DOCUMENTATION OF DECISION-AIDING SOFTWARE:

DECISION SYSTEM SPECIFICATION

DECISIONS AND DESIGNS INC.

Linda B. Allardyce
Dorothy M. Amey
Phillip H. Feuerwerger
Roy M. Gulick

November 1979

N00014-79-C-0069

ADVANCED DECISION TECHNOLOGY PROGRAM

CYBERNETICS TECHNOLOGY OFFICE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY
Office of Naval Research • Engineering Psychology Programs

DOCUMENTATION OF DECISION-AIDING SOFTWARE:
DECISION SYSTEM SPECIFICATION

by

Linda B. Allardyce, Dorothy M. Amey, Phillip H. Feuerwerger, and Roy M. Gulick

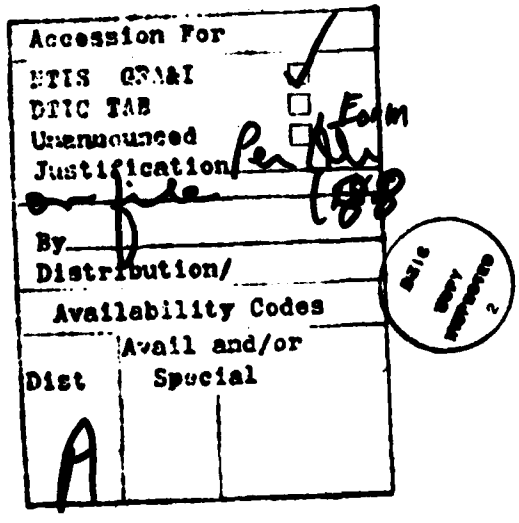
Sponsored by

Defense Advanced Research Projects Agency
ARPA Order 3469

November 1979

DECISIONS and DESIGNS, INC.
Suite 600, 8400 Westpark Drive
P.O.Box 907
McLean, Virginia 22101
(703) 821-2828

CONTENTS



FIGURES

DECISION SYSTEM SPECIFICATION

1.0 INTRODUCTION

1.1 Purpose of the System Specification

The DECISION System Specification is a technical document written for software development personnel. Together with the DECISION Functional Description, it guides the software development effort by identifying the functional requirements and by providing structured logic diagrams that depict the flow, control, and processing of information within the system.

The System Specification is generic and is intended to guide and facilitate the preparation of the language-specific program documentation and coding that are necessary to implement and operate DECISION at an installation.

1.2 References

1.2.1 IBM, HIPO--A Design Aid and Documentation Technique. Technical Publication GC20-1851-0. White Plains, New York: IBM, October 1974.

1.2.2 Allardyce, Linda B.; Amey, Dorothy M.; Feuerwerger, Phillip H.; Gulick, Roy M. Documentation of Decision-Aiding Software: DECISION Functional Description. McLean, Virginia: Decisions and Designs, Inc., November 1979.

1.2.3 Allardyce, Linda B.; Amey, Dorothy M.; Feuerwerger, Phillip H.; Gulick, Roy M. Documentation of Decision-Aiding Software: DECISION Users Manual. McLean, Virginia: Decisions and Designs, Inc., November 1979.

1.3 Terms

1.3.1 DECISION - DECISION is an abbreviation for Decision Tree Models, reflecting the system's major area of applicability.

1.3.2 HIPO - The Specification uses the standard Hierarchy plus Input-Process-Output (HIPO) diagramming technique to depict the structural design and logical flow of the system. A legend explaining the HIPO diagramming symbols is included. Reference 1.2.1 provides a complete description of the HIPO documentation technique.

2.0 DESIGN DETAILS

2.1 Background

Systems development personnel should refer to the DECISION Functional Description, reference 1.2.2, in conjunction with the documentation contained in this Specification. The Functional Description details the decision tree model implemented by DECISION and discusses the specific functions that the software performs. In addition, systems development personnel may wish to refer to the DECISION Users Manual, reference 1.2.3.

2.2 General Operating Procedures

DECISION is a menu-driven system. That is, the system is designed to interact with the user by presenting a sequential hierarchy of menus and asking the user to respond by selecting one option from the current menu. If the user does not select one of the menu options, the system displays the previous menu. In this manner, the user moves up and down the hierarchy, as desired. Whenever data entry is required as a result of option selection, the system specifically requests the data and specifies the format.

The system is also designed to anticipate and be generally forgiving of procedural errors by the user.

2.3 System Logical Flow

DECISION is a hierarchically structured, modular system. The system structure and logical flow lends itself to presentation in the form of HIPO diagrams, which are contained in this document.

The main purpose of the HIPO diagrams is to provide, in a pictorial manner, the complete set of modular elements necessary to the operation of DECISION including all input, output, and internal functional processing. This is done by displaying input items to the process step which uses them, defining the process, and showing the resulting output of the process step.

The documentation diagrams are designed and drawn in a hierarchical fashion from the main calling routines to the detail-level operation/calculation routines. Extended written descriptions are given below a HIPO diagram whenever it is deemed necessary.

A complete explanation of the symbolic notation used in the HIPO diagrams is given in reference 1.2.1. An abbreviated legend for the symbols used in this specification is given in Figure 2-1. Note that:

a. External subroutines appear partly in the process block and partly out. Internal subroutines are shown within the process block.

b. Overview diagrams show general inputs and outputs only, whereas detail/subroutine-level diagrams show specific input/output tables and/or displays.

c. Rectangular boxes inside the input/output block areas are generally used to denote single data items. Two or more boxes are grouped to show that several data items are input/output.

d. Rectangular boxes inside the process block indicate repetitive subprocesses.

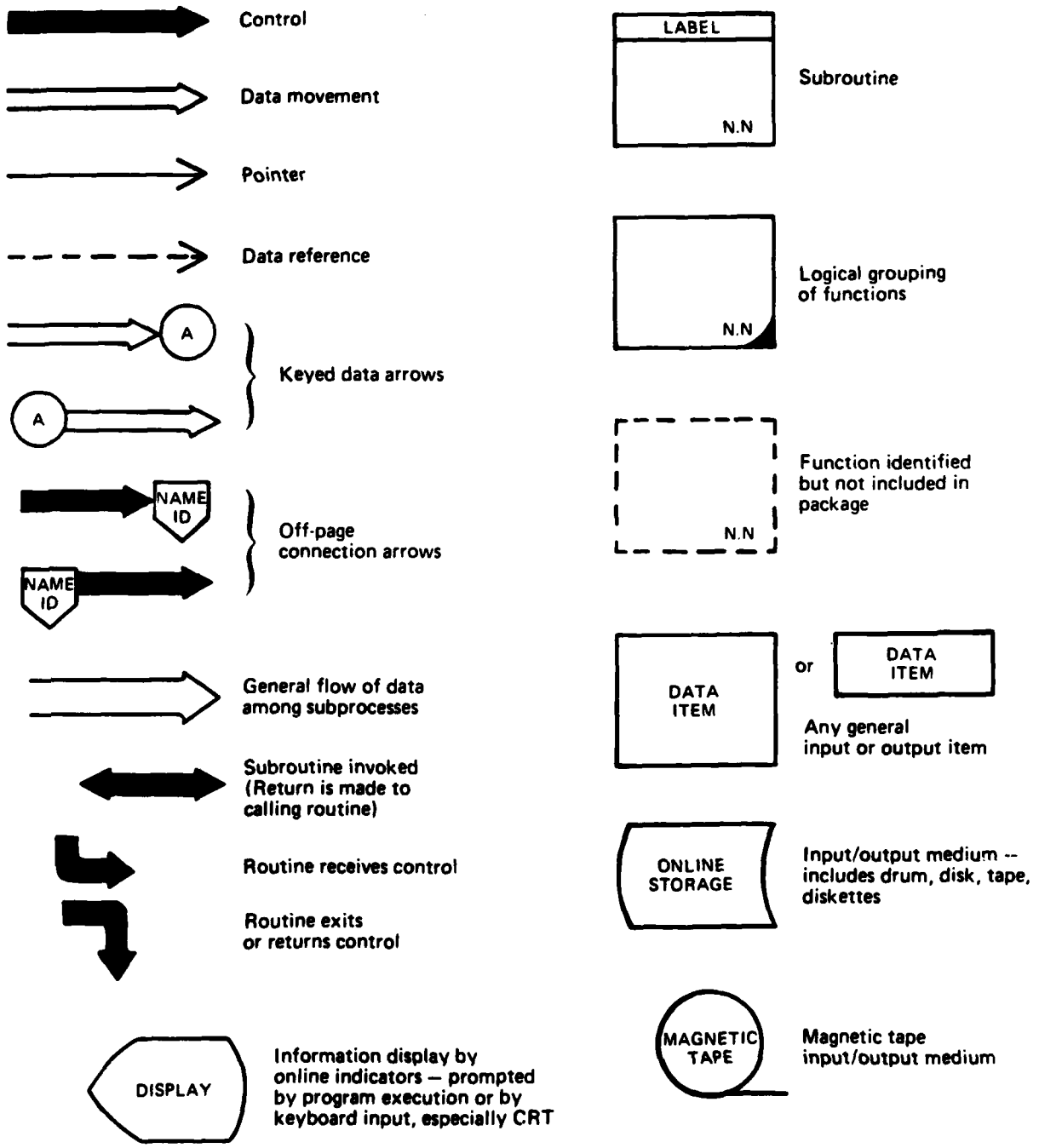


Figure 2-1
LEGEND OF HIPO SYMBOLS

The HIPO diagrams appear in the next section, which completes the System Specification.

2.4 HIPO Documentation

The HIPO diagram identification numbers and figure numbers used in this section stand alone; i.e., they start with 1.0, increase hierarchically, and are independent of the numbering scheme used to this point in this document.

The DECISION system comprises two subsystems: STRUCTURE, which builds and refines the decision tree model, and RUN, which produces various results based on the model and its data. Figure 2-2 is the system structure chart. Figures 2-3 and 2-4 are the subsystem charts and represent the overall program logic flows in visual tables of contents. The Visual Tables of Contents show the hierarchical structure, the functional description labels, and the diagram (chart) identifiers of the functions implemented by the DECISION subsystems.

