About Major Axis Due To Temperature Effect	M_{XT}	*
414	FMYD			Moment About Minor Axis Due To Dead Load	M_{yD}	*
415	FMYL			Moment About Minor Axis Due To Live Load	M_{yL}	*
416	FMYQ			Moment About Minor Axis Due To Earthquake Load	M_{yQ}	*
417	FMYT			Moment About Minor Axis Due To Temperature Effect	M_{yT}	*
418	VD			Shear Force Due To Dead Load	V_D	*
419	VL			Shear Force Due To Live Load	V_L	*
420	VQ			Shear Force Due To Earthquake Load	V_Q	*
421	VT			Shear Force Due To Temperature Effect	V_T	*
422	$FACT1			Overturning, Uplift, Or Stress Reversal Case		*
423	$FACT2		78	Only One Of L, T, Q Act		*
424	$FACT3		78	Two Of L, T, Q Act		*
425	$FACT4		78	All Of L, T, Q Act		*
426	$FACT5			Farm Building Of Low Occupancy Rate/ /Building Collapse Not Likely To Cause Injury		*

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
427	B)			Bearing Load Due To Dead Load	B_D	*
428	BL			Bearing Load Due To Live Load	B_L	*
429	BQ			Bearing Load Due To Earthquake Load	B_Q	*
430	BT			Bearing Load Due To Temperature Effect	B_T	*
431	BF	7.2.2		Factored Bearing Load	B_f	*
432	BR	13.10.A.1		Factored Bearing Resistance Of A Member Or Component	B_T	*
433	ACONT			Area Of Contact In Bearing	$A_{contact}$	*
434	RO			Diameter Of Roller Or Rocker	D	*
435	RL			Length Of Roller Or Rocker	L	*
436	$BRA		80	On The Contact Area Of Machined, Acurately Sawn Or Thread Parts		*
437	$BRB		80	On Expansion Rollers Or Rockers		*
438	RB	13.10.A.1		B_f/B_r	R_B	
439	$RB			RB $\leq$ 1.0		
440	SY			Section Modulus About the Weak Axis	s_y	
441	ZY			Plastic Modulus About the Weak Axis	z_y	

DATA NUMBER	DATA NAME	TABLE ADDRESS	SET NO.	DATA DESCRIPTION	DATA SYMBOL	DATA DEPD.
500	CHECKN	13.8.R		The Number Of Times A Table Is To Be Recycled		
501	CHECKI	X.6.R		Counter For Recycling Of Table		
502	FMRXA	X.7.R		Factored Moment Of Resistance About The Major Axis	M_{rxA}	
503	FMRYA	x.7.R		Factored Moment Of Resistance About The Minor Axis	M_{ryA}	
504	FPRCA	X.7.R		Factored Compressive Resistance	P_{rcA}	
505	OMGXA	X.7.R			ω_{xA}	
506	OMGYA	X.7.R			ω_{yA}	
507	THETAX	X.7.R		$1-P_f/C_{ex}$	θ_x	
508	THETAY	X.7.R		$1-P_f/C_{ey}$	θ_y	
509	$RATIO			$\frac{P_f}{P_{rcA}} + \frac{M_{fx}\omega_{xA}}{M_{rxA}\theta_x} + \frac{M_{fy}\omega_{yA}}{M_{ryA}\theta_y} \leq 1.0$		
510	$CK1		82	CHECKI=1		
511	$CK2		82	CHECKI=2		
512	$CK3		82	CHECKI=3		
513	$CN1		84	CHECKN=2		
514	$CN2		84	CHECKN=3		
515	$RATI1			$\frac{M_{fx}}{M_{rxA}} + \frac{M_{fy}}{M_{ryA}} \leq 1.0$		
516	$RATI2			$\frac{P_f}{P_{rcA}} + \frac{M_{fx}\omega_{xA}}{M_{rxA}\theta_x} + \frac{M_{fy}\omega_{yA}}{M_{ryA}\theta_y} \leq 1.0$		

NON COMPACTED LIST OF DEPENDENTS

DATA NO.	NO. OF DEPENDENTS	LIST OF DEPENDENTS
1	(1)	326
2	(1)	326
19	(1)	45
20	(2)	131 150
21	(1)	123
22	(1)	123
23	(5)	219 217 215 214 185
24	(5)	219 217 215 214 185
25	(5)	219 217 215 214 185
26	(2)	194 310
27	(2)	194 310
28	(1)	194
29	(1)	194
30	(1)	194
40	(1)	194
41	(1)	194
45	(3)	283 131 326
46	(9)	45 283 131 145 297 298 311 312 326
47	(3)	134 295 326
48	(4)	134 195 295 368
58	(23)	236 237 238 239 241 242 243 244 245 246 247 248 249 254 255 256 257 250 251 252
		253 258 75
59	(19)	241 242 243 244 245 246 247 248 249 254 255 256 257 250 251 252 253 258 75
60	(4)	236 237 238 239
61	(1)	45
62	(1)	45
63	(34)	275 260 261 262 263 264 265 266 267 268 276 269 270 271 274 272 134 273 295 284
		265 286 287 327 332 333 328 329 330 382 383 384 376 391
64	(29)	275 260 261 262 263 264 265 266 267 268 276 269 270 271 274 272 134 273 295 326
		264 285 286 287 327 383 376 391
65	(2)	195 326
66	(1)	45
67	(1)	372
75	(8)	274 272 134 135 273 295 296 299
76	(67)	236 237 238 239 241 242 243 244 245 246 247 248 249 275 260 261 262 263 264 265
		266 267 268 276 269 270 271 254 255 256 257 250 251 252 253 280 281 282 283 131
		230 145 134 274 258 75 135 295 296 272 273 299 297 313 298 326 284 285 286 287
		327 354 356 384 376 377 432
77	(6)	280 281 282 283 131 354
78	(3)	230 137 313
79	(2)	134 135
80	(6)	134 135 137 295 296 299
81	(1)	137
82	(1)	137

NOW COMPACTED LIST OF DEPENDENTS

DATA NO.	NO. OF DEPENDENTS	LIST OF DEPENDENTS
83	(1)	137
95	(9)	230 134 151 152 135 295 296 300 299
96	(9)	230 134 151 152 135 295 296 300 299
97	(9)	230 134 151 152 135 295 296 300 299
98	(9)	230 134 151 152 135 295 296 300 299
99	(4)	97 98 134 151
100	(4)	97 98 134 151
101	(4)	97 98 134 151
102	(4)	97 98 134 151
110	(6)	97 98 134 151 295 296
111	(6)	97 98 134 151 295 296
112	(6)	97 98 134 151 295 296
113	(6)	97 98 134 151 295 296
119	(9)	150 151 153 152 133 132 325 120 431
120	(15)	136 266 267 268 276 269 270 271 150 336 337 338 339 340 341
121	(9)	150 151 153 152 133 132 325 120 431
122	(9)	150 151 153 152 133 132 325 120 431
123	(9)	150 151 153 152 133 132 325 120 431
124	(9)	150 151 153 152 133 132 325 120 431
125	(7)	95 96 97 211 212 213 150
126	(7)	95 96 97 211 212 213 150
127	(1)	135
128	(1)	135
129	(1)	135
130	(3)	95 96 97
131	(3)	150 340 341
132	(8)	151 335 336 337 338 339 340 341
133	(7)	152 335 336 338 339 340 341
134	(2)	151 135
135	(1)	152
137	(4)	139 134 146 299
138	(6)	139 134 295 296 299 300
139	(3)	134 151 299
140	(3)	194 358 359
141	(3)	194 358 359
142	(4)	146 134 299 300
143	(1)	310
144	(1)	310
145	(1)	150
146	(3)	134 151 299
150	(1)	154
151	(1)	154
152	(1)	154
165	(9)	110 111 112 113 97 98 137 295 296
166	(9)	110 111 112 113 97 98 137 295 296
167	(9)	110 111 112 113 97 98 137 295 296
168	(9)	110 111 112 113 97 98 137 295 296
169	(9)	110 111 112 113 97 98 137 295 296
170	(9)	110 111 112 113 97 98 137 295 296
171	(9)	110 111 112 113 97 98 137 295 296

NON COMPACTED LIST OF DEPENDENTS

DATA NO.	NO. OF DEPENDENTS	LIST OF DEPENDENTS
172	(9)	110 111 112 113 97 98 137 295 296
173	(9)	110 111 112 113 97 98 137 295 296
174	(9)	110 111 112 113 97 98 137 295 296
175	(9)	110 111 112 113 97 98 137 295 296
177	(2)	95 96
178	(3)	95 96 97
179	(1)	97
180	(3)	135 295 296
185	(4)	219 230 311 397
186	(5)	219 211 230 311 397
187	(3)	137 195 196
188	(5)	217 212 230 312 356
189	(1)	196
190	(8)	136 266 267 268 276 269 270 271
191	(2)	258 75
192	(1)	137
194	(3)	137 337 339
195	(1)	137
196	(1)	137
197	(13)	131 145 134 135 295 296 299 297 298 300 326 391 432
198	(1)	218
199	(1)	218
200	(1)	218
201	(1)	218
203	(1)	218
204	(1)	218
205	(1)	218
208	(3)	219 211 311
211	(1)	213
212	(1)	213
214	(2)	217 312
215	(1)	313
216	(3)	217 212 312
217	(1)	215
218	(1)	45
219	(1)	215
220	(6)	219 217 215 214 185 194
221	(6)	219 217 215 214 185 310
222	(5)	219 217 215 214 185
223	(5)	219 217 215 214 185
230	(5)	231 232 233 234 145
231	(2)	145 150
232	(2)	145 150
233	(2)	145 150
234	(2)	145 150
236	(4)	95 96 97 98
237	(4)	95 96 97 98
238	(4)	95 96 97 98
239	(4)	95 96 97 98
241	(2)	97 98

NON COMPACTED LIST OF DEPENDENTS

DATA NO.	NO. OF DEPENDENTS	LIST OF DEPENDENTS
242	(2)	97 98
243	(4)	97 98 112 113
244	(4)	97 98 112 113
245	(6)	95 97 98 110 112 113
246	(6)	95 97 98 110 112 113
247	(6)	95 97 98 110 112 113
248	(2)	97 98
249	(2)	97 98
250	(4)	110 111 112 113
251	(4)	110 111 112 113
252	(4)	110 111 112 113
253	(4)	110 111 112 113
254	(6)	110 111 112 113 97 98
255	(6)	110 111 112 113 97 98
256	(6)	110 111 112 113 97 98
257	(6)	110 111 112 113 97 98
258	(1)	75
260	(2)	99 102
261	(2)	99 102
262	(4)	99 100 101 102
263	(4)	99 100 101 102
264	(4)	99 100 101 102
265	(4)	99 100 101 102
266	(3)	99 100 102
267	(3)	99 100 102
268	(3)	99 100 102
269	(4)	99 100 101 102
270	(4)	99 100 101 102
271	(4)	99 100 101 102
272	(4)	134 151 295 296
273	(4)	134 151 295 296
274	(4)	134 151 295 296
276	(4)	99 100 101 102
280	(1)	45
281	(1)	45
282	(1)	45
283	(2)	131 150
284	(1)	327
285	(1)	327
286	(1)	327
287	(1)	327
295	(3)	335 336 338
296	(6)	335 336 337 338 339 341
297	(2)	337 339
298	(2)	336 338
299	(5)	295 296 337 339 341
300	(1)	340
310	(2)	337 339
311	(2)	337 339
312	(2)	337 339

NON COMPACTED LIST OF DEPENDENTS

DATA NO.	NO. OF DEPENDENTS	LIST OF DEPENDENTS
313	(5)	314 315 316 317 297
314	(1)	297
315	(1)	297
316	(1)	297
317	(1)	297
320	(1)	310
321	(1)	310
322	(1)	310
323	(1)	310
324	(1)	310
325	(1)	155
326	(3)	155 390 379
327	(1)	326
328	(6)	284 265 286 287 327 376
329	(1)	327
330	(1)	327
331	(7)	332 333 328 329 330 383 391
332	(3)	328 329 330
333	(3)	328 329 330
334	(3)	328 329 330
342	(1)	326
343	(1)	326
344	(1)	326
355	(1)	357
356	(1)	357
358	(1)	356
359	(1)	356
363	(2)	368 369
364	(1)	369
365	(1)	369
366	(1)	369
367	(1)	369
370	(2)	372 46
371	(2)	372 46
372	(1)	46
373	(1)	46
374	(1)	382
375	(1)	383
376	(1)	383
377	(1)	383
378	(1)	383
379	(2)	383 384
380	(1)	384
381	(1)	384
385	(1)	377
386	(1)	378
387	(1)	378
388	(1)	378
389	(2)	390 379
390	(1)	379

NON COMPACTED LIST OF DEPENDENTS

DATA NO.	NO. OF DEPENDENTS	LIST OF DEPENDENTS
391	(1)	393
392	(1)	393
394	(1)	391
395	(1)	391
396	(1)	391
400	(5)	120 132 133 325 431
401	(5)	120 132 133 325 431
402	(5)	120 132 133 325 431
403	(5)	120 132 133 325 431
404	(5)	120 132 133 325 431
405	(5)	120 132 133 325 431
406	(1)	120
407	(1)	120
408	(1)	120
409	(1)	120
410	(1)	132
411	(1)	132
412	(1)	132
413	(1)	132
414	(1)	133
415	(1)	133
416	(1)	133
417	(1)	133
418	(1)	325
419	(1)	325
420	(1)	325
421	(1)	325
422	(4)	401 402 403 404
423	(1)	405
424	(1)	405
425	(1)	405
426	(1)	400
427	(1)	431
428	(1)	431
429	(1)	431
430	(1)	431
431	(1)	438
432	(1)	438
433	(1)	432
434	(1)	432
435	(1)	432
436	(2)	432 438
437	(2)	432 438

DECISION TABLE X.1 (1)

Data Requirement

Elastic Analysis	x	
Plastic Analysis	x	

Elastic Analysis	Y	N
Plastic Analysis	N	Y
Execute Table X.2	Y	
Execute Table 8.5		Y

DECISION TABLE X.2 (2)

Data Requirement

Structural Steel Member	x	
Open Web Steel Joist	x	
Composite Construction Member	x	
Built Up Member	x	
Connection	x	
Girder	x	
Bearing Stiffener	x	
Intermediate Transverse Stiffener	x	

Structural Steel Member	Y	N	N	N	N	N	N	N
Open-Web Steel Joists	N	Y	N	N	N	N	N	N
Composite Constr·ction Member	N	N	Y	N	N	N	N	N
Built-Up Members	N	N	N	Y	N	N	N	N
Connections	N	N	N	N	Y	N	N	N
Girder	N	N	N	N	N	Y	N	N
Bearing Stiffener	N	N	N	N	N	N	Y	N
Intermediate Transverse Stiffener	N	N	N	N	N	N	N	Y
Execute Table X.3	Y							
Execute Table ___		Y						
Execute Table ___			Y					
Execute Table ___				Y				
Execute Table ___					Y			
Execute Table 15.						Y		
Execute Table 15.5							Y	
Execute Table 15.6								Y

DECISION TABLE X.3 (3)

Data Requirement

$P_f = 0$	x	
$M_{fx} = 0$	x	
$M_{fy} = 0$	x	
$V_f = 0$	x	
$B_f = 0$	x	

DECISION TABLE X.3 (3)

$P_f = 0$	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y
$M_{fx} = 0$	Y	N	N	Y	N	N	Y	N	Y	Y	N	Y	N	N	Y	Y	Y	N	N	Y	N	N	Y	N	Y	Y	N	Y	N	N	Y	Y
$M_{fy} = 0$	N	Y	N	N	Y	N	Y	Y	N	Y	Y	N	N	N	Y	Y	N	Y	N	N	Y	N	Y	Y	N	Y	Y	N	N	N	Y	Y
$V_f = 0$	Y	Y	Y	N	N	N	Y	Y	Y	N	N	N	Y	N	N	Y	Y	Y	Y	N	N	N	Y	Y	Y	N	N	N	Y	N	N	Y
$B_f = 0$	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
$R_1 = 0$	Y	Y	Y	Y	Y	Y									Y	Y	Y	Y	Y	Y	Y	Y									Y	Y
$R_2 = 0$	Y			Y			Y		Y	Y		Y			Y	Y	Y			Y			Y		Y	Y		Y			Y	Y
$R_3 = 0$		Y			Y		Y	Y		Y	Y				Y	Y		Y			Y		Y	Y		Y	Y				Y	Y
$R_4 = 0$	Y	Y	Y				Y	Y	Y				Y			Y	Y	Y	Y				Y	Y	Y				Y			Y
Execute Table X.4	Y	Y	Y	Y	Y	Y	Y			Y							Y	Y	Y	Y	Y	Y	Y			Y						
Execute Table X.6								Y	Y		Y	Y	Y	Y										Y	Y		Y	Y	Y	Y		
Execute Table 13.4				Y	Y	Y				Y	Y	Y		Y	Y					Y	Y	Y				Y	Y	Y		Y	Y	
Execute Table 13.10																	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Msh: No Stress Resultants																																
No Checks Made																Y																

DECISION TABLE 9.3 (4)

Data Requirement

Compression Member	x	
Compression Member in Truss	x	
Sway Effects About The Major Axis Included in Analysis/ / Sway Effects About the Major Axis Resisted by Bracing or Shear Wall	x	
Sway Effects About the Minor Axis Included in Analysis/ / Sway Effects About the Minor Axis Resisted by Bracing or Shear Wall	x	
Rotational restraints at the ends of the unbraced length show k can be < 1.0 About the Major Axis	x	
Rotational restraints at the ends of the unbraced length show k can be < 1.0 about the Minor Axis	x	
K_x		Table 9.3 (4)
K_y		Table 9.3 (4)
L_x	x	
r_x	x	
r_y	x	
SLRAX		Table 9.3 (4)
SLRAY		Table 9.3 (4)

DECISION TABLE 9.3 (4)

	1	2	3	4	5	6	7	8	9	10	
Compression Member	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	
Compression Member in Truss	N	N	N	N	N	N	N	N	N	Y	
Sway Effects About the Major Axis Included in Analysis/ Sway Effects About the Major Axis Resisted by Bracing or Shear Wall	Y	Y	Y	Y	N	N	Y	Y	N	I	
Sway Effects About the Minor Axis Included in Analysis/ Sway Effects About the Minor Axis Resisted by Bracing or Shear Wall	Y	Y	Y	Y	Y	Y	N	N	N	I	
Rotational Restraints at the Ends of the Unbraced Length Shows K can be < 1.0 Along the Major Axis	Y	Y	N	N	I	I	Y	N	I	I	
Rotational Restraints at the Ends of the Unbraced Length Shows K can be < Along the Minor Axis	Y	N	Y	N	Y	N	I	I	I	I	
K_x = 1.0			Y	Y				Y		Y	
K_y = 1.0		Y		Y		Y				Y	
MSG: K_x = MIN (1.0 or Calculated from Appendix C, Sidesway Prevented Case)	Y	Y					Y				
MSG: K_y = MIN (1.0 or Calculated from Appendix C, Sidesway Prevented Case)	Y		Y		Y						
MSG: K_x = MAX (Calculated from Appendix C, Sidesway Permitted Case or 1.0)					Y	Y			Y		
MAG: K_y = MAX (Calculated from Appendix C, Sidesway Permitted Case or 1.0)							Y	Y	Y		
SLRAX = $K_x L_x / r_x$	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	$K_x L_x r_x$
SLRAY = $K_y L_y / r_y$	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	$K_y L_y r_y$
UKLR = MAX (SLRAX, SLRAY)†	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	SLRAX, SLRAY

† In Subroutine, check if KLR > 200. If yes, print out message, then stop execution of program.

TABLE 11. (5)

Data Requirement

Web Class = 1		Table 11.A (6)
Web Class = 2		Table 11.A (6)
Web Class = 3		Table 11.A (6)
Web Class = 4		Table 11.A (6)
Flange Class = 1		Table 11.B (20)
Flange Class = 2		Table 11.B (20)
Flange Class = 3		Table 11.B (20)
Flange Class = 4		Table 11.B (20)
Circular Hollow Section	x	
Single Angle	x	
Double Angle	x	
Flange Cover Plate/Diaphram Plate	x	
Perforated Cover Plate	x	
Plate Girder Intermediate	x	
Transverse Stiffener/Bearing Stiffener		

DECISION TABLE 11. (5)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Double Angle	Y	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Circular Hollow Section	I	Y	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Single Angle	I	I	Y	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Flange Cover Plate/ Diaphram Plate	I	I	I	Y	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Perforated Cover Plate	I	I	I	I	Y	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Plate Girder Int. Transverse Stiffener/ Bearing Stiffener	I	I	I	I	I	Y	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Web Class = 1	I	I	I	I	I	I	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N
Web Class = 2	I	I	I	I	I	I	N	N	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N
Web Class = 3	I	I	I	I	I	I	N	N	N	N	N	N	N	N	Y	Y	Y	Y	N	N	N	N
Web Class = 4	I	I	I	I	I	I	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y
Flange Class = 1	I	I	I	I	I	I	Y	N	N	N	Y	N	N	N	Y	N	N	N	Y	N	N	N
Flange Class = 2	I	I	I	I	I	I	N	Y	N	N	N	Y	N	N	N	Y	N	N	N	Y	N	N
Flange Class = 3	I	I	I	I	I	I	N	N	Y	N	N	N	Y	N	N	N	Y	N	N	N	Y	N
Flange Class = 4	I	I	I	I	I	I	N	N	N	Y	N	N	N	Y	N	N	N	Y	N	N	N	Y
Execute Table 11.B.1	Y																					
Execute Table 11.A.3		Y																				
Execute Table 11.A.4			Y																			
Execute Table 11.A.6				Y																		
Execute Table 11.A.7					Y																	
Execute Table 11.A.5						Y																
Execute Table 11.A.1							Y															
Execute Table 11.A.2								Y			Y	Y										
Class = 3									Y				Y		Y	Y	Y					
Class = 4										Y				Y				Y	Y	Y	Y	Y

DECISION TABLE 11.A (6)

Data Requirement

T Section	x
I Section	x
] Section	x
Rectangular Hollow Section	x
Box Section	x

T Section	Y	N	N	N	N
I Section	N	Y	N	N	N
] Section	N	N	Y	N	N
Rectangular Hollow Section	N	N	N	Y	N
Box Section	N	N	N	N	Y
Execute Table 11.A.8	Y	Y	Y	Y	Y

DECISION TABLE 11.A.1 (7)

Data Requirement

$P_f > 0$	x
M_{fx}.and.or.$M_{fy} > 0$	x
Section Doubly Symmetric	x
Section's Axis of Symmetry in Plane of Loading	x

$P_f > 0$	N	Y	Y	N	N	N	Y	Y	Y
M_{fx}.and.or.$M_{fy} > 0$	N	N	N	Y	Y	Y	Y	Y	Y
Section Doubly Summetric	I	Y	N	Y	N	N	Y	N	N
Section's Axis of Sym. in Plane of Loading	I	I	I	I	Y	N	I	Y	N
Class = 1	Y	Y		Y	Y		Y		
Class = 2			Y			Y		Y	Y

DECISION TABLE 11.A.2 (8)

Data Requirement

$P_f > 0$	x	
M_{fx}.and.or.$M_{fy} > 0$	x	
Section's Axis of Symmetry in Plane of Loading	x	

$P_f > 0$	N	Y	N	N	Y	Y
M_{fx}.and.or.$M_{fy} > 0$	N	N	Y	Y	Y	Y
Section's Axis of Sym. in Plane of Loading	I	I	Y	N	Y	N
Class = 2	Y	Y	Y		Y	
Class = 3				Y		Y

DECISION TABLE 11.A.3 (9)

Data Requirement

Dl	x	
t	x	
Fy	x	

$Dl/t \leq 260/\sqrt{F_y}$	Y	N	N	N	Dl t F_y
$260/\sqrt{F_y} < Dl/t \leq 365/\sqrt{F_y}$	N	Y	N	N	Dl t F_y
$365/\sqrt{F_y} < Dl/t \leq 3300/F_y$	N	N	Y	N	Dl t F_y
$Dl/t > 3300/F_y$	N	N	N	Y	Dl t F_y
Class = 1	Y				
Class = 2		Y			
Class = 3			Y		
Class = 4				Y	

DECISION TABLE 11.A.4 (10)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \leq 75/\sqrt{F_y}$	Y	N	b t F_y
$b/t > 75/\sqrt{F_y}$	N	Y	b t F_y
Class = 3	Y		
Class = 4		Y	

DECISION TABLE 11.A.5 (11)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \leq 100/\sqrt{F_y}$	Y	N	b t F_y
$b/t > 100/\sqrt{F_y}$	N	Y	b t F_y
Class = 3	Y		
Class = 4		Y	

DECISION TABLE 11.A.6 (12)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \le 200/\sqrt{F_y}$	Y N N	b t F_y
$200/\sqrt{F_y} < b/t \le 255/\sqrt{F_y}$	N Y N	b t F_y
$b/t > 255/\sqrt{F_y}$	N N Y	b t F_y
Class = 1	Y	
Class = 3	Y	
Class = 4	Y	

DECISION TABLE 11.A.7 (13)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \le 320/\sqrt{F_y}$	Y N	b t F_y
$b/t > 320/\sqrt{F_y}$	N Y	b t F_y
Class = 3	Y	
Class = 4	Y	

DECISION TABLE 11.A.8 (14)

Data Requirement

Web in Axial Compression	x	
Web in Flexural Compression	x	
Web in Combine Axial and Flexural Compression	x	
h	x	
w	x	
F_y	x	

Web in Axial Compression	Y	N	N	I	
Web in Flexural Compression	N	Y	N	I	
Web in Combine Axial and Flexural Compression	N	N	Y	I	
$h/w > 12000/F_y$	N	N	N	Y	h w F_y
Execute Table 11.A.9	Y				
Execute Table 11.A.10		Y			
Execute Table 11.A.11			Y		
MSG: Max h/w of the web exceeded				Y	

DECISION TABLE 11.A.9 (15)

Date Requirement

h	x	
w	x	
F_y	x	

$h/w \leq 255/\sqrt{F_y}$	Y	N	h w F_y
$h/w > 255/\sqrt{F_y}$	N	Y	h w F_y
Web Class = 1	Y		
Web Class = 4		Y	

DECISION TABLE 11.A.10 (16)

Data Requirement

h	x	
w	x	
F_y	x	

$h/w \le 420/\sqrt{F_y}$	Y	N	N	N	h w F_y
$420/\sqrt{F_y} < h/w \le 520/\sqrt{F_y}$	N	Y	N	N	h w F_y
$520/\sqrt{F_y} < h/w \le 690/\sqrt{F_y}$	N	N	Y	N	h w F_y
$h/w > 690/\sqrt{F_y}$	N	N	N	Y	h w F_y
Web Class = 1	Y				
Web Class = 2		Y			
Web Class = 3			Y		
Web Class = 4				Y	

DECISION TABLE 11.A.11 (17)

Data Requirement

P_f C_y	x	Table 7.2.2 (93)

$\frac{P_f}{C_y} > 0.15$	Y	N	P_f C_y
Execute Table 11.A.12	Y		
Execute Table 11.A.13		Y	

DECISION TABLE 11.A.12 (18)

Data Requirement

h	x	
w	x	
F_y	x	Table 7.2.2 (93)
P_f		
C_y	x	

$h/w \le 420/\sqrt{F_y}\ (1-1.4\ \frac{P_f}{C_y})$	Y	N	N	$h\ w\ F_y\ P_f\ C_y$
$420/\sqrt{F_y}(1-1.4\ \frac{P_f}{C_y}) < \frac{h}{w} \le 450/\sqrt{F_y}\ (1-0.43\ \frac{P_f}{C_y})$	N	Y	N	$h\ w\ F_y\ P_f\ C_y$
$h/w > 450/\sqrt{F_y}\ (1-0.43\ \frac{P_f}{C_y})$	N	N	Y	$h\ w\ F_y\ P_f\ C_y$
Web Class = 1	Y			
Web Class = 2		Y		
Web Class = 4			Y	

DECISION TABLE 11.A.13 (19)

Data Requirement

h	x	
w	x	
F_y	x	Table 7.2.2 (93)
P_f		
C_y	x	

$h/w \le 420/\sqrt{F_y}\ (1-1.4\ \frac{P_f}{C_y})$	Y	N	N	N	$h\ w\ F_y\ P_f\ C_y$
$420/\sqrt{F_y}(1-1.4\ \frac{P_f}{C_y}) < \frac{h}{w} \le 520/\sqrt{F_y}(1-1.28\ \frac{P_f}{C_y})$	N	Y	N	N	$h\ w\ F_y\ P_f\ C_y$
$520/\sqrt{F_y}(1-1.28\ \frac{P_f}{C_y}) < \frac{h}{w} \le 690/\sqrt{F_y}(1-2.6\ \frac{P_f}{C_y})$	N	N	Y	N	$h\ w\ F_y\ P_f\ C_y$
$h/w > 690/\sqrt{F_y}(1-2.6\ \frac{P_f}{C_y})$	N	N	N	Y	$h\ w\ F_y\ P_f\ C_y$
Web Class = 1	Y				
Web Class = 2		Y			
Web Class = 3			Y		
Web Class = 4				Y	

DECISION TABLE 11.B (20)

Data Requirement

T-Section	x	
I-Section	x	
]-Section	x	
Rectangular Hollow Section	x	
Box Section	x	

T-Section	Y	N	N	N	N
I-Section	N	Y	N	N	N
]-Section	N	N	Y	N	N
Rectangular Hollow Section	N	N	N	Y	N
Box Section	N	N	N	N	Y
Execute Table 11.B.1	Y	Y			
Execute Table 11.B.2			Y		
Execute Table 11.B.3				Y	
Execute Table 11.B.4					Y

DECISION TABLE 11.B.1 (21)

Data Requirement

T-Section	x	
I-Section	x	
Double Angle	x	
b	x	
t	x	
F_y	x	

T-Section	Y	Y	Y	Y	N	N	N	N	N	N	N	N	
I-Section	N	N	N	N	Y	Y	Y	Y	N	N	N	N	
Double Angle	N	N	N	N	N	N	N	N	Y	Y	Y	Y	
$b/t \leq 54/\sqrt{F_y}$	Y	N	N	N	Y	N	N	N	Y	N	N	N	b t F_y
$54/\sqrt{F_y} < b/t \leq 64/\sqrt{F_y}$	N	Y	N	N	N	Y	N	N	N	Y	N	N	b t F_y
$64/\sqrt{F_y} < b/t \leq 100/\sqrt{F_y}$	N	N	Y	N	N	N	Y	N	N	N	Y	N	b t F_y
$b/t > 100/\sqrt{F_y}$	N	N	N	Y	N	N	N	Y	N	N	N	Y	b t F_y
Flange Class = 1	Y				Y								
Flange Class = 2		Y				Y							
Flange Class = 3			Y				Y						
Flange Class = 4				Y				Y					
Execute Table 1'.B.5									Y	Y			
Class = 3											Y		
Class = 4												Y	

DECISION TABLE 11.B.2 (22)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \le 100/\sqrt{F_y}$	Y	N	b t F_y
$b/t > 100/\sqrt{F_y}$	N	Y	b t F_y
Flange Class = 3	Y		
Flange Class = 4		Y	

DECISION TABLE 11.B.3 (23)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \le 160/\sqrt{F_y}$	Y	N	N	N	b t F_y
$160/\sqrt{F_y} < b/t \le 200/\sqrt{F_y}$	N	Y	N	N	b t F_y
$200/\sqrt{F_y} < b/t \le 255/\sqrt{F_y}$	N	N	Y	N	b t F_y
$b/t > 255/\sqrt{F_y}$	N	N	N	Y	b t F_y
Flange Class = 1	Y				
Flange Class = 2		Y			
Flange Class = 3			Y		
Flange Class = 4				Y	

DECISION TABLE 11.B.4 (24)

Data Requirement

b	x	
t	x	
F_y	x	

$b/t \le 200/\sqrt{F_y}$	Y	N	N	b t F_y
$200/\sqrt{F_y} < b/t \le 255/\sqrt{F_y}$	N	Y	N	b t F_y
$b/t > 255/\sqrt{F_y}$	N	N	Y	b t F_y
Flange Class = 1	Y			
Flange Class = 3		Y		
Flange Class = 4			Y	

DECISION TABLE 12.3 (25)

Data Requirement

Tension Membe· without Holes	x	Table 12.3.A.1 (26)
F_y	x	
F_u	x	
t_p	x	
REDUTN		
n (no. of holes)	x	
d (dia. of rivet or bolt)	x	
A_g	x	

Tension Member without Holes	N	N	N	N	N	N	N	N	N	Y	
$F_y/F_u \le 0.75$	Y	N	N	Y	N	N	Y	N	N	I	F_y F_u
$0.75 < F_y/F_u \le 0.85$	N	Y	N	N	Y	N	N	Y	N	I	F_y F_u
$0.85 < F_y/F_u$	N	N	Y	N	N	Y	N	N	Y	I	F_y F_u
A_n=Min(0.85 A_g, $A_g-t_p\{n(d+\frac{1}{8})$-†REDUTN})	Y			Y			Y				A_g t_p n d REDUTN
A_n=Min(0.90 A_g, $A_g-t_p\{n(d+\frac{1}{8})$- REDUTN})		Y			Y			Y			A_g t_p n d REDUTN
A_n=Min(0.95 A_g, $A_g-t_p\{n(d+\frac{1}{8})$- REDUTN})			Y			Y			Y		A_g t_p n d REDUTN
$A_n=A_g$										Y	A_g

† $\text{REDUTN} = \sum_k \frac{S_k^2}{4g_k}$

DECISION TABLE 12.3.A.1 (26)

Data Requirement

S_K (K = 1 to 4) g_K (K = 1 to 4)	x x	

Condition	I	
$REDUTN = \sum_K \frac{S_K^2}{4g_K}$	Y	S(1) S(2) S(3) S(4) g(1) g(2) g(3) g(4)

DECISION TABLE X.4 (27)

Data Requirement

R_1 R_2 R_3		Table 13.2 (28) Table X.5 (42) Table X.5 (42)

$R_1 + R_2 + R_3 \leq 1.0$	Y	N	R_1 R_2 R_3
MSG: Strength Criterion Satisfied	Y		
MSG: Strength Criterion Not Satisfied		Y	

DECISION TABLE 13.2 (28)

Data Requirement

$P_f > 0$	x	
$P_f < 0$	x	
L_x	x	
L_y	x	
r_x	x	
r_y	x	
TSLRAX		Table 13.2 (28)
TSLRAY		Table 13.2 (28)

$P_f > 0$ (Comp)	Y	N	N	
$P_f < 0$ (Tension)	N	Y	N	
Execute Table 13.3.1	Y			
TSLRAX = L_x/r_x †		Y		L_x r_x
TSLRAY = L_y/r_y †		Y		L_y r_y
TLR = MAX(TSLRAX, TSLRAY)		Y		TSLRAX TSLRAY
Execute Table 13.2.A.1		Y		
$R_1 = 0$			Y	

† In subroutine, check if SLRAX, SLRAY > 300.0.
If yes, print out message and stop execution of program.