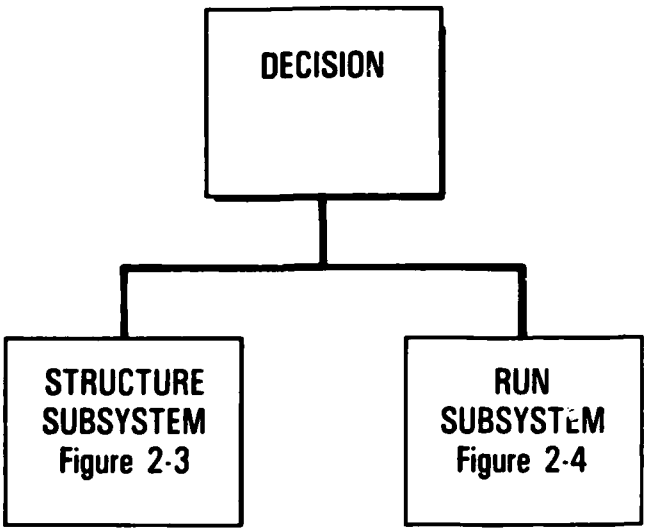


Figure 2-2
DECISION SYSTEM STRUCTURE CHART

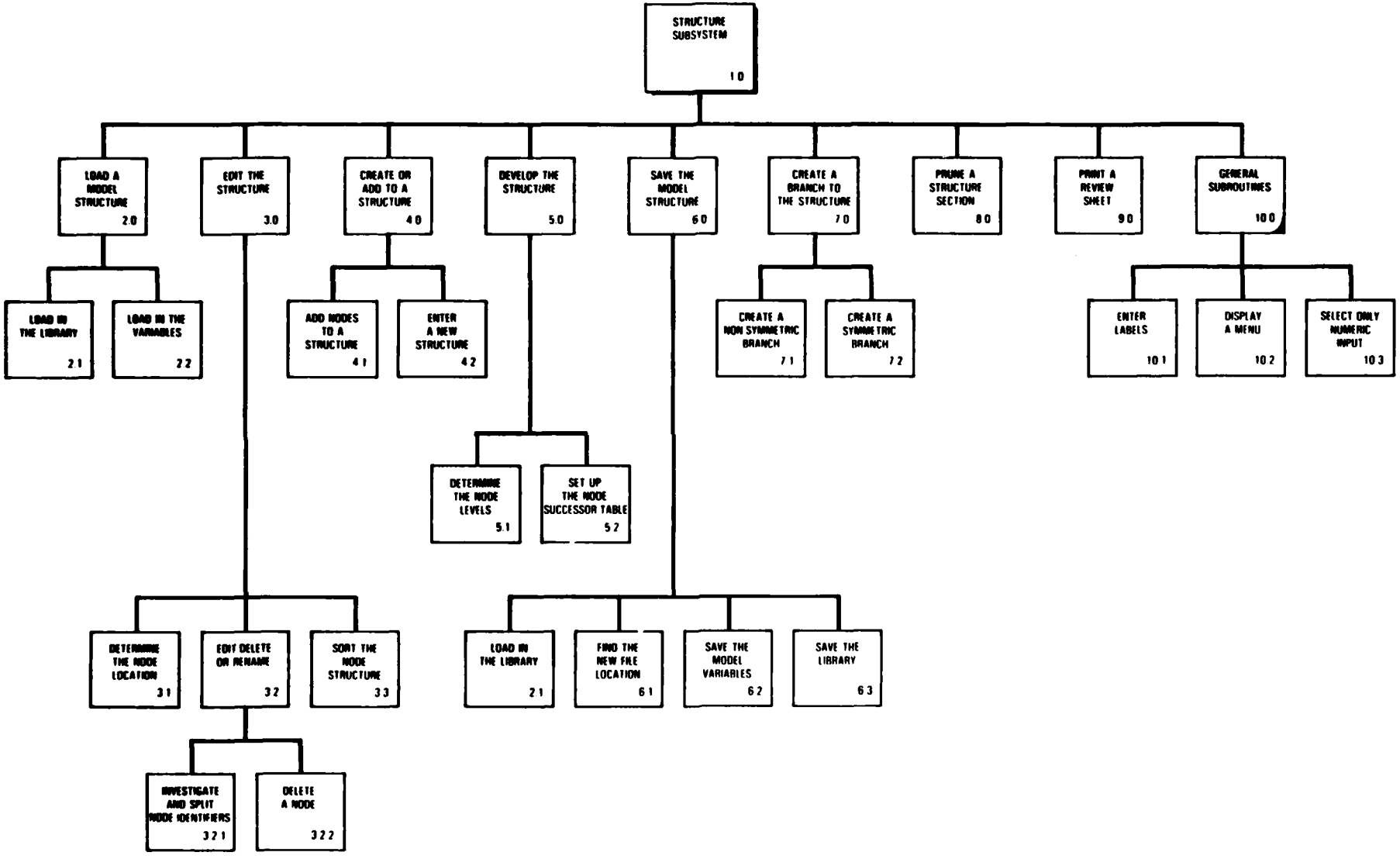


Figure 2-3

DECISION STRUCTURE SUBSYSTEM VISUAL TABLE OF CONTENTS

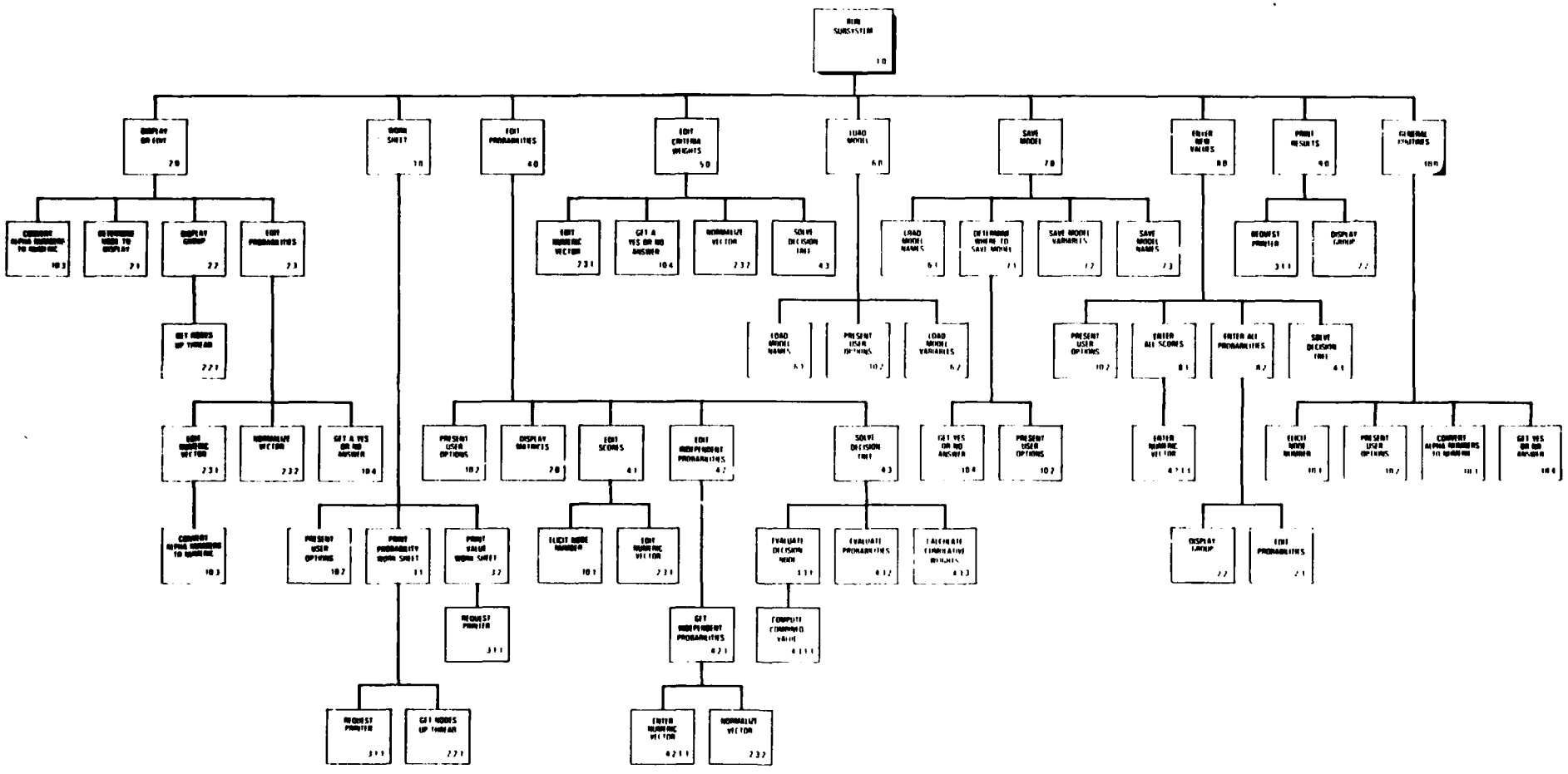
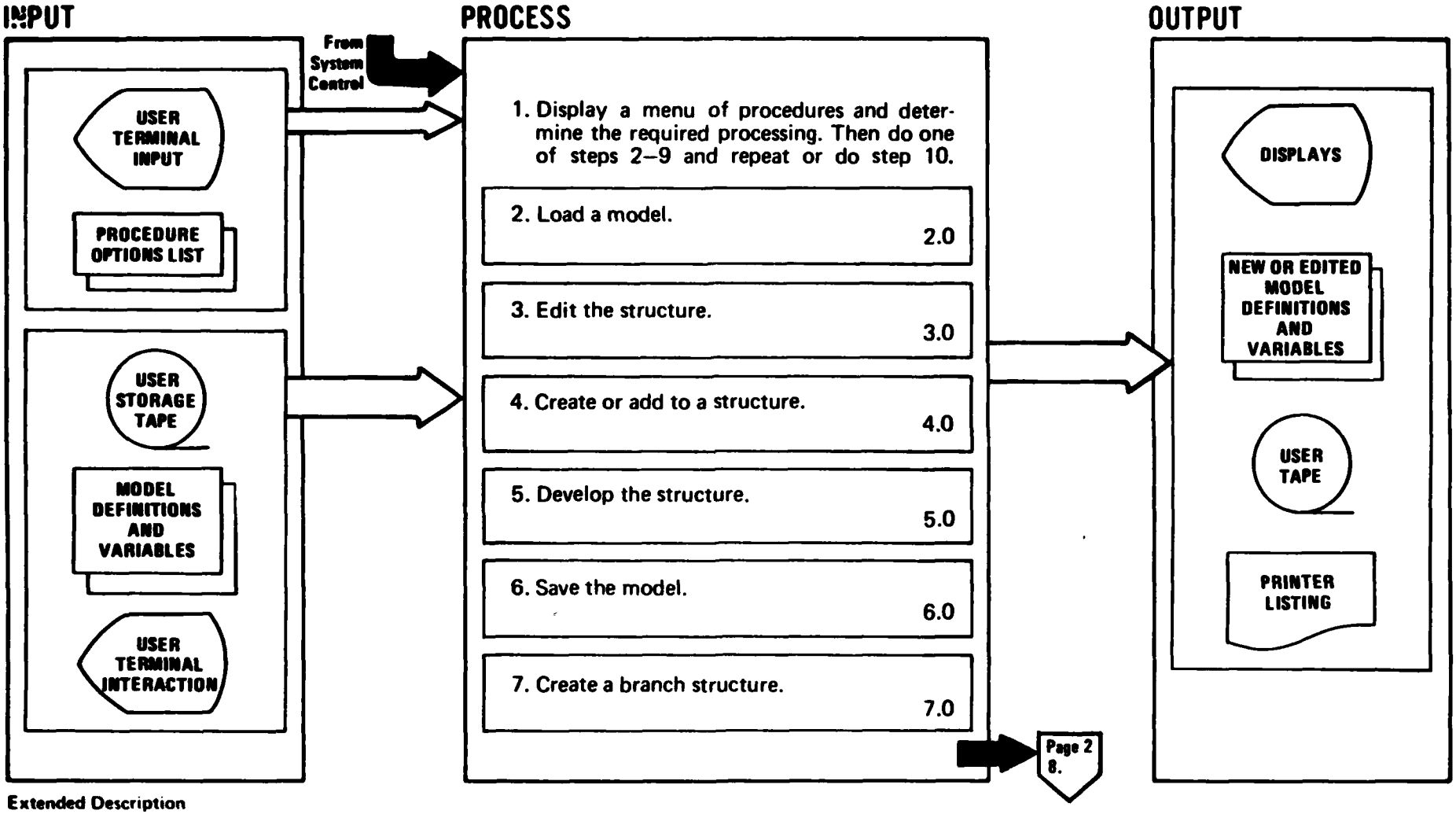


Figure 2-4

DECISION RUN SUBSYSTEM VISUAL TABLE OF CONTENTS

System Program STRUCTURE Name:

Diagram ID 1.0 Description Main Routine Page 1 of 2



Extended Description

The list of program procedures is displayed so that the user may select the next process to be performed. The list is displayed in menu format which allows the association of position numbers with different options in the list.

1. The user is prompted for a choice of operations. The chosen procedure is invoked via one of steps 2–9. If the user responds with blank or null input, then step 10 is executed.

2. The existence of RUN/STRUCTURE models on tape (storage) is determined and a selected model is read.

3. The structure (or model) currently defined by the program variables may be changed at this point.

4. A new structure may be entered via user interaction or nodes may be added to an existing structure.

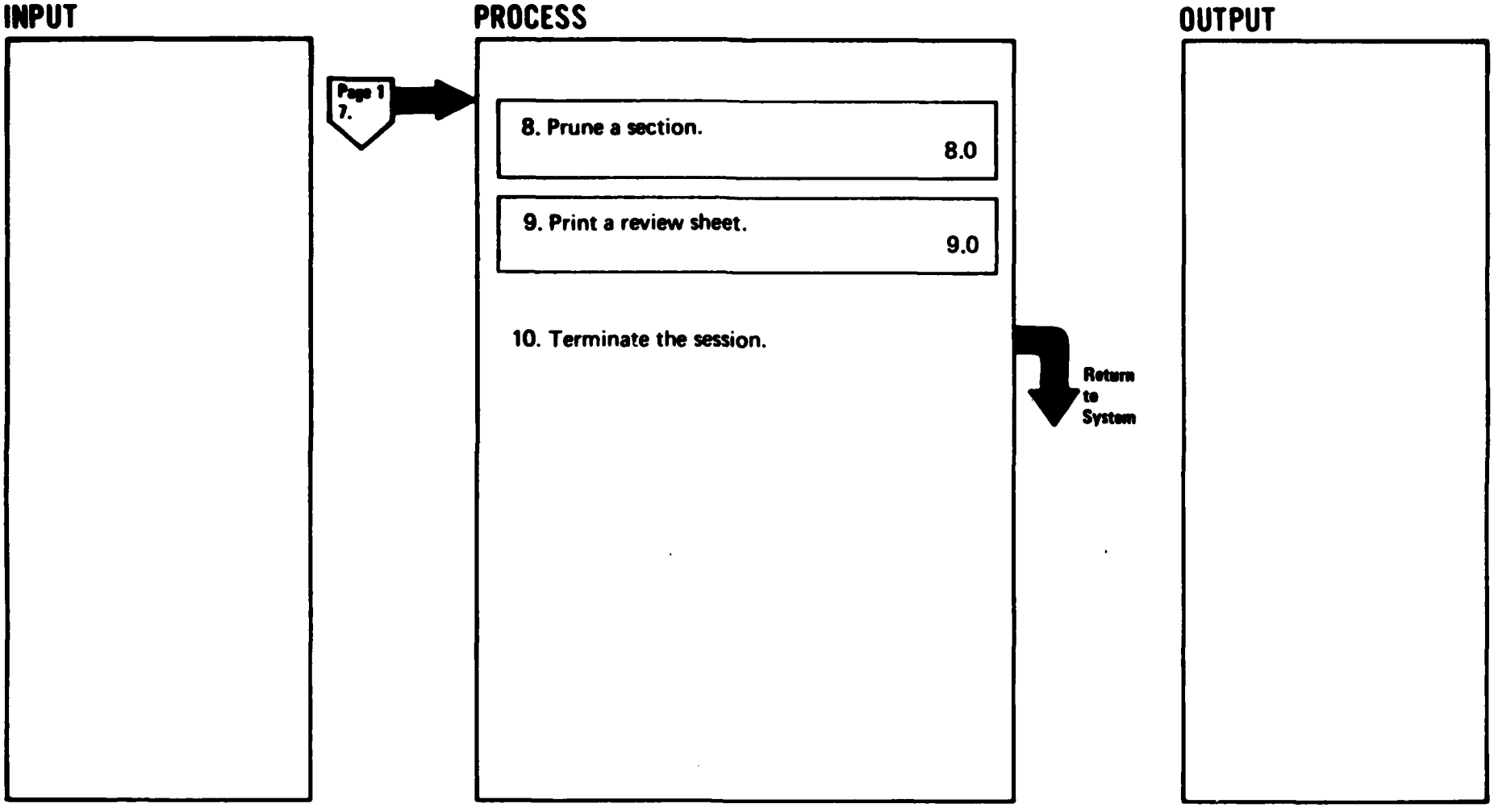
5. This step causes the completion of the model structure by setting up variables which interface with the RUN program. This step should always be performed before step 6.

6. The currently defined model structure may be stored via this step.

7. A branch or subtree may be defined and later added to a structure in procedure 4.

System Program: STRUCTURE Name:

Diagram ID: 1.0 Description: Main Routine Page 2 of 2



Extended Description

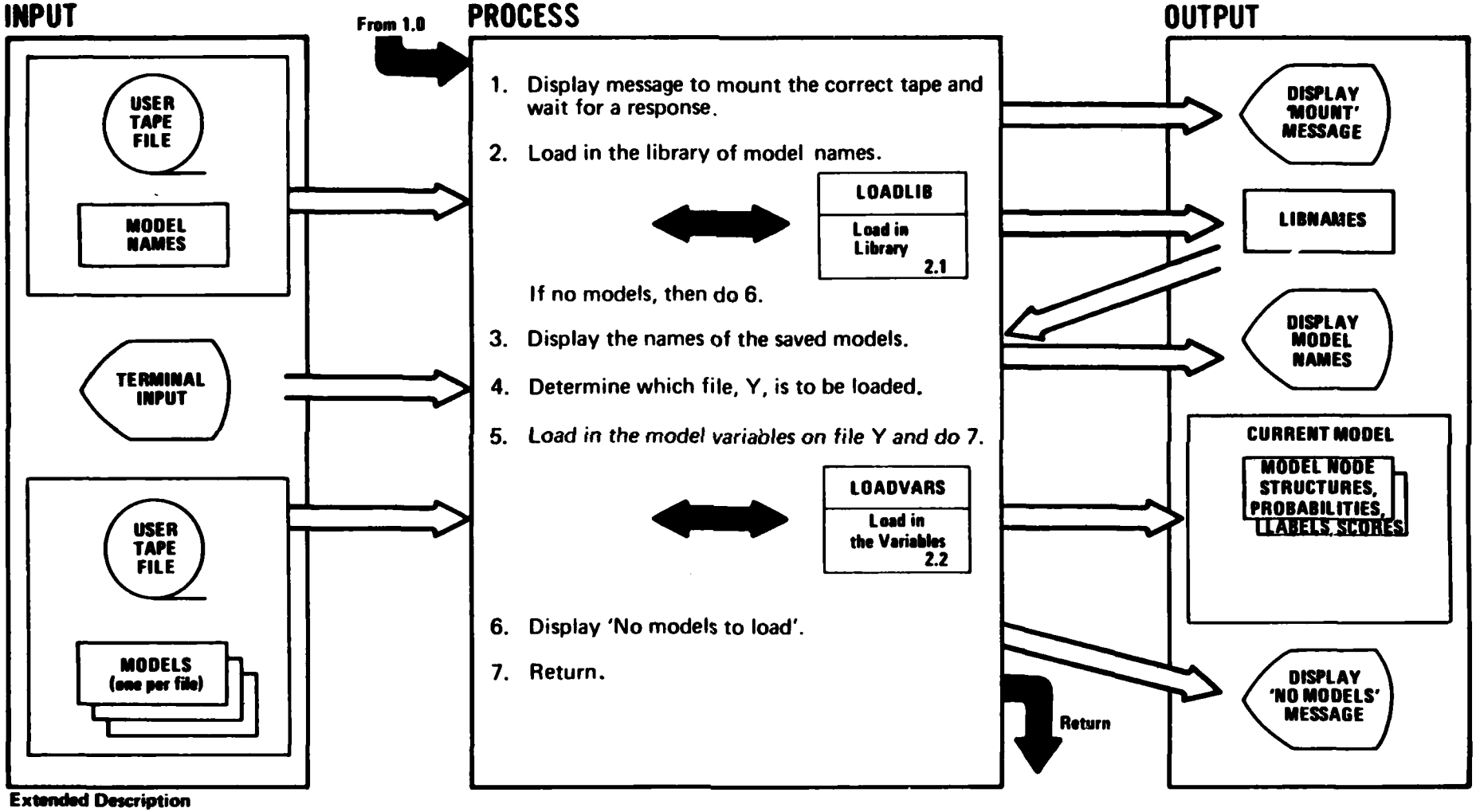
8. Groups of nodes may be deleted from the currently defined structure.

9. A printout of the structure as it is currently defined is obtained.

10. The program ends here: a restart option will cause step 1 to be executed again. When a session is terminated, all branch structures or subtrees defined are deleted.

System/Program: STRUCTURE Name:

Diagram ID: 2.0 Description Load A Model Structure Page: ___ of ___



Extended Description

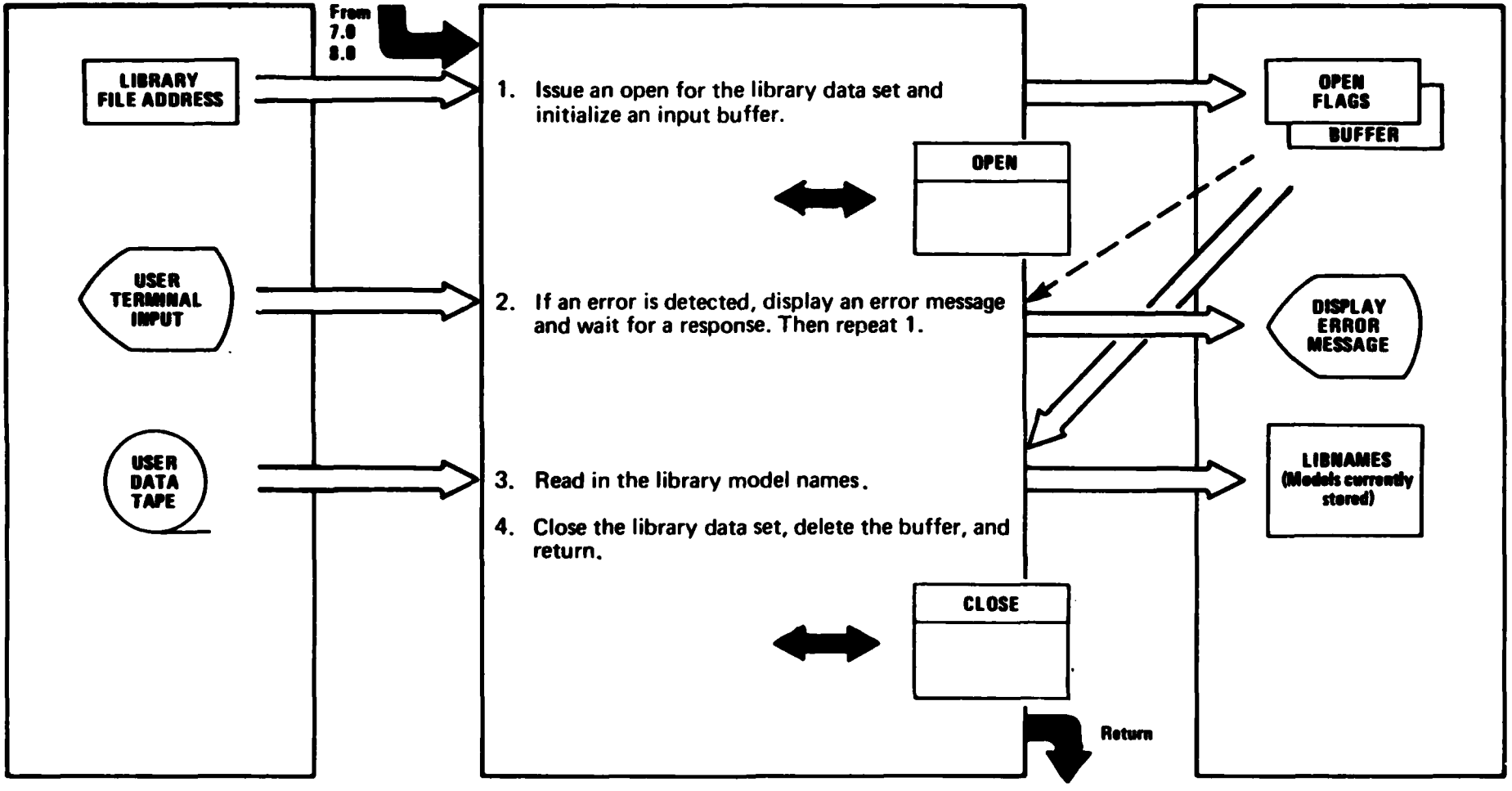
1. The user may have many tape files on which formatted models are stored. In this step, the user is prompted for a response indicating the desired tape is mounted and online.

3. The names of the models existing on the mounted tape are displayed in list or MENU format so that the user may select a model for loading.

4. The user is prompted for a model selection: the response may be the list item number or the model name. The requested model is stored in the same tape file as its position relative to the other model names in the displayed list.

System/Program: STRUCTURE Name: LOADLIB

Diagram ID: 2.1 Description Load in the Library Page: ___ of ___



Extended Description

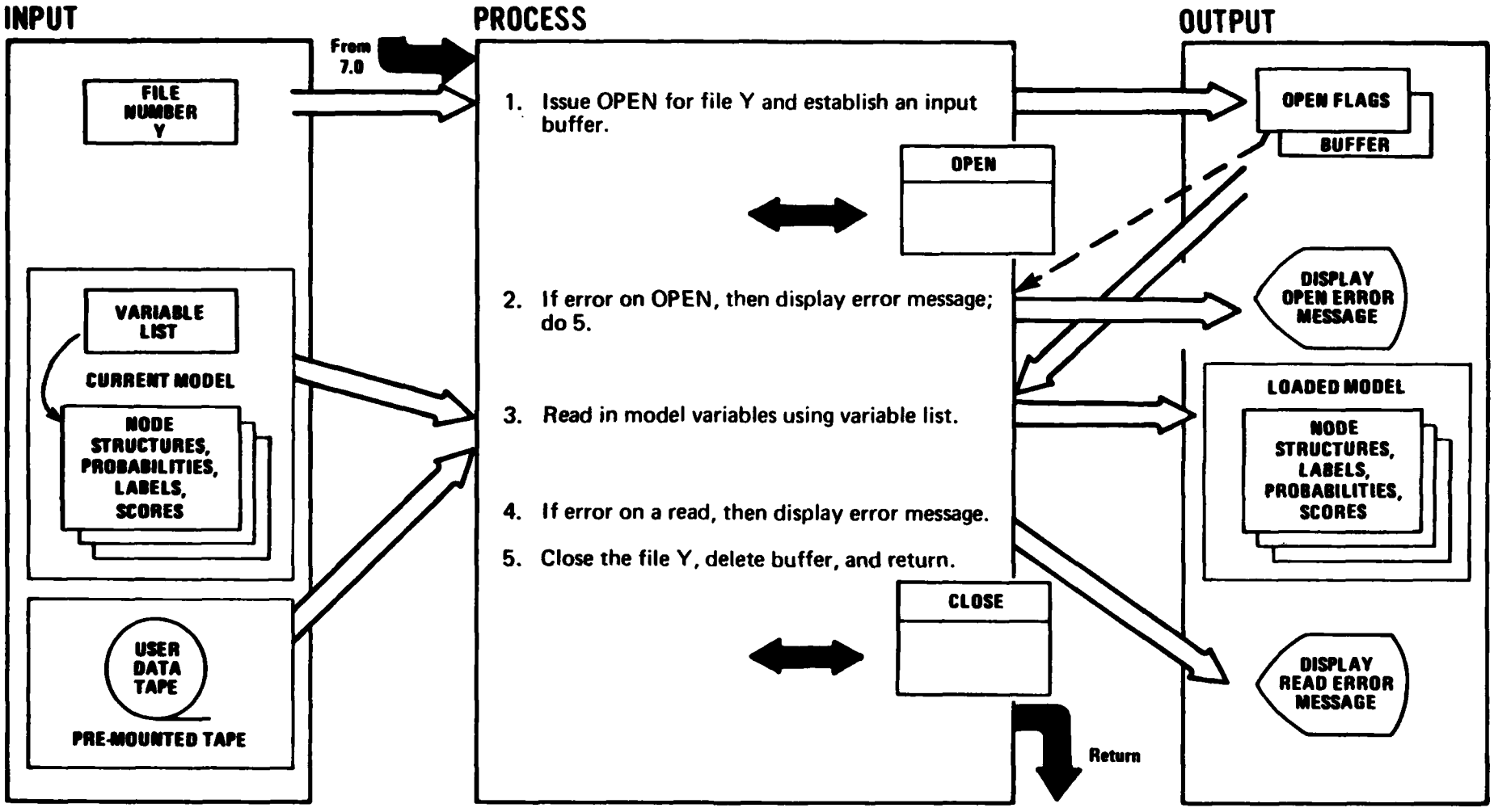
2. The library file of model names is available on each formatted data tape. The file is usually stored and retrieved as a character array and resides on the same device with model data and structure variables. A system OPEN command is needed to ensure that the data file is online and accessible for reading. An input buffer is needed and provides the link between stored information and program addressable information.

3. The library model names are retrieved from storage. The character array used for holding these model names, LIBNAMES, is of a form which facilitates display; thus, the names may all be of equal character length.

4. A system CLOSE command is issued to free the data file for later use.

System Program: STRUCTURE Name: LOADVARS

Diagram ID: 2.2 Description: Load in the Variables Page: 1 of 3



Extended Description

3. A list of variable Names or identifiers is kept so that load and store routines will always process the variables in the same sequence order.

4. The Model variables retrieved from storage are used in all other program functions (see Diagram 1.0). The variables which must be loaded are the following:

- OUTLINE TABLE
- LABELS OF NODES
- SCORES
- PROBABILITIES
- CUMULATIVE PROBABILITIES
- NODE TYPES
- NODE INDEPENDENT PROBABILITY TAGS
- DATA LEVEL MASK
- AGGREGATE NODE INDICES
- SUCCESSOR TABLE
- LABELS OF CRITERIA
- CRITERIA WEIGHTS

1. The OUTLINE TABLE contains an element for each node in the model, sorted in increasing numerical sequence order. The value is an encoded representation of the node outline number supplied for a node when the model structure is created.

System Program: STRUCTURE Name: LOADVARS

Diagram ID: 2.2 Description: Load in the Variables

INPUT

PROCESS

OUTPUT

Extended Description

2. The NODE LABELS contain descriptions (one per node in the same order as the outline table) of nodes that are supplied when the model structure is created.

3. SCORES is a numeric array which contains a set of values for each node of the structure. Each set of values consists of one number per criterion defined in the model.

4. PROBABILITIES are contained in a numeric vector with a value assigned to each node in the model structure. The elements must appear in the same order as the associated outline numbers. When a model structure is created, the vector is null or contains zeros.

5. For each element in the node outline table, there is an associated element in the CUMULATIVE PROBABILITIES vector. The vector will contain the normalized values of all nodes with respect to the entire model when all PROBABILITIES have been entered.

6. The NODE TYPES are indicators of the type of calculation that is to be used in assessing final SCORES and PROBABILITIES.

7. The independent probability tags indicate groups of events that occur more than once in the tree and the probabilities of which can be assessed all at once. The number and order of elements is the same as that for OUTLINE elements.

System/Program: STRUCTURE Name: LOADVARS

Diagram ID: 2.2 Description: Load in the Variables Page: 3 of 3

INPUT

PROCESS

OUTPUT

Extended Description

8. The **DATA LEVEL MASK** indicates which nodes are at the data level (bottom level) versus the nodes that are aggregate or non-bottom-level nodes.

9. The **AGGREGATE NODE INDICES** contain the sequence numbers of elements in the model variables which correspond to only the aggregate nodes. An Aggretate node is a node which has one or more subsequent nodes contributing to it.