DECISION TABLE 13.2.A.1 (29)

Data Requirement

A_n		Table 12.3
A_g	x	
F_y	x	
F_u	x	
ϕ	x	
P_f		Table 7.2.2
Member Using Pin Connections	x	
P_{rt}		Table 13.2.A.1

Member Using Pin-Connections	I	I	Y	
$A_n/A_g \geq F_y/F_u$	Y	N	I	A_n A_g F_y F_u
$P_{rt} = \text{Min}(\phi A_n F_y,\ 0.85\ \phi A_n F_u)$	Y			ϕ A_n F_y F_u
$P_{rt} = \text{Min}[\phi(F_u \frac{A_n}{A_g}) A_n,\ 0.85\ \phi A_n F_u]$		Y		ϕ A_n A_g F_u
$P_{rt} = 0.75\ \phi\ A_n\ F_y$			Y	ϕ A_n F_y
$F_1 = \frac{P_f}{P_{rt}}$	Y	Y	Y	P_f P_{rt}

DECISION TABLE 13.3.1 (30)

Data Requirement

Class = 1		Table 11.(5)
Class = 2		Table 11.(5)
Class = 3		Table 11.(5)
Class = 4		Table 11.(5)
UKLR		Table 9.3 (4)
F_y	x	
E	x	

Class = 1	Y	N	N	N	
Class = 2	N	Y	N	N	
Class = 3	N	N	Y	N	
Class = 4	N	N	N	Y	
λ = UKLR $\sqrt{F_y/\pi^2 E}$	Y	Y	Y	Y	UKLR F_y E
Execute Table 13.3.1.A.1	Y	Y	Y		
MSG: This section is a Class 4 section. Checking should use Clause 12 of CSA S136. "Cold Formed Steel Structural Members". Any other messages produced from now on relating to design should be ignored.				Y	

DECISION TABLE 13.3.1.A.1 (31)

Data Requirement

λ		Table 13.3.1 (30)
ϕ	x	
A_g	x	
F_y	x	
P_f		Table 7.2.2 (93)
P_{rc}		Table 13.3.1.A.1

$0 \leq \lambda \leq 1.0$	Y	N	N	N	λ
$1.0 < \lambda \leq 2.0$	N	Y	N	N	λ
$2.0 < \lambda \leq 3.6$	N	N	Y	N	λ
$3.6 < \lambda$	N	N	N	Y	λ
$P_{rc} = \phi A_g F_y (1.035 - 0.201\lambda - 0.224\lambda^2)$	Y				ϕ A_g F_y λ
$P_{rc} = \phi A_g F_y (-0.111 + 0.63\lambda^{-1} + 0.094\lambda^{-2})$		Y			ϕ A_g F_y λ
$P_{rc} = \phi A_g F_y (0.012 + 0.867\lambda^{-2})$			Y		ϕ A_g F_y λ
$P_{rc} = \phi A_g F_y \lambda^{-2}$				Y	ϕ A_g F_y λ
$R_1 = \frac{P_f}{P_{rc}}$	Y	Y	Y	Y	P_f P_{rc}

DECISION TABLE 11.B.5 (32)

Data Requirements

Is the Double Angle Continuously Connected by Adequate Mechanical Fasteners or Welds	x	

Is the Double Angle Continuously Connected by Adequate Mechanical Fasteners or Welds	Y	N
Execute Table 11.A.2	Y	
Class = 3		Y

DECISION TABLE X.5 (42)

Data Requirement

Continuous Laterally Supported Member	x	
Laterally Unsupported Member	x	

Continuous Laterally Supported Member	Y	N
Laterally Unsupported Member	N	Y
$M_{fy} = 0$	Y	
Execute Table 13.5.A	Y	
Execute Table 13.6.A		Y

DECISION TABLE 13.5.A (43)

Data Requirement

Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table T1. (5)
Class = 4		Table 11. (5)
ϕ	x	
Z	x	
F_y	x	
s	x	
M_{fx}		Table 7.2.2 (93)
M_{rx}		Tablè 13.6.A.1 (47)

Class = 1	Y	N	N	N	
Class = 2	N	Y	N	N	
Class = 3	N	N	Y	N	
Class = 4	N	N	N	Y	
$M_{rx} = \phi zF_y = (\phi M_p)$	Y	Y			ϕ z F_y
$M_{rx} = \phi sF_y = (\phi M_y)$			Y		ϕ s F_y
Execute Table 13.5.A.1				Y	
$R_2 = M_{fx}/M_{rx}$	Y	Y	Y		M_{fx} M_{rx}
$R_3 = 0$	Y	Y	Y	Y	

DECISION TABLE 13.5.A.1 (44)

Data Requirement

Flange Class = 1		Table 11.B (20)
Flange Class = 2		Table 11.B (20)
Flange Class = 3		Table 11.B (20)
Flange Class = 4		Table 11.B (20)
Web Class = 1		Table 11.A (6)
Web Class = 2		Table 11.A (6)
Web Class = 3		Table 11.A (6)
Web Class = 4		Table 11.A (6)
h	x	
w	x	
F_y	x	
F_{cr}		Table 13.5.3 (45)
ϕ	x	
s	x	
A_w	x	
A_f	x	
M_{fx}		Table 7.2.2 (93)
M_{rx}		Table 13.6.A.1 (47)

DECISION TABLE 13.5.A.1 (44)

Flange Class = 1	Y	N	N	Y	N	N	N	N	N	N	
Flange Class = 2	N	Y	N	N	Y	N	N	N	N	N	
Flange Class = 3	N	N	Y	N	N	Y	N	N	N	N	
Flange Class = 4	N	N	N	N	N	N	Y	Y	Y	Y	
Web Class = 1	N	N	N	N	N	N	Y	N	N	N	
Web Class = 2	N	N	N	N	N	N	N	Y	N	N	
Web Class = 3	N	N	N	N	N	N	N	N	Y	N	
Web Class = 4	Y	Y	Y	Y	Y	Y	N	N	N	Y	
$12000/F_y \geq \frac{h}{w} > 690/\sqrt{F_{cr}}$	Y	Y	Y	N	N	N	I	I	I	I	$h\ w\ F_y\ F_{cr}$
$\frac{h}{w} \leq 690/\sqrt{F_{cr}}$	N	N	N	Y	Y	Y	I	I	I	I	$h\ w\ F_{cr}$
$M_{rx} = \text{Min}\ (\phi s F_{cr}, \phi s F_y)$				Y	Y	Y	Y	Y	Y		$\phi\ s\ F_{cr}\ F_y$
$M_{rx} = \text{Min}(\phi s F_y, \phi s F_{cr})[1 - 0.0005\ \frac{A_w}{A_f}\ (\frac{h}{w} - 690/\sqrt{F_{cr}})]$	Y	Y	Y								$\phi\ s\ F_{cr}\ A_w\ A_f\ h\ w\ F_y$
MSG: M_{rx} to be determined from Clause 12, CSA S136										Y	
$R_2 = M_{fx}/M_{rx}$	Y	Y	Y	Y	Y	Y	Y	Y	Y		$M_{fx}\ M_{rx}$

DECISION TABLE 13.5.3 (45)

Data Requirement

b	x	
t	x	
F_y	x	
K_b	x	

$b/t \leq 201 \sqrt{K_b/F_y}$	Y	N	K_b F_y b t
$F_{cr} = F_y[1.46-0.004 \sqrt{F_y/K_b}\ (b/t)]$	Y		F_y K_b b t
$F_{cr} = 26200\ K_b/(b/t)^2$		Y	K_b b t

DECISION TABLE 13.6.A (46)

Data Requirements

M_{rx}		Table 13.6.A.1 (47)
$M_{fx} > 0$	x	
$M_{fy} > 0$	x	
M_{fx}.and.$M_{fy} > 0$	x	
Doubly Symmetrical Hollow Section (Circular Hollow/ Square Hollow)	x	

Doubly Symmetrical Hollow Section (Circular Hollow/ Square Hollow)	N	N	N	Y	Y	Y	
$M_{fx} > 0$	Y	N	Y	Y	N	Y	
$M_{fy} > 0$	N	Y	Y	N	Y	Y	
M_{fx}.and.$M_{fy} > 0$	N	N	Y	N	N	Y	
$M_{ry} = M_{rx}$					Y	Y	M_{rx}
Execute Table 13.6.A.1	Y		Y	Y			
Execute Table 13.6.A.2		Y	Y				

DECISION TABLE 13.6.A.1 (47)

Data Requirement

Double Sym. Section	x	
Channel Prevented from Twisting	x	
Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)

	1	2	3	4	5	6	7	8	9
Double Sym. Section	Y	Y	Y	Y	N	N	N	N	N
Channel Prevented From Twisting	N	N	N	N	Y	Y	Y	Y	N
Class = 1	Y	N	N	N	Y	N	N	N	I
Class = 2	N	Y	N	N	N	Y	N	N	I
Class = 3	N	N	Y	N	N	N	Y	N	I
Class = 4	N	N	N	Y	N	N	N	Y	I
Execute Table 13.6.1	Y	Y							
Execute Table 13.6.2			Y	Y			Y	Y	
MSG: Section Not a Double Sym. Section A rational Method of Analysis such as given in the Column Research Council's "Guild to Design Criteria for Compression Members" should be used to Calculate Resistant Moments.					Y	Y			Y

DECISION TABLE 13.6.1 (48)

Data Requirement

M_u		Table 13.6.2.A (51)
M_p	x	
ϕ	x	
z	x	
F_y	x	
M_{fx}		Table 7.2.2 (93)
M_{rx}		Table 13.6.A.1 (47)

	1	2	
$M_u > 2/3\ M_p$	Y	N	$M_u\ M_p$
$M_{rx} = \text{Min}[\phi z F_y,\ 1.15\phi M_p(1 - \frac{0.28\ M_p}{M_u})]$	Y		$\phi\ z\ F_y\ M_p\ M_u$
$M_{rx} = \phi M_u$		Y	$\phi\ M_u$
$R_2 = M_{fx}/M_{rx}$	Y	Y	$M_{fx}\ M_{rx}$

DECISION TABLE 13.6.2 (49)

Data Requirement

Class = 3		Table 11. (5)
Class = 4		Table 11. (5)
M_u		Table 13.6.2.A (51)
M_y	x	
ϕ	x	
s	x	
F_y	x	
F_{cr}		Table 13.5.3 (45)
M_{fx}		Table 7.2.2 (93)
M_{rx}		Table 13.6.A.1 (47)

$M_u > \frac{2}{3} M_y$	Y	Y	N	N	M_u M_y
Class = 3	Y	N	Y	N	
Class = 4	N	Y	N	Y	
M_{rx}=Min[$\phi s F_y$, $1.15\phi M_y(1-\frac{0.28\ M_y}{M_u})$]	Y				ϕ s F_y M_y M_u
$M_{rx} = \phi M_u$			Y		ϕ M_u
M_{rx}=Min[$\phi s F_{cr}$, $1.15\phi M_y(1-\frac{0.28\ M_y}{M_u})$]		Y			ϕ s F_{cr} M_y M_u
$M_{rx} = \phi M_u$				Y	ϕ M_u
$R_2 = M_{fx}/M_{rx}$	Y	Y	Y	Y	M_{fx} M_{rx}

DECISION TABLE 13.6.A.2 (50)

Data Requirement

Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)
ϕ	x	
z_y	x	
F_y	x	
s_y	x	
F_{cr}		Table 13.5.3 (45)
M_{fy}		Table 7.2.2 (93)
M_{ry}		

Class = 1	Y	N	N	N	
Class = 2	N	Y	N	N	
Class = 3	N	N	Y	N	
Class = 4	N	N	N	Y	
$M_{ry} = \phi z_y F_y$ $(=\phi M_p)$	Y	Y			ϕ z_y F_y
$M_{ry} = \phi s_y F_y$ $(=\phi M_y)$			Y		ϕ s_y F_y
$M_{ry} = \phi s_y F_{cr}$				Y	ϕ s_y F_{cr}
$R_3 = M_{fy}/M_{ry}$	Y	Y	Y	Y	M_{fy} M_{ry}

DECISION TABLE 13.6.2.A (51)

Data Requirement

I-Section	x	
ω_x		Table 13.6.2.A.1
L	x	
E	x	
I_y	x	
G	x	
J	x	
C_w	x	
s	x	
σ_1		Table 13.6.1.A.1
σ_2		Table 13.6.1.A.1

I-Section	Y	N	
$M_u = \frac{\pi}{\omega_x L} \sqrt{EI_y GJ + \frac{\pi^2}{L^2} E^2 I_y C_w}$		Y	ω_x L E I_y G J C_w
$M_u = \frac{s}{\omega_x} \sqrt{\sigma_1^2 + \sigma_2^2}$	Y		s ω_x σ_1 σ_2

DECISION TABLE 13.6.2.A.1 (52)

Data Requirement

Beam with Both Ends with Effective Lateral Support For the Compression Flange	x	
Beam with Only One End with Effective Lateral Support For The Compression Flange	x	

Beam with Both Ends with Effective Lateral Support For the Compression Flange	Y	N
Beam with Only One End with Effective Lateral Support for the Compression Flange	N	Y
Execute Table 13.8.3.a	Y	
$\omega_x = 1.0$		Y

DECISION TABLE 13.8.3.a (53)

Data Requirement

Member not Subjected to Transverse Loads in the Minor Axis Plane Between Supports	x	
Member Subjected to Distributed Load or Series of Point Loads in the Minor Axis Plane between Supports	x	
Member Subjected to a Concentrated Load or Moment in the Minor Axis Plane Between Supports	x	

Member Not Subjected to Transverse Loads in Minor Axis Plane Between Supports	Y	N	N
Member Subjected to Distributed Load or Series of Point Loads in Minor Axis Plane Between Supports	N	Y	N
Member Subjected to a Concentrated Load or Moment in Minor Axis Plane Between Supports	N	N	Y
Execute Table 13.8.3.a.A.1	Y		
$\omega_x = 1.0$		Y	
$\omega_x = 0.85$			Y

DECISION TABLE 13.8.3.a.A.1 (54)

Data Requirement

Sway Effects About the Major Axis Included in Analysis	x	
Member Bent in Single Curvature About the Major Axis	x	
Member Bent in Double Curvature About the Major Axis	x	
M_{f1}	x	
M_{f2}	x	

Sway Effects About the Major Axis Included in Analysis	N	N	Y	Y	
Member Bent in Single Curvature About the Major Axis	Y	N	Y	N	
Member Bent in Double Curvature About the Major Axis	N	Y	N	Y	
$\omega_x = 0.6 + 0.4\ M_{f1}/M_{f2}$	Y				M_{f1} M_{f2}
$\omega_x = 0.6 - 0.4\ M_{f1}/M_{f2}$		Y			M_{f1} M_{f2}
$\omega_x = 1.0$			Y		
$\omega_x = 0.85$				Y	

DECISION TABLE 13.6.1.A.1 (55)

Data Requirement

A_f	x	
L	x	
d	x	
r_t	x	

Condition	I	
$\sigma_1 = 20000\ A_f/Ld$	Y	A_f L d
$\sigma_2 = 250000\ r_t^2/L^2$	Y	r_t L

DECISION TABLE X.6 (56)

Data Requirement

Axial Compression and Bending	x	
Axial Tension and Bending	x	

Axial Compression and Bending	Y	N
Axial Tension and Bending	N	Y
Execute Table 13.8	Y	
Execute Table 13.9		Y

DECISION TABLE 13.8 (57)

Data Requirement

Double Symmetrical Hollow Section (Circular Hollow/Square Hollow)	x	
I-Section	x	
Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)
M_{rx2}		Table 13:8.A.3 (62)

Double Symmetrical Hollow Section (Circular Hollow/Square Hollow)	N	N	N	N	N	N	N	N	Y	
I-Section	Y	Y	Y	Y	N	N	N	N	N	
Class = 1	Y	N	N	N	Y	N	N	N	I	
Class = 2	N	Y	N	N	N	Y	N	N	I	
Class = 3	N	N	Y	N	N	N	Y	N	I	
Class = 4	N	N	N	Y	N	N	N	Y	I	
Execute Table 13.8.2	Y	Y								
$M_{rx1} = M_{rx2}$									Y	M_{rx2}
$M_{ry1} = M_{rx2}$									Y	M_{rx2}
Execute Table 13.8.1			Y	Y	Y	Y	Y	Y	Y	

DECISION TABLE 13.8.2 (58)

Data Requirement

M_{fx}			Table 7.2.2 (93)
M_{rx1}			Table 13.8.A.1 (60)
M_{fy}			Table 7.2.2 (93)
M_{ry1}			Table 13.8.A.1 (60)
P_f			Table 7.2.2 (93)
P_{rc2}			Table 13.8.A.8 (67)
P_{rc1}			Table 13.8.A.6 (65)
ω_x			Table 13.6.2.A.1 (52)
M_{rx2}			Table 13.8.A.3 (62)
C_{ex}			Table 13.8.A.9 (68)
C_{ey}			Table 13.8.A.9 (68)
ω_y			Table 13.8.A.10 (69)

DECISION TABLE 13.8.2 (58)

$\frac{M_{fx}}{M_{rx1}} + \frac{M_{fy}}{M_{ry1}} \leq 1.0$	Y	Y	Y	N	N	N	N	Y	M_{fx} M_{rx1} M_{fy} M_{ry1}
$\frac{P_f}{P_{rc2}} + \frac{0.85\ M_{fx}}{M_{rx1}} + \frac{0.6\ M_{fy}}{M_{ry1}} \leq 1.0$	N	Y	Y	Y	N	N	Y	N	P_f P_{rc2} M_{fx} M_{rx1} M_{fy} M_{ry1}
$\frac{P_f}{P_{rc1}} + \frac{\omega_x M_{fx}}{M_{rx2}(1-\frac{P_f}{C_{ex}})} + \frac{\omega_y\ M_{fy}}{M_{ry1}(1-\frac{P_f}{C_{ey}})} \leq 1.0$	N	N	Y	Y	Y	N	N	Y	P_f P_{rc1} M_{fx} ω_x M_{rx2} C_{ex} M_{ry1} C_{ey} ω_y
MSG: Both Strength and Stability Criteria Not Satisfied	Y					Y	Y	Y	
MSG: Strength Satisfied; Stability not Satisfied		Y							
MSG: Strength and Stability Critera Satisfied			Y						
MSG: Stability Satisfied; Strength Not Satisfied				Y	Y				

DECISION TABLE 13.8.1 (59)

Data Requirement

P_f		Table 7.2.2 (93)
P_{rc2}		Table 13.8.A.8 (67)
M_{fx}		Table 7.2.2 (93)
M_{rx1}		Table 13.8.A.1 (60)
M_{fy}		Table 7.2.2 (93)
M_{ry1}		Table 13.8.A.1 (60)
P_{rc1}		Table 13.8.A.6 (65)
ω_x		Table 13.6.2.A.1(52)
M_{rx2}		Table 13.8.A.3 (62)
C_{ex}		Table 13.8.A.9 (68)
ω_y		Table 13.8.A.10(69)
C_{ey}		Table 13.8.A.9 (68)

DECISION TABLE 13.8.1 (59)

$\frac{P_f}{P_{rc2}} + \frac{M_{fx}}{M_{rx1}} + \frac{M_{fy}}{M_{ry1}} \leq 1.0$	Y	Y	N	N	P_f P_{rc2} M_{fx} M_{rx1} M_{fy} M_{ry1}
$\frac{P_f}{P_{rc1}} + \frac{\omega_x M_{fx}}{M_{rx2}(1-\frac{P_f}{C_{ex}})} + \frac{\omega_y M_{fy}}{M_{ry1}(1-\frac{P_f}{C_{ey}})} \leq 1.0$	Y	N	Y	N	P_f P_{rc1} ω_x M_{fx} M_{rx2} C_{ex} ω_y M_{fy} M_{ry1} C_{ey}
MSG: Both Strength and Stability Criteria Satisfied	Y				
MSG: Strength Satisfied; Stability not Satisfied Design Unsatisfactory		Y			
MSG: Strength not Satisfied; Stability Satisfied Design Unsatisfactory			Y		
MSG: Both Strength and Stability Criteria Not Satisfied Design Unsatisfactory				Y	

DECISION TABLE 13.8.A.1 (60)

Data Requirement

Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)
ϕ	x	
M_p	x	
s	x	
F_y	x	
z_y	x	
s_y	x	

Class = 1	Y	N	N	N	
Class = 2	N	Y	N	N	
Class = 3	N	N	Y	N	
Class = 4	N	N	N	Y	
$M_{rx1} = \phi M_p$	Y	Y			ϕ M_p
$M_{rx1} = \phi s F_y$			Y		ϕ s F_y
$M_{ry1} = \phi z_y F_y$	Y	Y			ϕ z_y F_y
$M_{ry1} = \phi s_y F_y$			Y		ϕ s_y F_y
Execute Table 13.8.A.2				Y	

DECISION TABLE 13.8.A.2 (61)

Data Requirement

Flange Class = 3		Table 11.8 (20)
Flange Class = 4		Table 11.8 (20)
F_y	x	
h	x	
w	x	
F_{cr}		Table 13.5.3 (45)
ϕ	x	
s	x	
A_w	x	
A_f	x	
s_y	x	

Flange Class = 3	Y	Y	N	N	N	
Flange Class = 4	N	N	Y	Y	Y	
$h/w \le 690/\sqrt{F_y}$	N	N	Y	N	N	F_y h w
$690/\sqrt{F_y} < h/w \le 690/\sqrt{\phi s F_{cr}/\phi s}$	Y	N	N	Y	N	F_y F_{cr} h w
$690/\sqrt{\phi s F_{cr}/\phi s} < h/w \le 12000/F_y$	N	Y	N	N	Y	F_y F_{cr} h w
$M_{rx1} = \phi s F_{cr}$	Y		Y			ϕ s F_{cr}
$M_{rx1} = \phi s F_{cr}[1-0.005\frac{A_w}{A_f}(\frac{h}{w}-690/\sqrt{\phi s F_{cr}/\phi s})]$		Y				ϕ s F_{cr} A_w A_f h w
MSG: M_{rx1} to be Determined by Clause 12, CSA S136				Y	Y	
$M_{ry1} = \phi s_y F_{cr}$	Y		Y			ϕ s_y F_{cr}
$M_{ry1} = \phi s_y F_{cr}[1-0.005\frac{A_w}{A_f}(\frac{h}{w}-690/\sqrt{\phi s_y F_{cr}/\phi s_y})]$		Y				ϕ s_y F_{cr}

DECISION TABLE 13.8.A.3 (62)

Data Requirement

Double Symmetrical Section	x	
Channel Prevented from Twisting	x	
Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)

	1	2	3	4	5	6	7	8	9
Double Symmtrical Section	Y	Y	Y	Y	N	N	N	N	N
Channel Prevented from Twisting	N	N	N	N	Y	Y	Y	Y	N
Class = 1	Y	N	N	N	Y	N	N	N	I
Class = 2	N	Y	N	N	N	Y	N	N	I
Class = 3	N	N	Y	N	N	N	Y	N	I
Class = 4	N	N	N	Y	N	N	N	Y	I
Execute Table 13.8.A.4	Y	Y							
Execute Table 13.8.A.5			Y	Y			Y	Y	
MSG: Section Not a Doubly Symmetrical Section A Rational Method of Analysis Such as Given in CRC "Guide to Design Criteria for Metal Comp. Members" Should be Used. To Calculate Resistance Moment.					Y	Y			Y

DECISION TABLE 13.8.A.4 (63)

Data Requirement

M_u		Table 13.6.2.A (51)
M_p	x	
ϕ	x	

$M_u > 2/3\ M_p$	Y	N	M_p M_u
$M_{rx2} = \text{Min}[\phi\ M_p,\ 1.15\phi M_p(1-\frac{0.28M_p}{M_u})]$	Y		ϕ M_p M_u
$M_{rx2} = \phi M_u$		Y	ϕ M_u

DECISION TABLE 13.8.A.5 (64)

Data Requirement

M_u		Table 13.6.2.A (51)
M_y	x	
ϕ	x	
s	x	
F_y	x	
F_{cr}		Table 13.5.3 (45)

$M_u > \frac{2}{3} M_y$	Y	Y	N	N	M_y M_u
Class = 3	Y	N	Y	N	
Class = 4	N	Y	N	Y	
$M_{rx2} = \mathrm{Min}[\phi s F_y,\ 1.15\phi M_y(1-\frac{0.28M_y}{M_u})]$	Y				ϕ s F_y M_y M_u
$M_{rx2} = \phi M_u$			Y	Y	ϕ M_u
$M_{rx2} = \mathrm{Min}[\phi s F_{cr},\ 1.15\phi M_y(1-\frac{0.28M_y}{M_u})]$		Y			ϕ s F_{cr} M_y M_u

DECISION TABLE 13.8.A.6 (65)

Data Requirement

$\lambda 1$		Table 13.8.A.7 (66)
ϕ	x	
A_g	x	
F_y	x	

$0 \leq \lambda \leq 1.0$	Y	N	N	N	$\lambda 1$
$1.0 < \lambda \leq 2.0$	N	Y	N	N	$\lambda 1$
$2.0 < \lambda \leq 3.6$	N	N	Y	N	$\lambda 1$
$3.6 < \lambda 1$	N	N	N	Y	$\lambda 1$
$P_{rc1} = \phi A_g F_y(1.035-0.201\lambda 1-0.224\lambda 1^2)$	Y				ϕ A_g F_y $\lambda 1$
$P_{rc1} = \phi A_g F_y(-0.111+0.63\lambda^{-1} + 0.094\lambda 1^{-2}$		Y			ϕ A_g F_y $\lambda 1$
$P_{rc1} = \phi A_g F_y(0.012 + 0.867\lambda 1^{-2})$			Y		ϕ A_g F_y $\lambda 1$
$P_{rc1} = \phi A_g F_y \lambda 1^{-2}$				Y	ϕ A_g F_y $\lambda 1$

DECISION TABLE 13.8.A.7 (66)

Data Requirement

UKLR		Table 9.3 (4)
F_y	x	
E	x	

Condition	I	
$\lambda 1 = UKLR\sqrt{F_y/\pi^2 E}$	Y	UKLR F_y E

DECISION TABLE 13.8.A.8 (67)

Data Requirement

ϕ	x	
A_g	x	
F_y	x	

Condition	I	
$P_{rc2} = \phi A_g F_y$	Y	ϕ A_g F_y

DECISION TABLE 13.8.A.9 (68)

Data Requirement

A_g	x	
K_x		Table 9.3 (4)
L_x	x	
r_x	x	
K_y		Table 9.3 (4)
L_y	x	
r_y	x	

Condition	I	
$C_{ex} = 286000\ A_g/(K_xL_x/r_x)^2$	Y	$A_g\ K_x\ L_x\ r_x$
$C_{ey} = 286000\ A_g/(K_yL_y/r_y)^2$	Y	$A_g\ K_y\ L_y\ r_y$

DECISION TABLE 13.8.A.10 (69)

Data Requirement

Beam With Both Ends With Effective Lateral Support For the Compression Flange	x	
Beam With Only One End With Effective Lateral Support For the Compression Flange	x	

Beam With Both Ends With Effective Lateral Support For the Compression Flange	Y	N
Beam With Only One End With Effective Lateral Support For The Compression Flange	N	Y
Execute Table 13.8.A.11	Y	
$\omega_y = 1.0$		Y

DECISION TABLE 13.8.A.11 (70)

Data Requirement

Member Not Subjected To Transverse Loads in the Major Axis Plane Between Supports	x	
Member Subjected to Distributed Load or Series of Point Loads In the Major Axis Plane Between Supports.	x	
Member Subjected to a Concentrated Load or Moment In the Major Axis Plane Between Supports	x	

Member Not Subjected to Transverse Loads in the Major Axis Plane Between Supports	Y	N	N
Member Subjected to Distributed Load or Series of Point Loads in the Major Axis Plane Between Supports	N	Y	N
Member Subjected to a Concentrated Load or Moment in the Major Axis Plane Between Supports	N	N	Y
Execute Table 13.8.A.12	Y		
$\omega_y = 1.0$		Y	
$\omega_y = 0.85$			Y

DECISION TABLE 13.8.A.12 (71)

Data Requirement

Sway Effects About the Minor Axis Included in Analysis/Resisted by Bracing or Shear Wall	x	
Member Bent in Single Curvature About the Minor Axis	x	
Member Bent in Double Curvature About the Minor Axis	x	
M_{f1y}	x	
M_{f2y}	x	

Sway Effects About the Minor Axis Included in Analysis/Resisted by Bracing or Shear Wall	N	N	Y	Y	
Member Bent in Single Curvature About the Minor Axis.	Y	N	Y	N	
Member Bent in Double Curvature About the Minor Axis.	N	Y	N	Y	
$\omega_y = 0.6 + 0.4\, M_{f1y}/M_{f2y}$	Y				M_{f1y} M_{f2y}
$\omega_y = 0.6 - 0.4\, M_{f1y}/M_{f2y}$		Y			M_{f1y} M_{f2y}
$\omega_y = 1.0$			Y		
$\omega_y = 0.85$				Y	

DECISION TABLE 13.9 (72)

Data Requirement

P_f		Table 7.2.2 (93)
P_{rt}		Table 13.2.A.1 (29)
M_{fx}		Table 7.2.2(93)
M_{r1}		Table 13.9.A.1 (73)
M_{fy}		Table 7.2.2 (93)
M_{rx2}		Table 13.8.A.3 (62)
M_{ry1}		Table 13.8.A.1 (60)

$\frac{P_f}{P_{rt}} + \frac{M_{fx}}{M_{r1}} + \frac{M_{fy}}{M_{r1}} \leq 1.0$	Y	N	Y	N	P_f P_{rt} M_{fx} M_{r1} M_{fy}
$\frac{M_{fx}}{M_{rx2}} + \frac{M_{fy}}{M_{ry1}} - \frac{P_f}{P_{rt}} \leq 1.0$	Y	N	N	Y	M_{fx} M_{rx2} M_{fy} M_{ry1} P_f P_{rt}
Tensile and Compressive Stress Criteria Satisfied	Y				
Tensil. Stress Criterion Not Satisfied				Y	
Compressive Stress Criterion Not Satisfied			Y		
Tensile and Compressive Stress Criteria Not Satisfied		Y			

DECISION TABLE 13.9.A.1 (73)

Data Requirement

Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)

Class = 1	Y				
Class = 2		Y			
Class = 3			Y		
Class = 4				Y	
$M_{r1} = \phi M_p$	Y	Y			$\phi\ M_p$
$M_{r1} = \phi M_y$			Y	Y	$\phi\ M_y$

DECISION TABLE 13.4 (74)

Data Requirement

V_f		Table 7.2.2 (93)
V_r		Table 13.4.A.1 (75)

$\frac{V_f}{V_r} \le 1.0$	Y	N	V_f V_r
Shear Criterion Satisfied	Y		
Shear Criterion Not Satisfied		Y	

DECISION TABLE 13.4.A.1 (75)

Data Requirement

Elastic Analysis	x	Table 13.4.1 (76)
Plastic Analysis	x	
Flexural Member Subjected to Shear	x	
Pin	x	
A_w	x	
F_s		
ϕ	x	
A_g	x	
F_y	x	
w	x	
d	x	
Gusset Plate	x	

Elastic Analysis	Y	Y	Y	N	
Plastic Analysis	N	N	N	Y	
Flexural Member Subjected to Shear	Y	N	N	Y	
Gusset Plate	N	Y	N	I	
Pin	N	N	Y	I	
$V_r = \phi A_w F_s$	Y				A_w F_s
$V_r = 0.5 \phi A_g F_y$		Y			ϕ A_g F_y
$V_r = 0.66 \phi A_g F_y$			Y		ϕ A_g F_y
$V_r = 0.55 \phi w d F_y$				Y	ϕ w d F_y