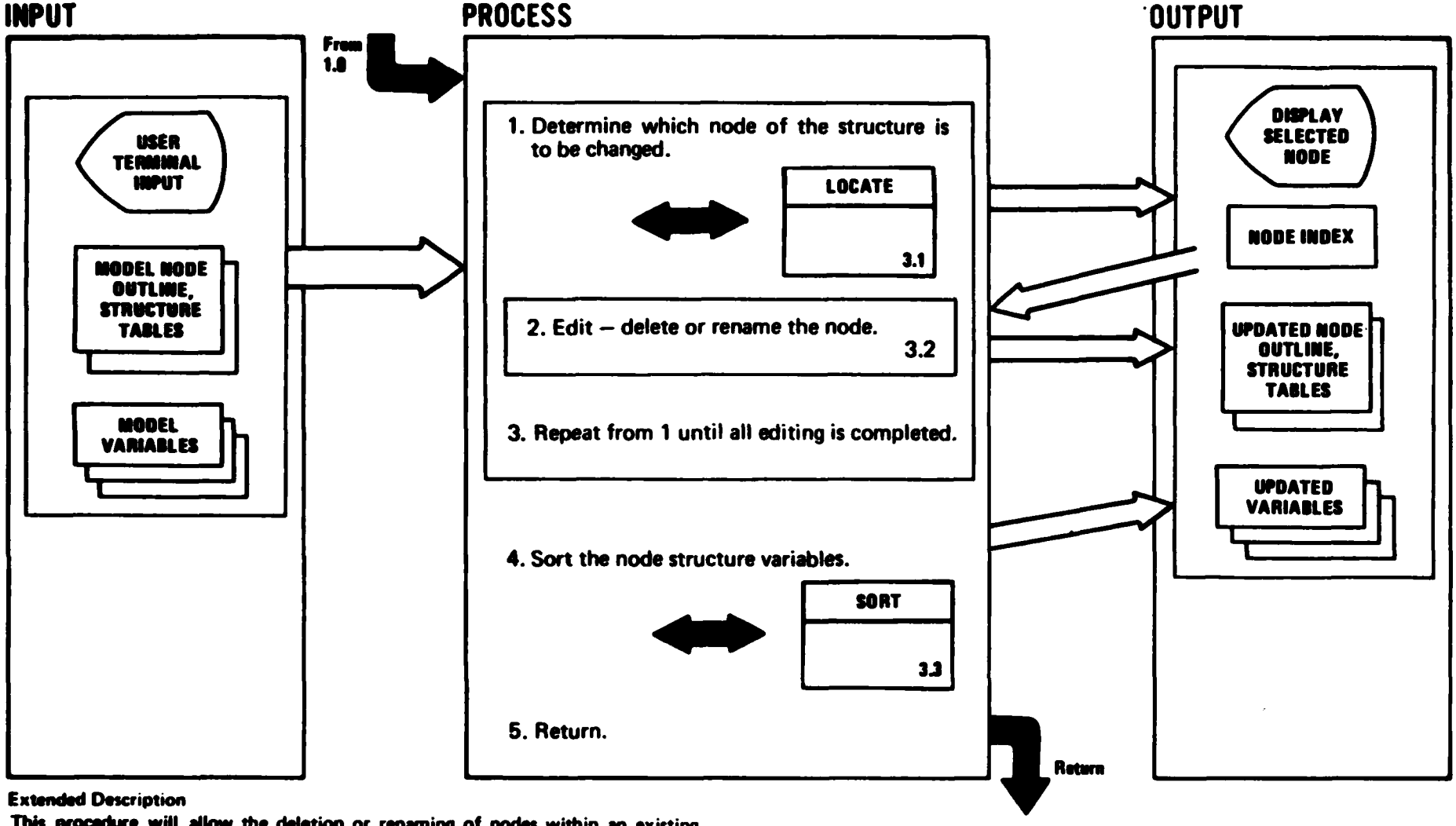
10. The **SUCCESSOR TABLE** is an array which contains, for each aggregate node, the set of indices of nodes which contribute to a node.

11. The **CRITERIA LABELS** contain the user-specified character descriptions of the criteria that are being evaluated.

12. The **CRITERIA WEIGHTS** contain the weights that are to be applied to the criteria when the decision tree is solved. The number of elements is equal to the number of criteria plus one for the total.

System Program: STRUCTURE Name:

Diagram ID: 3.0 Description: Edit the Structure Page: ___ of ___



Extended Description

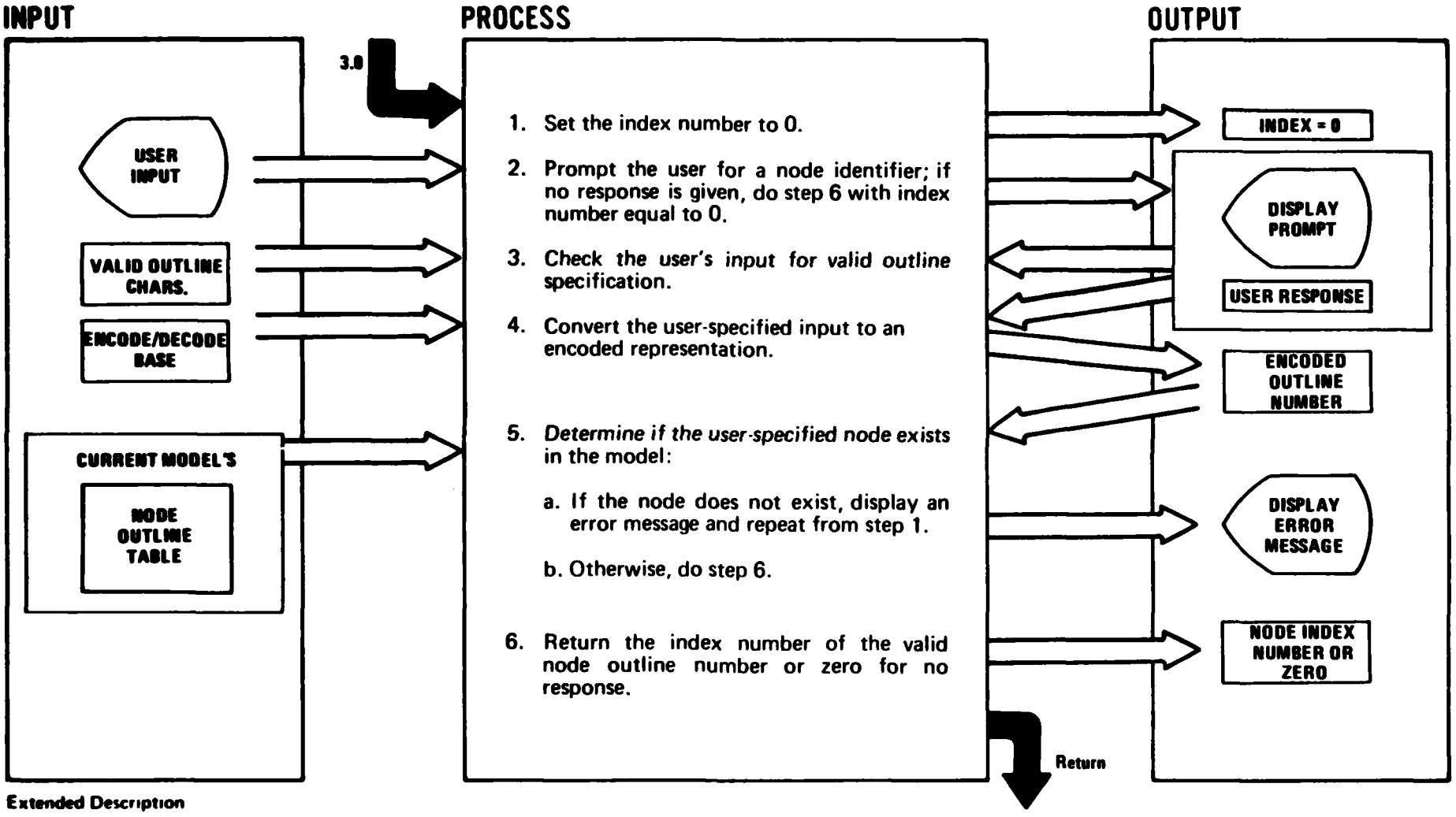
This procedure will allow the deletion or renaming of nodes within an existing structure and operates on a single node at a time. If a group or subtree of nodes is to be deleted, the user should select the "Prune a section" procedure described in diagram 8.0.

1. The user is prompted for a node identifier. This identifier corresponds to the manner in which the node was named when it was placed in the structure. The outline number is a shortened form of the node's identification. An associated index number is determined which is relative to the node outline and structure tables.

4. The node structure variables are reorganized so that associated nodes are always grouped together after the structure has been edited.

System Program STRUCTURE Name LOCATE

Diagram ID 3.1 Description Determine the Node Location Page: ___ of ___

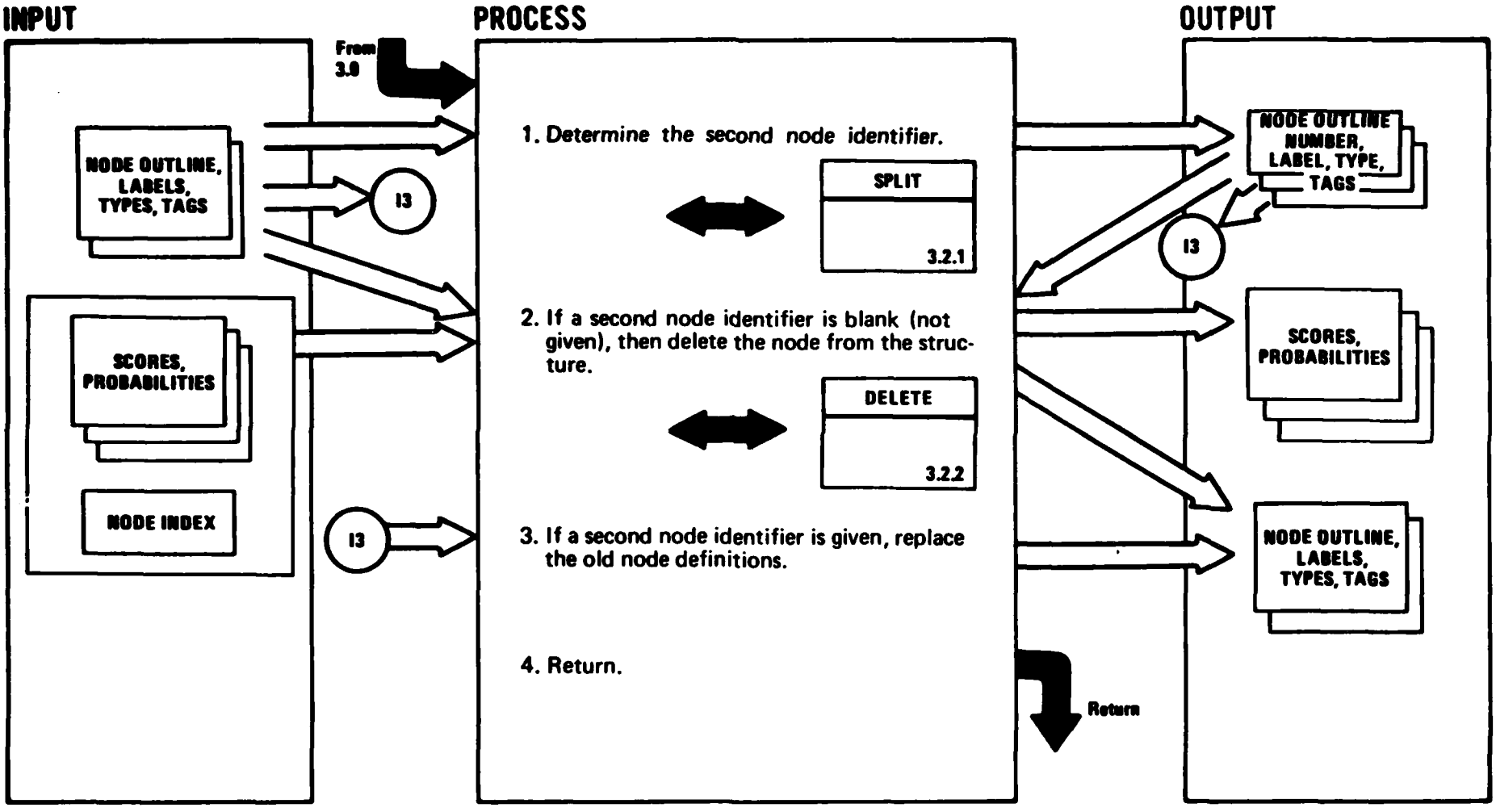


Extended Description

5. The existing outline table is searched for a matching encoded outline number. It is the index into this table of the matching outline number which is returned to the calling routine in step 6.

System Program: STRUCTURE Name:

Diagram ID: 3.2 Description EDIT - Delete or Rename Page: ___ of ___



Extended Description

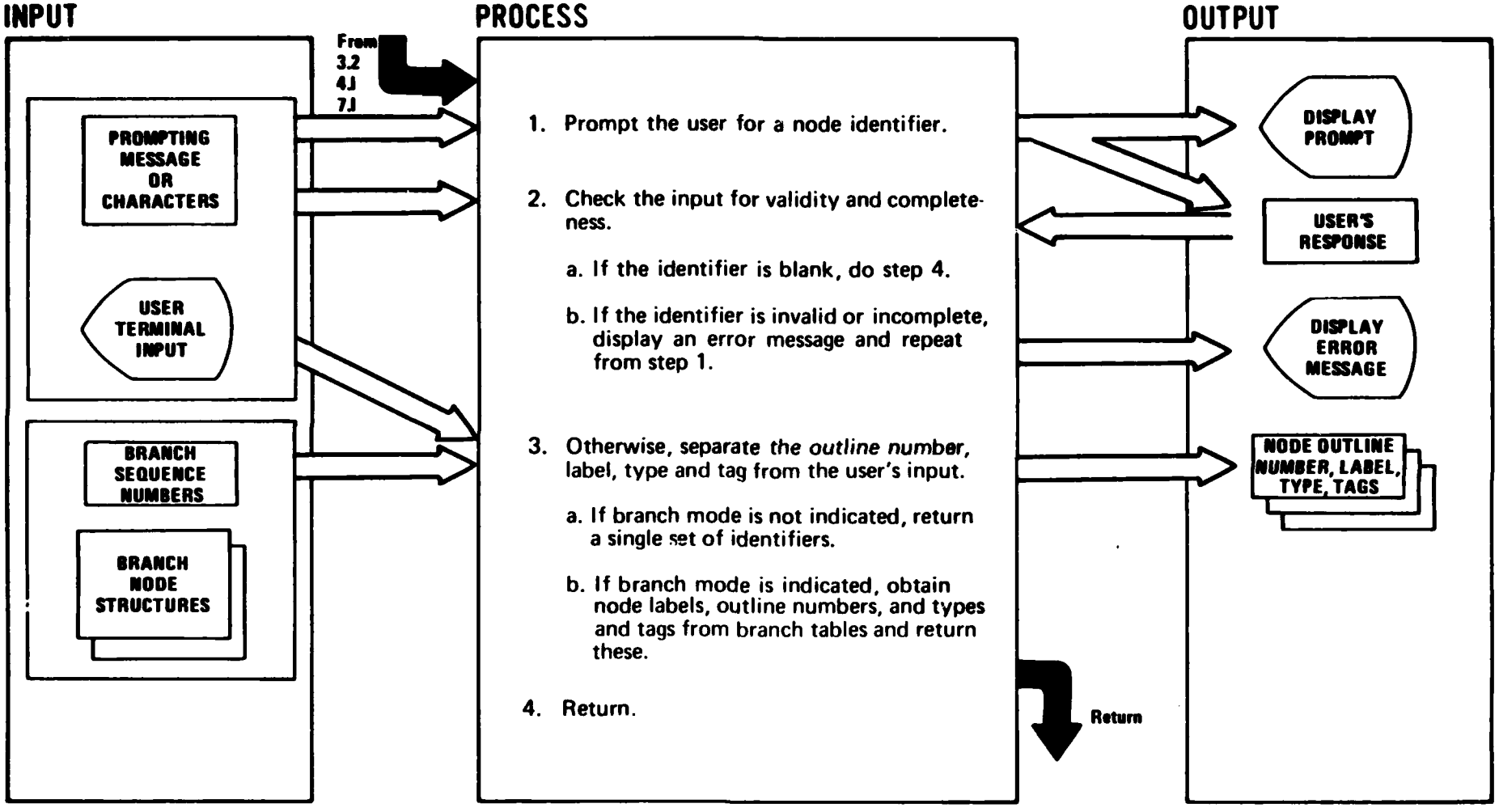
1. The user is prompted for all node identification information – the node outline number, the node label, type and probability tag. (See diagram 2.2 for a description of these items.)

2. A null entry or blank response from the user indicates that the node is to be deleted from the current structure.

3. Replace the outline number, the node label, type and tag in the appropriate arrays with the new ones.

System Program STRUCTURE Name SPLIT

Program ID 3.2.1 Description Investigate and Split Node Identifiers Page ___ of ___



Extended Description

1. The user is required to input the identifying information for a particular node in either an existing structure or one that is currently being defined.

2. Proper node identification consists of an outline sequence number which has a hierarchical relationship to other nodes in the structure, a label or descriptive name, a node "type" and tag indicators. (The node type and probability tag indicators are optional input with default type = W for probability node and tag = blank.) The three variables are usually entered with commas or some other punctuation separating each one from the other.

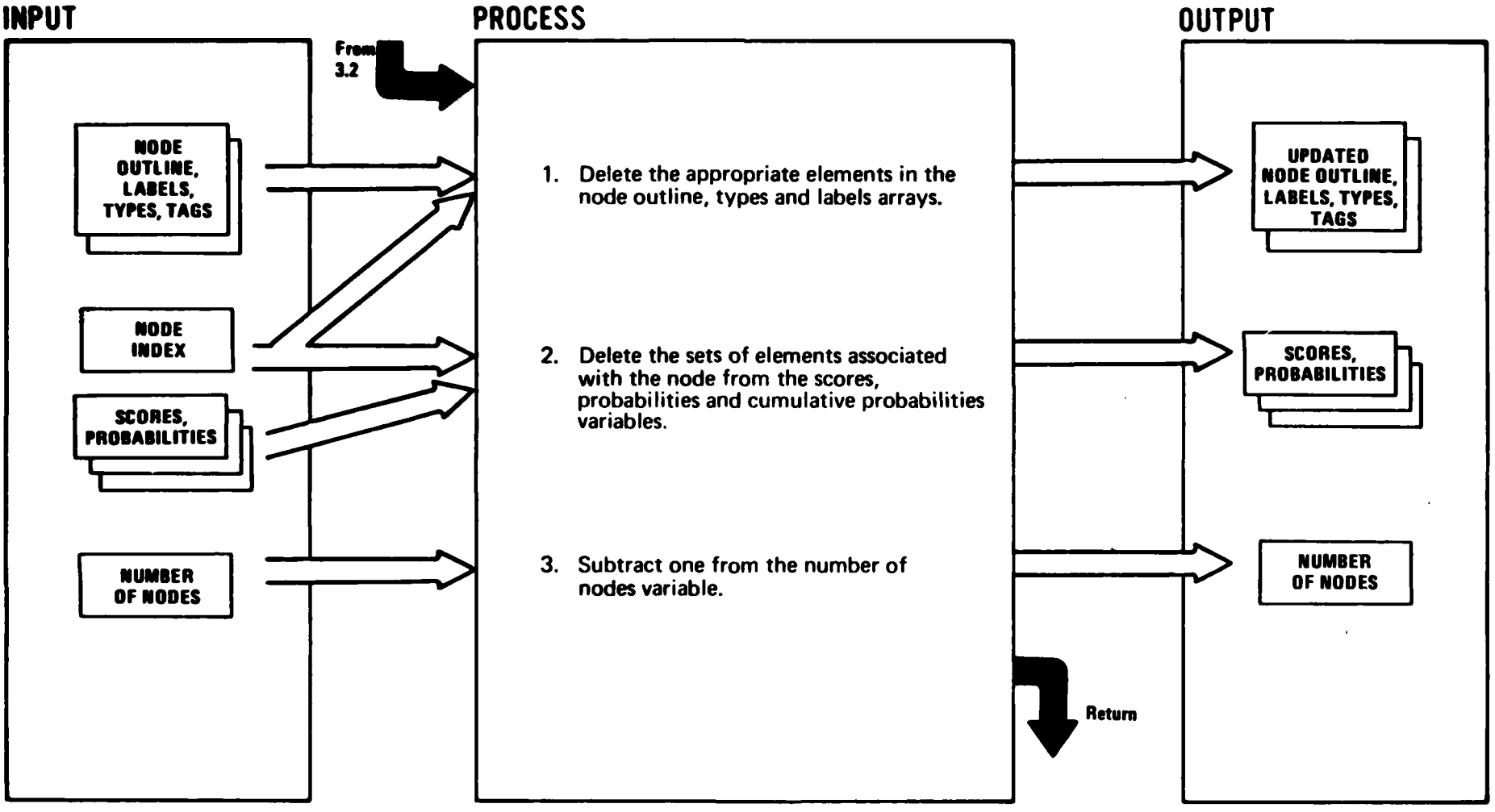
A special character, such as an asterisk (*) or pound sign (#), should be used to designate that a group or subtree is being specified. The special character would be the first in the input line of the user's response.

3. The outline number – numerically encoded to a sufficiently large number, the label, type and tag are returned as separate variables.

If a branch or subtree is being specified, the appropriate node labels, outline numbers and types are obtained from the branch structure tables. A group of encoded outline numbers, a group of labels and the group types are all returned to the calling routine. The new outline numbers have been encoded again to agree with the node after which the branch or subtree is being placed in hierarchical fashion.

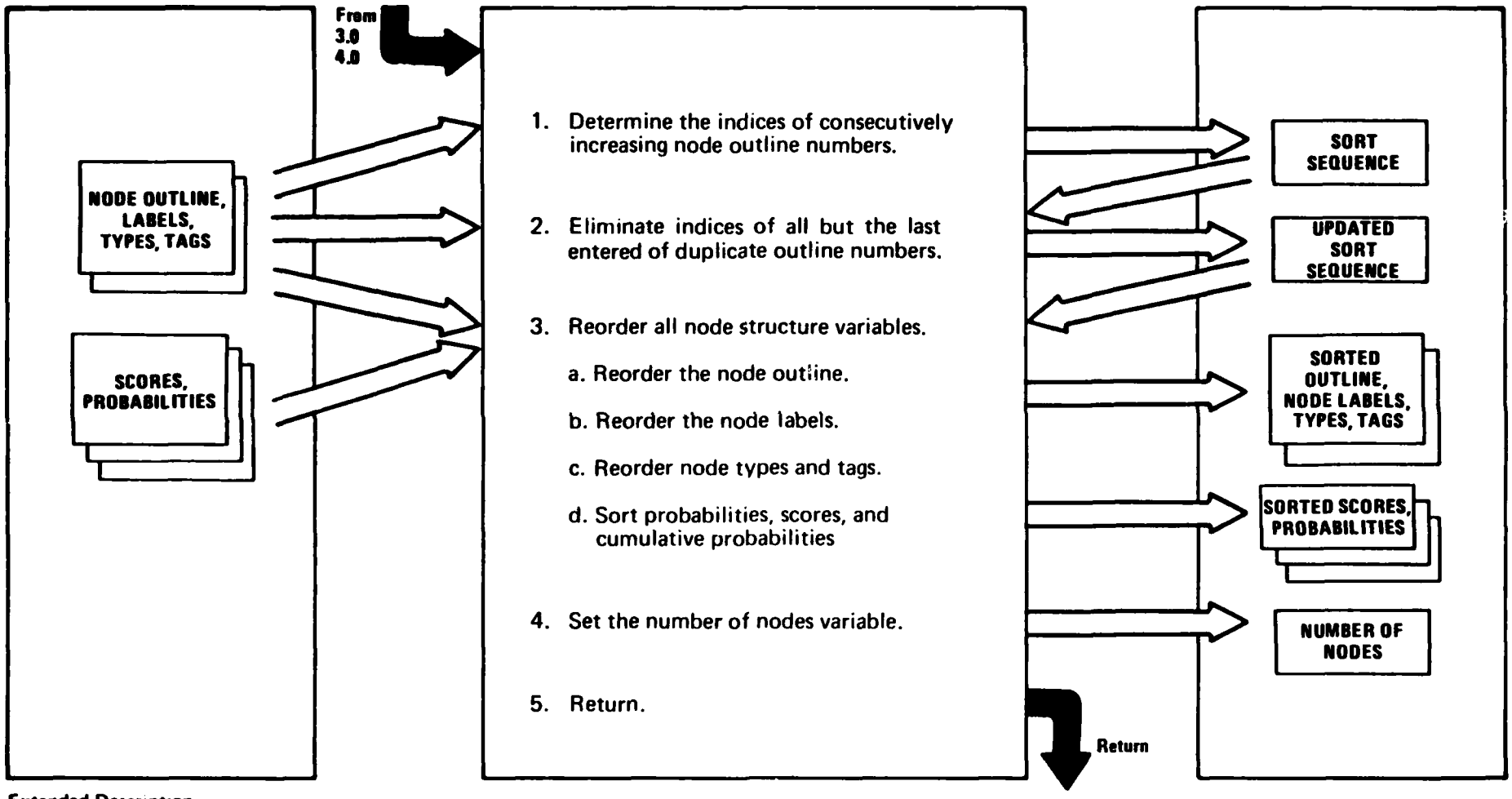
System Program STRUCTURE Name

Diagram ID 3.2.2 Description Delete a Node Page ___ of ___



System Program STRUCTURE Name SORT

Diagram ID 3.3 Description SORT the Node Structure Page ___ of ___

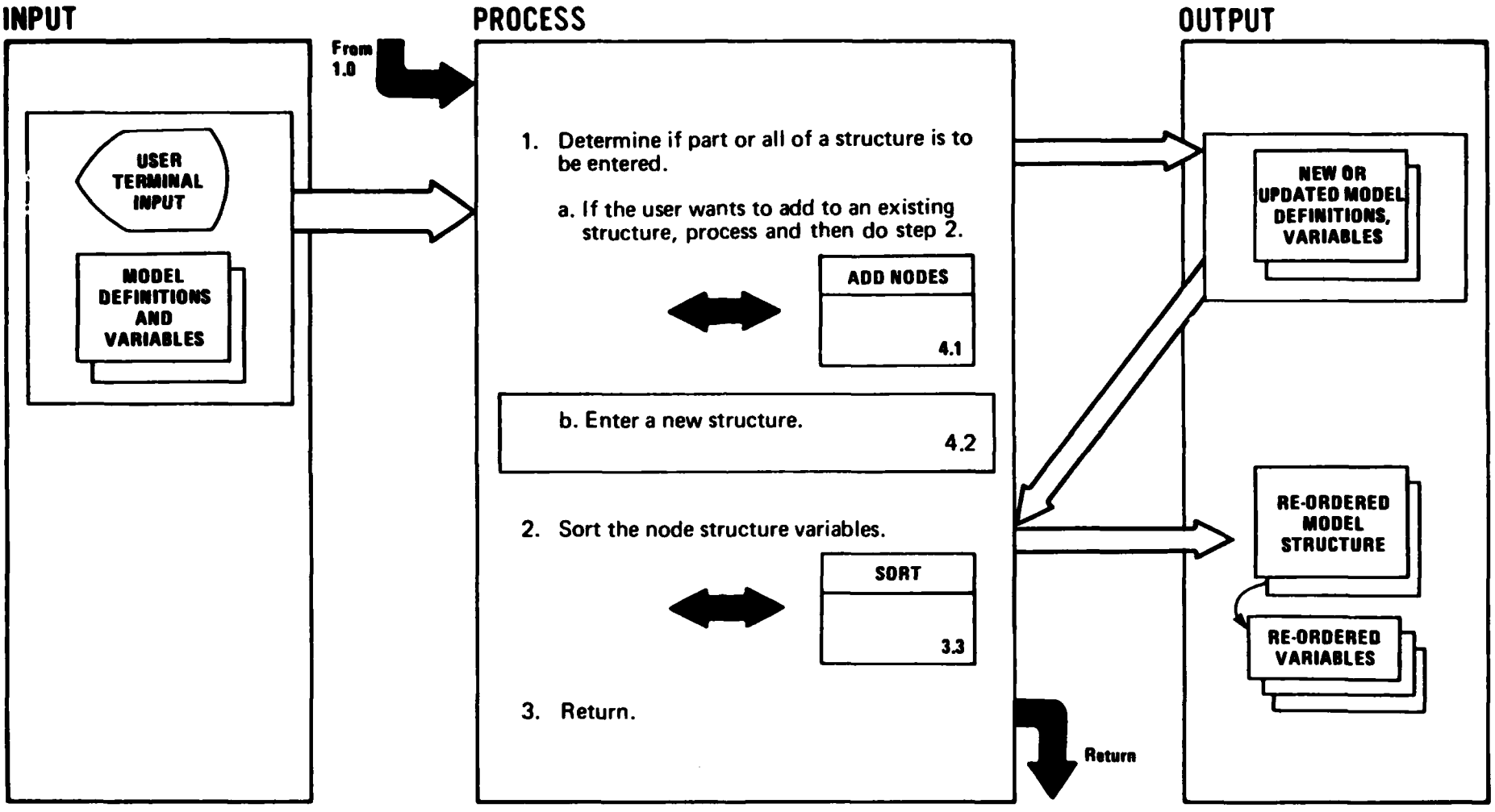


Extended Description

1. The relative indices or locations in the numerically encoded set of outline numbers in increasing value are determined. These indices constitute the sort sequence and will be used to rearrange the structure variables.