DECISION TABLE 13.4.1 (76)

Data Requirement

h	x	
w	x	
K_v		Table 13.4.1.A.1 (77)
F_y	x	
T		Table 13.4.1.A.1 (77)
η		Table 13.4.1.A.1 (77)

	1	2	3	4	
$\frac{h}{w} \le 167\sqrt{K_v/F_y}$	Y	N	N	N	h w K_v F_y
$167\sqrt{K_v/F_y} < \frac{h}{w} \le 190\sqrt{K_v/F_y}$	N	Y	N	N	h w K_v F_y
$190\sqrt{K_v/F_y} < \frac{h}{w} \le 239\sqrt{K_v/F_y}$	N	N	Y	N	h w K_v F_y
$239\sqrt{K_v/F_y} < \frac{h}{w}$	N	N	N	Y	h w K_v F_y
$F_s = 0.66\ F_y$	Y				F_y
$F_s = 110\sqrt{F_y K_v}/(h/w)$		Y			F_y K_v h
$F_s = 110\sqrt{F_y K_v}.T/(h/w) + \eta\ F_y$			Y		F_y K_v T h w η
$F_s = 26200\ K_v.T/(h/w)^2 + \eta\ F_y$				Y	K_v T h w η F_y

DECISION TABLE 13.4.1.A.1 (77)

Data Requirement

Stiffened Webs	x	
a	x	
h	x	

	1	2	3	
Stiffened Webs	N	Y	Y	
a/h < 1.0	I	Y	N	a h
a/h ≥ 1.0	I	N	Y	a h
$K_v = 5.34$	Y			
$T = 1.0$	Y			
$\eta = 0.0$	Y			
$K_v = 4 + 5.34/(a/h)^2$		Y		a h
$T = 1\text{-}0.866/\sqrt{1+(a/h)^2}$		Y	Y	a h
$\eta = 0.5/\sqrt{1+(a/h)^2}$		Y	Y	a h
$K_v = 5.34 + 4/(a/h)^2$			Y	a h

DECISION TABLE 8.5 (78)

Data Requirement

Web Stiffener Supplied On The Member At A Point Of Load Application Where A Plastic Hinge Would Form	x	
Splices In the Member Are Designed To Transmit 1.1 Times The Maximum Computed Moment Under Factored Loads At The Splice Location Or 0.25 M_p Whichever is Greater	x	
Member Is Not Subjected To Repeated Heavy Impact Or Fatigue	x	
The Influence Of Inelastic Deformation On The Strength Of The Structure Shall Be Taken Into Account	x	

DECISION TABLE 8.5 (78) PLASTIC ANALYSIS

Web Stiffener Are Supplied On The Member At A Point Of Load Application Where A Plastic Hinge Would Form	Y	N	Y	Y	Y
Splices In The Member Are Designed To Transmit 1.1 Times The Max. Computed Moment Under Factored Loads At The Splice Location Or 0.25 M_p Whichever is Greater	Y	I	N	Y	Y
Member Is Not Subjected To Repeated Heavy Impact Or Fatigue	Y	I	I	N	Y
The Influence Of Inelastic Deformation On The Strength Of The Structure Shall Be Taken Into Account	Y	I	I	I	N
Execute Table 8.5a	Y				
MSG: Limitations On Plastic Analysis Not Satisfied. Check Clause 8.5(d,e,f,g)		Y	Y	Y	Y

DECISION TABLE 8.5.a (79)

Data Requirement

F_y	x	
F_u	x	

$F_y \leq 0.8\ F_u$	Y N	F_y F_u
Execute Table 8.5.b	Y	
MSG: Limitations on Plastic Analysis Not Satisfied. Check Clause 8.5(a)	Y	

DECISION TABLE 8.5.b (80)

Data Requirement

Class = 1		Table 11. (5)

Class = 1	Y N
Execute Table 8.5.c	Y
MSG: Limitations on Plastic Analysis Not Satisfied. Check Clause 8.5(b)	Y

DECISION TABLE 8.5.c (81)

Data Requirement

L_{ph}	x	
L_{cr}		Table 13.7 (82)

Laterally Unbraced Length Between Plastic Hinges $\leq L_{cr}$	Y	N	L_{ph} L_{cr}
Execute Table X.2	Y		
MSG: Limitations on Plastic Analysis Not Satisfied Check Clause 13.7		Y	

DECISION TABLE 13.7 (82)

Data Requirement

M_{f1}	x	
M_{f2}	x	
r_y	x	
F_y	x	

$\frac{M_{f1}}{M_{f2}} > 0.5$	Y	N	M_{f1} M_{f2}
$\frac{M_{f1}}{M_{f2}} \leq 0.5$	N	Y	M_{f1} M_{f2}
$L_{cr} = 210\ r_y/\sqrt{F_y}$	Y		r_y F_y
$L_{cr} = 375\ r_y/\sqrt{F_y}$		Y	r_y F_y

DECISION TABLE 15. (83)

Data Requirement

Cover Plate Used	x	

Cover Plate Used	Y	N
Execute Table 15.A.1	Y	
Execute Table X.3		Y

DECISION TABLE 15.A.1 (84)

Data Requirement

Bolted Girder	x	
Welded Girder	x	
A_{cov}	x	
A_f	x	
P_{cov}	x	
M_{fc}	x	
Y	x	
I_{cov}	x	

	1	2	3	4	5	
Bolted Girder	Y	Y	Y	Y	N	
Welded Girder	N	N	N	N	Y	
$A_{cov} \leq 70\% A_F$	Y	Y	N	N	I	A_{cov} A_f
$P_{cov} \geq \frac{A_{cov} M_{fc} Y}{I_{cov}}$	Y	N	Y	N	I	P_{cov} A_{cov} M_{fc} Y I_{cov}
MSG: Clause 15.4.2 and 15.4.4 Satisfied	Y					
MSG: Clause 15.4.4 Not Satisfied		Y		Y		
MSG: Clause 15.4.2 Not Satisfied			Y	Y		
Execute Table X.3	Y				Y	

DECISION TABLE 15.5 (85)

Data Requirement

Interior Bearing Stiffener	x	
End Bearing Stiffener	x	
T_{web}	x	Table 15.5 (85)
A_{eweb}		
A_{bst}	x	

Interior Bearing Stiffener	Y	N	
End Bearing Stiffener	N	Y	
$A_{eweb} = T_{web} \times 25$	Y		T_{web}
$A_{eweb} = T_{web} \times 12$		Y	T_{web}
$A_g = A_{eweb} + A_{bst}$	Y	Y	A_{eweb} A_{bst}
Execute Table 15.5.A.1	Y	Y	

DECISON TABLE 15.5.A.1 (86)

Data Requirement

K_x		Table 9.3 (4)
L_x	x	

$K_xL_x > 3/4\ L_x$	Y	N	K_x L_x
MSG: Clause 15.5.2, KL of Stiffener Satisfied	Y		
MSG: K_xL_x of Stiffener Does Not Satisfy Clause 15.5.2		Y	
Execute Table X.3	Y		

DECISION TABLE 15.6 (87)

Data Requirement

b	x		
t	x		
F_y	x		
FI_{ywb}	x		
h	x		
A_{ist}	x		
a	x		
w	x		
C		Table 15.6.A.1	(88)
YRAT		Table 15.6.A.1	(88)
SFACT		Table 15.6.A.2	(89)
REFAC		Table 15.6.A.3	(90)
V_c	x		
TLOAD	x		

DECISION TABLE 15.6 (87)

Condition / Action	1	2	3	4	5	Data
$b/t \leq 100/\sqrt{F_y}$	Y	N	Y	Y	Y	$b\ t\ F_y$
$FI_{ywb} \geq (h/50)^4$	Y	I	N	Y	Y	$FI_{ywb}\ h$
$A_{ist} \geq \frac{aw}{2} [1- \frac{a/h}{\sqrt{1+(a/h)^2}}$ Cx YRAT x SFACT x REFAC	Y	I	I	N	Y	A_{ist} a w h C YRAT SFACT REFAC
$V_c \geq \max (0.0026h\ F_y^{3/2}$ x REFAC, $\frac{TLOAD}{h})$	Y	I	I	I	N	V_c h F_y REFAC TLOAD
MSG: Intermediate Stiffener Design Satisfactory	Y					
MSG: b/t Limit for Intermediate Stiffener Exceeded		Y				
MSG: I_{ywB} Less Than $(h/50)^4$			Y			
MSG: Minimum A_{ist} Not Satisfied				Y		
MSG: Minimum V_c/h Not Satisfied					Y	

DECISION TABLE 15.6.A.1 (88)

Data Requirement

K_v		Table 13.4.1.A.1 (77)
F_y	x	
h	x	
w	x	
F_{ystif}	x	

Condition	I	
$C = MAX(1.0 - \frac{45000.0\ K_v}{F_y(h/w)^2}, 0.1)$	Y	K_v F_y h w
$YRAT = F_y/F_{ystif}$	Y	F_y F_{ystif}

DECISION TABLE 15.6.A.2 (89)

Data Requirement

Intermediate Stiffener Furnished in Pairs	x	
Intermediate Single Angle Stiffener	x	
Intermediate Single Plate Stiffener	x	

Intermediate Stiffener Furnished in Pairs	Y	N	N
Intermediate Single Angle Stiffener	N	Y	N
Intermediate Single Plate Stiffener	N	N	Y
SFACT = 1.0	Y		
SFACT = 1.8		Y	
SFACT = 2.4			Y

DECISION TABLE 15.6.A.3 (90)

Data Requirement

V_{fadj} V_r	x	Table 13.4.A.1 (75)

If $\frac{V_{fadj}}{V_r}$ < 1.0	Y N	V_{fadj} V_r
REFAC = $\frac{V_{fadj}}{V_r}$	Y	V_{fadj} V_r
REFAC = 1.0	Y	

DECISION TABLE 13.9 (91)

Data Requirement

B_{r1} SLOAD	x	Table 13.9.A.1 (92)

B_{r1} > SLOAD	Y N	B_{r1} SLOAD
MSG: Clause 15.9 (Stability of Thin Webs) Satisfied	Y	
MSG: Clause 15.9 Not Satisfied	Y	

DECISION TABLE 13.9.A.1 (92)

Data Requirement

Flange Is Restrained Against Rotation	x	
Flange is not Restrainted Against Rotation	x	
ϕ	x	
h	x	
w	x	
a	x	
ATW	x	

Flange Is Restrained Against Rotation	Y	N	
Flange is Not Restrained Against Rotation	N	Y	
$B_{r1} = \phi \frac{16700}{(h/w)^2} [5.5 + \frac{4}{(a/h)^2}] \times ATW$	Y		ϕ h w a ATW
$B_{r1} = \phi \frac{16700}{(h/w)^2} [2 + \frac{4}{(a/h)^2}] \times ATW$		Y	ϕ h w a ATW

NOTE: For Distributed Load
ATW = Panel Length x Web Thickness

For Concentrated Loads and Loads Distributed Partially
ATW = Less of a or h x Web Thickness

DECISION TABLE 7.2.2 (93)

Data Requirement

$P_f = 0$	x	
$M_{fx} = 0$	x	
$M_{fy} = 0$	x	
$V_f = 0$	x	
$B_f = 0$	x	
γ		Table 7.2.5 (96)
α_D		Table 7.2.3 (94)
α_L		Table 7.2.3 (94)
α_Q		Table 7.2.3 (94)
α_T		Table 7.2.3 (94)
P_D	x	
P_L	x	
P_Q	x	
P_T	x	
M_{xD}	x	
M_{xL}	x	
M_{xQ}	x	
M_{xT}	x	
M_{yD}	x	
M_{yL}	x	
M_{yQ}	x	
M_{yT}	x	
V_D	x	
V_L	x	
V_Q	x	
V_T	x	
B_D	x	
B_L	x	
B_Q	x	
B_T	x	
ψ		Table 7.2.4 (95)

DECISION TABLE 7.2.2 (93)

$P_f=0$	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y	
$M_{fx}=0$	Y	N	N	Y	N	N	Y	N	Y	Y	N	Y	N	N	Y	Y	Y	N	N	Y	N	N	Y	N	Y	Y	N	Y	N	N	Y	Y	
$M_{fy}=0$	N	Y	N	N	Y	N	Y	Y	N	Y	Y	N	N	N	Y	Y	N	Y	N	N	Y	N	Y	Y	N	Y	Y	N	N	N	Y	Y	
$V_f=0$	Y	Y	Y	N	N	N	Y	Y	Y	N	N	N	Y	N	N	Y	Y	Y	Y	N	N	N	Y	Y	Y	N	N	N	Y	N	N	Y	
$B_f=0$	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	
$P_f=\gamma[\alpha_D P_D+\psi(\alpha_L P_L+\alpha_Q P_Q+\alpha_T P_T)]$							Y	Y	Y	Y	Y	Y	Y	Y									Y	Y	Y	Y	Y	Y	Y	Y			$\gamma\ \alpha_D\ \alpha_L\ \alpha_Q\ \alpha_T\ \psi\ P_D\ P_L\ P_\alpha\ P_T$
$M_{fx}=\gamma[\alpha_D M_{xD}+\psi(\alpha_L M_{xL}+\alpha_Q M_{x\alpha}+\alpha_T M_{xT})]$		Y	Y		Y	Y		Y			Y		Y	Y				Y	Y		Y	Y		Y			Y		Y	Y			$\gamma\ \alpha_D\ \alpha_L\ \alpha_Q\ \alpha_T\ \psi\ M_{xD}\ M_{xL}\ M_{x\alpha}\ M_{xT}$
$M_{fy}=\gamma[\alpha_D M_{yD}+\psi(\alpha_L M_{yL}+\alpha_Q M_{yQ}+\alpha_T M_{yT})]$	Y		Y	Y		Y			Y			Y	Y	Y			Y		Y	Y		Y			Y			Y	Y	Y			$\gamma\ \alpha_D\ \alpha_L\ \alpha_Q\ \alpha_T\ \psi\ M_{yD}\ M_{yL}\ M_{yQ}\ M_{yT}$
$V_f=\gamma[\alpha_D V_D+\psi(\alpha_L V_L+\alpha_Q V_Q+\alpha_T V_T)]$				Y	Y	Y				Y	Y	Y		Y	Y					Y	Y	Y				Y	Y	Y		Y	Y		$\gamma\ \alpha_D\ \alpha_L\ \alpha_Q\ \alpha_T\ \psi\ V_D\ V_L\ V_Q\ V_T$
$B_f=\gamma[\alpha_D B_D+\psi(\alpha_L B_L+\alpha_Q B_Q+\alpha_T B_T)]$																	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	$\gamma\ \alpha_D\ \alpha_L\ \alpha_Q\ \alpha_T\ \psi\ B_D\ B_L\ B_Q\ B_T$
MSG: No Stress Resultants																Y																	

DECISION TABLE 7.2.3 (94)

Data Requirement

Overturning, Uplift Or Stress Reversal Case	x	

Overturning, Uplift Or Stress Reversal Case	Y	N
α_D = 1.25		Y
α_D = 0.85	Y	
α_L = 1.5	Y	Y
α_Q = 1.5	Y	Y
α_T = 1.25	Y	Y

DECISION TABLE 7.2.4 (95)

Data Requirement

When One of L, T, Q Act	x	
When Two of L, T, Q Act	x	
When All of L, T, Q Act	x	

When One of L, T, Q Act	Y	N	N
When Two of L, T, Q Act	N	Y	N
When All of L, T, Q Act	N	N	Y
ψ = 1.0	Y		
ψ = 0.7		Y	
ψ = 0.6			Y

DECISION TABLE 7.2.5 (96)

Data Requirement

Farm Building of Low Occupancy Rate, Density < One Person/500 sq. ft. During Normal Periods of Use of 4 hrs. or Longer/Buildings For Which It Can Be Shown That Collapse Is Not Likely To Cause Injury	x	

Farm Building of Low Occupancy Rate, Density < One Person/500 sq.ft. During Normal Periods of Use of 4 hrs. or Longer/Buildings For which It Can Be Shown That Collapse Is Not Likely To Cause Injury.	N	Y
$\gamma = 1.0$	Y	
$\gamma = 0.8$		Y

DECISION TABLE 13.10 (97)

Data Requirement

R_B		Table 13.10.A.1 (98)

$R_B \leq 1.0$	Y	N	R_B
MSG: Clause 13.10 Bearing Resistance Satisfied	Y		
MSG: Clause 13.10 Bearing Resistance Not Satisfied		Y	

DECISION TABLE 13.10.A.1 (98)

Data Requirement

On Contact Area Of Machined, Accurately Sawn Or Fitted Parts	x	
On Expansion Rollers Or Rockers	x	
ϕ	x	
F_y	x	
$A_{contact}$	x	
D	x	
L	x	
B_f	x	Table 7.2.2 (93)
B_r		Table 13.10.A.1 (98)

On Contact Area of Machined, Accurately Sawn or Fitted Parts	Y	N	
On Expansion Rollers or Rockers	N	Y	
$B_r = 1.5\phi\ F_y\ A_{contact}$	Y		$\phi\ F_y\ A_{contact}$
$B_r = 0.0008\phi DLF_y{}^2$		Y	ϕ D L F_y
$R_B = \frac{B_f}{B_r}$	Y	Y	B_f B_r

DECISION TABLE X.6.R (56R)

Data Requirement

Axial Compression And Bending	x	
Axial Tension And Bending	x	

Axial Compression And Bending	Y	N
Axial Tension And Bending	N	Y
CHECKI=0.0	Y	Y
Execute Table 13.8.R	Y	
Execute Table 13.9		Y

DECISION TABLE 13.8.R (57R)

Data Requirement

Double Symmetric Hollow Section (Circular Hollow/Square Hollow) .	x	
I-Section	x	
Class = 1		Table 11. (5)
Class = 2		Table 11. (5)
Class = 3		Table 11. (5)
Class = 4		Table 11. (5)

Double Sym. Hollow Section (Circular Hollow/Square Hollow)	N	N	N	N	N	N	N	N	Y	
I-Section	Y	Y	Y	Y	N	N	N	N	N	
Class = 1	Y	N	N	N	Y	N	N	N	I	
Class = 2	N	Y	N	N	N	Y	N	N	I	
Class = 3	N	N	Y	N	N	N	Y	N	I	
Class = 4	N	N	N	Y	N	N	N	Y	I	
CHECKN=2.0			Y	Y	Y	Y	Y	Y	Y	
CHECKN=3.0	Y	Y								
CHECKI=CHECKI+1.0	Y	Y	Y	Y	Y	Y	Y	Y	Y	
Execute Table 13.8.2.R	Y	Y								
M_{rx1} = M_{rx2}									Y	M_{rx2}
M_{ry1} = M_{rx2}									Y	M_{rx2}
Execute Table 13.8.1.R			Y	Y	Y	Y	Y	Y	Y	
CHECKI Eq. CHECKN?	Y	Y	Y	Y	Y	Y	Y	Y	Y	

DECISION TABLE 13.8.2.R (5BR)

Data Requirement

CHECKI		Table 13.B (56R)
M_{fx}		Table 7.2.2 (93)
M_{fy}		Table 7.2.2 (93)
M_{rxA}		Table X.7 (99R)
M_{ryA}		Table X.7 (99R)
P_f		Table 7.2.2 (93)
P_{rcA}		Table X.7 (99R)
ω_{xA}		Table X.7 (99R)
ω_{yA}		Table X.7 (99R)
θ_x		Table X.7 (99R)
θ_y		Table X.7 (99R)

DECISION TABLE 13.8.2.R (5BR)

	1	2	3	4	5	6	
CHECKI=1.0	Y	Y	N	N	N	N	CHECKI
CHECKI=2.0	N	N	Y	Y	N	N	CHECKI
CHECKI=3.0	N	N	N	N	Y	Y	CHECKI
$\frac{M_{fx}}{M_{rxA}} + \frac{M_{fy}}{M_{ryA}} \leq 1.0$	Y	N	I	I	I	I	M_{fx} M_{fy} M_{rxA} M_{ryA} CHECKI
$\frac{P_f}{P_{rcA}} + \frac{M_{fx}\omega_{xA}}{M_{rxA}\theta_x} + \frac{M_{fy}\omega_{yA}}{M_{ryA}\theta_y} \leq 1.0$	I	I	Y	N	Y	N	P_f M_{fx} M_{fy} CHECKI P_{rcA} M_{rxA} M_{ryA} ω_{xA} ω_{yA} θ_x θ_y
MSG: First Strength Criterion (When CHECKI=1) Satisfied	Y						
MSG: First Strength Criterion (When CHECKI=1) Not Satisfied		Y					
MSG: Second Strength Criterion (When CHECKI=2) Satisfied.			Y				
MSG: Second Strength Criterion (When CHECKI=2) Not Satisfied				Y			
MSG: Stability Criterion (When CHECKI=3) Satisfied					Y		
MSG: Stability Criterion (When CHECKI=3) Not Satisfied						Y	

DECISION TABLE 13.8.1.R (59R)

Data Requirement

CHECKI		Table 13.B (56R)
P_f		Table 7.2.2 (93)
M_{fx}		Table 7.2.2 (93)
M_{fy}		Table 7.2.2 (93)
M_{rxA}		Table X.7 (99R)
M_{ryA}		Table X.7 (99R)
P_{rcA}		Table X.7 (99R)
ω_{xA}		Table X.7 99R)
ω_{yA}		Table X.7 (99R)
θ_x		Table X.7 (99R)
θ_y		Table X.7 (99R)

DECISION TABLE 13.8.1.R (59R)

CHECKI=1	Y	N	Y	N	CHECKI
CHECKI=2	N	Y	N	Y	CHECKI
$\frac{P_f}{P_{rcA}} + \frac{M_{fx}\omega_{xA}}{M_{rxA}\theta_x} + \frac{M_{fy}\omega_{yA}}{M_{ryA}\theta_y} \leq 1.0$	Y	Y	N	N	P_f M_{fx} M_{fy} M_{rxA} M_{ryA} ω_{xA} ω_{yA} θ_x θ_y
MSG: Strength Criterion Satisfied	Y				
MSG: Stability Criterion Satisfied		Y			
MSG: Strength Criterion Not Satisfied			Y		
MSG: Stability Criterion Not Satisfied				Y	

DECISION TABLE X.7.R (99R)

Data Requirement

CHECKN		Table 13.8.R (57R)
CHECKI		Table X.6.R (56R)
M_{rx1}		Table 13.8.A.1 (60)
M_{rx2}		Table 13.8.A.3 (62)
M_{ry1}		Table 13.8.A.1 (60)
P_{rc1}		Table 13.8.A.6 (65)
P_{rc2}		Table 13.8.A.8 (67)
ω_x		Table 13.6.2.A.1 (52)
ω_y		Table 13.8.A.10 (69)
P_f		Table 7.2.2 (93)
C_{ex}		Table 13.8.A.9 (68)
C_{ey}		Table 13.8.A.9 (68)

DECISION TABLE X.7.R (99R)

	1	2	3	4	5	
CHECKN=2.0	Y	Y	N	N	N	CHECKN
CHECKN=3.0	N	N	Y	Y	Y	CHECKN
CHECKI=1.0	Y	N	Y	N	N	CHECKI
CHECKI=2.0	N	Y	N	Y	N	CHECKI
CHECKI=3.0	I	I	N	N	Y	CHECKI
$M_{rxA}=M_{rx1}$	Y		Y	Y		M_{rx1}
$M_{rxA}=M_{rx2}$		Y			Y	M_{rx2}
$M_{ryA}=M_{ry1}$	Y	Y	Y	Y	Y	M_{ry1}
$P_{rcA}=P_{rc1}$		Y			Y	P_{rc1}
$P_{rcA}=P_{rc2}$	Y			Y		P_{rc2}
$\omega_{xA}=1.0$	Y					
$\omega_{xA}=0.85$				Y		
$\omega_{xA}=\omega_x$		Y			Y	ω_x
$\omega_{yA}=1.0$	Y					
$\omega_{yA}=0.6$				Y		
$\omega_{yA}=\omega_y$		Y			Y	ω_y
$\theta_x=1.0$	Y			Y		
$\theta_x=(1-P_f/c_{ex})$		Y			Y	P_f c_{ex}
$\theta_y=1.0$	Y					
$\theta_y=(1-P_f/c_{ey})$		Y		Y	Y	P_f C_{ry}

APPENDIX C

USER'S GUIDE

Introduction

This appendix discusses the processing program limitations, Input/Output procedures for interactive and batch mode and contains a description of MTS files. The source programs are listed in Appendices D and E.

C.1 Program Limitations

(a) Interactive Mode

Maximum number of tables	= 120
Maximum number of rules per table	= 40
Maximum number of ingredients per condition or action	= 11
Maximum number of data elements	= 700

(b) Batch Mode

Maximum number of tables	= 120
Maximum number of rules per table	= 40
Maximum number of ingredients per condition or action	= 9
Maximum number of data elements	= 700

C.2 Batch Mode Control Cards

(1) $SIGNON XXXXXX P=[ ] T=[ ] PRIO= [R / H / N / L]

(2) PASSWORD

(3) $RUN *FORTG SCARDS=BATCHMODE+ROUTINE1+ROUTINE2+
ROUTINE3+ROUTINE4+ROUTINE5+
ROUTINE6+ROUTINE7+ROUTINE8

(4) $RUN -LOAD# 8=CSAS16

(5) DATA DECK

(6) $SIGNOFF

See Sect. C.5.5 for description of source and object files in item 3.

See Sect. C.5 for MTS I/O Units in item 4.

See Sect. C.3 for contents of data deck in item 5.

C.3 Data Input For Batch Mode

(a) Control Cards:

(1)

	11	14
	THEMAP	

FORMAT(T11,A4)

THEMAP Prints out the contents of the permanent and temporary arrays.

= YES or NO

(2)

10	
ISAVE	

FORMAT(I10)

ISAVE=1 indicates decision table information has to be input

ISAVE=2 indicates decision table information has been processed from previous runs, and the following items 3 to 8 will not be necessary. This informationstorage capability, on average, saves 25 seconds of CPU time per run.

(b) <u>Decision Table Input:</u> Items (3) to (5) to be supplied for each decision table.

(3)

10	20	30	40	
T	LT	MT	NT	

FORMAT (4I10)

T = decision table number

LT = number of rules

MT = number of actions

NT = number of conditions

(4)

1	5	11	50	51	79	80
INDEX		ENTRIES		INGREDIENTS		FLAG

FORMAT(T51,5(I4,1X),I4,A1,T1,I5,5X,40I1)

INDEX = condition or action stub subscript
(condition stub: condition data subscripts)
(action stub: action data subscripts; or
blank if action is a message;
or a table number if action is
to execute a table)

ENTRIES = condition or action entries
(condition entry: 1=YES 2=NO)
(action entry: 1= calculating value of
an element or printing
message
2= execution of another
table)

INGREDIENTS = the data subscripts of condition or
action ingredients

FLAG = 'c', if more ingredients on the next card
= 'blank' if no more ingredients on the next
card.

(5) Read this card if FLAG=c in 4

51 79

| | INGREDIENTS | |

FORMAT(T51,5(I4,1X),I4)

(6) BLANK CARD

this indicates the end of decision table input

(c) <u>Data Element Properties:</u> Item (7) to be supplied for each data element referenced by decision tables

(7)

10	20	30	
KGLOB	TABDK	NSET	

FORMAT(3I10)

KGLOB = data subscript

TABDK = table number of KGLOB

NSET = mutually exclusive set number of KGLOB

(8)

BLANK CARD

This indicates the end of the data element information

(d) <u>Problem Data:</u> Values of data elements required to define problem

(9)

5	11	20	
KGLOB		DATAK	

FORMAT(I5,5X,F10.0)

KGLOB = external input data subscript

DATAK = corresponding data value.

(10)

BLANK CARD

This indicates the end of the external input data

(e) <u>Instructions for Execution:</u>

(11)

10	
TFIRST	

FORMAT(I10)

TFIRST = the number of first decision table to be executed

```
           11   14
(12)  |   | TRACE |   }      FORMAT(T11,A4)
```

TRACE = 'YES', if a trace of the tables executed for the problem is desired.
= 'NO', if no trace is desired.

```
           11   14
(13)  |   | WARAY |   }      FORMAT(T11,A4)
```

WARAY = 'YES', if listing of compacted arrays is desired
= 'NO', if no listing is desired

A schematic arrangement of the batch-mode input cards is illustrated in Fig. C.1.

C.4 Interactive Mode Control Commands

```
$signon XXXX
password
$run obcombine 5=filename1 9=filename2 8=filename3
               2=filename4 6=*print*   4=*source*
               7=*sink*
```

For details on I/O units:

I/O unit 5 (see Sect. C.5.1)

I/O unit 2 (see Sect. C.5.3)

I/O unit 9 (see Sect. C.5.2)

I/O unit 8,6,4,7 (see Sect. C.5.4)

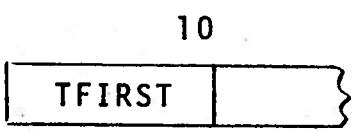
The files associated with the I/O units are discussed in Sect. C.6. If decision table and data property information has already been processed and stored, the only data required to execute a problem are those associated with I/O units 5, and 2.

C.5 I/O Units For Interactive Mode

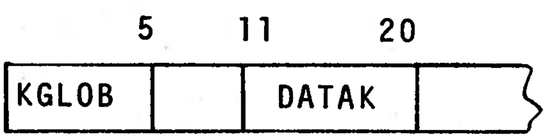
(Note: For definition of variable names see Sect. C.3).

C.5.1 Data In I/O Unit 5

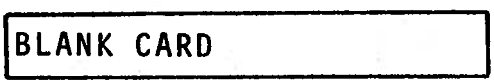
(1)



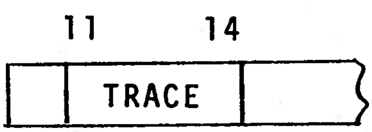
(2)



(3)



(4)



(5)

	11	14	
	WARAY		

C.5.2 Data In I/O Unit 9

This file contains decision table information and data properties in coded form.

(1)

10	20	30	40	
T	LT	MT	NT	

(2)

1 5	11	50 51	79	80
INDEX		ENTRIES	INGREDIENTS	FLAG

(3)

	51	79
	INGREDIENTS	

(4)

BLANK CARD

(5)

10	20	30	
KGLOB	TABDK	NSET	

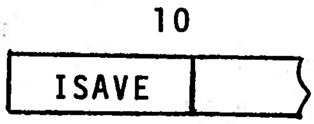
(6)

BLANK CARD

C.5.3 Data In I/O Unit 2

(1)

	11	14	
	THEMAP		

(2) 

C.5.4 Data In I/O Units 8, 6, 4, 7

I/O Unit 8 - The file which this unit refers to contains the processed decision table information in compacted binary form.

I/O Unit 6 - 6 = *print* refers to output for the line printer.

I/O Unit 4 - 4 = *SOURCE* refers to input from the terminal.

I/O Unit 7 - 7=*sink* refers to output for the terminal.

C.6 Description Of MTS Files

(a) OBCOMBINE - this file is an MTS control file. The contents of which is listed in Fig. C.2. This file is in object form.

(b) OBCOMBINER - this file is used with the recursive execution scheme. It's contents and function are similar to that of OBCOMBINE. A listing of this file is presented in Fig. C.3.

(c) DECIDATA1 - this file contains coded but unprocessed decision tables and data property information. It is referenced by I/O unit 9 during execution.

(d) DECIDATA1R - this file is used with the recursive execution scheme. Its content and function are similar to that of DECIDATA1.

(e) CSAS16 - this file contains processed decision tables and data property information in binary form. It is referenced by I/O unit 8 during execution.

(f) CSAS16R - this file is used with the recursive execution scheme. Its content and function are similar to that of CSAS16.

(g) MAPNSAVE - this file contains the values of variables THEMAP and ISAVE (Sect. C.5.3).

(h) BATCHMODE - this file contains the subroutines SPECHK, SETUP, INITIAL, SETS, OUTPUT and STAK in source form for batch mode execution.

(i) LUB - this file contains the external input data for example 1 in Sect. 8.1. A listing of this file is presented in Fig. C.4. The file is referenced by I/O unit 5.

(j) AXCOM - this file contains the externally input data for example 2 in Sect. 8.2. A listing of this file is presented in Fig. C.5. The file is referenced by I/O unit 5.

(k) COMBEN - this file contains the externally input data for example 3 in Sect. 8.3. A listing of this file is presented in Fig. C.6. The file is referenced by I/O unit 5.

(l) ROUTINE1 TO ROUTINE8 - these files contain the condition and action subroutines of the decision tables compiled in Appendix B.