System Program STRUCTURE Name

Diagram ID 4.0 Description Create or Add to a Structure Page ___ of ___



Extended Description

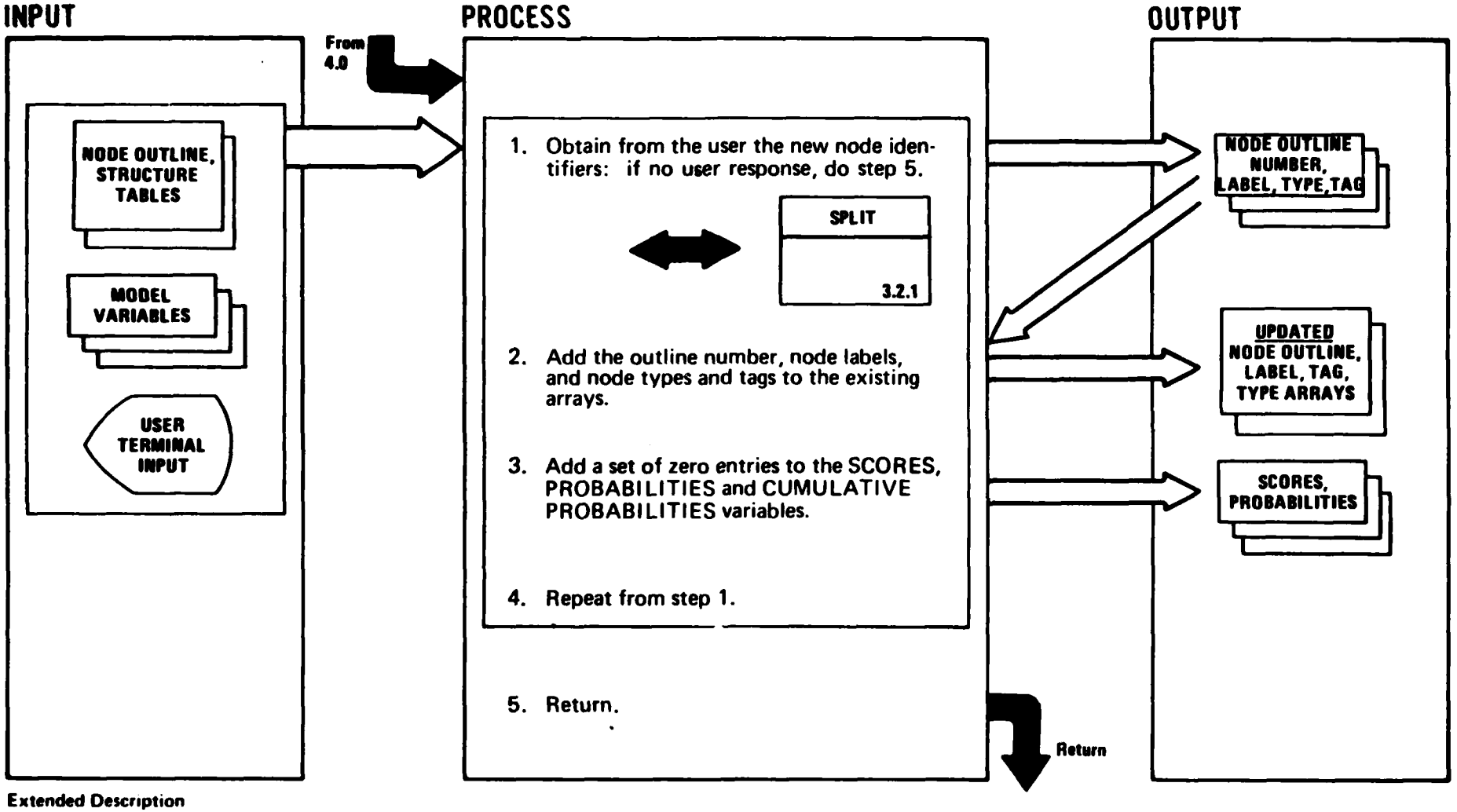
1. Request a "yes" or "no" response directly from the user to determine whether a new structure is to be entered or nodes are to be added to an existing structure.

b. If a new structure is entered, all currently defined variables of the old structure are deleted.

2. An explanation of the sorting function is given in diagram 3.3 of the STRUCTURE System Specifications.

System Program: STRUCTURE Name: ADDNODES

Diagram ID: 4.1 Description: Add Nodes to a Structure Page: ____ of ____

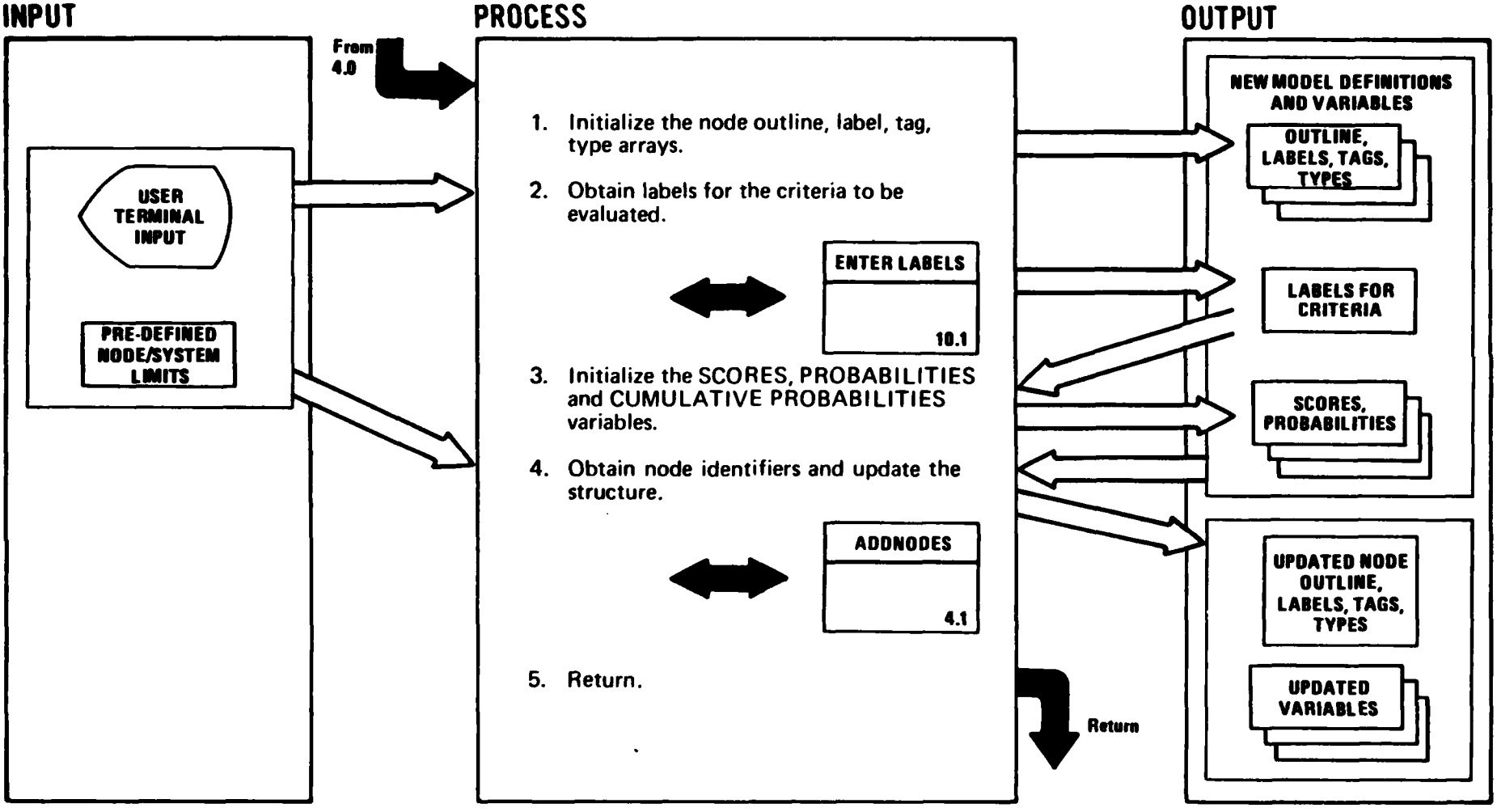


Extended Description

2 - 3. Additions to previously initialized or existing variables are accomplished by extending the arrays such that the corresponding orders of associated labels, scores, types, tags, weights and decoded outline numbers are the same.

System Program STRUCTURE Name

Diagram ID 4.2 Description Enter a New Structure Page: ___ of ___



Extended Description

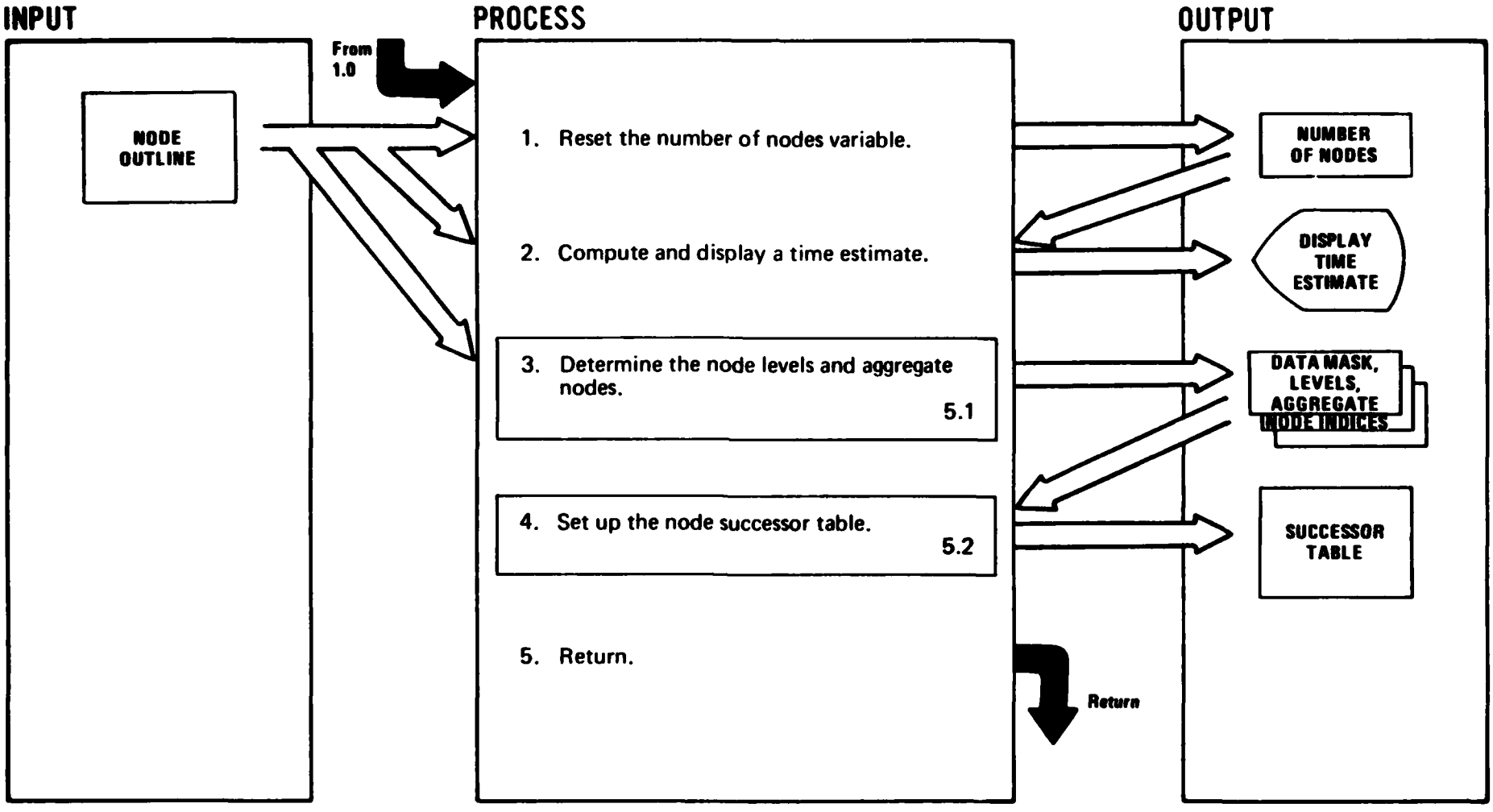
1. Initialization is caused by establishing null or blank vectors for the specified variables.

2. Labels for the criteria to be evaluated are obtained from later storage and for the determination of the length of any set of SCORES.

4. The user is prompted for input which will be used to define a hierarchical tree structure described by outline numbers, labels, tags and types of nodes within the structure.