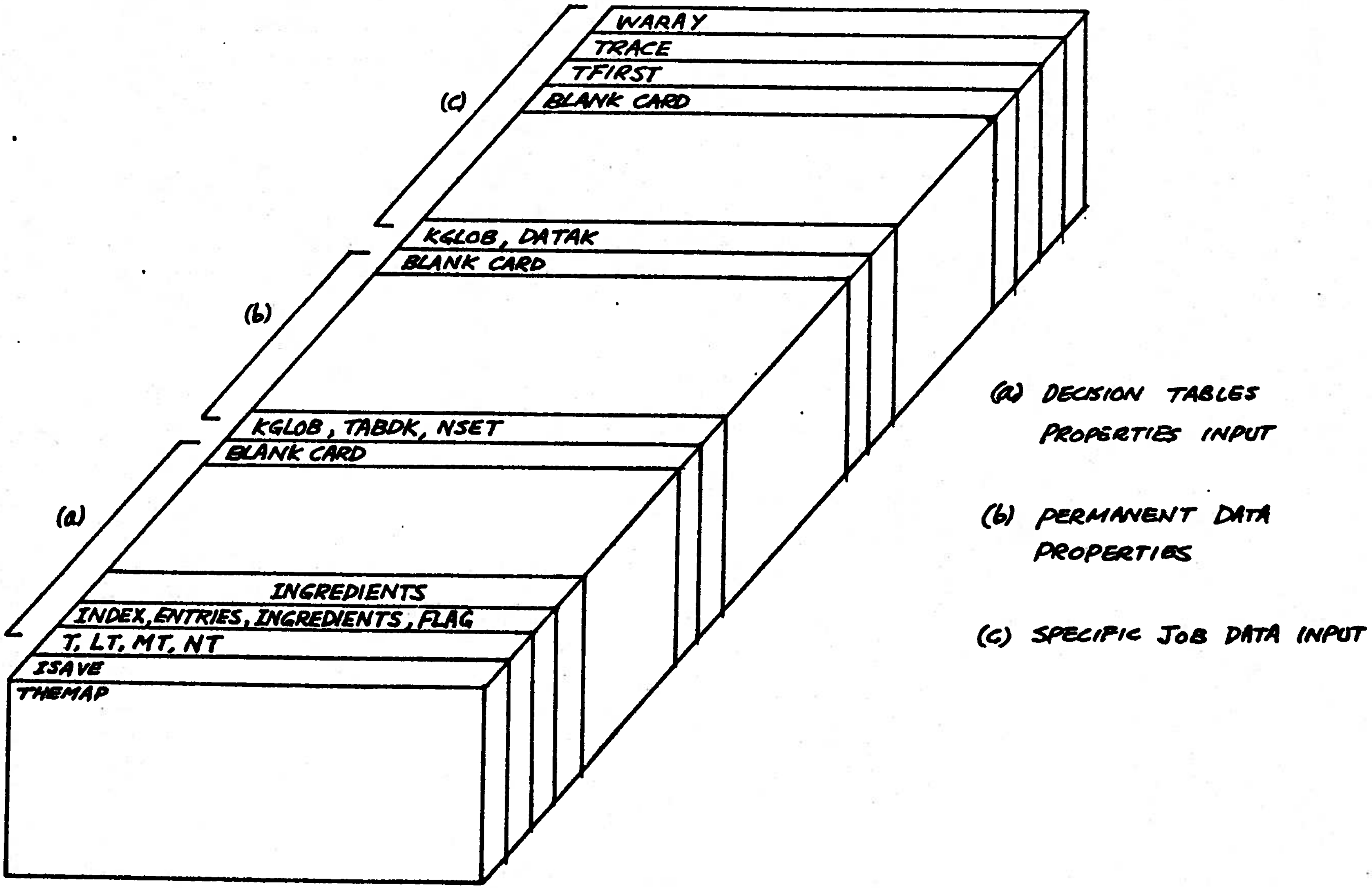


FIG. C.1 ARRANGEMENT OF DATA INPUT CARDS FOR BATCH MODE

```
$list obcombine
>      1      $CONTINUE WITH OBSPECHK RETURN
>      2      $CONTINUE WITH OBSTAK RETURN
>      3      $CONTINUE WITH OBSETUP RETURN
>      4      $CONTINUE WITH OBINITIAL RETURN
>      5      $CONTINUE WITH OBINPUT RETURN
>      6      $CONTINUE WITH OBSETS RETURN
>      7      $CONTINUE WITH OBOUTPUT RETURN
>      7.1    $CONTINUE WITH OBREADIN RETURN
>      8      $CONTINUE WITH OBROUTINE1 RETURN
>      9      $CONTINUE WITH OBROUTINE2 RETURN
>     10      $CONTINUE WITH OBROUTINE3 RETURN
>     11      $CONTINUE WITH OBROUTINE4 RETURN
>     12      $CONTINUE WITH OBROUTINE5 RETURN
>     13      $CONTINUE WITH OBROUTINE6 RETURN
>     14      $CONTINUE WITH OBROUTINE7 RETURN
>     15      $CONTINUE WITH OBROUTINE8 RETURN
#END OF FILE
```

FIG. C.2 FILE OBCOMBINE

```
$list obcombiner
>      1      $CONTINUE WITH OBSPECHKR RETURN
>      2      $CONTINUE WITH OBSTAK RETURN
>      3      $CONTINUE WITH OBSETUP RETURN
>      4      $CONTINUE WITH OBINITIAL RETURN
>      5      $CONTINUE WITH OBINPUT RETURN
>      6      $CONTINUE WITH OBSETS RETURN
>      7      $CONTINUE WITH OBOUTPUT RETURN
>      8      $CONTINUE WITH OBREADIN RETURN
>      8.1    $CONTINUE WITH OBCLEAR RETURN
>      9      $CONTINUE WITH OBROUTINE1 RETURN
>     10      $CONTINUE WITH OBROUTINE2 RETURN
>     11      $CONTINUE WITH OBROUTINE3 RETURN
>     12      $CONTINUE WITH OBROUTINE4R RETURN
>     13      $CONTINUE WITH OBROUTINE5R RETURN
>     14      $CONTINUE WITH OBROUTINE6 RETURN
>     15      $CONTINUE WITH OBROUTINE7 RETURN
>     16      $CONTINUE WITH OBROUTINE8R RETURN
#END OF FILE
```

FIG. C.3 FILE OBCOMBINER

```
$list lub
>     1      1,
>     2      1,     1.0,
>     3      10,     1.0,
>     4      22,     1.0,
>     5      26,     1.0,
>     6      29,     1.0,
>     7      40,     1.0,
>     8      47,     4.05,
>     9      48,     5.18,
>    10      58,     0.641,
>    11      59,     4.04,
>    12      64,     0.371,
>    13      65,     12.19,
>    14      76,     44.0,
>    15      78,     29000.0,
>    16      79,     72.5,
>    17      80,     64.8,
>    18      119,     1.0,
>    19      121,     1.0,
>    20      122,     0.0,
>    21      123,     1.0,
>    22      124,     0.0,
>    23      127,     0.0,
>    24      128,     1.0,
>    25      129,     0.0,
>    26      130,     1.0,
>    27      138,     3190.0,
>    28      169,     1.0,
>    29      176,     0.0,
>    30      177,     1.0,
>    31      180,     0.0,
>    32      187,     288.0,
>    33      189,     2.2,
>    34      197,     0.9,
>    35      225,     1.0,
>    36      320,     1.0,
>    37      342,     1.0,
>    38      334,     0.0,
>    39      400,     0.9,
>    40      401,     1.25,
>    41      402,     1.5,
>    42      403,     1.5,
>    43      404,     1.25,
>    44      405,     1.0,
>    45      410,     1036.8,
>    46      411,     518.4,
>    47      412,     0.0,
>    48      413,     0.0,
>    49      418,     14.4,
>    50      419,     7.2,
>    51      420,     0.0,
>    52      421,     0.0,
>    53      125,     0.0,
>    54      0,
>    55                YES
>    56                YES
#END OF FILE
```

FIG. C.4 DATA FILE LUB

```
$list axcom
>     1     1,
>     2     1,      1.0,
>     3     10,      1.0,
>     4     23,      1.0,
>     5     25,      0.0,
>     6     46,      16.7,
>     7     58,      0.63,
>     8     59,      8.0,
>     9     76,      44.0,
>    10     78,      29000.0,
>    11     119,      1.0,
>    12     121,      0.0,
>    13     122,      1.0,
>    14     123,      1.0,
>    15     124,      1.0,
>    16     125,      1.0,
>    17     166,      1.0,
>    18     179,      1.0,
>    19     186,      144.0,
>    20     188,      2.42,
>    21     197,      0.9,
>    22     208,      2.54,
>    23     216,      144.0,
>    24     400,      0.9,
>    25     401,      1.25,
>    26     402,      1.5,
>    27     403,      1.5,
>    28     404,      1.25,
>    29     405,      1.0,
>    30     406,      300.0,
>    31     407,      85.0,
>    32     408,      0.0,
>    33     409,      0.0,
>    34     220,      1.0,
>    35     221,      1.0,
>    36     0,
>    37                YES
>    38                YES
#END OF FILE
```

FIG. C.5 DATA FILE AXCOM

```
$list comben
>   1,
>   1,     1.0,
>   10,    1.0,
>   23,    1.0,
>   25,    0.0,
>   26,    1.0,
>   28,    1.0,
>   40,    1.0,
>   46,    14.4,
>   48,    5.58,
>   58,    0.558,
>   59,    5.0,
>   63,    8.88,
>   64,    0.34,
>   76,    44.0,
>   78,    29000.0,
>   79,    60.3,
>   80,    54.6,
>   119,   1.0,
>   121,   0.0,
>   122,   0.0,
>   123,   0.0,
>   124,   1.0,
>   125,   1.0,
>   127,   1.0,
>   128,   1.0,
>   129,   1.0,
>   138,   2490.39,
>   169,   1.0,
>   177,   1.0,
>   187,   114.0,
>   188,   2.54,
>   189,   2.77,
>   190,   633.6,
>   197,   0.9,
>   208,   4.35,
>   220,   1.0,
>   221,   1.0,
>   226,   1.0,
>   290,   1.0,
>   320,   1.0,
>   342,   1.0,
>   400,   0.9,
>   401,   1.25,
>   402,   1.5,
>   403,   1.5,
>   404,   1.25,
>   405,   1.0,
>   406,   90.0,
>   407,   21.0,
>   408,   0.0,
>   409,   0.0,
>   410,   0.0,
>   411,   1067.0,
>   412,   0.0,
>   413,   0.0,
>   414,   0.0,
>   415,   890.0,
>   416,   0.0,
>   417,   0.0,
>   186,   114.0,
>   216,   114.0,
>   180,   0.0,
>   130,   1.0,
>   178,   1.0,
>   222,   0.0,
>   223,   0.0,
>   176,   0.0,
>   65,    10.0,
>   323,   0.0,
>   324,   1.0,
>   0,
>                YES
>                YES
#END OF FILE
```


C.6 DATA FILE COMBEN

APPENDIX D

INTERACTIVE MODE PROCESSING PROGRAM - SOURCE LISTING

This appendix presents a source listing of the Main routine and the subroutines of the processing program for interactive mode procedure. In order to conserve space, the condition and action subroutines of each decision table are not presented. An example of them is presented in Fig. 5.5.

```
$LIST SPECHK
CC     THIS IS THE MAIN SUBROUTINE DOING THE BULK OF DECISION TABLE
CC     PROCESSING INCLUDING IDENTIFYING THE APPLICABLE RULE,
CC     AND CHECKING THE PRESENCE OF DATA ETC.
C
CC     DECLARATIONS
C
       IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
       INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,THPSET
       COMMON /MICA/DATA,PRD
       COMMON /MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
      1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
       COMMON /MNSTIN/ISET,MEXSET,MARCA,TRACE,THEMAP
       COMMON/STUPIN/ICLEAR,IARROW
       COMMON/STINTL/INDEX,ENTRY,INGR,IDEPND,THPSET
       COMMON/MINCE/ICYCLE
       COMMON/DUMB/WARAY
       DIMENSION
      1 LARRY1(600),LARRY2(600),LARRY3(5000),LARRY4(5000),IBASE(120,4),
      2 LARRY5( 600),LARRY6( 600),IPNTRC(600),IPNTRA(600),
      3 DATA(700),PRD(700),TABD(700),ISET(700),
      4L(120),M(120),N(120),STACK(20,5),RESULT(2)
       DIMENSION MEXSET(150),MARCA(100),IARROW(700),ICLEAR(2000)
       DATA YES/'YES'/,NO/'NO'/
C      READ IF A MAP OF PERMANENT DATA STORAGE IS DESIRED OR NOT
       READ(2,110) THEMAP
   110 FORMAT(T11,A4)
C      IF ISAVE=1,ENTER SUBROUTINE SETUP,CALCULATE DECISION TABLES
C      PERMANENT DATA,THEN STORE IN UNIT 8
C      IF ISAVE=2,SKIP SUBROUTINE SETUP,READ DECISXON TABLES
C      PERMANENT DATA FROM UNIT 8
       READ(2,100) ISAVE
   100 FORMAT(I10)
       CALL SETDSR(8,11000,11000)
       GO TO (700,710),ISAVE
   710 CALL INTIAL
       READ(8) LARRY1,LARRY2,LARRY5,LARRY6,IPNTRC,IPNTRA
       READ(8) LARRY3
       READ(8) LARRY4
       READ(8) ISET,TABD,L,M,N,TFIRST,IBASE
       READ(8) MEXSET,MARCA,TRACE,ICLEAR,IARROW
C      WHEN ISAVE=2
C      READ TFIRST FROM UNIT 5 AGAIN IN CASE IT IS DIFFERENT
C      THAN THAT READ FROM UNIT 8
       READ(5,102) TFIRST
   102 FORMAT(I10)
C      IF WARAY EQUAL YES,WRITE THE ARRAYS IF STATMENT 900 TO 928
       IF (WARAY.NE.YES) GO TO 730
       WRITE (6,900) (LARRY1(I),I=1,600)
   900 FORMAT (1H1,5X,'LARRY1'//30(5X,10I5,5X,10I5/))
       WRITE (6,902) (LARRY3(I),I=1,5000)
   902 FORMAT (1H1,5X,'LARRY3'//50(5X,10(10I1,1X)/))
       WRITE (6,904) (LARRY2(I),I=1,600)
   904 FORMAT (1H1,5X,'LARRY2'//30(5X,10I5,5X,10I5/))
       WRITE (6,906) (LARRY4(I),I=1,5000)
   906 FORMAT (1H1,5X,'LARRY4'//50(5X,10(10I1,1X)/))
       WRITE (6,908) (J,(IBASE(I,J),I=1,120),J=1,4)
   908 FORMAT (1H1/4(5X,'IBASE',I1//6(5X,10I5,5X,10I5/)////))
       WRITE(6,910) (LARRY5(I),I=1,600)
   910 FORMAT (1H1,5X,'LARRY5'//30(5X,10I5,5X,10I5/))
       WRITE (6,912) (LARRY6(I),I=1,600)
   912 FORMAT (1H1,5X,'LARRY6'//30(5X,10I5,5X,10I5/))
       WRITE (6,914) (IPNTRC(I),I=1,600)
   914 FORMAT (1H1,5X,'IPNTRC'//30(5X,10I5,5X,10I5/))
       WRITE (6,916) (IPNTRA(I),I=1,600)
   916 FORMAT (1H1,5X,'IPNTRA'//30(5X,10I5,5X,10I5/))
       WRITE (6,918) (IARROW(I),I=1,700)
   918 FORMAT (1H1,5X,'IARROW'/35(5X,10I5,5X,10I5/))
       WRITE (6,920) (ICLEAR(I),I=1,2000)
   920 FORMAT (1H1,5X,'ICLEAR'//20(5(5X,10I5,5X,10I5/)/))
       WRITE (6,922) (MEXSET(I),I=1,150)
   922 FORMAT (1H1,20X,'MEXSET'/15(20X,10I5/))
       WRITE (6,924) (MARCA(I),I=1,100)
   924 FORMAT (1H0,20X'MARCA'/5(20X,10I5/))
       WRITE(6,926) (ISET(I),I=1,700)
```

```
  926 FORMAT(1H1,5X,'ISET'/35(5X,10I5,5X,10I5/))
      WRITE(6,928)  (L(I),I=1,120)
  928 FORMAT(1H1,20X,'ARRAY L'/12(20X,10I5/))
      GO TO 730
  700 CALL SETUP
  730 ICYCLE = 1
    1  CALL INPUT(ICYCLE)
C     PRINT DATA AGAIN TO CHECK ITS VALIDITY
      WRITE (6,799)
  799 FORMAT (1H1,15X,'DATA PRINTED AGAIN FOR CHECKING. ONLY ',
     1 'THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE'//
     2 31X,'KGLOB',10X,'DATAK',10X,'PRD'//)
      DO 801 KGLOB = 1,700
      IF (.NOT. PRD(KGLOB)) GO TO 801
      WRITE (6,800) KGLOB,DATA(KGLOB),PRD(KGLOB)
  800 FORMAT (25X,I10,1X,F14.4,5X,L7)
  801 CONTINUE
C
CC    GIVE A MESSAGE THAT EXECUTION OF CYCLE NUMBER (= ICYCLE) IS ABOUT
CC    TO COMMENCE
C
      WRITE (6,172) ICYCLE,TFIRST
  172 FORMAT (1H1,10X,'CYCLE NUMBER',I3,  5X,'***   START',
     1 ' EXECUTION WITH TABLE',I4,'    ***')
C     INITIALISE BEFORE STARTING EXECUTION OF THE TABLES
      TRUE = 1.0
      T = TFIRST
      ISTACK = 0
      TABNO = 0
   18 J = 1
   19 I = 1
C     GET THE ADDRESS OF THE CONDITION ENTRY
   24 IJ = IBASE(T,3) + (J-1)*N(T) + I
C     IF THE CONDITION ENTRY IS IMMATERIAL, SKIP CHECKING IT
      IF (LARRY3(IJ)  .EQ. 0) GO TO 52
   25 I1 = IBASE(T,1) + I
      KGLOB = LARRY1(I1)
C
CC    FIRST CHECK IF THE CONDITION HAS BEEN SUPPLIED WITH ITS VALUE
C
 2006 IF (PRD(KGLOB)) GO TO 50
C
CC    CHECK IF ANY OTHER TABLE CAN BE EXECUTED TO GET THIS CONDITION
C
      IF (TABD(KGLOB) .NE. 0) GO TO 45
C
CC    OTHERWISE SEE IF THIS CONDITION CAN BE ESTABLISHED BY
CC    SUBROUTINE CC OF THIS TABLE. THIS IS INDICATED BY
CC    HAVING ATLEAST ONE INGREDIENT FOR THIS CONDITION
C
      IF ((IPNTRC(I1+1) - IPNTRC(I1)) .NE. 0) GO TO 13
C
CC    ELSE AN ERROR MESSAGE
C
   12 WRITE (7,174) I,T,KGLOB
  174 FORMAT (1H0,10X,'CONDITION NUMBER',I3,'OF TABLE ',I3,
     1 'IS NOT AVAILABLE. THIS CORRESPONDS TO DATA NUMBER',I4/
     2 11X,'SUBROUTINE READIN IS CALLED TO INPUT THIS DATA ITEM')
      NUMBER=1
 2000 KG=KGLOB
      CALL READIN(KG,NUMBER,KGLOB)
      IF(KG.NE.KGLOB) GO TO 2004
      GO TO 2006
 2004 WRITE(7,1000)
 1000 FORMAT(1X,'YOU HAVE INPUT THE INCORRECT VALUE OF KG',
     * ' PLEASE TRY AGAIN.GOOD LUCK')
      GO TO 2000
C
CC    CHECK IF ALL THE INGREDIENTS OF THIS CONDITION ARE PRESENT
C
   13 IR = IPNTRC(I1) + 1
   23 IDATA = LARRY5(IR)
```

```
 2007 IF (.NOT. PRD(IDATA)) GO TO 39
   15 IR = IR + 1
      IF (IR .LE. IPNTRC(I1+1)) GO TO 23
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT ALL THE
CC    DATA NECESSARY TO SET THIS CONDITION IS PRESENT AND SO
CC    ITS SUBROUTINE CC CAN BE CALLED
C
      GO TO(888,888,888,888,888,888,888,888,209,210,
     *      211,212,213,214,215,216,217,218,219,888,
     *      221,222,223,224,225,888,227,888,229,888,
     *      231,888,888,888,888,888,888,888,888,888,
     *      888,888,888,244,245,888,888,248,249,888,
     5      888,888,888,888,888,888,888,258,259,888,
     6      261,888,263,264,265,888,888,888,888,888,
     7      888,272,888,274,888,276,277,888,279,888,
     8      281,282,888,284,888,286,287,888,888,290,
     9      291,888,888,888,888,888,297,888,888,888,
     A      888,888,888,888,888,888,888,888,888,888,
     B      888,888,888,888,888,888,888,888,888,888),T
  209 CALL CC9(I)
      GO TO 41
  210 CALL CC10(I)
      GO TO 41
  211 CALL CC11(I)
      GO TO 41
  212 CALL CC12(I)
      GO TO 41
  213 CALL CC13(I)
      GO TO 41
  214 CALL CC14(I)
      GO TO 41
  215 CALL CC15(I)
      GO TO 41
  216 CALL CC16(I)
      GO TO 41
  217 CALL CC17(I)
      GO TO 41
  218 CALL CC18(I)
      GO TO 41
  219 CALL CC19(I)
      GO TO 41
  221 CALL CC21(I)
      GO TO 41
  222 CALL CC22(I)
      GO TO 41
  223 CALL CC23(I)
      GO TO 41
  224 CALL CC24(I)
      GO TO 41
  225 CALL CC25(I)
      GO TO 41
  227 CALL CC27(I)
      GO TO 41
  229 CALL CC29(I)
      GO TO 41
  231 CALL CC31(I)
      GO TO 41
  244 CALL CC44(I)
      GO TO 41
  245 CALL CC45(I)
      GO TO 41
  248 CALL CC48(I)
      GO TO 41
  249 CALL CC49(I)
      GO TO 41
  258 CALL CC58(I)
      GO TO 41
  259 CALL CC59(I)
      GO TO 41
  261 CALL CC61(I)
      GO TO 41
  263 CALL CC63(I)
      GO TO 41
  264 CALL CC64(I)
      GO TO 41
```

```
      265 CALL CC65(I)
          GO TO 41
      272 CALL CC72(I)
          GO TO 41
      274 CALL CC74(I)
          GO TO 41
      276 CALL CC76(I)
          GO TO 41
      277 CALL CC77(I)
          GO TO 41
      279 CALL CC79(I)
          GO TO 41
      281 CALL CC81(I)
          GO TO 41
      282 CALL CC82(I)
          GO TO 41
      284 CALL CC84(I)
          GO TO 41
      286 CALL CC86(I)
          GO TO 41
      287 CALL CC87(I)
          GO TO 41
      290 CALL CC90(I)
          GO TO 41
      291 CALL CC91(I)
          GO TO 41
      297 CALL CC97(I)
          GO TO 41
      888 WRITE (6,1888) T
     1888 FORMAT (1H0,10X,'ERROR SITUATION. ATTEMPT TO CALL SUBROUTINE CC',
         1 I3,'WHERE IT IS NOT SUPPOSED TO BE SO'/11X,'NO SUCH ',
         2 'SUBROUTINE EXISTS')
          WRITE(7,1889) T
     1889 FORMAT(10X,'ERROR SITUATION. ATTEMPT TO CALL SUBROUTINE CC',
         * I3,'WHERE IT IS NOT SUPPOSED TO BE SO'/11X,'NO SUCH',
         * 'SUBROUTINE EXISTS')
          GO TO 76
C         CHECK IF THE CONDITION HAS GOT ITS VALUE NOW
CD 41 WRITE(6,444) KGLOB,DATA(KGLOB),PRD(KGLOB)
CD444 FORMAT(1X,'DATA CHECK JUST BEFORE STATEMENT 41 IN SPECHK'//
CD    120X,I10,1X,F14.4,5X,L7)
       41 IF (PRD(KGLOB)) GO TO 50
          GO TO 12
C
CC        FOLLOWING IS A CHECK WHETHER THE MISSING INGREDIENT
CC        IS OBTAINABLE BY EXECUTING ANY TABLE
C
       39 IF (TABD(IDATA) .NE. 0) GO TO 46
C
CC        WRITE THE ERROR MESSAGE THAT THIS DATA IS NOT AVAILABLE
C
          WRITE (7,112) IDATA
      112 FORMAT (1H0,10X,'ERROR MESSAGE; DATA NUMBER',I5/
         1'IS NOT AVAILABLE.THIS IS AN INGREDIENT OF A CONDITION')
          NUMBER=2
     2008 KG=IDATA
          CALL READIN(KG,NUMBER,IDATA)
          IF(KG.NE.IDATA) GO TO 2010
          GO TO 2007
     2010 WRITE(7,1020)
     1020 FORMAT(1X,'YOU HAVE INPUT THE INCORRECT VALUE OF KG',
         * ' PLEASE TRY AGAIN.GOOD LUCK')
          GO TO 2008
       46 TABNO = TABD(IDATA)
C
CC        START THE STACKING PROCEDURE TO EXECUTE THE APPROPRIATE TABLE
CC        THE VALUE OF IFLAG = 1 INDICATES THAT THE STACKING IS REQUIRED
CC        BECAUSE OF SOME MISSING INGREDIENT OF A CONDITION
C
          IFLAG = 1
          CALL STAK(STACK,ISTACK,IFLAG,T,I,J,IR,TABNO,IDATA,TRACE)
          GO TO 18
C
CC        START THE STACKING PROCEDURE TO EXECUTE THE APPROPRIATE TABLE
```

```
CC     THE VALUE OF IFLAG = 2 INDICATES THAT THE STACKING
CC     IS REQUIRED BECAUSE THE MISSING CONDITION IS OBTAINABLE
CC     BY EXECUTING SOME OTHER TABLE
C
    45 TABNO = TABD(KGLOB)
       IFLAG = 2
       IDATA = 0
       CALL STAK(STACK,ISTACK,IFLAG,T,I,J,KGLOB,TABNO :DATA,TRACE)
       GO TO 18
C
CC     MATCH THE RULE
C
    50 IF (LARRY3(IJ) .EQ. 1 .AND. DATA(KGLOB) .NE. TRUE) GO TO 55
       IF (LARRY3(IJ) .EQ. 2 .AND. DATA(KGLOB) .EQ. TRUE) GO TO 55
C
CC     CONTINUE MATCHING CONDITIONS IN THIS RULE
C
CD 52 WRITE(6,440) I
CD440 FORMAT(1X,'AT STATEMENT 52 IN SPECHK,THE CONDITION NO. IS',
CD    *5X,I3)
    52 IF (I .EQ. N(T)) GO TO 31
       I = I+1
       GO TO 24
C      CONTINUE THE SEARCH WITH THE NEXT RULE
    55 IF (J.EQ. L(T)) GO TO 30
       J = J+1
CD     WRITE(6,445) J
CD445 FORMAT(1X,'AFTER STATEMENT 55 IN SPECHK,THE RULE NO. IS',5X,I3)
       GO TO 19
C
CC     MESSAGE FOR UNSUCCESSFUL MATCH IN THE TABLE
C
    30 WRITE (6,130) T
   130 FORMAT (1H0,10X,'NO RULE IN TABLE',I4,' IS MATCHING ',
      1 'THE CONDITION STUB'/11X,' CYCLE TERMINATED')
       WRITE (7,132) T
   132 FORMAT(10X,'NO RULE IN TABLE',I4,'IS MATCHING ',
      * 'THE CONDITION STUB'/11X,' CYCLE TERMINATED')
       GO TO 76
C
CC     THE APPLICABLE RULE HAS BEEN IDENTIFIED
CC     CHECK IF THIS INFORMATION IS DESIRED TO BE PRINTED OUT
C
    31 IF (TRACE .NE. YES) GO TO 57
       WRITE (6,177) T,J
   177 FORMAT (1H0,15X,'SCANNING OF TABLE ',I3,' IS COMPLETE. RULE ',
      1 'NO.',I3,' APPLIES')
C
CC     NOW FIND WHICH ACTION ENTRY IS APPLICABLE
CC     CODE: 0 FOR NO ACTION, 1 FOR CONDITIONAL EVALUATION,
CC     2 FOR DIRECT EXECUTION ACTIONS AND 3 FOR THE ELSE RULE
C
    57 K = 1
    58 KJ = IBASE(T,4)  +  (J-1)*M(T) + K
       IFLAG = LARRY4(KJ) + 2
       K1 = IBASE(T,2) + K
       GO TO (49,59,69,79,89),IFLAG
C
CC     THE FOLLOWING IS A DUMMY STATEMENT AND SHOULD NEVER BE REACHED
C
    49 STOP
C
CC     CONDITIONAL EVALUATION
C
    69 KGLOB = LARRY2(K1)
C
CC     BEFORE CALLING SUBROUTINE AA OF THIS TABLE, CHECK IF
CC     THERE ARE ANY INGREDIENTS FOR THIS ACTION.
CC     IF YES, THEN CHECK THEIR PRESENCE
C
       IF ((IPNTRA(K1+1) - IPNTRA(K1)) .EQ. 0) GO TO 93
       IR = IPNTRA(K1) + 1
    27 IDATA = LARRY6(IR)
  2011 IF (.NOT. PRD(IDATA)) GO TO 35
    16 IR = IR + 1
       IF (IR .LE. IPNTRA(K1+1)) GO TO 27
```

```
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT ALL THE
CC    INGREDIENTS NEEDED FOR EVALUATING THIS ACTION ARE PRESENT
CC    AND SO SUBROUTINE AA FOR THIS TABLE CAN BE CALLED
C
   93 GO TO (999,999,503,504,505,999,507,508,509,510,
     1       511,512,513,514,515,516,999,518,519,999,
     2       521,522,523,524,525,526,527,528,529,530,
     3       531,532,999,999,999,999,999,999,999,999,
     4       999,542,543,544,545,546,547,548,549,550,
     5       551,552,553,554,555,999,557,558,559,560,
     6       561,562,563,564,565,566,567,568,569,570,
     7       571,572,573,574,575,576,577,578,579,580,
     8       581,582,999,584,585,586,587,588,589,590,
     9       591,592,593,594,595,596,597,598,999,999,
     A       999,999,999,999,999,999,999,999,999,999,
     B       999,999,999,999,999,999,999,999,999,999) ,T
  503 CALL AA3(K)
      GO TO 60
  504 CALL AA4(K)
      GO TO 60
  505 CALL AA5(K)
      GO TO 60
  507 CALL AA7(K)
      GO TO 60
  508 CALL AA8(K)
      GO TO 60
  509 CALL AA9(K)
      GO TO 60
  510 CALL AA10(K)
      GO TO 60
  511 CALL AA11(K)
      GO TO 60
  512 CALL AA12(K)
      GO TO 60
  513 CALL AA13(K)
      GO TO 60
  514 CALL AA14(K)
      GO TO 60
  515 CALL AA15(K)
      GO TO 60
  516 CALL AA16(K)
      GO TO 60
  518 CALL AA18(K)
      GO TO 60
  519 CALL AA19(K)
      GO TO 60
  521 CALL AA21(K)
      GO TO 60
  522 CALL AA22(K)
      GO TO 60
  523 CALL AA23(K)
      GO TO 60
  524 CALL AA24(K)
      GO TO 60
  525 CALL AA25(K)
      GO TO 60
  526 CALL AA26(K)
      GO TO 60
  527 CALL AA27(K)
      GO TO 60
  528 CALL AA28(K)
      GO TO 60
  529 CALL AA29(K)
      GO TO 60
  530 CALL AA30(K)
      GO TO 60
  531 CALL AA31(K)
      GO TO 60
  532 CALL AA32(K)
      GO TO 60
  542 CALL AA42(K)
      GO TO 60
  543 CALL AA43(K)
```

```
      GO TO 60
  544 CALL AA44(K)
      GO TO 60
  545 CALL AA45(K)
      GO TO 60
  546 CALL AA46(K)
      GO TO 60
  547 CALL AA47(K)
      GO TO 60
  548 CALL AA48(K)
      GO TO 60
  549 CALL AA49(K)
      GO TO 60
  550 CALL AA50(K)
      GO TO 60
  551 CALL AA51(K)
      GO TO 60
  552 CALL AA52(K)
      GO TO 60
  553 CALL AA53(K)
      GO TO 60
  554 CALL AA54(K)
      GO TO 60
  555 CALL AA55(K)
      GO TO 60
  557 CALL AA57(K)
      GO TO 60
  558 CALL AA58(K)
      GO TO 60
  559 CALL AA59(K)
      GO TO 60
  560 CALL AA60(K)
      GO TO 60
  561 CALL AA61(K)
      GO TO 60
  562 CALL AA62(K)
      GO TO 60
  563 CALL AA63(K)
      GO TO 60
  564 CALL AA64(K)
      GO TO 60
  565 CALL AA65(K)
      GO TO 60
  566 CALL AA66(K)
      GO TO 60
  567 CALL AA67(K)
      GO TO 60
  568 CALL AA68(K)
      GO TO 60
  569 CALL AA69(K)
      GO TO 60
  570 CALL AA70(K)
      GO TO 60
  571 CALL AA71(K)
      GO TO 60
  572 CALL AA72(K)
      GO TO 60
  573 CALL AA73(K)
      GO TO 60
  574 CALL AA74(K)
      GO TO 60
  575 CALL AA75(K)
      GO TO 60
  576 CALL AA76(K)
      GO TO 60
  577 CALL AA77(K)
      GO TO 60
  578 CALL AA78(K)
      GO TO 60
  579 CALL AA79(K)
      GO TO 60
  580 CALL AA80(K)
      GO TO 60
  581 CALL AA81(K)
      GO TO 60
  582 CALL AA82(K)
      GO TO 60
  584 CALL AA84(K)
      GO TO 60
  585 CALL AA85(K)
      GO TO 60
  586 CALL AA86(K)
      GO TO 60
  587 CALL AA87(K)
      GO TO 60
  588 CALL AA88(K)
      GO TO 60
  589 CALL AA89(K)
      GO TO 60
  590 CALL AA90(K)
      GO TO 60
  591 CALL AA91(K)
      GO TO 60
  592 CALL AA92(K)
      GO TO 60
  593 CALL AA93(K)
      GO TO 60
  594 CALL AA94(K)
      GO TO 60
  595 CALL AA95(K)
```

```
      GO TO 60
  596 CALL AA96(K)
      GO TO 60
  597 CALL AA97(K)
      GO TO 60
  598 CALL AA98(K)
      GO TO 60
  999 WRITE (6,1999) T
 1999 FORMAT (1H0,10X,'ERROR SITUATION. ATTEMPT TO CALL SUBROUTINE AA',
     1 I3,'WHERE IT IS NOT SUPPOSED TO BE SO'/11X,'NO SUCH ',
     2 'SUBROUTINE EXISTS')
      WRITE (7,3000) T
 3000 FORMAT(10X,'ERROR SITUATION. ATTEMPT TO CALL SUBROUTINE AA',
     * I3,'WHERE IT IS NOT SUPPOSED TO BE SO'/11X,'NO SUCH ',
     *' SUBROUTINE EXISTS')
      GO TO 76
C     CHECK IF THIS ACTION IS COMPLETE
   60 IF (KGLOB .EQ. 0) GO TO 59
      IF (PRD(KGLOB)) GO TO 56
C
CC    ERROR MESSAGE
C
      WRITE (6,141) K,T,J
  141 FORMAT (1H0,10X,'ACTION NUMBER',I3,' OF TABLE NUMBER',I3,
     1 ' CAN NOT BE COMPLETED.'/5X,' THE CURRENT RULE NUMBER IS',I3,
     2 ' FURTHER EXECUTION WILL HAVE TO STOP')
      WRITE (7,143) K,T,J
  143 FORMAT(10X,'ACTION NUMBER',I3,' OF TABLE NUMBER',I3,
     * ' CAN NOT BE COMPLETED.'/5X,' THE CURRENT RULE NUMBER IS',I3,
     * ' FURTHER EXECUTION WILL HAVE TO STOP')
      GO TO 76
C
CC    THE FOLLOWING APPLIES IF THE MISSING INGREDIENT OF THE ACTION
CC    IS ADDRESSED TO SOME TABLE FROM WHICH IT CAN BE RETRIEVED
C
   35 IF (TABD(IDATA) .NE. 0) GO TO 36
C
CC    ERROR MESSAGE
C
      WRITE (7,181)K,T,IDATA
  181 FORMAT (1H0,15X,'ACTION NUMBER',I3,' OF TABLE ',I3,
     1 ' CANNOT BE COMPLETED BECAUSE DATA NUMBER',I3,' IS NOT ',
     2 'PRESENT.'/16X,'SUBROUTINE READIN IS CALLED')
      NUMBER=3
 2012 KG=IDATA
      CALL READIN(KG,NUMBER,IDATA)
      IF(KG.NE.IDATA) GO TO 2014
      GO TO 2011
 2014 WRITE(7,1040)
 1040 FORMAT(1X,'YOU HAVE INPUT THE INCORRECT VALUE OF KG',
     * ' PLEASE TRY AGAIN.GOOD LUCK')
      GO TO 2012
C
CC    OBTAIN THE MISSING INGREDIENT BY EXECUTING THE TABLE TABD(IR)
C
   36 TABNO = TABD(IDATA)
C
CC    STACK-UP BEFORE STARTING EXECUTION OF ANOTHER TABLE
C
      CALL STAK(STACK,ISTACK,IFLAG,T,K,J,IR,TABNO,IDATA,TRACE)
      GO TO 18
C
CC    THE FOLLOWING APPLIES IN CASE OF DIRECT EXECUTION COMMANDS
CC    THE ADDRESS OF THE TABLE WHICH IS DESIRED TO BE EXECUTED
CC    IS AVAILABLE AT LARRY2(K1)
CC    AT LARRY2(K1)
C
   79 TABNO = LARRY2(K1)
      IR = 0
      IDATA = 0
      CALL STAK(STACK,ISTACK,IFLAG,T,K,J,IR,TABNO,IDATA,TRACE)
      GO TO 18
C
```

```
CC    THE FOLLOWING APPLIES IN CASE THE ELSE RULE IS APPLICABLE
C
   89 WRITE (6,189) T
  189 FORMAT (1H0,15X,'ELSE RULE IS APPLICABLE IN TABLE NO.',I4,
     1 '. FURTHER EXECUTION IS NOT POSSIBLE')
      WRITE (7,190) T
  190 FORMAT(15X,'ELSE RULE IS APPLICABLE IN TABLE NO.',I4,
     * ' FURTHER EXECUTION IS NOT POSSIBLE')
      GO TO 76
C     CALL SUBROUTINE SETS TO CHECK IF KGLOB BELONGS
C     TO A MUTUALLY EXCLUSIVE SET.IF IT DOES,SET THE
C     OTHER ELEMENTS OF THE SET TO NO.
   56 CALL SETS(KGLOB)
   59 K = K + 1
      IF (K .GT. N(T)) GO TO 61
      GO TO 58
C
CC    UNSTACKING GOES AS FOLLOWS
C
   61 IF (ISTACK .EQ. 0) GO TO 76
      IFLAG = STACK(ISTACK,1)
      T     = STACK(ISTACK,2)
      J     = STACK (ISTACK,4)
C
CC    IF THE VALUE OF IFLAG IS 1 OR 2, THEN THE STACKING WAS DONE
CC    IN THE CONDITION SECTION; OTHERWISE IN THE ACTION SECTION
C
      GO TO (63,64,65,66),IFLAG
   63 I     = STACK(ISTACK,3)
      IR    = STACK(ISTACK,5)
      GO TO 176
   64 I     = STACK(ISTACK,3)
      KGLOB = STACK(ISTACK,5)
C     CHECK IF A TRACE OF THE UNSTACKING IS DESIRED OR NOT
  176 IF(TRACE .NE. YES) GO TO 67
      WRITE(6,178) T,I,J
  178 FORMAT(1H0,15X,'RESTART EXECUTION OF TABLE ',I3,
     *3X,'AT CONDITION',I3,3X,'OF RULE',I3)
      GO TO 67
   65 IR    = STACK(ISTACK,5)
   66 K     = STACK(ISTACK,3)
C     CHECK IF A TRACE OF THE UNSTACKING IS DESIRED OR NOT
      IF(TRACE .NE. YES) GO TO 67
      WRITE(6,179) T,K,J
  179 FORMAT(1H0,15X,'RESTART EXECUTION OF TABLE ',I3,
     *3X,'AT ACTION',I3,3X,'OF RULE',I3)
   67 ISTACK = ISTACK - 1
C
CC    FOLLOWING IS A CHECK WHETHER THE VALUE OF THE MISSING
CC    INGREDIENT HAS BEEN OBTAINED OR NOT
C
   68 GO TO (70,71,72,59),IFLAG
   70 I1 = IBASE(T,1) + I
      KGLOB = LARRY1(I1)
      IDATA = LARRY5(IR)
      IJ=IBASE(T,3)+(J-1)*N(T)+I
 2020 IF (PRD(IDATA)) GO TO 15
C
CC    ERROR MESSAGE
C
      WRITE (7,116) IDATA,TABD(IDATA),I,T
  116 FORMAT (1H0,10X,'VALUE OF DATA NUMBER',I4,' COULD NOT BE OBTAINED
     1 EVEN BY EXECUTING TABLE NUMBER',I3,/11X,'THIS DATA IS AN ',
     2 'INGREDIENT OF CONDITION NUMBER',I3,' OF TABLE NUMBER',I3/
     3 11X,'SUBROUTINE READIN IS CALLED')
      NUMBER=2
 2016 KG=IDATA
      CALL READIN(KG,NUMBER,IDATA)
      IF(KG.NE.IDATA) GO TO 2018
      GO TO 2020
 2018 WRITE(7,1060)
 1060 FORMAT(1X,'YOU HAVE INPUT THE INCORRECT VALUE OF KG',
     * ' PLEASE TRY AGAIN.GOOD LUCK')
      GO TO 2016
   71 IJ = IBASE(T,3) + (J-1)*N(T) + I
 2026 IF (PRD(KGLOB)) GO TO 50
C
```

```
CC     ERROR MESSAGE
C
       WRITE (7,117) KGLOB,TABD(KGLOB),I,T
   117 FORMAT (1H0,10X,'DATA NUMBER',I5, ' COULD NOT BE ESTABLISHED EVEN
     1 BY EXECUTING TABLE NUMBER',I3/11X,'THIS DATA IS CONDITION ',
     2 'NUMBER',I3,' OF TABLE NUMBER',I3/11X,'SUBROUTINE READIN',
     3  'IS CALLED')
       NUMBER=1
  2022 KG=KGLOB
       CALL READIN(KG,NUMBER,KGLOB)
       IF(KG.NE.KGLOB) GO TO 2024
       GO TO 2026
  2024 WRITE(7,1080)
  1080 FORMAT(1X,'YOU HAVE INPUT THE INCORRECT VALUE OF KG',
     + ' PLEASE TRY AGAIN.GOOD LUCK')
       GO TO 2022
    72 K1 = IBASE(T,2) + K
       KGLOB = LARRY2(K1)
       IDATA = LARRY6(IR)
  2032 IF (PRD(IDATA)) GO TO 16
C
CC     ERROR MESSAGE
C
       WRITE (7,118) IDATA,TABD(IDATA),K,T
   118 FORMAT (1H0,10X,'VALUE OF DATA NUMBER',I4,' COULD NOT BE OBTAINED
     1 EVEN BY EXECUTING TABLE NUMBER',I3,/11X,'THIS DATA IS AN ',
     2 'INGREDIENT OF  ACTION   NUMBER',I3,' OF TABLE NUMBER',I3/
     3 11X,'SUBROUTINE READIN IS CALLED')
       NUMBER=3
  2028 KG=IDATA
       CALL READIN(KG,NUMBER,IDATA)
       IF(KG.NE.IDATA) GO TO 2030
       GO TO 2032
  2030 WRITE(7,1100)
  1100 FORMAT(1X,'YOU HAVE INPUT THE INCORRECT VALUE OF KG',
     + ' PLEASE TRY AGAIN.GOOD LUCK')
       GO TO 2028
    76 CALL OUTPUT(ICYCLE)
       ICYCLE = ICYCLE + 1
       GO TO 1
       END
```

```
      SUBROUTINE SETUP
C
CC    THIS SUBROUTINE READS THE DECISION TABLES AND THE PROPERTIES
CC    OF DATA FOR PERMANENT STORAGE AND STORES THEM IN COMPACTED FORM
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,THPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,NEXSET,MARCA,TRACE,THEMAP
      COMMON /STUPIN/ ICLEAR,IARROW
      COMMON /STINTL/ INDEX,ENTRY,INGR,IDEPND,THPSET
      DIMENSION
     1 LARRY1(600),LARRY2(600),LARRY3(5000),LARRY4(5000),IBASE(120,4),
     2 LARRY5( 600),LARRY6( 600),IPNTRC(600),IPNTRA(600),
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4IARROW(700) ,ICLEAR(2000),
     5L(120),M(120),N(120),STACK(20,5),RESULT(2),
     6INDEX(25),ENTRY(25,40),INGR(25,12),IDEPND(700,100),THPSET(100,20)
      DIMENSION NEXSET(150),MARCA(100)
C
C     INITIALISE THE ARRAYS
C
      CALL INTIAL
      DATA C/'C'/,YES/'YES'/,NO/'NO'/
      IBASE1 = 0
      IBASE2 = 0
      IBASE3 = 0
      IBASE4 = 0
      IBASE5 = 0
      IBASE6 = 0
C
CC    INPUT THE TABLES FIRST FROM DATA SET NUMBER 9
CC    FIRST READ THE TABLE NUMBER AND ITS SIZE
C
      KOUNT = 1
    1 READ (9,101) T,LT,MT,NT
  101 FORMAT (4I10)
C
C     A BLANK CARD SIGNIFIES END OF THE LAST TABLE
C
      IF (T .EQ. 0) GO TO 4
C
CC    CHECK THAT THE TABLE NUMBER HAS NOT EXCEEDED THE DIMENSION OF
CC    ARRAYS L,M,AND N, WHICH IS 120
CC    OTHERWISE GIVE AN ERROR MESSAGE HERE TO THIS EFFECT
C
      IF (T .LE. 120) GO TO 12
      WRITE (6,124) T
  124 FORMAT (20X,'CAREFUL:  YOU ARE EXCEEDING THE DIMENSION OF L,M,N
     * WHICH IS 120.' / 20X, 'CURRENT VALUE OF T IS' ,I4)
      STOP
   12 L(T) = LT
      M(T) = MT
      N(T) = NT
      IF (THEMAP .NE. YES) GO TO 13
C     PRINT OUT THE INFORMATION ON TABLE NUMBER AND ITS SIZE
      WRITE (6,151) T,LT,MT,NT
  151 FORMAT (1H0,10X, 'TABLE NO.',I4,',  LT =',I3,',  MT =',I3,
     1 ',  NT =',I3//)
C
C     NOW READ  THE CONDITION ENTERIES
C
   13 IF (KOUNT .NE. 1) GO TO 14
C
CC    STORE THE DESIGNATION OF THE FIRST TABLE READ IN AS ITABLE
C
      ITABLE = T
      KOUNT = 1000
   14 DO 5 I = 1,NT
      READ (9,102) (INGR(I,J),J=1,6),FLAG,INDEX(I),(ENTRY(I,J),J=1,LT)
  102 FORMAT (T51,5(I4,1X),I4,A1,T1,I5,5X,40I1)
```

```
C
C     CHECK IF THE NEXT CARD IS A CONTINUATION OF THIS CARD
C
      IF (FLAG .NE. C) GO TO 17
      READ (9,103)  (INGR(I,J),J=7,12)
  103 FORMAT (T51,5(I4,1I),I4)
      GO TO 19
C
CC    INCASE THE NEXT CARD IS NOT IN CONTINUATION OF THE PREVIOUS CARD
C
   17 DO 18 J = 7,12
      INGR(I,J) = 0
   18 CONTINUE