System Program STRUCTURE Name

Diagram ID 5.0 Description Develop the Structure Page ___ of ___



Extended Description

1. The number of nodes is equal to the number of entries in the outline array.

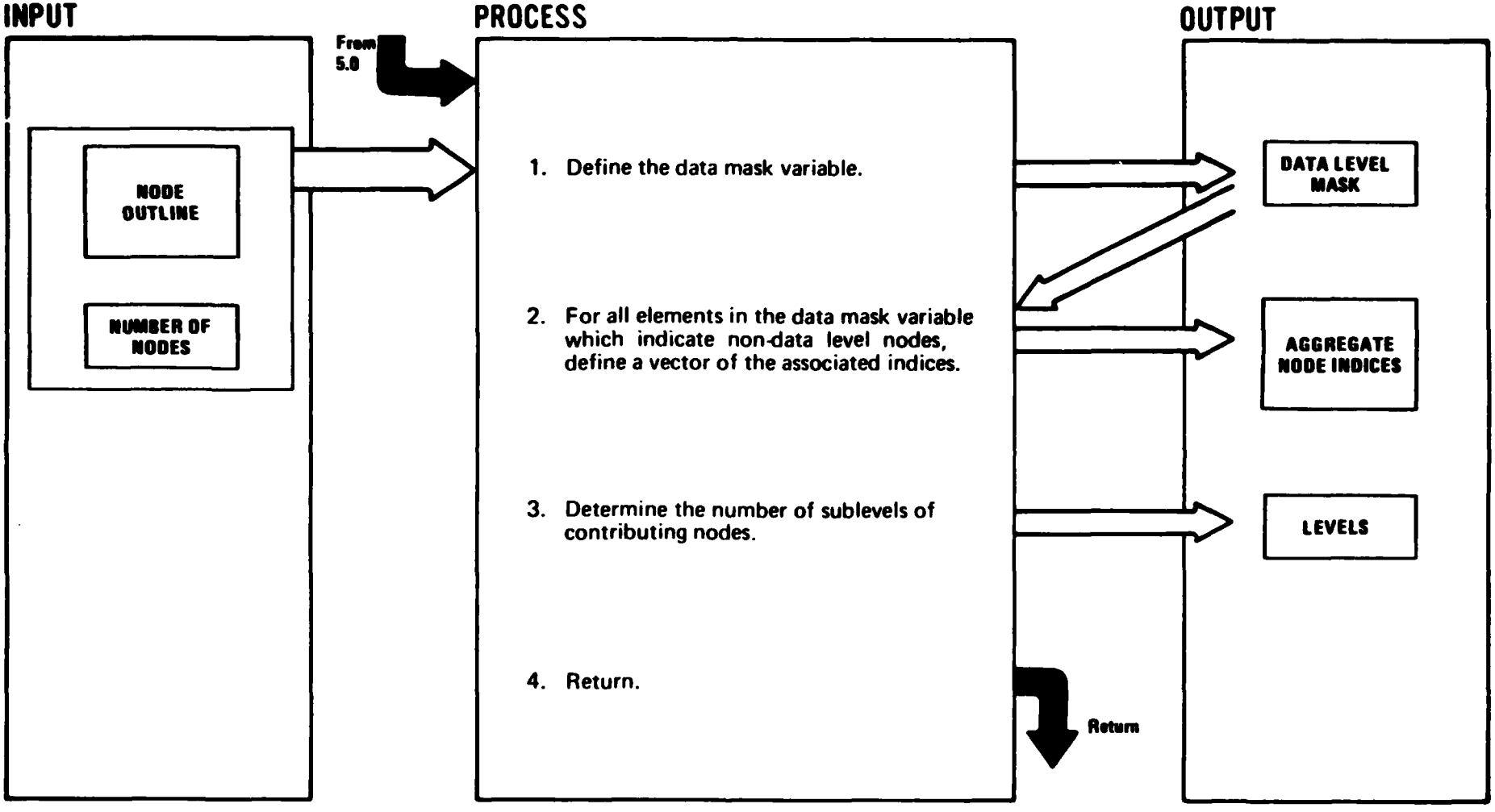
2. A rough estimate of the amount of time required to perform the developing operation may be displayed. The estimate is derived from the number of nodes in the model.

3. The data level mask indicates which nodes in the model are at the data level and which nodes are aggregate nodes. The aggregate node indices are indices into the node outline of nodes which are not at the data level. The LEVELS variable shows how far away a particular node is from the lowest level.

4. The successor table provides a set of contributing node indices for each aggregate node in the same order as aggregate node appearance in the outline.

System Program STRUCTURE Name

Diagram ID 5.1 Description Determine the Node Levels Page: ___ of ___



Extended Description

1. For each node in the model outline, an element is placed in a vector to indicate that node is a data level node or that it is an aggregate node having other contributing nodes.

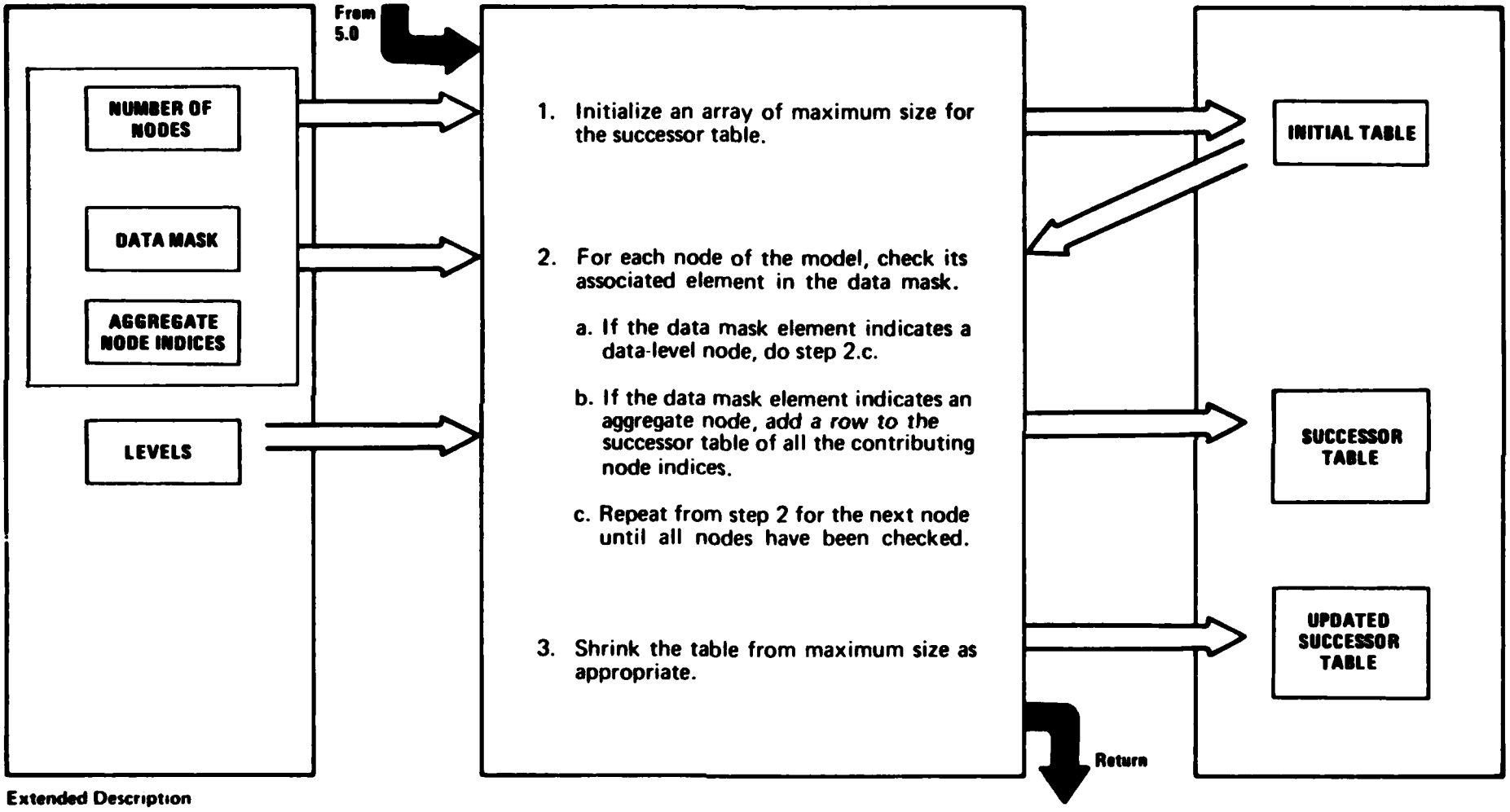
The indicator may be 0 for data level and 1 for the aggregate level or vice versa.

2. The data level mask indicator setting for each node in the outline is used to determine the aggregate node indices – indices into the node outline.

3. The farthest element or data level node from the topmost node is determined. The topmost node is assigned the number of levels between it and the data level farthest away (the depth of the path with the most sub-level tree branches). All other nodes are assigned a value equal to the top-level's minus its distance (number of levels) from the top.

System Program: STRUCTURE Name:

Diagram ID: 5.2 Description: Set up the Node Successor Table Page ___ of ___



Extended Description

1. The maximum size table is prescribed by the number of aggregate nodes and the predefined limit to the number of contributing nodes on any single level.

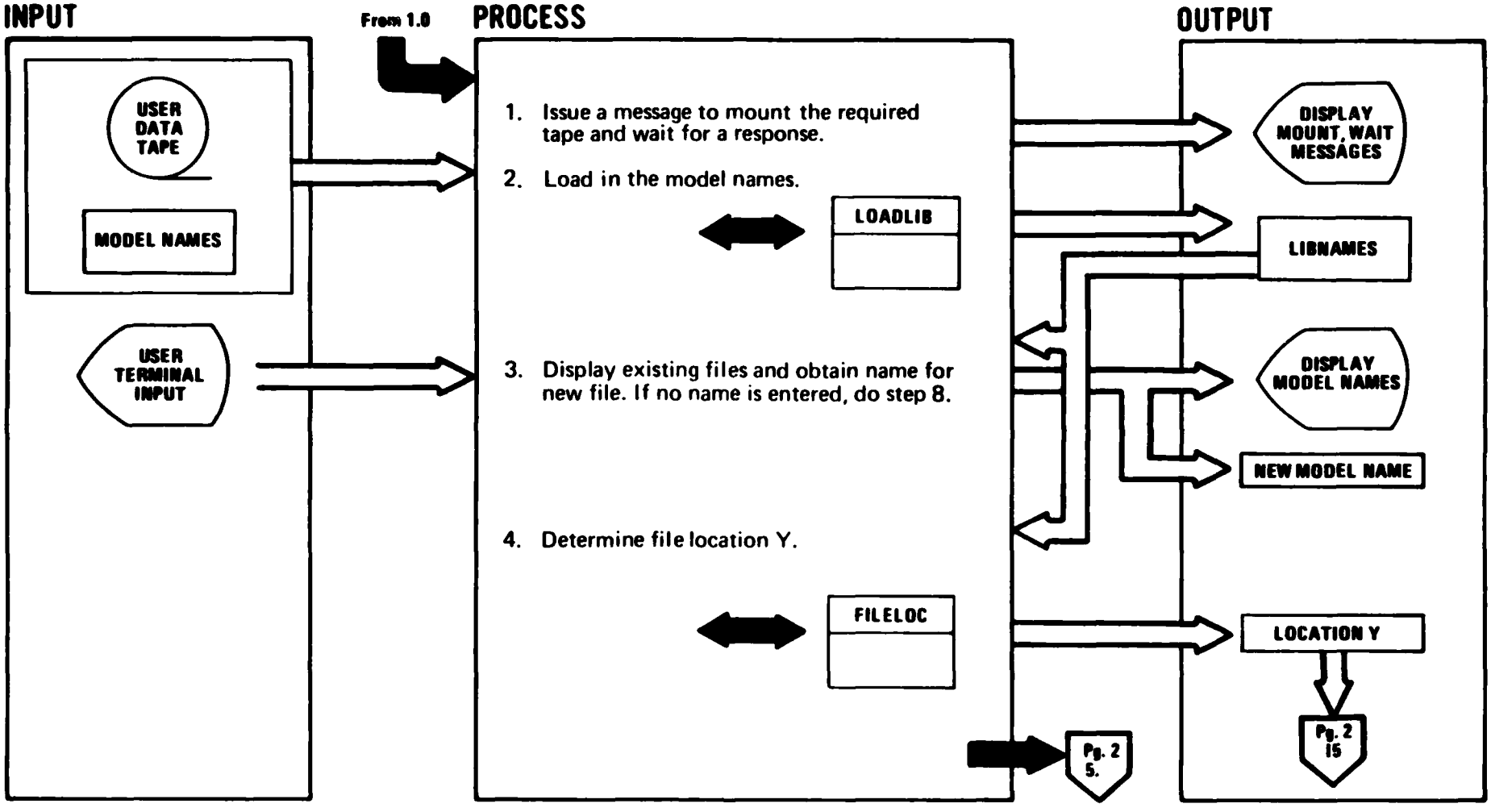
2. This procedure steps through the data mask variable in sequential order: the contributing nodes of the topmost aggregate node will be added to the successor table first.

2.b. If the nodes' associated data mask element indicates an aggregate node, then the contributing nodes are all the nodes which follow in sequential order that have an associated LEVELS number that is less than the selected nodes LEVELS numbers, provided these nodes occur before any node with equal or higher LEVELS number.

3. Since the number of elements in any set of contributing nodes may be less than the predefined limit, the number of columns (or characters) in the table may be diminished.