   19 IF (THEMAP .NE. YES) GO TO 5
C     PRINT THE CONDITION ENTRIES SIMULTANEOUSLY
      WRITE (6,162)  (INGR(I,J),J=1,12),INDEX(I),(ENTRY(I,J),J=1,LT)
  162 FORMAT (T61,12I5,T11,I5,5X,40I1)
    5 CONTINUE
C
CC    FILL THE CONDITION STUB IN THE LINEAR ARRAY "LARRY1"
CC    FILL THE CONDITION ENTRIES THE LINEAR ARRAY "LARRY3" COLUMN-WISE
C
CC    THE BASE ADDRESSES FOR THIS TABLE IN LARRY1 AND LARRY3 ARE
CC    AVAILABLE AS IBASE(T,1) AND IBASE(T,3) RESPECTIVELY
C
      IBASE(T,1) = IBASE1
      IBASE(T,3) = IBASE3
C     FIRST FILL THE CONDITION ENTRIES IN LARRY3 COLUMN-WISE
      DO 41 J = 1,LT
C     BASE ADDRESS FOR THE COLUMN
      IJJ = IBASE3 + (J-1)*NT
      DO 41 I = 1,NT
      IJ = IJJ + I
C     CHECK THAT IJ IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY3
      IF (IJ .GT. 5000) GO TO 42
      LARRY3(IJ) = ENTRY(I,J)
   41 CONTINUE
C
CC    FILL LARRY1 WITH CONDITION STUB, LARRY5 WITH INGREDIENTS AND
CC    GENERATE THE DEPENDENCE LIST FOR THE CONDITIONS OF THIS TABLE
C
      DO 48 I = 1,NT
      I1 = IBASE1 + I
C     CHECK THAT I1 IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY1
      IF (I1 .GT. 600) GO TO 43
      LARRY1(I1) = INDEX(I)
C
CC    THE INGREDIENT LIST FOR THIS ROW OF CONDITION STUB HAS ITS
CC    BASE ADDRESS STORED AS FOLLOWS
C
      IPNTRC(I1) = IBASE5
      DO 46 J = 1,12
C     IF THERE IS NO INGREDIENT, IT WILL BE INDICATED BY ZERO
      IF (INGR(I,J) .EQ. 0) GO TO 47
      IJ = IPNTRC(I1) + J
C     CHECK THAT THIS IS NOT MORE THAN THE DIMENSION OF LARRY5
      IF (IJ .GT. 600) GO TO 49
      LARRY5(IJ) = INGR(I,J)
C
CC    GENERATE THE DEPENDENCE ARRAY RIGHT HERE
CC    NOTICE THAT INDEX(I) IS DEPENDENT OF INGR(I,J)
C
      KGLOB = INGR(I,J)
C
CC    SEARCH IF INDEX(I) ALREADY EXISTS IN IDEPND(KGLOB,*)
CC    IF NOT THEN PLACE IT IN; OTHERWISE SKIP IT
C
      DO 45 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ. 0) GO TO 44
      IF (IDEPND(KGLOB,NUM) .EQ. INDEX(I)) GO TO 46
      GO TO 45
   44 IDEPND(KGLOB,NUM) = INDEX(I)
      GO TO 46
```

```
   45 CONTINUE
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT THE 100 ELEMENTS
CC    OF IDEPND(KGLOB,*) ARE ALL THERE AND SO ITS DIMENSION
CC    SHOULD BE INCREASED
C
   99 WRITE (6,119) KGLOB
  119 FORMAT (20X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION OF ',
     1 'IDEPND(',I4,',  ). THIS MESSAGE PRINTED BY FORMAT 119' )
      STOP
   46 CONTINUE
   47 IBASE5 = IPNTRC(I1) + J - 1
   48 CONTINUE
C
C     KEEP THE BASE ADDRESS READY FOR THE NEXT TABLE
C
      IBASE1 = IBASE1 + NT
      IBASE3 = IBASE3 + NT*LT
C     READ THE ACTION ENTRIES AND PRINT THEM, IF DESIRED
      IF (THEMAP .NE. YES) GO TO 750
      WRITE (6,153)
  153 FORMAT (1X )
  750 DO 6 K = 1,NT
      READ (9,102) (INGR(K,J),J=1,6),FLAG,INDEX(K),(ENTRY(K,J),J=1,LT)
C     CHECK IF THE NEXT CARD IS A CONTINUATION OF THIS CARD
      IF (FLAG .NE. C) GO TO 25
      READ (9,103) (INGR(K,J),J=7,12)
      GO TO 29
C
CC    INCASE THE NEXT CARD IS NOT IN CONTINUATION OF THE PREVIOUS CARD
C
   25 DO 26 J = 7,12
      INGR(K,J) = 0
   26 CONTINUE
C     PRINT THE ACTION ENTRIES ETC. SIMULTANEOUSLY, IF SO DESIRED
   29 IF (THEMAP .NE. YES) GO TO 6
      WRITE (6,162) (INGR(K,J),J=1,12),INDEX(K),(ENTRY(K,J),J=1,LT)
    6 CONTINUE
C
CC    FILL THE ACTION STUB IN THE LINEAR ARRAY "LARRY2"
CC    FILL THE ACTION ENTRIES IN THE LINEAR ARRAY "LARRY4" COLUMN-WISE
C
CC    THE BASE ADDRESSES FOR THIS TABLE IN LARRY2 AND LARRY4 ARE
CC    AVAILABLE AS IBASE(T,2) AND IBASE(T,4) RESPECTIVELY
C
      IBASE(T,2) = IBASE2
      IBASE(T,4) = IBASE4
C     FIRST FILL THE ACTION ENTRIES IN THE LARRY4 COLUMN-WISE
      DO 51 J = 1,LT
C     BASE ADDRESS FOR THE COLUMN
      IJJ = IBASE4 + (J-1)*NT
      DO 51 K = 1,NT
      KJ = IJJ + K
C     CHECK THAT KJ IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY4
      IF (KJ .GT. 5000) GO TO 52
      LARRY4(KJ) = ENTRY(K,J)
   51 CONTINUE
C
CC    FILL LARRY2 WITH ACTION STUB, LARRY6 WITH INGREDIENTS AND
CC    GENERATE THE DEPENDENCE LIST FOR ACTIONS OF THIS TABLE
C
      DO 58 K = 1,NT
      K1 = IBASE2 + K
C     CHECK THAT K1 IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY2
      IF (K1 .GT. 600) GO TO 53
      LARRY2(K1) = INDEX(K)
C     THE INGREDIENT LIST FOR THIS ROW OF ACTION STUB IS AS FOLLOWS
      IPNTRA(K1) = IBASE6
      DO 56 J = 1,12
C     IF THERE IS NO INGREDIENT, IT WILL BE INDICATED BY ZERO
      IF (INGR(K,J) .EQ. 0) GO TO 57
      KJ = IPNTRA(K1) + J
C     CHECK THAT THIS IS NOT MORE THAN THE DIMENSION OF LARRY6
      IF (KJ .GT. 600) GO TO 59
      LARRY6(KJ) = INGR(K,J)
```

```
C
CC    GENERATE THE DEPENDENCE ARRAY RIGHT HERE
CC    NOTICE THAT INDEX(K) IS A DEPENDENT OF INGR(K,J)
C
      KGLOB = INGR(K,J)
C
CC     SEARCH IF INDEX(K) ALREADY EXISTS IN IDEPND(KGLOB,*)
CC    IF NOT THEN PLACE IT IN; OTHERWISE SKIP IT
C
      DO 55 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ.0) GO TO 54
      IF (IDEPND(KGLOB,NUM) .EQ.INDEX(K)) GO TO 56
      GO TO 55
   54 IDEPND(KGLOB,NUM) = INDEX(K)
      GO TO 56
   55 CONTINUE
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT THE 100 ELEMENTS
CC    OF IDEPND(KGLOB,*) ARE ALL THERE AND SO ITS DIMENSION
CC    SHOULD BE INCREASED
C
      WRITE (6,120) KGLOB
  120 FORMAT (20X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION OF ',
     1 'IDEPND(',I4,',  ). THIS MESSAGE PRINTED BY FORMAT 120' )
      STOP
   56 CONTINUE
   57 IBASE6 = IPNTRA(K1) + J - 1
   58 CONTINUE
C     KEEP THE BASE ADDRESS READY FOR THE NEXT TABLE
      IBASE2 = IBASE2 + NT
      IBASE4 = IBASE4 + NT*LT
C     APPEND LOGICAL DATA TO THE DEPENDENCE LIST
      DO 69 J = 1,LT
      DO 68 K = 1,NT
      IF (ENTRY(K,J) .EQ. 1) GO TO 63
      GO TO 68
C
CC    CHECK IF THIS ACTION STORES ANY VALUE IN ANY LOCATION
C
   63 IF (INDEX(K) .EQ. 0) GO TO 68
C
CC    INDEX(K) IS DEPENDENT OF ALL THE LOGICAL CONDITIONS WHICH ARE
CC    NOT IMMATERIAL FOR THIS RULE
C
      DO 66 I = 1,NT
      IJ = IBASE(T,3) + (J-1)*NT + I
      IF (LARRY3(IJ) .EQ. 0) GO TO 66
C
CC    OTHERWISE SEARCH IF INDEX(K) IS ALREADY IN THE LIST OF
CC    DEPENDENTS OF THIS CONDITION
C
      I1 = IBASE(T,1) + I
      KGLOB = LARRY1(I1)
      DO 65 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ. 0) GO TO 64
      IF (IDEPND(KGLOB,NUM) .EQ. INDEX(K)) GO TO 66
      GO TO 65
   64 IDEPND(KGLOB,NUM) = INDEX(K)
      GO TO 66
   65 CONTINUE
C     A NORMAL EXIT IS AN ERROR AS BEFORE
      WRITE (6,121) KGLOB
  121 FORMAT (20X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION OF ',
     1 'IDEPND(',I4,',  ). THIS MESSAGE PRINTED BY FORMAT 121' )
      STOP
   66 CONTINUE
   68 CONTINUE
   69 CONTINUE
C     TO READ THE NEXT TABLE
      GO TO 1
C
CC    COMPACT THE ARRAY IDEPND"INTO A LINEAR ARRAY AND NAME IT ICLEAR
CC    BECAUSE IT WILL BE USED IN CLEARING THE EFFECT OF CHANGES OF DATA.
CC    EACH ELEMENT IN THE LIST OF DATA WILL HAVE AN ARROW POINTING INTO
CC    "ICLEAR"; THESE ARROWS ARE STORED AS IARROW(KGLOB)
```

```
C
CC    RANGES OF THE TWO DO LOOPS BELOW ARE SAME AS THE
CC    TWO DIMENSIONS OF IDEPND
C
    3 J1 = 0
      DO 80 KGLOB = 1,700
      IARROW(KGLOB) = J1
C
CC    CHECK IF KGLOB BELONGS TO A SET
C
      IF (ISET(KGLOB) .EQ. 0) GO TO 82
      NSET = ISET(KGLOB)
C
CC    CHECK IF KGLOB IS THE FIRST ELEMENT OF THE SET.
CC    IF YES THEN IT GOES INTO ICLEAR; OTHERWISE NOT
C
      I1 = MARCA(NSET) + 1
      IF (NEXSET(I1) .EQ. KGLOB) GO TO 282
C
CC    MAKE THE DEPENDENTS OF KGLOB SAME AS THAT OF THE FIRST
CC    ELEMENT OF THE SET
C
      JGLOB = NEXSET(I1)
      DO 202 NUM = 1,100
      IF (IDEPND(JGLOB,NUM) .EQ. 0) GO TO 80
      IDEPND(KGLOB,NUM) = IDEPND(JGLOB,NUM)
  202 CONTINUE
      GO TO 80
C
CC    FILL THE DEPENDENTS OF THE FIRST ELEMENT OF THE SET WITH
CC    ALL THE POSSIBLE DEPENDENTS INDICATED FOR THE ELEMENTS
CC    OF THE SET
C
  282 N1 = MARCA(NSET) + 2
      N2 = MARCA(NSET + 1)
      DO 220 ID = N1,N2
      NN = NEXSET(ID)
      DO 215 JN = 1,100
      IF (IDEPND(NN,JN) .EQ. 0) GO TO 220
C
CC    SEARCH IF THIS DEPENDENT IS ALREADY IN THE LIST OF
CC    DEPENDENTS OF THE FIRST ELEMENT
C
      DO 210 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ. 0) GO TO 205
      IF (IDEPND(KGLOB,NUM) .EQ. IDEPND(NN,JN)) GO TO 215
      GO TO 210
  205 IDEPND(KGLOB,NUM) = IDEPND(NN,JN)
      GO TO 215
  210 CONTINUE
  215 CONTINUE
  220 CONTINUE
   82 DO 70 J = 1,100
      IF (IDEPND(KGLOB,J) .EQ. 0) GO TO 80
      J1 = J1 + 1
C
CC    CHECK THAT J1 IS NOT MORE THAN THE DIMENSIONED VALUE OF
CC    THE ARRAY "ICLEAR" WHICH CURRENTLY IS 2,000
C
      IF (J1 .GT. 2000) GO TO 33
      ICLEAR(J1) = IDEPND(KGLOB,J)
      GO TO 70
   33 WRITE (6,133)
  133 FORMAT (1H0,10X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION',
     1 ' OF THE ARRAY "ICLEAR" WHICH IS 2000'/10X,'REMEDY IS TO',
     2 ' INCREASE THIS DIMENSION')
      STOP
   70 CONTINUE
   80 CONTINUE
      GO TO 95
C
CC    FILL-UP THE BLANKS IN THE ARRAY  IBASE
C
    4 DO 444 T = 1,120
      IF (IBASE(T,1) .NE. 0 .OR. T .EQ. ITABLE) GO TO 444
```

```
          IBASE(T,1) = IBASE1
          IBASE(T,2) = IBASE2
          IBASE(T,3) = IBASE3
          IBASE(T,4) = IBASE4
      444 CONTINUE
C
CC        FILL-UP THE BLANKS AT THE TAIL END OF THE ARRAY IPNTRC
C
          I1 = I1 + 1
          DO 445 I = I1,600
          IPNTRC(I) = IBASE5
      445 CONTINUE
C
CC        FILL-UP THE BLANKS AT THE TAIL END OF THE ARRAY IPNTRA
C
          K1 = K1 + 1
          DO 446 K = K1,600
          IPNTRA(K) = IBASE6
      446 CONTINUE
C
CC        INPUT PROPERTIES OF THE ELEMENTS OF DATA E.G. ADDRESSES
CC        OF TABLE NUMBERS FROM WHICH THEY CAN BE DERIVED
CC        AND THEIR MEMBERSHIP OF SETS ETC. FROM DATA SET NUMBER 9
C
          IF (THEMAP .NE. YES) GO TO 2
C
CC        HEADING FOR THE NEXT OUTPUT
C
          WRITE (6,154)
      154 FORMAT (1H1,30X, 'KGLOB',10X, 'TABDK',10X, 'NSET'//)
        2 READ (9,105) KGLOB,TABDK,NSET
      105 FORMAT (3I10)
C         A BLANK IS A SIGNAL OF END OF THIS DATA
          IF (KGLOB .EQ.0) GO TO 15
          TABD(KGLOB) = TABDK
          ISET(KGLOB) = NSET
C
CC        PRINT THIS INFORMATION ABOUT THE ELEMENTS OF DATA, IF DESIRED
C
          IF (THEMAP .NE. YES) GO TO 73
          WRITE (6,155) KGLOB,TABDK,NSET
      155 FORMAT (25X,3(I10,5X))
C
CC        GENERATE THE MUTUALLY EXCLUSIVE SETS IN A TEMPERORARY ARRAY
CC        AND NAME IT TMPSET
C
       73 IF (NSET .EQ. 0) GO TO 2
          DO 75 J = 1,20
          IF (TMPSET(NSET,J) .EQ. 0) GO TO 74
          IF (TMPSET(NSET,J) .EQ. KGLOB) GO TO 2
          GO TO 75
       74 TMPSET(NSET,J) = KGLOB
          GO TO 2
       75 CONTINUE
C
CC        A NORMAL EXIT FROM THIS LOOP INDICATES THAT THE TEN ELEMENTS
CC        OF TMPSET(NSET,*) ARE FULL AND ITS DIMENSION SHOULD BE INCREASED
C
          WRITE (6,175) NSET
      175 FORMAT (10X,'CAREFUL; YOU ARE EXCEEDING THE DIMENSION OF TMPSET(',
         1 I2,',*). THIS MESSAGE GENERATED BY FORMAT NUMBER 175')
          STOP
C
CC        COMPACT THE CONTENTS OF TMPSET IN ARRAY MEXSET.
CC        POINTER FROM MUTUALLY EXCLUSIVE SET TO MEXSET IS NAMED AS MARCA
C
       15 J1 = 0
          DO 78 NSET = 1,100
          MARCA(NSET) = J1
          DO 76 J = 1,20
          IF (TMPSET(NSET,J) .EQ. 0) GO TO 78
          J1 = J1 + 1
C
CC        CHECK THAT J1 IS NOT MORE THAN THE DIMENSIONED VALUE
CC        OF THE ARRAY MEXSET WHICH IS CURRENTLY 150
```

```
C
      IF (J1 .GT. 150) GO TO 79
      NEXSET(J1) = TMPSET(NSET,J)
      GO TO 76
   79 WRITE (6,179)
  179 FORMAT (1H0,10X,'CAREFUL; YOU ARE EXCEEDING THE DIMENSION ',
     1 'OF THE ARRAY NEXSET WHICH IS 150'/
     211X,'REMEDY IS TO INCREASE THIS DIMENSION')
      STOP
   76 CONTINUE
   78 CONTINUE
C
CC    NOW TO COMPACT THE ARRAY IDEPND
C
      GO TO 3
C     READ WHICH TABLE MUST BE EXECUTED FIRST
   95 READ (5,107) TFIRST
  107 FORMAT (I10)
      GO TO 39
   42 WRITE (6,142)
  142 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY3',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 5000'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   43 WRITE (6,143)
  143 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY1',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   49 WRITE (6,149)
  149 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY5',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   52 WRITE (6,152)
  152 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY4',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 5000'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   53 WRITE (6,150)
  150 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY2',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   59 WRITE (6,159)
  159 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY6',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
C
CC    CHECK IF THE MAP OF THE PERMANENT STORAGE IS DESIRED
C
   39 IF (THEMAP .NE. YES) GO TO 999
      WRITE (6,500) (LARRY1(I),I=1,600)
  500 FORMAT (1H1,5X,'LARRY1'//30(5X,10I5,5X,10I5/))
      WRITE (6,501) (LARRY3(I),I=1,5000)
  501 FORMAT (1H1,5X,'LARRY3'//50(5X,10(10I1,1X)/))
      WRITE (6,502) (LARRY2(I),I=1,600)
  502 FORMAT (1H1,5X,'LARRY2'//30(5X,10I5,5X,10I5/))
      WRITE (6,503) (LARRY4(I),I=1,5000)
  503 FORMAT (1H1,5X,'LARRY4'//50(5X,10(10I1,1X)/))
      WRITE (6,505) (J,(IBASE(I,J),I=1,120),J=1,4)
  505 FORMAT (1H1/4(5X,'IBASE',I1//6(5X,10I5,5X,10I5/)////))
      WRITE(6,506) (LARRY5(I),I=1,600)
  506 FORMAT (1H1,5X,'LARRY5'//30(5X,10I5,5X,10I5/))
      WRITE (6,507) (LARRY6(I),I=1,600)
  507 FORMAT (1H1,5X,'LARRY6'//30(5X,10I5,5X,10I5/))
      WRITE (6,508) (IPNTRC(I),I=1,600)
  508 FORMAT (1H1,5X,'IPNTRC'//30(5X,10I5,5X,10I5/))
      WRITE (6,509) (IPNTRA(I),I=1,600)
  509 FORMAT (1H1,5X,'IPNTRA'//30(5X,10I5,5X,10I5/))
      WRITE (6,511)
  511 FORMAT (1H1,10X,'NON COMPACTED LIST OF DEPENDENTS'//
     1 T10,'DATA',T17,'NO. OF',T30,'LIST OF DEPENDENTS'/
     2 T10,'NO.',T15,'DEPENDENTS'//)
```

```
      DO 514 I = 1,700
      DO 513 J = 1,100
      IF (IDEPND(I,J) .EQ. 0) GO TO 613
  513 CONTINUE
  613 IF (J .EQ. 1) GO TO 514
      J1 = J-1
      WRITE (6,512) I,J1,(IDEPND(I,J),J=1,J1)
  512 FORMAT (10X,I3,5X,'(',I2,')',4(T30,4(5I4,2X)/))
  514 CONTINUE
      WRITE (6,520) (IARROW(I),I=1,700)
  520 FORMAT (1H1,5X,'IARROW'/35(5X,10I5,5X,10I5/))
      WRITE (6,521) (ICLEAR(I),I=1,2000)
  521 FORMAT (1H1,5X,'ICLEAR'//20(5(5X,10I5,5X,10I5/)/))
      WRITE (6,522) ((TMPSET(I,J),J=1,20),I=1,100)
  522 FORMAT (1H1,20X,'TMPSET'//5 (10(20X,10I5/)/))
      WRITE (6,523) (NEXSET(I),I=1,150)
  523 FORMAT (1H1,20X,'NEXSET'/15(20X,10I5/))
      WRITE (6,524) (MARCA(I),I=1,100)
  524 FORMAT (1H0,20X'MARCA'/5(20X,10I5/))
      WRITE(6,525) (ISET(I),I=1,700)
  525 FORMAT(1H1,5X,'ISET'/35(5X,10I5,5X,10I5/))
  999 WRITE (6,172) TFIRST
  172 FORMAT (1H0,15X,'EXECUTION WILL START WITH TABLE NO.',I4)
      WRITE(8) LARRY1,LARRY2,LARRY5,LARRY6,IPNTRC,IPNTRA
      WRITE(8) LARRY3
      WRITE(8) LARRY4
      WRITE(8) ISET,TABD,L,M,N,TFIRST,IBASE
      WRITE(8) NEXSET,MARCA,TRACE,ICLEAR,IARROW
      RETURN
      END
```

```
      SUBROUTINE INTIAL
CC    THIS SUBROUTINE IS USED TO INITIALIZE THE VARIOUS ARRAYS
CC    USED BY THE PROGRAM
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,MEXSET,MARCA,TRACE
      COMMON /STUPIN/ ICLEAR,IARROW
      COMMON /STINTL/ INDEX,ENTRY,INGR,IDEPND,TMPSET
      DIMENSION
     1 LARRY1(600),LARRY2(600),LARRY3(5000),LARRY4(5000),IBASE(120,4),
     2 LARRY5( 600),LARRY6( 600),IPNTRC(600),IPNTRA(600),
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4 IARROW(700) ,ICLEAR(2000),
     4 L(120),M(120),N(120),STACK(20,5),RESULT(2),
     5 INDEX(25),ENTRY(25,40),INGR(25,12),IDEPND(700,100),TMPSET(100,20)
      DIMENSION MEXSET(150),MARCA(100)
      DO 10 I = 1,600
      LARRY1(I) = 0
      LARRY2(I) = 0
      IPNTRC(I) = 0
      IPNTRA(I) = 0
   10 CONTINUE
      DO 20 I = 1,700
      DATA(I)    = 0.
      PRD(I)     = .FALSE.
      TABD(I)    = 0
      ISET(I) = 0
      IARROW(I) = 0
      DO 22 J = 1,100
      IDEPND(I,J) = 0
   22 CONTINUE
   20 CONTINUE
      DO 21 I = 1,1000
      LARRY5(I) = 0
      LARRY6(I) = 0
   21 CONTINUE
      DO 30 I = 1,5000
      LARRY3(I) = 0
      LARRY4(I) = 0
   30 CONTINUE
      DO 31 I = 1,2000
      ICLEAR(I) = 0
   31 CONTINUE
      DO 40 I = 1,100
      MARCA(I) = 0
      DO 40 J = 1,20
      TMPSET(I,J) = 0
   40 CONTINUE
      DO 41 I = 1,120
      L(I) = 0
      M(I) = 0
      N(I) = 0
      DO 41 J = 1,4
      IBASE(I,J) = 0
   41 CONTINUE
      DO 50 I = 1,25
      INDEX(I) = 0
      DO 48 J = 1,40
      ENTRY(I,J) = 0
   48 CONTINUE
      DO 49 J = 1,12
      INGR(I,J) = 0
   49 CONTINUE
   50 CONTINUE
      DO 60 I = 1,20
      DO 60 J = 1,5
      STACK(I,J) = 0
   60 CONTINUE
      DO 70 I = 1,150
      MEXSET(I) = 0
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE STAK(STACK,ISTACK,IFLAG,T,I,J,IR,TABNO,IDATA,TRACE)
C
CC    THIS SUBROUTINE PERFORMS STACKING OF DECISION TABLES FOR
CC    CONDITIONAL EXECUTION AND GENERATES MESSAGES TO THIS EFFECT
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ISTACK,IFLAG,T,I,J,IR,TABNO,IDATA,TABD
      COMMON/MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N
      DIMENSION STACK(20,5),L(120),LARRY1(600),LARRY2(600),
     1LARRY3(5000),LARRY4(5000),LARRY5(600),LARRY6(600),
     2IBASE(120,4),IPNTRC(600),IPNTRA(600),TABD(700),
     3M(120),N(120)
      DATA YES/'YES'/,NO/'NO'/
      ISTACK = ISTACK + 1
      STACK(ISTACK,1) = IFLAG
      STACK(ISTACK,2) = T
      STACK(ISTACK,3) = I
      STACK(ISTACK,4) = J
      STACK(ISTACK,5) = IR
C     CHECK IF A TRACE OF THE STACKING INFORMATION IS DESIRED OR NOT
      IF (TRACE .NE. YES) GO TO 10
C
C     PRINT THE APPROPRIATE MESSAGE
C
      GO TO (4,5,6,7),IFLAG
C
CC    MISSING INGREDIENT OF A CONDITION
C
    4 WRITE (6,104) T,I,J,IDATA,TABNO
  104 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT CONDITION',I3,' OF RULE',I3/11X,' REASON: MISSING ',
     2 'INGREDIENT CORRESPONDING TO DATA NUMBER ',I3/
     3 11X,' STARTED EXECUTION OF TABLE ',I3)
      GO TO 10
C
CC    MISSING VALUE OF THE CONDITION ITSELF
C
    5 WRITE (6,105)T,I,J,TABNO,IR
  105 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE '.I3,
     1 ' AT CONDITION',I3,' OF RULE',I3/11X,' STARTED EXECUTION',
     2 ' OF TABLE ',I3,' TO OBTAIN VALUE OF DATA NUMBER',I4)
      GO TO 10
C
CC    MISSING INGREDIENT OF AN ACTION
C
    6 WRITE (6,106)T,I,J,IDATA,TABNO
  106 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT ACTION',I3,' OF RULE',I3/11X,' REASON: MISSING ',
     2 'INGREDIENT CORRESPONDING TO DATA NUMBER ',I3/
     3 11X,' STARTED EXECUTION OF TABLE ',I3)
      GO TO 10
C
CC    DIRECT EXECUTION
C
    7 WRITE (6,107) T,I,J,TABNO
  107 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT ACTION',I3,' OF RULE',I3/11X,' STARTED EXECUTION',
     2 ' OF TABLE ',I3,' FOR DIRECT EXECUTION')
   10 T = TABNO
C     IF TABLE T DOES NOT EXIST ,TERMINATE EXECUTION.
CD    WRITE(7,110) L(T)
CD110 FORMAT(1X,'L(T)=',I4)
      IF(L(T).EQ.0) GO TO 9999
      TABNO = 0
      RETURN
 9999 WRITE(7,109) T
      WRITE(6,109) T
  109 FORMAT(1X,'TABLE NUMBER',2X,I4,'DOES NOT EXIST,',
     *' EXECUTION TERMINATED.')
      STOP
      END
```

```
      SUBROUTINE INPUT(ICYCLE)
CC    THIS SUBROUTINE IS USED FOR READING THE DATA VALUES AND FOR
CC    CLEARING THE DEPENDENT DATA FOR SECOND OR SUBSEQUENT CYCLES
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TNPSET
      COMMON /MICA/DATA,PRD
      COMMON/MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,MEXSET,MARCA,TRACE,TNBMAP
      COMMON /STUPIN/ ICLEAR,IARROW
      COMMON/STINTL/INDEX,ENTRY,INCR,IDEPND,TNPSET
      COMMON/DOOP/IPDL
      COMMON/DUMB/NARAY
      DIMENSION
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4 IARROW(700) ,ICLEAR(2000),
     5 IPDL(20,2)
      DIMENSION MEXSET(150),MARCA(100)
C
C     IF NOT THE FIRST CYCLE,CHECK IF THERE ARE
C     ANY DATA INPUT FOR THIS CYCLE
C
      IF (ICYCLE.EQ.1) GO TO 9
      WRITE (7,208)
  208 FORMAT(1H1,10X,'PLEASE INPUT A VALUE OF 1 OR 2 FOR INDIC')
      WRITE (7,210)
  210 FORMAT(10X,'1 INDICATES THERE ARE FURTHER CYCLES')
      WRITE (7,212)
  212 FORMAT(10X,'2 INDICATES NO FURTHER CYCLES')
      READ (4,200) INDIC
  200 FORMAT(I4)
C
C     IF INDIC=1, READ IN DATA ITEMS FOR THIS CYCLE
C                 FROM THE TERMINAL
C     IF INDIC=2, NO DATA ITEMS FOR THIS CYCLE,EXECUTION COMPLETED
      IF (INDIC.EQ.1) GO TO 201
      IF (INDIC.EQ.2) GO TO 9999
  201 WRITE (6,170) ICYCLE
  202 WRITE (7,203)
  203 FORMAT(10X,'AWAITING NEXT DATA ITEM')
      READ (4,204) KGLOB,DATAK
  204 FORMAT(I5,F10.0)
C
C     SIGNAL LAST DATA INPUT BY A ZERO INPUT
C
      IF (KGLOB.EQ.0) GO TO 110
      GO TO 207
C     READ THE NUMERICAL DATA FOR ALL THE TABLES
C     HEADING FOR THE NEXT PRINTED ITEM
C
    9 WRITE (6,170) ICYCLE
  170 FORMAT (1H1,15X,'THE FOLLOWING NUMERICAL DATA HAS BEEN SUPPLIED ',
     1 'FOR CYCLE NUMBER',I3//31X,'KGLOB',10X,'DATAK'//)
    1  READ (5,103,END=9999) KGLOB,DATAK
  103 FORMAT (I5,5X,F10.0)
C     SIGNAL LAST CARD IN THIS DATA BY A BLANK CARD
      IF (KGLOB .EQ. 0) GO TO 15
C
CC    IF THE DATA BELONGS TO A SET, FIX THE VALUE OF ITS
CC    ELEMENTS FIRST
C
  207 IF (ISET(KGLOB) .NE. 0) CALL SETS(KGLOB)
      DATA(KGLOB) = DATAK
      PRD(KGLOB) = .TRUE.
      WRITE (6,171) KGLOB,DATAK
  171 FORMAT (25X,I10,5X,F10.4)
C
CC    IF THIS IS THE FIRST CYCLE THEN GO TO READ THE NEXT CARD;
CC    OTHERWISE CLEAR THE EFFECT OF CHANGING THIS DATA FIRST
C
      IF (ICYCLE .EQ. 1) GO TO 1
C
```

```
CC     FOR THE PURPOSE OF CLEARING THE EFFECTS OF READING NEW DATA, THE
CC     LIST OF DEPENDENTS IS STORED WITH THE FIRST ELEMENT OF THE SET
C
       NSET = ISET(KGLOB)
CD     WRITE(6,8) KGLOB,NSET
CD   8 FORMAT(1X,'KGLOB=',I4,8X,'NSET=',I4)
       IF (NSET .EQ. 0) GO TO 11
       KGLOB = MARCA(NSET) + 1
CD     WRITE(6,10) KGLOB
CD  10 FORMAT(1X,'KGLOB****=',I4)
C
CC     FIRST FIND OUT IF THIS DATA HAS ANY DEPENDENT DATA.IF NO THEN READ
CC     THE NEXT DATA; IF YES THEN CLEAR IT
C
    11 IF ((IARROW(KGLOB+1) - IARROW(KGLOB)) .EQ. 0) GO TO 202
C
CC     START CLEARING
C
       IZ = 0
    12 LG = 1
    13 I1 = IARROW(KGLOB) + LG
       KDEP = ICLEAR(I1)
       IF(.NOT.PRD(KDEP)) GO TO 21
       PRD(KDEP) = .FALSE.
CD     WRITE(6,2000) KDEP
C2000 FORMAT(1X,'KDEP=',I4)
C
CC     CHECK IF DATA ITEM KDEP HAS ANY DEPENDENT COMPONENTS. IF YES THEN
CC     SUSPEND CLEARING DEPENDENTS OF KGLOB BY STACKING PROCEDURE AND
CC     START CLEARING DEPENDENTS OF KDEP; OTHERWISE CONTINUE CLEARING
CC     DEPENDENTS OF KGLOB
C
       IF ((IARROW(KDEP+1) - IARROW(KDEP)) .EQ. 0) GO TO 21
       IZ = IZ + 1
       IPDL(IZ,1) = KGLOB
       IPDL(IZ,2) = LG
CD     WRITE(6,2001) KGLOB,LG
C2001 FORMAT(1X,'KGLOB=',I4,5X,'LG=',I3)
       KGLOB = KDEP
       GO TO 12
    21 LG = LG + 1
CD     WRITE(6,2002) LG
C2002 FORMAT(1X,'LG IN LOOP 21=',I3)
       IF (LG .LE. (IARROW(KGLOB+1) - IARROW(KGLOB))) GO TO 13
C
CC     ALL THE DEPENDENT DATA FOR THIS KGLOB HAVE BEEN CLEARED AND SO
CC     UNSTACKING CAN BE STARTED
CD     WRITE(6,2003) IZ
C2003 FORMAT(1X,'IZ BEFORE STATEMENT 20=',I3)
    20 IF (IZ .EQ. 0) GO TO 202
       KGLOB = IPDL(IZ,1)
       LG    = IPDL(IZ,2)
       IZ    = IZ - 1
CD     WRITE(6,2004) KGLOB,LG,IZ
C2004 FORMAT(1X,'VALUES AFTER STATEMENT 20,KGLOB=',I4,
CD   13X,'LG=',I3,3X,'IZ=',I3)
       GO TO 21
C
CC     READ IF A TRACE OF THE TABLES EXECUTED IS DESIRED OR NOT
C
    15 READ (5,105) TRACE
   105 FORMAT (T11,A4)
C      READ IF WRITING OUT OF THE FOLLOWING ARRAYS IS DESIRED:
C      LARRY1,LARRY3,LARRY2,LARRY4,LARRY5,LARRY6
C      IBASE,IPNTRC,IPNTRA,IARROW,ICLEAR,NEXSET,
C      MARCA,ISET,L
C
       READ(5,108) WARAY
   108 FORMAT(T11,A4)
   110 RETURN
  9999 WRITE(6,2010)
  2010 FORMAT(1X,'EXECUTION OF PROGRAM IS COMPLETED')
       WRITE(7,2012)
  2012 FORMAT(1X,'EXECUTION OF PROGRAM IS COMPLETED.'/
      *1X,'COLLECT YOUR OUTPUT FROM THE COMPUTING CENTER.COME BACK SOON')
       STOP

       END
```

```
      SUBROUTINE SETS(KGLOB)
C
CC    THIS SUBROUTINE IS USED TO EVALUATE THE DATA IN MUTUALLY
CC    EXCLUSIVE SETS AT THE TIME OF EXTERNAL INPUT
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTIN/ISET,MEXSET,MARCA
      DIMENSION DATA(700),PRD(700),ISET(700)
      DIMENSION MEXSET(150),MARCA(100)
C
CC    CHECK IF KGLOB BELONGS TO A SET. IF YES THEN
CC    FIX THE VALUES OF ALL THE ELEMENTS OF THE SET TO NO
C
      NSET = ISET(KGLOB)
      IF (NSET .EQ. 0) GO TO 99
      I1 = MARCA(NSET) + 1
      I2 = MARCA(NSET + 1)
      DO 10 I = I1,I2
      IGLOB = MEXSET(I)
    3 DATA(IGLOB) = 0.0
    4 PRD(IGLOB)  = .TRUE.
   10 CONTINUE
      DATA(KGLOB)=1.0
   99 RETURN
      END
```

READIN

```
C23456789012345678901234567890123456789012345678901234567890123456789012
C
C      THIS SUBROUTINE IS FOR READING IN MISSING
C      DATA ITEMS FROM THE TERMINAL. THIS ENABLES
C      THE PROGRAM TO BE RUN UNDER INTERACTIVE MODE.
C      A LARGE NUMBER OF ABORTIVE RUNS DUE TO MISSING
C      DATA ITEMS CAN THEREFORE BE ELIMINATED.
C
C
       SUBROUTINE READIN(KG,NUMBER,KK)
       IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
       COMMON/MICA/DATA,PRD
       COMMON/MUSTIN/ISET,MEXSET,MARCA,TRACE,THEMAP
       COMMON/STUPIN/ICLEAR,IARROW
       COMMON/DOOP/IPDL
       COMMON/MINCE/ICYCLE
       DIMENSION DATA(700),PRD(700),ISET(700),MEXSET(150),
      *MARCA(100),ICLEAR(2000),IARROW(700),IPDL(20,2)
C
       KLOSS=KK
       GO TO (10,20,30),NUMBER
C
C      WRITING ON TERMINAL TO LET USER KNOW THAT
C      IT IS AWAITING INPUT.
C
    10 WRITE(7,100) KG
   100 FORMAT(1X,'*****AWAITING INPUT FOR DATA ITEM WITH SUBSCRIPT=',I4/
      *1X,'THIS DATA ITEM IS A CONDITION.')
       GO TO 500
C
    20 WRITE(7,120) KG
   120 FORMAT(1X,'*****AWAITING INPUT FOR DATA ITEM WITH SUBSCRIPT=',I4/
      *1X,'THIS DATA ITEM IS A MISSING INGREDIENT OF A CONDITION')
       GO TO 500
C
    30 WRITE(7,140) KG
   140 FORMAT(1X,'*****AWAITING INPUT FOR DATA ITEM WITH SUBSCRIPT=',I4/
      *1X,'THIS DATA ITEM IS A MISSING INGREDIENT OF AN ACTION')
       GO TO 500
C
C      READ IN MISSING DATA SUBSCRIPT AND ITS VALUE.
C
   500 READ(4,520) KG,DATA(KG)
   520 FORMAT(I10,F10.4)
C
C      IF WISH TO TERMINATE EXECUTION,INPUT A NEGATIVE VALUE OF KG
C
       IF (KG.LT.0) GO TO 9999
C
C      CHECK IF THE CORRECT DATA SUBSCRIPT HAS BEEN INPUT
C
       IF (KG.NE.KK) GO TO 600
C      SET PRESENCE OF INPUT DATA TO .TRUE.
C
       PRD(KG)=.TRUE.
C      IF INPUT DATA ITEM BELONGS TO A MUTUALLY EXCLUSIVE SET,
C      AND IF THE DATA=1.0 . SET THE OTHER DATA ITEMS
C      OF THAT SET TO 0.0 AND THEIR PRESENCE TO .TRUE.
       IF(DATA(KG).EQ.0.0) GO TO 540
       IF(ISET(KG).NE.0) CALL SETS(KG)
   540 IF(ICYCLE.EQ.1) GO TO 600
C
C      WHEN A MISSING DATA ITEM IS READ IN IN
C      THIS ROUTINE,ANY DEPENDENTS THAT THIS DATA
C      ITEM HAS IS TO BE CLEARED IN CYCLES OTHER
C      THAN CYCLE 1.
CC     FOR THE PURPOSE OF CLEARING THE EFFECTS OF READING NEW DATA, THE
CC     LIST OF DEPENDENTS IS STORED WITH THE FIRST ELEMENT OF THE SET
C
       NSET = ISET(KLOSS)
       WRITE(6,620) KLOSS,NSET
   620 FORMAT(1X,'KLOSS=',I4,8X,'NSET=',I4)
       IF (NSET .EQ. 0) GO TO 590
       KLOSS = MARCA(NSET) + 1
       WRITE(6,640) KLOSS
   640 FORMAT(1X,'KLOSS****=',I4)
```

```
C
CC     FIRST FIND OUT IF THIS DATA HAS ANY DEPENDENT DATA.IF NO THEN READ
CC     THE NEXT DATA; IF YES THEN CLEAR IT
C
   590 IF ((IARROW(KLOSS+1) - IARROW(KLOSS)) .EQ. 0) GO TO 600
C
CC     START CLEARING
C
       IZ = 0
   660 LG = 1
   680 I1 = IARROW(KLOSS) + LG
       KDEP = ICLEAR(I1)
       IF(.NOT.PRD(KDEP)) GO TO 740
       PRD(KDEP) = .FALSE.
CD     WRITE(6,700 ) KDEP
CD700  FORMAT(1X,'KDEP=',I4)
C
CC     CHECK IF DATA ITEM KDEP HAS ANY DEPENDENT COMPONENTS. IF YES THEN
CC     SUSPEND CLEARING DEPENDENTS OF KLOSS BY STACKING PROCEDURE AND
CC     START CLEARING DEPENDENTS OF KDEP; OTHERWISE CONTINUE CLEARING
CC     DEPENDENTS OF KLOSS
C
       IF ((IARROW(KDEP+1) - IARROW(KDEP)) .EQ. 0) GO TO 740
       IZ = IZ + 1
       IPDL(IZ,1) = KLOSS
       IPDL(IZ,2) = LG
CD     WRITE(6,720)  KLOSS,LG
CD720  FORMAT(1X,'KLOSS=',I4,5X,'LG=',I3)
       KLOSS = KDEP
       GO TO 660
   740 LG = LG + 1
CD     WRITE(6,760)  LG
CD760  FORMAT(1X,'LG IN LOOP 740=',I3)
       IF (LG .LE. (IARROW(KLOSS+1) - IARROW(KLOSS))) GO TO 680
C
CC     ALL THE DEPENDENT DATA FOR THIS KLOSS HAVE BEEN CLEARED AND SO
CC     UNSTACKING CAN BE STARTED
CD     WRITE(6,780)  IZ
CD780  FORMAT(1X,'IZ BEFORE STATEMENT 800=';I3)
   800 IF (IZ .EQ. 0) GO TO 600
       KLOSS = IPDL(IZ,1)
       LG    = IPDL(IZ,2)
       IZ    = IZ - 1
CD     WRITE(6,820)  KLOSS,LG,IZ
CD820  FORMAT(1X,'VALUES AFTER STATEMENT 800,KLOSS=',I4,
CD    13X,'LG=',I3,3X,'IZ=',I3)
       GO TO 740
   600 RETURN
C
  9999 WRITE(7,560)
   560 FORMAT(1X,'PROGRAM EXECUTION TERMINATED BY AN INPUT OF NEGETIVE',
      *' VALUE OF KG')
       WRITE(6,560)
       STOP
       END
```

```
      SUBROUTINE OUTPUT(ICYCLE)
C
CC    THROUGH THIS SUBROUTINE, IT IS POSSIBLE TO OUTPUT ANY DATA
CC    VALUE FOR CHECKING AND DIOGNASIS
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      DIMENSION DATA(700),PRD(700)
      WRITE (6,100) ICYCLE
  100 FORMAT (1H1,15X,'DATA VALUES AT THE END OF CYCLE NO.',I3/
     1 16X,'ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE'
     2//31X,'KGLOB',10X,'DATAK',10X,'PRD'//)
      DO 801 KGLOB = 1,700
      IF (.NOT. PRD(KGLOB)) GO TO 801
      WRITE (6,800) KGLOB,DATA(KGLOB),PRD(KGLOB)
  800 FORMAT (25X,I10,1X,F14.4,5X,L7)
  801 CONTINUE
      RETURN
      END
```

```
CLEAR
      C      THIS IS THE SUBROUTINE FOR PRINTING AND CLEARING
      C      DATAS IF A TABLE HAS TO BE EXECUTED MORE THAN
      C      ONCE IN A PARTICULAR RUN.
      C      THE DATAS WHICH ARE TO BE PRINTED AND CLEARED ARE
      C      SPECIFIED BY THE DATA STATEMENT IN THE ACTION
      C      SUBROUTINE OF THAT TABLE.
      C      THE NUMBER OF TIMES THAT THE TABLE IS TO BE EXECUTED
      C      IS SPECIFIED BY  NCHECK .
      C
      C
             SUBROUTINE CLEAR(MCLEAR)
             IMPLICIT LOGICAL*1(P), INTEGER*2 (I-N)
             COMMON/MICA/DATA,PRD
             COMMON/MUSTIN/ISET,MEXSET,MARCA,TRACE,THEMAP
             COMMON/STUPIN/ICLEAR,IARROW
             COMMON/DOOP/IPDL
             DIMENSION
            *DATA(700),PRD(700),ISET(700),MEXSET(150),MARCA(100),
            *ICLEAR(2000),IARROW(700),IPDL(20,2),MCLEAR(15)
      C
      C
      C      WRITE THE VALUE OF (DATA(501),CHECKI)
      C
             WRITE(6,100) DATA(501)
             WRITE(7,100) DATA(501)
         100 FORMAT(1H1,'******* CHECKI=',2X,F4.1,2X,'*******'///)
      C
      C      PRINT OUT THE DATAS THAT HAVE TO BE CLEARED BEFORE
      C      RENTERING THE TABLE.
      C
             WRITE(6,102)
             WRITE(7,102)
         102 FORMAT(31X,'KGLOB',10X,'DATA(KGLOB)',10X,'PRD(KGLOB)'///)
             DO 5 J=1,15
             KGLOB=MCLEAR(J)
             IF (KGLOB.EQ.0) GO TO 106
             WRITE(6,104) KGLOB,DATA(KGLOB),PRD(KGLOB)
             WRITE(7,104) KGLOB,DATA(KGLOB),PRD(KGLOB)
         104 FORMAT(25X,I10,5X,F14.4,10X,L7)
           5 CONTINUE
      C
      C
         106 I=0
          10 I=I+1
             KGLOB=MCLEAR(I)
      C      IF KGLOB IS 0 , ALL OF THE DATA HAVE BEEN CLEARED
             IF(KGLOB.EQ.0) GO TO 30
      C
```

```
C     SET KGLOB TO ZERO AND .FALSE.
C
      DATA(KGLOB)=0.0
      PRD(KGLOB)=.FALSE.
C
C     FOR THE PURPOSE OF CLEARING THE DATA, THE LIST OF
C     DEPENDENTS IS STORED WITH THE FIRST ELEMENT OF THE SET.
C
      NSET=ISET(KGLOB)
      IF(NSET.EQ.0) GO TO 11
      KGLOB=MARCA(NSET)+1
C
C     FIND OUT IF THIS DATA HAS ANY DEPENDENT DATA. IF NO ,
C     READ THE NEXT DATA. IF YES, THEN CLEAR IT.
C
   11 IF((IARROW(KGLOB+1)-IARROW(KGLOB)).EQ.0) GO TO 10
C
C     START CLEARING
C
      IZ=0
   12 LG=1
   13 I1=IARROW(KGLOB)+LG
      KDEP=ICLEAR(I1)
      IF(.NOT.PRD(KDEP)) GO TO 21
      PRD(KDEP)=.FALSE.
C
C     CHECK IF DATA ITEM KDEP HAS ANY DEPENDENT COMPONENTS.
C     IF YES THEN SUSPEND CLEARING DEPENDENTS OF KGLOB BY
C     STACKING PROCEDURE AND START CLEARING DEPENDENTS OF KDEP;
C     OTHERWISE CONTINUE CLEARING DEPENDENTS OF KGLOB.
C
      IF((IARROW(KDEP+1)-IARROW(KDEP)).EQ.0) GO TO 21
      IZ=IZ+1
      IPDL(IZ,1)=KGLOB
      IPDL(IZ,2)=LG
      KGLOB=KDEP
      GO TO 12
   21 LG=LG+1
      IF(LG.LE.(IARROW(KGLOB+1)-IARROW(KGLOB))) GO TO 13
C
C     ALL THE DEPENDENT DATA FOR THIS KGLOB HAVE BEEN CLEARED,
C     SO UNSTACKING CAN BE STARTED.
C
      IF(IZ.EQ.0) GO TO 10
      KGLOB=IPDL(IZ,1)
      LG=IPDL(IZ,2)
      IZ=IZ-1
      GO TO 21
   30 RETURN
      END
```

APPENDIX E

BATCH MODE PROCESSING PROGRAM
- SOURCE LISTING

This appendix contains a source listing of the program for batch mode processing of design specifications.

```
$LIST BATCHMODE

CC    THIS IS THE MAIN SUBROUTINE DOING THE BULK OF DECISION TABLE
CC    PROCESSING INCLUDING IDENTIFYING THE APPLICABLE RULE,
CC    AND CHECKING THE PRESENCE OF DATA ETC.
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,MEXSET,MARCA,TRACE,THEMAP
      COMMON/STUPIN/ICLEAR,IARROW
      COMMON/STINTL/INDEX,ENTRY,INGR,IDEPND,TMPSET
      COMMON/DUMB/WARAY
      DIMENSION
     1 LARRY1(600),LARRY2(600),LARRY3(5000),LARRY4(5000),IBASE(120,4),
     2 LARRY5( 600),LARRY6( 600),IPNTRC(600),IPNTRA(600) ,
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4L(120),M(120),N(120),STACK(20,5),RESULT(2)
      DIMENSION MEXSET(150),MARCA(100),IARROW(700),ICLEAR(2000)
      DATA YES/'YES'/,NO/'NO'/
C     READ IF A MAP OF PERMANENT DATA STORAGE IS DESIRED OR NOT
      READ(2,110) THEMAP
  110 FORMAT(T11,A4)
C     IF ISAVE=1,ENTER SUBROUTINE SETUP,CALCULATE DECISION TABLES
C     PERMANENT DATA,THEN STORE IN UNIT 8
C     IF ISAVE=2,SKIP SUBROUTINE SETUP,READ DECISION TABLES
C     PERMANENT DATA FROM UNIT 8
      READ(2,100) ISAVE
  100 FORMAT(I10)
      CALL SETDSB(8,11000,11000)
      GO TO (700,710),ISAVE
  710 CALL INTIAL
      READ(8) LARRY1,LARRY2,LARRY5,LARRY6,IPNTRC,IPNTRA
      READ(8) LARRY3
      READ(8) LARRY4
      READ(8) ISET,TABD,L,M,N,TFIRST,IBASE
      READ(8) MEXSET,MARCA,TRACE,ICLEAR,IARROW
C     WHEN ISAVE=2
C     READ TFIRST FROM UNIT 5 AGAIN IN CASE IT IS DIFFERENT
C     THAN THAT READ FROM UNIT 8
      READ(5,102) TFIRST
  102 FORMAT(I10)
C     IF WARAY EQUAL YES,WRITE THE ARRAYS IF STATMENT 900 TO 928
      IF (WARAY.NE.YES) GO TO 730
      WRITE (6,900) (LARRY1(I),I=1,600)
  900 FORMAT (1H1,5X,'LARRY1'//30(5X,10I5,5X,10I5/))
      WRITE (6,902) (LARRY3(I),I=1,5000)
  902 FORMAT (1H1,5X,'LARRY3'//50(5X,10(10I1,1X)/))
      WRITE (6,904) (LARRY2(I),I=1,600)
  904 FORMAT (1H1,5X,'LARRY2'//30(5X,10I5,5X,10I5/))
      WRITE (6,906) (LARRY4(I),I=1,5000)
  906 FORMAT (1H1,5X,'LARRY4'//50(5X,10(10I1,1X)/))
      WRITE (6,908) (J,(IBASE(I,J),I=1,120),J=1,4)
  908 FORMAT (1H1/4(5X,'IBASE',I1//6(5X,10I5,5X,10I5/)////))
      WRITE(6,910) (LARRY5(I),I=1,600)
  910 FORMAT (1H1,5X,'LARRY5'//30(5X,10I5,5X,10I5/))
      WRITE (6,912) (LARRY6(I),I=1,600)
  912 FORMAT (1H1,5X,'LARRY6'//30(5X,10I5,5X,10I5/))
      WRITE (6,914) (IPNTRC(I),I=1,600)
  914 FORMAT (1H1,5X,'IPNTRC'//30(5X,10I5,5X,10I5/))
      WRITE (6,916) (IPNTRA(I),I=1,600)
  916 FORMAT (1H1,5X,'IPNTRA'//30(5X,10I5,5X,10I5/))
      WRITE (6,918) (IARROW(I),I=1,700)
  918 FORMAT (1H1,5X,'IARROW'/35(5X,10I5,5X,10I5/))
      WRITE (6,920) (ICLEAR(I),I=1,2000)
  920 FORMAT (1H1,5X,'ICLEAR'//20(5(5X,10I5,5X,10I5/)/))
      WRITE (6,922) (MEXSET(I),I=1,150)
  922 FORMAT (1H1,20X,'MEXSET'/15(20X,10I5/))
      WRITE (6,924) (MARCA(I),I=1,100)
  924 FORMAT (1H0,20X'MARCA'/5(20X,10I5/))
      WRITE(6,926) (ISET(I),I=1,700)
  926 FORMAT(1H1,5X,'ISET'/35(5X,10I5,5X,10I5/))
      WRITE(6,928) (L(I),I=1,120)
  928 FORMAT(1H1,20X,'ARRAY L'/12(20X,10I5/))
      GO TO 730
  700 CALL SETUP
  730 ICYCLE = 1
    1 CALL INPUT(ICYCLE)
C     PRINT DATA AGAIN TO CHECK ITS VALIDITY
      WRITE (6,799)
  799 FORMAT (1H1,15X,'DATA PRINTED AGAIN FOR CHECKING. ONLY ',
     1 'THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE'//
     2 31X,'KGLOB',10X,'DATAK',10X,'PRD'//)
      DO 801 KGLOB = 1,700
      IF (.NOT. PRD(KGLOB)) GO TO 801
      WRITE (6,800) KGLOB,DATA(KGLOB),PRD(KGLOB)
  800 FORMAT (25X,I10,1X,F14.4,5X,L7)
  801 CONTINUE
C
CC    GIVE A MESSAGE THAT EXECUTION OF CYCLE NUMBER (= ICYCLE) IS ABOUT
CC    TO COMMENCE
C
      WRITE (6,172) ICYCLE,TFIRST
  172 FORMAT (1H1,10X,'CYCLE NUMBER',I3,  5X,'***   START',
     1 ' EXECUTION WITH TABLE',I4,'    ***')
C     INITIALISE BEFORE STARTING EXECUTION OF THE TABLES
```

```
      TRUE = 1.0
      T = TFIRST
      ISTACK = 0
      TABNO = 0
   18 J = 1
   19 I = 1
C     GET THE ADDRESS OF THE CONDITION ENTRY
   24 IJ = IBASE(T,3) + (J-1)*N(T) + I
C     IF THE CONDITION ENTRY IS IMMATERIAL, SKIP CHECKING IT
      IF (LARRY3(IJ)  .EQ. 0) GO TO 52
   25 I1 = IBASE(T,1) + I
      KGLOB = LARRY1(I1)
C
CC    FIRST CHECK IF THE CONDITION HAS BEEN SUPPLIED WITH ITS VALUE
C
      IF (PRD(KGLOB)) GO TO 50
C
CC    CHECK IF ANY OTHER TABLE CAN BE EXECUTED TO GET THIS CONDITION
C
      IF (TABD(KGLOB) .NE. 0) GO TO 45
C
CC    OTHERWISE SEE IF THIS CONDITION CAN BE ESTABLISHED BY
CC    SUBROUTINE CC OF THIS TABLE. THIS IS INDICATED BY
CC    HAVING ATLEAST ONE INGREDIENT FOR THIS CONDITION
C
      IF ((IPNTRC(I1+1) - IPNTRC(I1)) .NE. 0) GO TO 13
C
CC    ELSE AN ERROR MESSAGE
C
   12 WRITE (6,174) I,T,KGLOB
  174 FORMAT (1H0,10X,'CONDITION NUMBER',I3,'OF TABLE ',I3,
     1 'IS NOT AVAILABLE. THIS CORRESPONDS TO DATA NUMBER',I4/
     2 11X,'FURTHER EXECUTION OF THIS CYCLE IS NOT POSSIBLE')
      GO TO 76
C
CC    CHECK IF ALL THE INGREDIENTS OF THIS CONDITION ARE PRESENT
C
   13 IR = IPNTRC(I1) + 1
   23 IDATA = LARRY5(IR)
      IF (.NOT. PRD(IDATA)) GO TO 39
   15 IR = IR + 1
      IF (IR .LE. IPNTRC(I1+1)) GO TO 23
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT ALL THE
CC    DATA NECESSARY TO SET THIS CONDITION IS PRESENT AND SO
CC    ITS SUBROUTINE CC CAN BE CALLED
C
      GO TO(888,888,888,888,888,888,888,888,209,210,
     *      211,212,213,214,215,216,217,218,219,888,
     *      221,222,223,224,225,888,227,888,229,888,
     *      231,888,888,888,888,888,888,888,888,888,
     *      888,888,888,244,245,888,888,248,249,888,
     5      888,888,888,888,888,888,888,258,259,888,
     6      261,888,263,264,265,888,888,888,888,888,
     7      888,272,888,274,888,276,277,888,279,888,
     8      281,282,888,284,888,286,287,888,888,290,
     9      291,888,888,888,888,888,297,888,888,888,
     A      888,888,888,888,888,888,888,888,888,888,
     B      888,888,888,888,888,888,888,888,888,888),T
  209 CALL CC9(I)
      GO TO 41
  210 CALL CC10(I)
      GO TO 41
  211 CALL CC11(I)
      GO TO 41
  212 CALL CC12(I)
      GO TO 41
  213 CALL CC13(I)
      GO TO 41
  214 CALL CC14(I)
      GO TO 41
  215 CALL CC15(I)
      GO TO 41
  216 CALL CC16(I)
      GO TO 41
  217 CALL CC17(I)
      GO TO 41
  218 CALL CC18(I)
      GO TO 41
  219 CALL CC19(I)
      GO TO 41
  221 CALL CC21(I)
      GO TO 41
  222 CALL CC22(I)
      GO TO 41
  223 CALL CC23(I)
      GO TO 41
  224 CALL CC24(I)
      GO TO 41
  225 CALL CC25(I)
      GO TO 41
  227 CALL CC27(I)
      GO TO 41
  229 CALL CC29(I)
      GO TO 41
  231 CALL CC31(I)
      GO TO 41
  244 CALL CC44(I)
      GO TO 41
  245 CALL CC45(I)
```

```
      GO TO 41
  248 CALL CC48(I)
      GO TO 41
  249 CALL CC49(I)
      GO TO 41
  258 CALL CC58(I)
      GO TO 41
  259 CALL CC59(I)
      GO TO 41
  261 CALL CC61(I)
      GO TO 41
  263 CALL CC63(I)
      GO TO 41
  264 CALL CC64(I)
      GO TO 41
  265 CALL CC65(I)
      GO TO 41
  272 CALL CC72(I)
      GO TO 41
  274 CALL CC74(I)
      GO TO 41
  276 CALL CC76(I)
      GO TO 41
  277 CALL CC77(I)
      GO TO 41
  279 CALL CC79(I)
      GO TO 41
  281 CALL CC81(I)
      GO TO 41
  282 CALL CC82(I)
      GO TO 41
  284 CALL CC84(I)
      GO TO 41
  286 CALL CC86(I)
      GO TO 41
  287 CALL CC87(I)
      GO TO 41
  290 CALL CC90(I)
      GO TO 41
  291 CALL CC91(I)
      GO TO 41
  297 CALL CC97(I)
      GO TO 41
  888 WRITE (6,1888) T
 1888 FORMAT (1H0,10X,'ERROR SITUATION. ATTEMPT TO CALL SUBROUTINE CC',
     1 I3,'WHERE IT IS NOT SUPPOSED TO BE SO'/11X,'NO SUCH ',
     2 'SUBROUTINE EXISTS')
      GO TO 76
C     CHECK IF THE CONDITION HAS GOT ITS VALUE NOW
   41 WRITE(6,444) KGLOB,DATA(KGLOB),PRD(KGLOB)
  444 FORMAT(1X,'DATA CHECK JUST BEFORE STATEMENT 41 IN SPECHK'//
     120X,I10,1X,F14.4,5X,L7)
      IF (PRD(KGLOB)) GO TO 50
      GO TO 12
C
CC    FOLLOWING IS A CHECK WHETHER THE MISSING INGREDIENT
CC    IS OBTAINABLE BY EXECUTING ANY TABLE
C
   39 IF (TABD(IDATA) .NE. 0) GO TO 46
C
CC    WRITE THE ERROR MESSAGE THAT THIS DATA IS NOT AVAILABLE
C
      WRITE (6,112) IDATA
  112 FORMAT (1H0,10X,'ERROR MESSAGE; DATA NUMBER',I5,' IS NOT AVAILABLE
     1 AND IS NOT OBTAINABLE BY EXECUTING ANY TABLE EITHER')
      GO TO 76
   46 TABNO = TABD(IDATA)
C
CC    START THE STACKING PROCEDURE TO EXECUTE THE APPROPRIATE TABLE
CC    THE VALUE OF IFLAG = 1 INDICATES THAT THE STACKING IS REQUIRED
CC    BECAUSE OF SOME MISSING INGREDIENT OF A CONDITION
C
      IFLAG = 1
      CALL STAK(STACK,ISTACK,IFLAG,T,I,J,IR,TABNO,IDATA,TRACE)
      GO TO 18
C
CC    START THE STACKING PROCEDURE TO EXECUTE THE APPROPRIATE TABLE
CC    THE VALUE OF IFLAG = 2 INDICATES THAT THE STACKING
CC    IS REQUIRED BECAUSE THE MISSING CONDITION IS OBTAINABLE
CC    BY EXECUTING SOME OTHER TABLE
C
   45 TABNO = TABD(KGLOB)
      IFLAG = 2
      IDATA = 0
      CALL STAK(STACK,ISTACK,IFLAG,T,I,J,KGLOB,TABNO,IDATA,TRACE)
      GO TO 18
C
CC    MATCH THE RULE
C
   50 IF (LARRY3(IJ) .EQ. 1 .AND. DATA(KGLOB) .NE. TRUE) GO TO 55
      IF (LARRY3(IJ) .EQ. 2 .AND. DATA(KGLOB) .EQ. TRUE) GO TO 55
C
CC    CONTINUE MATCHING CONDITIONS IN THIS RULE
C
   52 WRITE(6,440) I
  440 FORMAT(1X,'AT STATEMENT 52 IN SPECHK,THE CONDITION NO. IS',
     *5X,I3)
      IF (I .EQ. N(T)) GO TO 31
      I = I+1
      GO TO 24
```

```
C     CONTINUE THE SEARCH WITH THE NEXT RULE
   55 IF (J.EQ. L(T)) GO TO 30
      J = J+1
      WRITE(6,445) J
  445 FORMAT(1X,'AFTER STATEMENT 55 IN SPECHK,THE RULE NO. IS',5X,I3)
      GO TO 19
C
CC    MESSAGE FOR UNSUCCESSFUL MATCH IN THE TABLE
C
   30 WRITE (6,130) T
  130 FORMAT (1H0,10X,'NO RULE IN TABLE',I4,' IS MATCHING ',
     1 'THE CONDITION STUB'/11X,' CYCLE TERMINATED')
      GO TO 76
C
CC    THE APPLICABLE RULE HAS BEEN IDENTIFIED
CC    CHECK IF THIS INFORMATION IS DESIRED TO BE PRINTED OUT
C
   31 IF (TRACE .NE. YES) GO TO 57
      WRITE (6,177) T,J
  177 FORMAT (1H0,15X,'SCANNING OF TABLE ',I3,' IS COMPLETE. RULE ',
     1 'NO.',I3,' APPLIES')
C
CC    NOW FIND WHICH ACTION ENTRY IS APPLICABLE
CC    CODE: 0 FOR NO ACTION, 1 FOR CONDITIONAL EVALUATION,
CC    2 FOR DIRECT EXECUTION ACTIONS AND 3 FOR THE ELSE RULE
C
   57 K = 1
   58 KJ = IBASE(T,4)  +  (J-1)*N(T) + K
      IFLAG = LARRY4(KJ) + 2
      K1 = IBASE(T,2) + K
      GO TO (49,59,69,79,89),IFLAG
C
CC    THE FOLLOWING IS A DUMMY STATEMENT AND SHOULD NEVER BE REACHED
C
   49 STOP
C
CC    CONDITIONAL EVALUATION
C
   69 KGLOB = LARRY2(K1)
C
CC    BEFORE CALLING SUBROUTINE AA OF THIS TABLE, CHECK IF
CC    THERE ARE ANY INGREDIENTS FOR THIS ACTION.
CC    IF YES, THEN CHECK THEIR PRESENCE
C
      IF ((IPNTRA(K1+1) - IPNTRA(K1)) .EQ. 0) GO TO 93
      IR = IPNTRA(K1) + 1
   27 IDATA = LARRY6(IR)
      IF (.NOT. PRD(IDATA)) GO TO 35
   16 IR = IR + 1
      IF (IR .LE. IPNTRA(K1+1)) GO TO 27
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT ALL THE
CC    INGREDIENTS NEEDED FOR EVALUATING THIS ACTION ARE PRESENT
CC    AND SO SUBROUTINE AA FOR THIS TABLE CAN BE CALLED
C
   93 GO TO (999,999,503,504,505,999,507,508,509,510,
     1       511,512,513,514,515,516,999,518,519,999,
     2       521,522,523,524,525,526,527,528,529,530,
     3       531,532,999,999,999,999,999,999,999,999,
     4       999,542,543,544,545,546,547,548,549,550,
     5       551,552,553,554,555,999,557,558,559,560,
     6       561,562,563,564,565,566,567,568,569,570,
     7       571,572,573,574,575,576,577,578,579,580,
     8       581,582,999,584,585,586,587,588,589,590,
     9       591,592,593,594,595,596,597,598,999,999,
     A       999,999,999,999,999,999,999,999,999,999,
     B       999,999,999,999,999,999,999,999,999,999),T
  503 CALL AA3(K)
      GO TO 60
  504 CALL AA4(K)
      GO TO 60
  505 CALL AA5(K)
      GO TO 60
  507 CALL AA7(K)
      GO TO 60
  508 CALL AA8(K)
      GO TO 60
  509 CALL AA9(K)
      GO TO 60
  510 CALL AA10(K)
      GO TO 60
  511 CALL AA11(K)
      GO TO 60
  512 CALL AA12(K)
      GO TO 60
  513 CALL AA13(K)
      GO TO 60
  514 CALL AA14(K)
      GO TO 60
  515 CALL AA15(K)
      GO TO 60
  516 CALL AA16(K)
      GO TO 60
  518 CALL AA18(K)
      GO TO 60
  519 CALL AA19(K)
      GO TO 60
  521 CALL AA21(K)
      GO TO 60
  522 CALL AA22(K)
```

```
      GO TO 60
  523 CALL AA23(K)
      GO TO 60
  524 CALL AA24(K)
      GO TO 60
  525 CALL AA25(K)
      GO TO 60
  526 CALL AA26(K)
      GO TO 60
  527 CALL AA27(K)
      GO TO 60
  528 CALL AA28(K)
      GO TO 60
  529 CALL AA29(K)
      GO TO 60
  530 CALL AA30(K)
      GO TO 60
  531 CALL AA31(K)
      GO TO 60
  532 CALL AA32(K)
      GO TO 60
  542 CALL AA42(K)
      GO TO 60
  543 CALL AA43(K)
      GO TO 60
  544 CALL AA44(K)
      GO TO 60
  545 CALL AA45(K)
      GO TO 60
  546 CALL AA46(K)
      GO TO 60
  547 CALL AA47(K)
      GO TO 60
  548 CALL AA48(K)
      GO TO 60
  549 CALL AA49(K)
      GO TO 60
  550 CALL AA50(K)
      GO TO 60
  551 CALL AA51(K)
      GO TO 60
  552 CALL AA52(K)
      GO TO 60
  553 CALL AA53(K)
      GO TO 60
  554 CALL AA54(K)
      GO TO 60
  555 CALL AA55(K)
      GO TO 60
  557 CALL AA57(K)
      GO TO 60
  558 CALL AA58(K)
      GO TO 60
  559 CALL AA59(K)
      GO TO 60
  560 CALL AA60(K)
      GO TO 60
  561 CALL AA61(K)
      GO TO 60
  562 CALL AA62(K)
      GO TO 60
  563 CALL AA63(K)
      GO TO 60
  564 CALL AA64(K)
      GO TO 60
  565 CALL AA65(K)
      GO TO 60
  566 CALL AA66(K)
      GO TO 60
  567 CALL AA67(K)
      GO TO 60
  568 CALL AA68(K)
      GO TO 60
  569 CALL AA69(K)
      GO TO 60
  570 CALL AA70(K)
      GO TO 60
  571 CALL AA71(K)
      GO TO 60
  572 CALL AA72(K)
      GO TO 60
  573 CALL AA73(K)
      GO TO 60
  574 CALL AA74(K)
      GO TO 60
  575 CALL AA75(K)
      GO TO 60
  576 CALL AA76(K)
      GO TO 60
  577 CALL AA77(K)
      GO TO 60
  578 CALL AA78(K)
      GO TO 60
  579 CALL AA79(K)
      GO TO 60
  580 CALL AA80(K)
      GO TO 60
  581 CALL AA81(K)
      GO TO 60
  582 CALL AA82(K)
      GO TO 60
  584 CALL AA84(K)
      GO TO 60
  585 CALL AA85(K)
      GO TO 60
  586 CALL AA86(K)
      GO TO 60
  587 CALL AA87(K)
      GO TO 60
  588 CALL AA88(K)
      GO TO 60
  589 CALL AA89(K)
      GO TO 60
  590 CALL AA90(K)
      GO TO 60
  591 CALL AA91(K)
      GO TO 60
  592 CALL AA92(K)
      GO TO 6C
  593 CALL AA93(K)
      GO TO 60
  594 CALL AA94(K)
      GO TO 60
  595 CALL AA95(K)
      GO TO 60
  596 CALL AA96(K)
      GO TO 60
  597 CALL AA97(K)
      GO TO 60
  598 CALL AA98(K)
      GO TO 60
  999 WRITE (6,1999) T
 1999 FORMAT (1H0,10X,'ERROR SITUATION. ATTEMPT TO CALL SUBROUTINE AA',
     1 I3,'WHERE IT IS NOT SUPPOSED TO BE SO'/11X,'NO SUCH ',
     2 'SUBROUTINE EXISTS')
      GO TO 76
C     CHECK IF THIS ACTION IS COMPLETE
   60 IF (KGLOB .EQ. 0) GO TO 59
      IF (PRD(KGLOB)) GO TO 59
C
CC    ERROR MESSAGE
C
      WRITE (6,141) K,T,J
  141 FORMAT (1H0,10X,'ACTION NUMBER',I3,' OF TABLE NUMBER',I3,
     1 ' CAN NOT BE COMPLETED. THE CURRENT RULE NUMBER IS',I3,
     2 ' FURTHER EXECUTION WILL HAVE TO STOP')
      GO TO 76
C
CC    THE FOLLOWING APPLIES IF THE MISSING INGREDIENT OF THE ACTION
CC    IS ADDRESSED TO SOME TABLE FROM WHICH IT CAN BE RETRIEVED
```

```
C
   35 IF (TABD(IDATA) .NE. 0) GO TO 36
C
CC    ERROR MESSAGE
C
      WRITE (6,181) K,T,IDATA
  181 FORMAT (1H0,15X,'ACTION NUMBER',I3,' OF TABLE ',I3,
     1 ' CANNOT BE COMPLETED BECAUSE DATA NUMBER',I3,' IS NOT ',
     2 'PRESENT.'/16X,'FURTHER EXECUTION OF THIS CYCLE TERMINATED')
      GO TO 76
C
CC    OBTAIN THE MISSING INGREDIENT BY EXECUTING THE TABLE TABD(IR)
C
   36 TABNO = TABD(IDATA)
C
CC    STACK-UP BEFORE STARTING EXECUTION OF ANOTHER TABLE
C
      CALL STAK(STACK,ISTACK,IFLAG,T,K,J,IR,TABNO,IDATA,TRACE)
      GO TO 18
C
CC    THE FOLLOWING APPLIES IN CASE OF DIRECT EXECUTION COMMANDS
CC    THE ADDRESS OF THE TABLE WHICH IS DESIRED TO BE EXECUTED
CC    IS AVAILABLE AT LARRY2(K1)
CC    AT LARRY2(K1)
C
   79 TABNO = LARRY2(K1)
      IR = 0
      IDATA = 0
      CALL STAK(STACK,ISTACK,IFLAG,T,K,J,IR,TABNO,IDATA,TRACE)
      GO TO 18
C
CC    THE FOLLOWING APPLIES IN CASE THE ELSE RULE IS APPLICABLE
C
   89 WRITE (6,189) T
  189 FORMAT (1H0,15X,'ELSE RULE IS APPLICABLE IN TABLE NO.',I4,
     1 '. FURTHER EXECUTION IS NOT POSSIBLE')
      GO TO 76
   59 K = K + 1
      IF (K .GT. N(T)) GO TO 61
      GO TO 58
C
CC    UNSTACKING GOES AS FOLLOWS
C
   61 IF (ISTACK .EQ. 0) GO TO 76
      IFLAG = STACK(ISTACK,1)
      T     = STACK(ISTACK,2)
      J     = STACK (ISTACK,4)
C
CC    IF THE VALUE OF IFLAG IS 1 OR 2, THEN THE STACKING WAS DONE
CC    IN THE CONDITION SECTION; OTHERWISE IN THE ACTION SECTION
C
      GO TO (63,64,65,66),IFLAG
   63 I      = STACK(ISTACK,3)
      IR     = STACK(ISTACK,5)
      GO TO 176
   64 I      = STACK(ISTACK,3)
      KGLOB = STACK(ISTACK,5)
C     CHECK IF A TRACE OF THE UNSTACKING IS DESIRED OR NOT
  176 IF(TRACE .NE. YES) GO TO 67
      WRITE(6,178) T,I,J
  178 FORMAT(1H0,15X,'RESTART EXECUTION OF TABLE ',I3,
     *3X,'AT CONDITION',I3,3X,'OF RULE',I3)
      GO TO 67
   65 IR     = STACK(ISTACK,5)
   66 K      = STACK(ISTACK,3)
C     CHECK IF A TRACE OF THE UNSTACKING IS DESIRED OR NOT
      IF(TRACE .NE. YES) GO TO 67
      WRITE(6,179) T,K,J
  179 FORMAT(1H0,15X,'RESTART EXECUTION OF TABLE ',I3,
     *3X,'AT ACTION',I3,3X,'OF RULE',I3)
   67 ISTACK = ISTACK - 1
C
CC    FOLLOWING IS A CHECK WHETHER THE VALUE OF THE MISSING
CC    INGREDIENT HAS BEEN OBTAINED OR NOT
C
   68 GO TO (70,71,72,59),IFLAG
   70 I1 = IBASE(T,1) + I
      KGLOB = LARRY1(I1)
      IDATA = LARRY5(IR)
      IJ=IBASE(T,3)+(J-1)*N(T)+I
      IF (PRD(IDATA)) GO TO 15
C
CC    ERROR MESSAGE
C
      WRITE (6,116) IDATA,TABD(IDATA),I,T
  116 FORMAT (1H0,10X,'VALUE OF DATA NUMBER',I4,' COULD NOT BE OBTAINED
     1 EVEN BY EXECUTING TABLE NUMBER',I3,/11X,'THIS DATA IS AN ',
     2 'INGREDIENT OF CONDITION NUMBER',I3,' OF TABLE NUMBER',I3/
     3 11X,'FURTHER EXECUTION OF THIS CYCLE IS STOPPED')
      GO TO 76
   71 IJ = IBASE(T,3) + (J-1)*N(T) + I
      IF (PRD(KGLOB)) GO TO 50
C
CC    ERROR MESSAGE
C
      WRITE (6,117) KGLOB,TABD(KGLOB),I,T
  117 FORMAT (1H0,10X,'DATA NUMBER',I5, ' COULD NOT BE ESTABLISHED EVEN
     1 BY EXECUTING TABLE NUMBER',I3/11X,'THIS DATA IS CONDITION ',
     2 'NUMBER',I3,' OF TABLE NUMBER',I3/11X,'FURTHER EXECUTION OF THIS
     3 CYCLE IS STOPPED')
```

```
      GO TO 76
   72 K1 = IBASE(T,2) + K
      KGLOB = LARRY2(K1)
      IDATA = LARRY6(IR)
      IF (PRD(IDATA)) GO TO 16
C
CC    ERROR MESSAGE
C
      WRITE (6,118) IDATA,TABD(IDATA),K,T
  118 FORMAT (1H0,10X,'VALUE OF DATA NUMBER',I4,' COULD NOT BE OBTAINED
     1 EVEN BY EXECUTING TABLE NUMBER',I3,/11X,'THIS DATA IS AN ',
     2 'INGREDIENT OF  ACTION   NUMBER',I3,' OF TABLE NUMBER',I3/
     3 11X,'FURTHER EXECUTION OF THIS CYCLE IS STOPPED')
   76 CALL OUTPUT(ICYCLE)
      ICYCLE = ICYCLE + 1
      GO TO 1
      END
      SUBROUTINE SETUP
C
CC    THIS SUBROUTINE READS THE DECISION TABLES AND THE PROPERTIES
CC    OF DATA FOR PERMANENT STORAGE AND STORES THEM IN COMPACTED FORM
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,NEXSET,MARCA,TRACE,THEMAP
      COMMON /STUPIN/ ICLEAR,IARROW
      COMMON /STINTL/ INDEX,ENTRY,INGR,IDEPND,TMPSET
      DIMENSION
     1 LARRY1(600),LARRY2(600),LARRY3(5000),LARRY4(5000),IBASE(120,4),
     2 LARRY5( 600),LARRY6( 600),IPNTRC(600),IPNTRA(600),
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4IARROW(700)  ,ICLEAR(2000),
     5L(120),M(120),N(120),STACK(20,5),RESULT(2),
     6INDEX(25),ENTRY(25,40),INGR(25,12),IDEPND(700,100),TMPSET(100,20)
      DIMENSION NEXSET(150),MARCA(100)
C
C     INITIALISE THE ARRAYS
C
      CALL INTIAL
      DATA C/'C'/,YES/'YES'/,NO/'NO'/
      IBASE1 = 0
      IBASE2 = 0
      IBASE3 = 0
      IBASE4 = 0
      IBASE5 = 0
      IBASE6 = 0
C
CC    INPUT THE TABLES FIRST FROM DATA SET NUMBER 9
CC    FIRST READ THE TABLE NUMBER AND ITS SIZE
C
      KOUNT = 1
    1 READ (9,101) T,LT,MT,NT
  101 FORMAT (4I10)
C
C     A BLANK CARD SIGNIFIES END OF THE LAST TABLE
C
      IF (T .EQ. 0) GO TO 4
C
CC    CHECK THAT THE TABLE NUMBER HAS NOT EXCEEDED THE DIMENSION OF
CC    ARRAYS L,M,AND N, WHICH IS 120
CC    OTHERWISE GIVE AN ERROR MESSAGE HERE TO THIS EFFECT
C
      IF (T .LE. 120) GO TO 12
      WRITE (6,124) T
  124 FORMAT (20X,'CAREFUL:  YOU ARE EXCEEDING THE DIMENSION OF L,M,N
     * WHICH IS 120.' / 20X, 'CURRENT VALUE OF T IS' ,I4)
      STOP
   12 L(T) = LT
      M(T) = MT
      N(T) = NT
      IF (THEMAP .NE. YES) GO TO 13
C     PRINT OUT THE INFORMATION ON TABLE NUMBER AND ITS SIZE
      WRITE (6,151) T,LT,MT,NT
  151 FORMAT (1H0,10X, 'TABLE NO.',I4,',  LT =',I3,',  MT =',I3,
     1 ',  NT =',I3//)
C
C     NOW READ  THE CONDITION ENTERIES
C
   13 IF (KOUNT .NE. 1) GO TO 14
C
CC    STORE THE DESIGNATION OF THE FIRST TABLE READ IN AS ITABLE
C
      ITABLE = T
      KOUNT = 1000
   14 DO 5 I = 1,NT
      READ (9,102) (INGR(I,J),J=1,6),FLAG,INDEX(I),(ENTRY(I,J),J=1,LT)
  102 FORMAT (T51,5(I4,1X),I4,A1,T1,I5,5X,40I1)
C
C     CHECK IF THE NEXT CARD IS A CONTINUATION OF THIS CARD
C
      IF (FLAG .NE. C) GO TO 17
      READ (9,103) (INGR(I,J),J=7,12)
  103 FORMAT (T51,5(I4,1X),I4)
      GO TO 19
C
```

```
CC    INCASE THE NEXT CARD IS NOT IN CONTINUATION OF THE PREVIOUS CARD
C
   17 DO 18 J = 7,12
      INGR(I,J) = 0
   18 CONTINUE
   19 IF (THEMAP .NE. YES) GO TO 5
C     PRINT THE CONDITION ENTRIES SIMULTANEOUSLY
      WRITE (6,162) (INGR(I,J),J=1,12),INDEX(I),(ENTRY(I,J),J=1,LT)
  162 FORMAT (T61,12I5,T11,I5,5X,40I1)
    5 CONTINUE
C
CC    FILL THE CONDITION STUB IN THE LINEAR ARRAY "LARRY1"
CC    FILL THE CONDITION ENTRIES THE LINEAR ARRAY "LARRY3" COLUMN-WISE
C
CC    THE BASE ADDRESSES FOR THIS TABLE IN LARRY1 AND LARRY3 ARE
CC    AVAILABLE AS IBASE(T,1) AND IBASE(T,3) RESPECTIVELY
C
      IBASE(T,1) = IBASE1
      IBASE(T,3) = IBASE3
C     FIPST FILL THE CONDITION ENTRIES IN LARRY3 COLUMN-WISE
      DO 41 J = 1,LT
C     BASE ADDRESS FOR THE COLUMN
      IJJ = IBASE3 + (J-1)*NT
      DO 41 I = 1,NT
      IJ = IJJ + I
C     CHECK THAT IJ IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY3
      IF (IJ .GT. 5000) GO TO 42
      LARRY3(IJ) = ENTRY(I,J)
   41 CONTINUE
C
CC    FILL LARRY1 WITH CONDITION STUB, LARRY5 WITH INGREDIENTS AND
CC    GENERATE THE DEPENDENCE LIST FOR THE CONDITIONS OF THIS TABLE
C
      DO 48 I = 1,NT
      I1 = IBASE1 + I
C     CHECK THAT I1 IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY1
      IF (I1 .GT. 600) GO TO 43
      LARRY1(I1) = INDEX(I)
C
CC    THE INGREDIENT LIST FOR THIS ROW OF CONDITION STUB HAS ITS
CC    BASE ADDRESS STORED AS FOLLOWS
C
      IPNTRC(I1) = IBASE5
      DO 46 J = 1,12
C     IF THERE IS NO INGREDIENT, IT WILL BE INDICATED BY ZERO
      IF (INGR(I,J) .EQ. 0) GO TO 47
      IJ = IPNTRC(I1) + J
C     CHECK THAT THIS IS NOT MORE THAN THE DIMENSION OF LARRY5
      IF (IJ .GT. 600) GO TO 49
      LARRY5(IJ) = INGR(I,J)
C
CC    GENERATE THE DEPENDENCE ARRAY RIGHT HERE
CC    NOTICE THAT INDEX(I) IS DEPENDENT OF INGR(I,J)
C
      KGLOB = INGR(I,J)
C
CC    SEARCH IF INDEX(I) ALREADY EXISTS IN IDEPND(KGLOB,*)
CC    IF NOT THEN PLACE IT IN; OTHERWISE SKIP IT
C
      DO 45 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ. 0) GO TO 44
      IF (IDEPND(KGLOB,NUM) .EQ. INDEX(I)) GO TO 46
      GO TO 45
   44 IDEPND(KGLOB,NUM) = INDEX(I)
      GO TO 46
   45 CONTINUE
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT THE 100 ELEMENTS
CC    OF IDEPND(KGLOB,*) ARE ALL THERE AND SO ITS DIMENSION
CC    SHOULD BE INCREASED
C
   99 WRITE (6,119) KGLOB
  119 FORMAT (20X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION OF ',
     1 'IDEPND(',I4,',  ). THIS MESSAGE PRINTED BY FORMAT 119' )
      STOP
   46 CONTINUE
   47 IBASE5 = IPNTRC(I1) + J - 1
   48 CONTINUE
C
C     KEEP THE BASE ADDRESS READY FOR THE NEXT TABLE
C
      IBASE1 = IBASE1 + NT
      IBASE3 = IBASE3 + NT*LT
C     READ THE ACTION ENTERIES AND PRINT THEM, IF DESIRED
      IF (THEMAP .NE. YES) GO TO 750
      WRITE (6,153)
  153 FORMAT (1X )
  750 DO 6 K = 1,NT
      READ (9,102) (INGR(K,J),J=1,6),FLAG,INDEX(K),(ENTRY(K,J),J=1,LT)
C     CHECK IF THE NEXT CARD IS A CONTINUATION OF THIS CARD
      IF (FLAG .NE. C) GO TO 25
      READ (9,103) (INGR(K,J),J=7,12)
      GO TO 29
C
CC    INCASE THE NEXT CARD IS NOT IN CONTINUATION OF THE PREVIOUS CARD
C
   25 DO 26 J = 7,12
      INGR(K,J) = 0
   26 CONTINUE
C     PRINT THE ACTION ENTRIES ETC. SIMULTANEOUSLY, IF SO DESIRED
```

```
   29 IF (THEMAP .NE. YES) GO TO 6
      WRITE (6,162) (INGR(K,J),J=1,12),INDEX(K),(ENTRY(K,J),J=1,LT)
    6 CONTINUE
C
CC    FILL THE ACTION STUB IN THE LINEAR ARRAY "LARRY2"
CC    FILL THE ACTION ENTRIES IN THE LINEAR ARRAY "LARRY4" COLUMN-WISE
C
CC    THE BASE ADDRESSES FOR THIS TABLE IN LARRY2 AND LARRY4 ARE
CC    AVAILABLE AS IBASE(T,2) AND IBASE(T,4) RESPECTIVELY
C
      IBASE(T,2) = IBASE2
      IBASE(T,4) = IBASE4
C     FIRST FILL THE ACTION ENTRIES IN THE LARRY4 COLUMN-WISE
      DO 51 J = 1,LT
C     BASE ADDRESS FOR THE COLUMN
      IJJ = IBASE4 + (J-1)*NT
      DO 51 K = 1,NT
      KJ = IJJ + K
C     CHECK THAT KJ IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY4
      IF (KJ .GT. 5000) GO TO 52
      LAPRY4(KJ) = ENTRY(K,J)
   51 CONTINUE
C
CC    FILL LARRY2 WITH ACTION STUB, LARRY6 WITH INGREDIENTS AND
CC    GENERATE THE DEPENDENCE LIST FOR ACTIONS OF THIS TABLE
C
      DO 58 K = 1,NT
      K1 = IBASE2 + K
C     CHECK THAT K1 IS NOT MORE THAN THE DIMENSIONED VALUE OF LARRY2
      IF (K1 .GT. 600) GO TO 53
      LARRY2(K1) = INDEX(K)
C     THE INGPEDIENT LIST FOR THIS ROW OF ACTION STUB IS AS FOLLOWS
      IPNTRA(K1) = IBASE6
      DO 56 J = 1,12
C     IF THERE IS NO INGREDIENT, IT WILL BE INDICATED BY ZERO
      IF (INGR(K,J) .EQ. 0) GO TO 57
      KJ = IPNTRA(K1) + J
C     CHECK THAT THIS IS NOT MORE THAN THE DIMENSION OF LARRY6
      IF (KJ .GT. 600) GO TO 59
      LARRY6(KJ) = INGR(K,J)
C
CC    GENERATE THE DEPENDENCE ARRAY RIGHT HERE
CC    NOTICE THAT INDEX(K) IS A DEPENDENT OF INGR(K,J)
C
      KGLOB = INGR(K,J)
C
CC     SEARCH IF INDEX(K) ALREADY EXISTS IN IDEPND(KGLOB,*)
CC    IF NOT THEN PLACE IT IN; OTHERWISE SKIP IT
C
      DO 55 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ.0) GO TO 54
      IF (IDEPND(KGLOB,NUM) .EQ.INDEX(K)) GO TO 56
      GO TO 55
   54 IDEPND(KGLOB,NUM) = INDEX(K)
      GO TO 56
   55 CONTINUE
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT THE 100 ELEMENTS
CC    OF IDEPND(KGLOB,*) ARE ALL THERE AND SO ITS DIMENSION
CC    SHOULD BE INCREASED
C
      WRITE (6,120) KGLOB
  120 FORMAT (20X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION OF ',
     1 'IDEPND(',I4,',  ). THIS MESSAGE PRINTED BY FORMAT 120' )
      STOP
   56 CONTINUE
   57 IBASE6 = IPNTRA(K1) + J - 1
   58 CONTINUE
C     KEEP THE BASE ADDRESS READY FOR THE NEXT TABLE
      IBASE2 = IBASE2 + NT
      IBASE4 = IBASE4 + NT*LT
C     APPEND LOGICAL DATA TO THE DEPENDENCE LIST
      DO 69 J = 1,LT
      DO 68 K = 1,NT
      IF (ENTRY(K,J) .EQ. 1) GO TO 63
      GO TO 68
C
CC    CHECK IF THIS ACTION STORES ANY VALUE IN ANY LOCATION
C
   63 IF (INDEX(K) .EQ. 0) GO TO 68
C
CC    INDEX(K) IS DEPENDENT OF ALL THE LOGICAL CONDITIONS WHICH ARE
CC    NOT IMMATERIAL FOR THIS RULE
C
      DO 66 I = 1,NT
      IJ = IBASE(T,3) + (J-1)*NT + I
      IF (LAPRY3(IJ) .EQ. 0) GO TO 66
C
CC    OTHERWISE SEARCH IF INDEX(K) IS ALREADY IN THE LIST OF
CC    DEPENDENTS OF THIS CONDITION
C
      I1 = IBASE(T,1) + I
      KGLOB = LARRY1(I1)
      DO 65 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ. 0) GO TO 64
      IF (IDEPND(KGLOB,NUM) .EQ. INDEX(K)) GO TO 66
      GO TO 65
   64 IDEPND(KGLOB,NUM) = INDEX(K)
      GO TO 66
   65 CONTINUE
```

```
C     A NORMAL EXIT IS AN ERROR AS BEFORE
      WRITE (6,121) KGLOB
  121 FORMAT (20X,'CAREFUL: YOU ARE EXCEEDING THE DIMENSION OF ',
     1 'IDEPND(',I4,',  ). THIS MESSAGE PRINTED BY FORMAT 121' )
      STOP
   66 CONTINUE
   68 CONTINUE
   69 CONTINUE
C     TO READ THE NEXT TABLE
      GO TO 1
C
CC    COMPACT THE ARRAY IDEPND"INTO A LINEAR ARRAY AND NAME IT ICLEAR
CC    BECAUSE IT WILL BE USED IN CLEARING THE EFFECT OF CHANGES OF DATA.
CC    EACH ELEMENT IN THE LIST OF DATA WILL HAVE AN ARROW POINTING INTO
CC    "ICLEAR"; THESE ARROWS ARE STORED AS IARROW(KGLOB)
C
CC    RANGES OF THE TWO DO LOOPS BELOW ARE SAME AS THE
CC    TWO DIMENSIONS OF IDEPND
C
    3 J1 = 0
      DO 80 KGLOB = 1,700
      IARROW(KGLOB) = J1
C
CC    CHECK IF KGLOB BELONGS TO A SET
C
      IF (ISET(KGLOB) .EQ. 0) GO TO 82
      NSET = ISET(KGLOB)
C
CC    CHECK IF KGLOB IS THE FIRST ELEMENT OF THE SET.
CC    IF YES THEN IT GOES INTO ICLEAR; OTHERWISE NOT
C
      I1 = MARCA(NSET) + 1
      IF (NEXSET(I1) .EQ. KGLOB) GO TO 282
C
CC    MAKE THE DEPENDENTS OF KGLOB SAME AS THAT OF THE FIRST
CC    ELEMENT OF THE SET
C
      JGLOB = NEXSET(I1)
      DO 202 NUM = 1,100
      IF (IDEPND(JGLOB,NUM) .EQ. 0) GO TO 80
      IDEPND(KGLOB,NUM) = IDEPND(JGLOB,NUM)
  202 CONTINUE
      GO TO 80
C
CC    FILL THE DEPENDENTS OF THE FIRST ELEMENT OF THE SET WITH
CC    ALL THE POSSIBLE DEPENDENTS INDICATED FOR THE ELEMENTS
CC    OF THE SET
C
  282 N1 = MARCA(NSET) + 2
      N2 = MARCA(NSET + 1)
      DO 220 ID = N1,N2
      NN = NEXSET(ID)
      DO 215 JN = 1,100
      IF (IDEPND(NN,JN) .EQ. 0) GO TO 220
C
CC    SEARCH IF THIS DEPENDENT IS ALREADY IN THE LIST OF
CC    DEPENDENTS OF THE FIRST ELEMENT
C
      DO 210 NUM = 1,100
      IF (IDEPND(KGLOB,NUM) .EQ. 0) GO TO 205
      IF (IDEPND(KGLOB,NUM) .EQ. IDEPND(NN,JN)) GO TO 215
      GO TO 210
  205 IDEPND(KGLOB,NUM) = IDEPND(NN,JN)
      GO TO 215
  210 CONTINUE
  215 CONTINUE
  220 CONTINUE
   82 DO 70 J = 1,100
      IF (IDEPND(KGLOB,J) .EQ. 0) GO TO 80
      J1 = J1 + 1
C
CC    CHECK THAT J1 IS NOT MORE THAN THE DIMENSIONED VALUE OF
CC    THE ARRAY "ICLEAR" WHICH CURRENTLY IS 2,000
C
      IF (J1 .GT. 2000) GO TO 33
      ICLEAR(J1) = IDEPND(KGLOB,J)
      GO TO 70
   33 WRITE (6,133)
  133 FORMAT (1H0,10X,'CAREFUL; YOU ARE EXCEEDING THE DIMENSION',
     1 ' OF THE ARRAY "ICLEAR" WHICH IS 2000'/10X,'REMEDY IS TO',
     2 ' INCREASE THIS DIMENSION')
      STOP
   70 CONTINUE
   80 CONTINUE
      GO TO 95
C
CC    FILL-UP THE BLANKS IN THE ARRAY  IBASE
C
    4 DO 444 T = 1,120
      IF (IBASE(T,1) .NE. 0 .OR. T .EQ. ITABLE) GO TO 444
      IBASE(T,1) = IBASE1
      IBASE(T,2) = IBASE2
      IBASE(T,3) = IBASE3
      IBASE(T,4) = IBASE4
  444 CONTINUE
C
CC    FILL-UP THE BLANKS AT THE TAIL END OF THE ARRAY IPNTRC
C
      I1 = I1 + 1
      DO 445 I = I1,600
```

```
      IPNTRC(I) = IBASE5
  445 CONTINUE
C
CC    FILL-UP THE BLANKS AT THE TAIL END OF THE ARRAY IPNTRA
C
      K1 = K1 + 1
      DO 446 K = K1,600
      IPNTRA(K) = IBASE6
  446 CONTINUE
C
CC    INPUT PROPERTIES OF THE ELEMENTS OF DATA E.G. ADDRESSES
CC    OF TABLE NUMBERS FROM WHICH THEY CAN BE DERIVED
CC    AND THEIR MEMBERSHIP OF SETS ETC. FROM DATA SET NUMBER 9
C
      IF (THEMAP .NE. YES) GO TO 2
C
CC    HEADING FOR THE NEXT OUTPUT
C
      WRITE (6,154)
  154 FORMAT (1H1,30X, 'KGLOB',10X, 'TABDK',10X, 'NSET'//)
    2 READ (9,105) KGLOB,TABDK,NSET
  105 FORMAT (3I10)
C     A BLANK IS A SIGNAL OF END OF THIS DATA
      IF (KGLOB .EQ.0) GO TO 15
      TABD(KGLOB) = TABDK
      ISET(KGLOB) = NSET
C
CC    PRINT THIS INFORMATION ABOUT THE ELEMENTS OF DATA, IF DESIRED
C
      IF (THEMAP .NE. YES) GO TO 73
      WRITE (6,155) KGLOB,TABDK,NSET
  155 FORMAT (25X,3(I10,5X))
C
CC    GENERATE THE MUTUALLY EXCLUSIVE SETS IN A TEMPERORARY ARRAY
CC    AND NAME IT THPSET
C
   73 IF (NSET .EQ. 0) GO TO 2
      DO 75 J = 1,20
      IF (THPSET(NSET,J) .EQ. 0) GO TO 74
      IF (THPSET(NSET,J) .EQ. KGLOB) GO TO 2
      GO TO 75
   74 THPSET(NSET,J) = KGLOB
      GO TO 2
   75 CONTINUE
C
CC    A NORMAL EXIT FROM THIS LOOP INDICATES THAT THE TEN ELEMENTS
CC    OF THPSET(NSET,*) ARE FULL AND ITS DIMENSION SHOULD BE INCREASED
C
      WRITE (6,175) NSET
  175 FORMAT (10X,'CAREFUL; YOU ARE EXCEEDING THE DIMENSION OF THPSET(',
     1 I2,',*). THIS MESSAGE GENERATED BY FORMAT NUMBER 175')
      STOP
C
CC    COMPACT THE CONTENTS OF THPSET IN ARRAY NEXSET.
CC    POINTER FROM MUTUALLY EXCLUSIVE SET TO NEXSET IS NAMED AS MARCA
C
   15 J1 = 0
      DO 78 NSET = 1,100
      MARCA(NSET) = J1
      DO 76 J = 1,20
      IF (THPSET(NSET,J) .EQ. 0) GO TO 78
      J1 = J1 + 1
C
CC    CHECK THAT J1 IS NOT MORE THAN THE DIMENSIONED VALUE
CC    OF THE ARRAY NEXSET WHICH IS CURRENTLY 150
C
      IF (J1 .GT. 150) GO TO 79
      NEXSET(J1) = THPSET(NSET,J)
      GO TO 76
   79 WRITE (6,179)
  179 FORMAT (1H0,10X,'CAREFUL; YOU ARE EXCEEDING THE DIMENSION ',
     1 'OF THE ARRAY NEXSET WHICH IS 150'/
     211X,'REMEDY IS TO INCREASE THIS DIMENSION')
      STOP
   76 CONTINUE
   78 CONTINUE
C
CC    NOW TO COMPACT THE ARRAY IDEPND
C
      GO TO 3
C     READ WHICH TABLE MUST BE EXECUTED FIRST
   95 READ (5,107) TFIRST
  107 FORMAT (I10)
      GO TO 39
   42 WRITE (6,142)
  142 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY3',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 5000'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   43 WRITE (6,143)
  143 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY1',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   49 WRITE (6,149)
  149 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY5',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   52 WRITE (6,152)
```

```
  152 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY4',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 5000'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   53 WRITE (6,150)
  150 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY2',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
   59 WRITE (6,159)
  159 FORMAT (1H0,10X,'CAREFUL, THE DIMENSION OF THE ARRAY LARRY6',
     1 ' IS BEING EXCEEDED. CURRENT SPECIFIED DIMENSION = 600'/
     2 11X,'REMEDY IS TO INCREASE THIS DIMENSION. JOB TERMINATED')
      STOP
C
CC    CHECK IF THE MAP OF THE PERMANENT STORAGE IS DESIRED
C
   39 IF (THEMAP .NE. YES) GO TO 999
      WRITE (6,500) (LARRY1(I),I=1,600)
  500 FORMAT (1H1,5X,'LARRY1'//30(5X,10I5,5X,10I5/))
      WRITE (6,501) (LARRY3(I),I=1,5000)
  501 FORMAT (1H1,5X,'LARRY3'//50(5X,10(10I1,1X)/))
      WRITE (6,502) (LARRY2(I),I=1,600)
  502 FORMAT (1H1,5X,'LARRY2'//30(5X,10I5,5X,10I5/))
      WRITE (6,503) (LARRY4(I),I=1,5000)
  503 FORMAT (1H1,5X,'LARRY4'//50(5X,10(10I1,1X)/))
      WRITE (6,505) (J,(IBASE(I,J),I=1,120),J=1,4)
  505 FORMAT (1H1/4(5X,'IBASE',I1//6(5X,10I5,5X,10I5/)////))
      WRITE(6,506) (LARRY5(I),I=1,600)
  506 FORMAT (1H1,5X,'LARRY5'//30(5X,10I5,5X,10I5/))
      WRITE (6,507) (LARRY6(I),I=1,600)
  507 FORMAT (1H1,5X,'LARRY6'//30(5X,10I5,5X,10I5/))
      WRITE (6,508) (IPNTRC(I),I=1,600)
  508 FORMAT (1H1,5X,'IPNTRC'//30(5X,10I5,5X,10I5/))
      WRITE (6,509) (IPNTRA(I),I=1,600)
  509 FORMAT (1H1,5X,'IPNTRA'//30(5X,10I5,5X,10I5/))
      WRITE (6,511)
  511 FORMAT (1H1,10X,'NON COMPACTED LIST OF DEPENDENTS'//
     1 T10,'DATA',T17,'NO. OF',T30,'LIST OF DEPENDENTS'/
     2 T10,'NO.',T15,'DEPENDENTS'//)
      DO 514 I = 1,700
      DO 513 J = 1,100
      IF (IDEPND(I,J) .EQ. 0) GO TO 613
  513 CONTINUE
  613 IF (J .EQ. 1) GO TO 514
      J1 = J-1
      WRITE (6,512) I,J1,(IDEPND(I,J),J=1,J1)
  512 FORMAT (10X,I3,5X,'(',I2,')',4(T30,4(5I4,2X)/))
  514 CONTINUE
      WRITE (6,520) (IARROW(I),I=1,700)
  520 FORMAT (1H1,5X,'IARROW'/35(5X,10I5,5X,10I5/))
      WRITE (6,521) (ICLEAR(I),I=1,2000)
  521 FORMAT (1H1,5X,'ICLEAR'//20(5(5X,10I5,5X,10I5/)/))
      WRITE (6,522) ((THPSET(I,J),J=1,20),I=1,100)
  522 FORMAT (1H1,20X,'THPSET'//5 (10(20X,10I5/)/))
      WRITE (6,523) (HEXSET(I),I=1,150)
  523 FORMAT (1H1,20X,'HEXSET'/15(20X,10I5/))
      WRITE (6,524) (HARCA(I),I=1,100)
  524 FORMAT (1H0,20X'HARCA'/5(20X,10I5/))
      WRITE(6,525) (ISET(I),I=1,700)
  525 FORMAT(1H1,5X,'ISET'/35(5X,10I5,5X,10I5/))
  999 WRITE (6,172) TFIRST
  172 FORMAT (1H0,15X,'EXECUTION WILL START WITH TABLE NO.',I4)
      WRITE(8) LARRY1,LARRY2,LARRY5,LARRY6,IPNTRC,IPNTRA
      WRITE(8) LARRY3
      WRITE(8) LARRY4
      WRITE(8) ISET,TABD,L,M,N,TFIRST,IBASE
      WRITE(8) HEXSET,HARCA,TRACE,ICLEAR,IARROW
      RETURN
      END
      SUBROUTINE INTIAL
CC    THIS SUBROUTINE IS USED TO INITIALIZE THE VARIOUS ARRAYS
CC    USED BY THE PROGRAM
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,THPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,HEXSET,HARCA,TRACE
      COMMON /STUPIN/ ICLEAR,IARROW
      COMMON /STINTL/ INDEX,ENTRY,INGR,IDEPND,THPSET
      DIMENSION
     1 LARRY1(600),LARRY2(600),LARRY3(5000),LARRY4(5000),IBASE(120,4),
     2 LARRY5( 600),LARRY6( 600),IPNTRC(600),IPNTRA(600),
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4 IARROW(700) ,ICLEAR(2000),
     4 L(120),M(120),N(120),STACK(20,5),RESULT(2),
     5 INDEX(25),ENTRY(25,40),INGR(25,12),IDEPND(700,100),THPSET(100,20)
      DIMENSION HEXSET(150),HARCA(100)
      DO 10 I = 1,600
      LARRY1(I) = 0
      LARRY2(I) = 0
      IPNTRC(I) = 0
      IPNTRA(I) = 0
   10 CONTINUE
      DO 20 I = 1,700
      DATA(I)   = 0.
      PRD(I)    = .FALSE.
      TABD(I)   = 0
      ISET(I) = 0
```

```
      IARROW(I) = 0
      DO 22 J = 1,100
      IDEPND(I,J) = 0
   22 CONTINUE
   20 CONTINUE
      DO 21 I = 1,1000
      LARRY5(I) = 0
      LARRY6(I) = 0
   21 CONTINUE
      DO 30 I = 1,5000
      LARRY3(I) = 0
      LARRY4(I) = 0
   30 CONTINUE
      DO 31 I = 1,2000
      ICLEAR(I) = 0
   31 CONTINUE
      DO 40 I = 1,100
      NARCA(I) = 0
      DO 40 J = 1,20
      THPSET(I,J) = 0
   40 CONTINUE
      DO 41 I = 1,120
      L(I) = 0
      M(I) = 0
      N(I) = 0
      DO 41 J = 1,4
      IBASE(I,J) = 0
   41 CONTINUE
      DO 50 I = 1,25
      INDEX(I) = 0
      DO 48 J = 1,40
      ENTRY(I,J) = 0
   48 CONTINUE
      DO 49 J = 1,12
      INGR(I,J) = 0
   49 CONTINUE
   50 CONTINUE
      DO 60 I = 1,20
      DO 60 J = 1,5
      STACK(I,J) = 0
   60 CONTINUE
      DO 70 I = 1,150
      NEXSET(I) = 0
   70 CONTINUE
      RETURN
      END
      SUBROUTINE INPUT(ICYCLE)
CC    THIS SUBROUTINE IS USED FOR READING THE DATA VALUES AND FOR
CC    CLEARING THE DEPENDENT DATA FOR SECOND OR SUBSEQUENT CYCLES
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,THPSET
      COMMON /NICA/DATA,PRD
      COMMON/MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTEC,IPNTRA,TABD,L,M,N,T,TFIRST
      COMMON /MNSTIN/ISET,NEXSET,NARCA,TRACE,THEMAP
      COMMON /STUPIN/ ICLEAR,IARROW
      COMMON/STINTL/INDEX,ENTRY,INGR,IDEPND,THPSET
      COMMON/DOOP/IPDL
      COMMON/DUMB/WARAY
      DIMENSION
     3 DATA(700),PRD(700),TABD(700),ISET(700),
     4 IARROW(700) ,ICLEAR(2000),
     5 IPDL(20,2)
      DIMENSION NEXSET(150),NARCA(100)
C
C     READ THE NUMERICAL DATA FOR ALL THE TABLES
C     HEADING FOR THE NEXT PRINTED ITEM
C
    9 WRITE (6,170) ICYCLE
  170 FORMAT (1H1,15X,'THE FOLLOWING NUMERICAL DATA HAS BEEN SUPPLIED ',
     1 'FOR CYCLE NUMBER',I3//31X,'KGLOB',10X,'DATAK'//)
    1 READ (5,103,END=9999) KGLOB,DATAK
  103 FORMAT (I5,5X,F10.0)
C     SIGNAL LAST CARD IN THIS DATA BY A BLANK CARD
      IF (KGLOB .EQ. 0) GO TO 15
C
CC    IF THE DATA BELONGS TO A SET, FIX THE VALUE OF ITS
CC    ELEMENTS FIRST
C
      IF (ISET(KGLOB) .NE. 0) CALL SETS(KGLOB)
      DATA(KGLOB) = DATAK
      PRD(KGLOB) = .TRUE.
      WRITE (6,171) KGLOB,DATAK
  171 FORMAT (25X,I10,5X,F10.4)
C
CC    IF THIS IS THE FIRST CYCLE THEN GO TO READ THE NEXT CARD;
CC    OTHERWISE CLEAR THE EFFECT OF CHANGING THIS DATA FIRST
C
      IF (ICYCLE .EQ. 1) GO TO 1
C
CC    FOR THE PURPOSE OF CLEARING THE EFFECTS OF READING NEW DATA, THE
CC    LIST OF DEPENDENTS IS STORED WITH THE FIRST ELEMENT OF THE SET
C
      NSET = ISET(KGLOB)
CD    WRITE(6,8) KGLOB,NSET
CD  8 FORMAT(1X,'KGLOB=',I4,8X,'NSET=',I4)
      IF (NSET .EQ. 0) GO TO 11
      KGLOB = NARCA(NSET) + 1
CD    WRITE(6,10) KGLOB
CD 10 FORMAT(1X,'KGLOB****=',I4)
```

```
C
CC    FIRST FIND OUT IF THIS DATA HAS ANY DEPENDENT DATA.IF NO THEN READ
CC    THE NEXT DATA; IF YES THEN CLEAR IT
C
   11 IF ((IARROW(KGLOB+1) - IARROW(KGLOB)) .EQ. 0) GO TO 1
C
CC    START CLEARING
C
      IZ = 0
   12 LG = 1
   13 I1 = IARROW(KGLOB) + LG
      KDEP = ICLEAR(I1)
      IF (.NOT.PRD(KDEP)) GO TO 21
      PRD(KDEP) = .FALSE.
CD    WRITE(6,2000) KDEP
C2000 FORMAT(1X,'KDEP=',I4)
C
CC    CHECK IF DATA ITEM KDEP HAS ANY DEPENDENT COMPONENTS. IF YES THEN
CC    SUSPEND CLEARING DEPENDENTS OF KGLOB BY STACKING PROCEDURE AND
CC    START CLEARING DEPENDENTS OF KDEP; OTHERWISE CONTINUE CLEARING
CC    DEPENDENTS OF KGLOB
C
      IF ((IARROW(KDEP+1) - IARROW(KDEP)) .EQ. 0) GO TO 21
      IZ = IZ + 1
      IPDL(IZ,1) = KGLOB
      IPDL(IZ,2) = LG
CD    WRITE(6,2001) KGLOB,LG
C2001 FORMAT(1X,'KGLOB=',I4,5X,'LG=',I3)
      KGLOB = KDEP
      GO TO 12
   21 LG = LG + 1
CD    WRITE(6,2002) LG
C2002 FORMAT(1X,'LG IN LOOP 21=',I3)
      IF (LG .LE. (IARROW(KGLOB+1) - IARROW(KGLOB))) GO TO 13
C
CC    ALL THE DEPENDENT DATA FOR THIS KGLOB HAVE BEEN CLEARED AND SO
CC    UNSTACKING CAN BE STARTED
CD    WRITE(6,2003) IZ
C2003 FORMAT(1X,'IZ BEFORE STATEMENT 20=',I3)
   20 IF (IZ .EQ. 0) GO TO 1
      KGLOB = IPDL(IZ,1)
      LG    = IPDL(IZ,2)
      IZ    = IZ - 1
CD    WRITE(6,2004) KGLOB,LG,IZ
C2004 FORMAT(1X,'VALUES AFTER STATEMENT 20,KGLOB=',I4,
CD   13X,'LG=',I3,3X,'IZ=',I3)
      GO TO 21
C
CC    READ IF A TRACE OF THE TABLES EXECUTED IS DESIRED OR NOT
C
   15 READ (5,105) TRACE
  105 FORMAT (T11,A4)
C     READ IF WRITING OUT OF THE FOLLOWING ARRAYS IS DESIRED:
C     LARRY1,LARRY3,LARRY2,LARRY4,LARRY5,LARRY6
C     IBASE,IPNTRC,IPNTRA,IARROW,ICLEAR,MEXSET,
C     MARCA,ISET,L
C
      READ(5,108) NARAY
  108 FORMAT(T11,A4)
      RETURN
 9999 WRITE(6,2010)
 2010 FORMAT(1X,'EXECUTION OF PROGRAM IS COMPLETED')
      WRITE(7,2012)
 2012 FORMAT(1X,'EXECUTION OF PROGRAM IS COMPLETED.'/
     *1X,'COLLECT YOUR OUTPUT FROM THE COMPUTING CENTER.COME BACK SOON')
      STOP
      END
      SUBROUTINE STAK(STACK,ISTACK,IFLAG,T,I,J,IR,TABNO,IDATA,TRACE)
C
CC    THIS SUBROUTINE PERFORMS STACKING OF DECISION TABLES FOR
CC    CONDITIONAL EXECUTION AND GENERATES MESSAGES TO THIS EFFECT
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ISTACK,IFLAG,T,I,J,IR,TABNO,IDATA,TABD
      COMMON/MNSTUP/LARRY1,LARRY2,LARRY3,LARRY4,LARRY5,LARRY6,IBASE,
     1 IPNTRC,IPNTRA,TABD,L,M,N
      DIMENSION STACK(20,5),L(120),LARRY1(600),LARRY2(600),
     1LARRY3(5000),LARRY4(5000),LARRY5(600),LARRY6(600),
     2IBASE(120,4),IPNTRC(600),IPNTRA(600),TABD(700),
     3M(120),N(120)
      DATA YES/'YES'/,NO/'NO'/
      ISTACK = ISTACK + 1
      STACK(ISTACK,1) = IFLAG
      STACK(ISTACK,2) = T
      STACK(ISTACK,3) = I
      STACK(ISTACK,4) = J
      STACK(ISTACK,5) = IR
C     CHECK IF A TRACE OF THE STACKING INFORMATION IS DESIRED OR NOT
      IF (TRACE .NE. YES) GO TO 10
C
C     PRINT THE APPROPRIATE MESSAGE
C
      GO TO (4,5,6,7),IFLAG
C
CC    MISSING INGREDIENT OF A CONDITION
C
    4 WRITE (6,104) T,I,J,IDATA,TABNO
  104 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT CONDITION',I3,' OF RULE',I3/11X,' REASON: MISSING ',
     2 'INGREDIENT CORRESPONDING TO DATA NUMBER ',I3/
     3 11X,' STARTED EXECUTION OF TABLE ',I3)
```

```
      GO TO 10
C
CC    MISSING VALUE OF THE CONDITION ITSELF
C
    5 WRITE (6,105)T,I,J,TABNO,IR
  105 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT CONDITION',I3,' OF RULE',I3/11X,' STARTED EXECUTION',
     2 ' OF TABLE ',I3,' TO OBTAIN VALUE OF DATA NUMBER',I4)
      GO TO 10
C
CC    MISSING INGREDIENT OF AN ACTION
C
    6 WRITE (6,106)T,I,J,IDATA,TABNO
  106 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT ACTION',I3,' OF RULE',I3/11X,' REASON: MISSING ',
     2 'INGREDIENT CORRESPONDING TO DATA NUMBER ',I3/
     3 11X,' STARTED EXECUTION OF TABLE ',I3)
      GO TO 10
C.
CC    DIPECT EXECUTION
C
    7 WRITE (6,107) T,I,J,TABNO
  107 FORMAT (1H0,10X,'SUSPENDED EXECUTION OF TABLE ',I3,
     1 ' AT ACTION',I3,' OF RULE',I3/11X,' STARTED EXECUTION',
     2 ' OF TABLE ',I3,' FOR DIRECT EXECUTION')
   10 T = TABNO
C     IF TABLE T DOES NOT EXIST ,TERMINATE EXECUTION.
CD    WRITE(7,110) L(T)
CD110 FORMAT(1X,'L(T)=',I4)
      IF(L(T).EQ.0) GO TO 9999
      TABNO = 0
      RETURN
 9999 WRITE(7,109) T
      WRITE(6,109) T
  109 FORMAT(1X,'TABLE NUMBER',2X,I4,'DOES NOT EXIST,',
     *' EXECUTION TERMINATED.')
      STOP
      END
      SUBROUTINE OUTPUT(ICYCLE)
C
CC    THROUGH THIS SUBROUTINE, IT IS POSSIBLE TO OUTPUT ANY DATA
CC    VALUE FOR CHECKING AND DIOGNASIS
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      DIMENSION DATA(700),PRD(700)
      WRITE (6,100) ICYCLE
  100 FORMAT (1H1,15X,'DATA VALUES AT THE END OF CYCLE NO.',I3/
     1 16X,'ONLY THAT DATA WHICH HAS A VALUE IS REPRODUCED HERE'
     2//31X,'KGLOB',10X,'DATAK',10X,'PRD'//)
      DO 801 KGLOB = 1,700
      IF (.NOT. PRD(KGLOB)) GO TO 801
      WRITE (6,800) KGLOB,DATA(KGLOB),PRD(KGLOB)
  800 FORMAT (25X,I10,1X,F14.4,5X,L7)
  801 CONTINUE
      RETURN
      END
      SUBROUTINE SETS(KGLOB)
C
CC    THIS SUBROUTINE IS USED TO EVALUATE THE DATA IN MUTUALLY
CC    EXCLUSIVE SETS AT THE TIME OF EXTERNAL INPUT
C
CC    DECLARATIONS
C
      IMPLICIT LOGICAL*1 (P), INTEGER*2 (I-N)
      INTEGER*2 STACK,ENTRY,T,TABNO,TABD,TABDK,TFIRST,TMPSET
      COMMON /MICA/DATA,PRD
      COMMON /MNSTIN/ISET,MEXSET,MARCA
      DIMENSION DATA(700),PRD(700),ISET(700)
      DIMENSION MEXSET(150),MARCA(100)
C
CC    CHECK IF KGLOB BELONGS TO A SET. IF YES THEN
CC    FIX THE VALUES OF ALL THE ELEMENTS OF THE SET TO NO
C
      NSET = ISET(KGLOB)
      IF (NSET .EQ. 0) GO TO 99
      I1 = MARCA(NSET) + 1
      I2 = MARCA(NSET + 1)
      DO 10 I = I1,I2
      IGLOB = MEXSET(I)
    3 DATA(IGLOB) = 0.0
    4 PRD(IGLOB)  = .TRUE.
   10 CONTINUE
      DATA(KGLOB)=1.0
   99 RETURN
      END
```