System Program STRUCTURE Name:

Diagram ID 6.0 Description Save the Model Structure Page 1 of 2



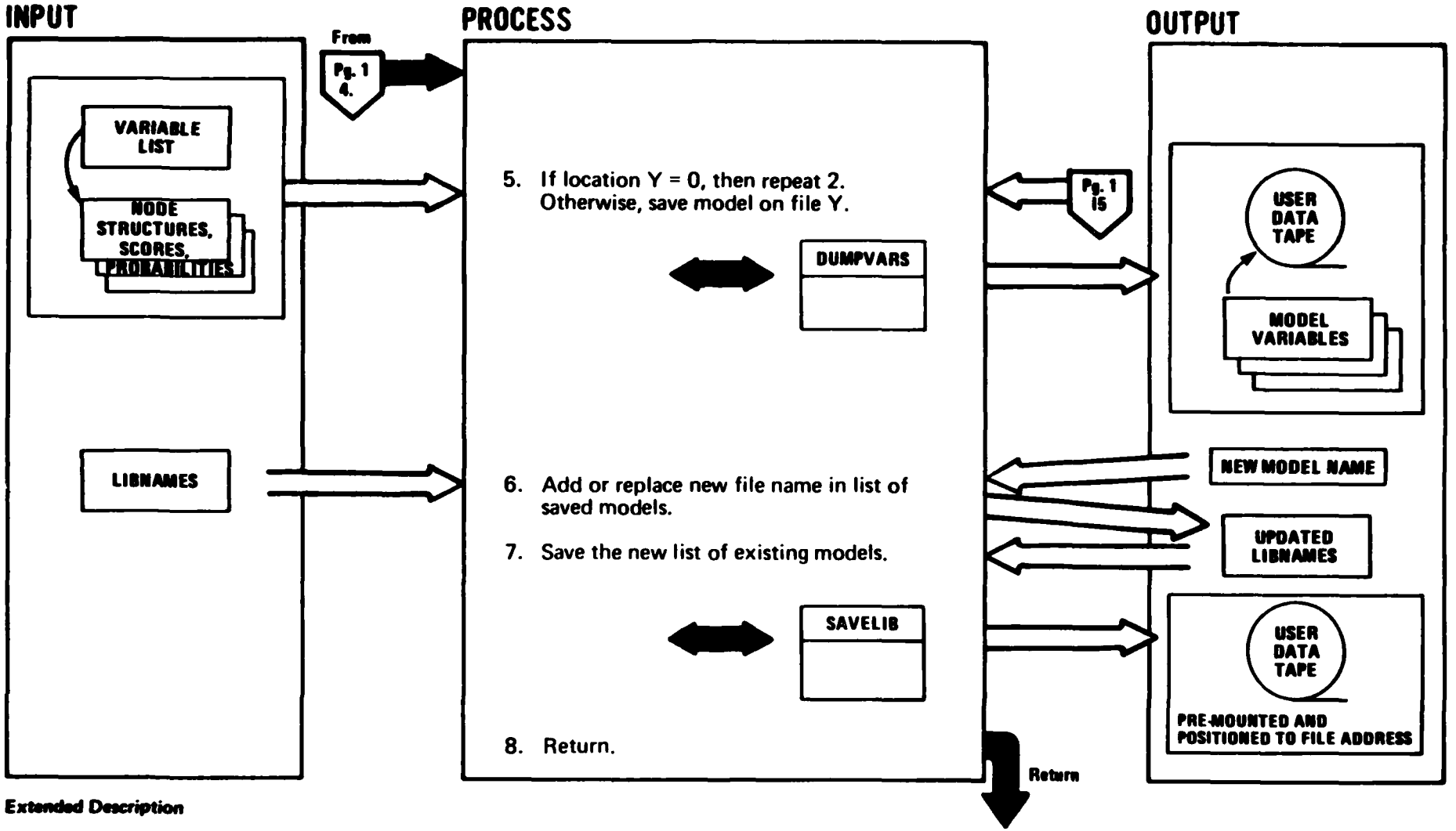
Extended Description

1. The computer program prompts for an indication that the desired storage file/device has been selected and placed online. Any response from the keyboard causes processing to resume.

4. The existing file structure and the amount of available space on the data tape are checked along with the user specification to determine where the model variables are to be stored.

System Program STRUCTURE Name

Diagram ID 6.0 Description Save the Model Structure Page 2 of 2

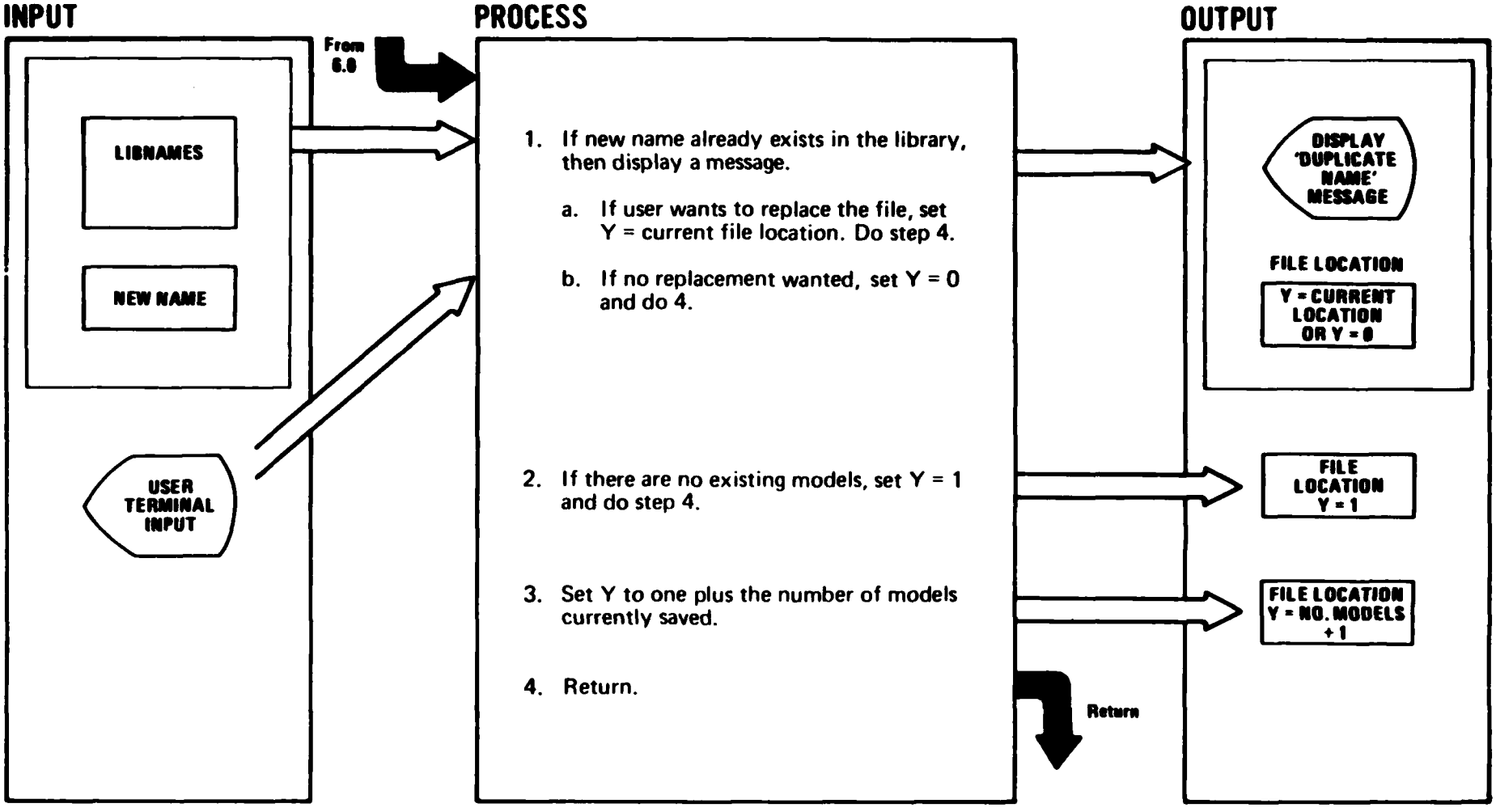


Extended Description

6. The library name list is updated to include the new file. The new model name's position in the LIBNAMES array must be the same relative position to other models stored on the device.

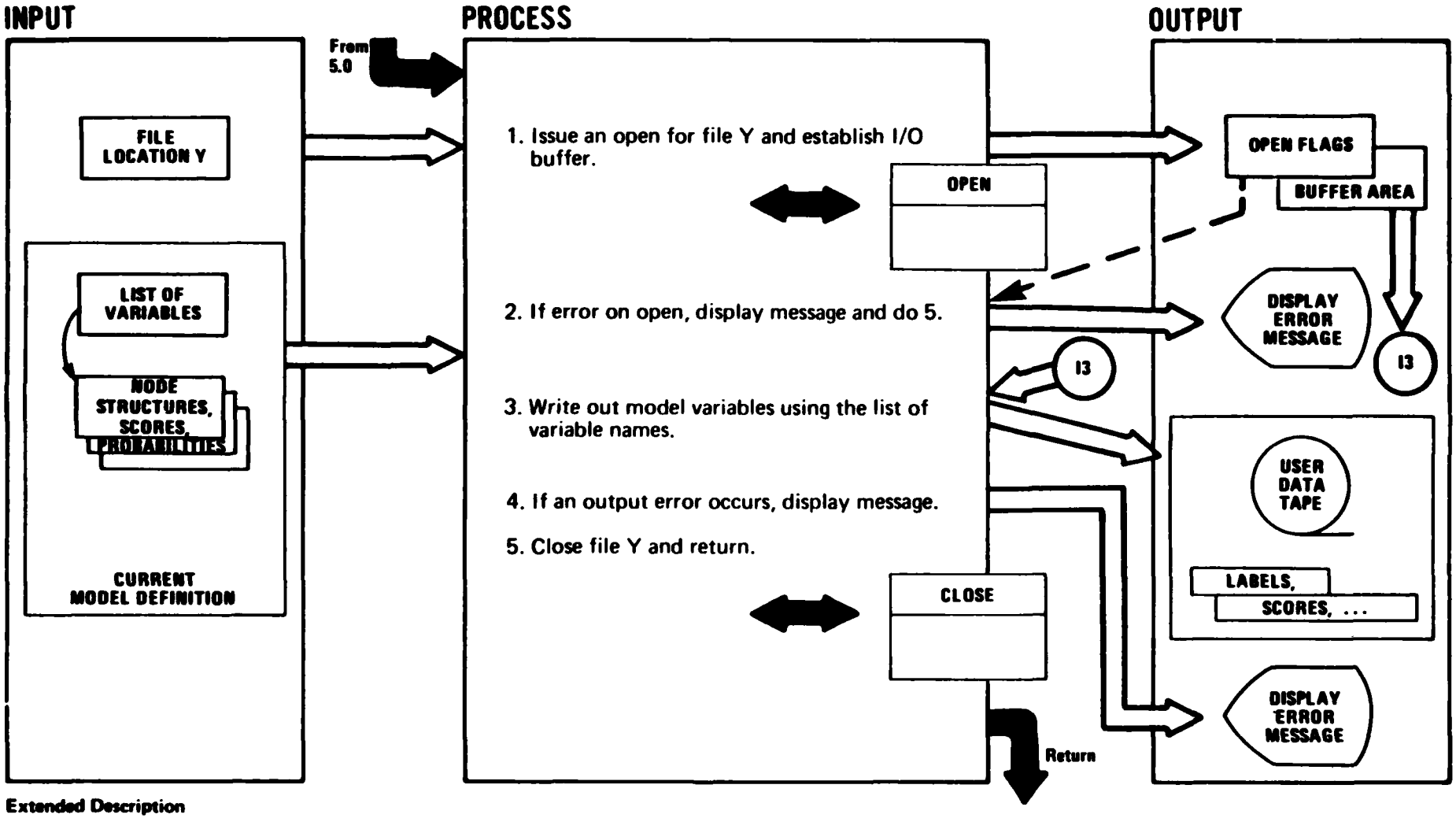
System Program: STRUCTURE Name: FILELOC

Diagram ID: 6.1 Description: Find the New File Location Page: ____ of ____



System/Program: STRUCTURE Name: DUMPVARS

Diagram ID: 6.2 Description: Save the Model Variables Page: ___ of ___



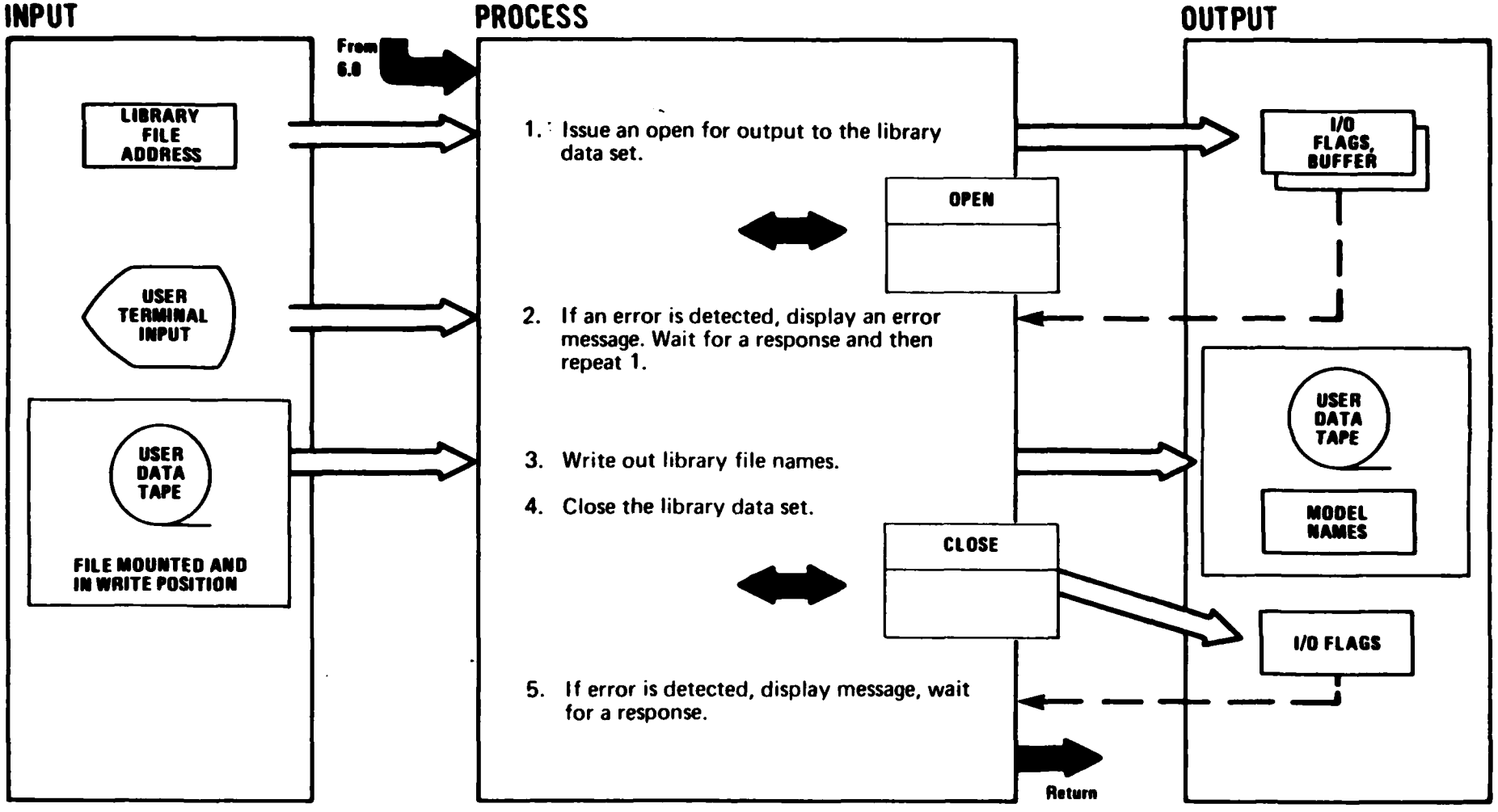
Extended Description

1. The file location Y is used to determine an exact storage position on the selected device.

3. The list of variable names is identical to the list of names used to Load a Model (see diagram 2.2)

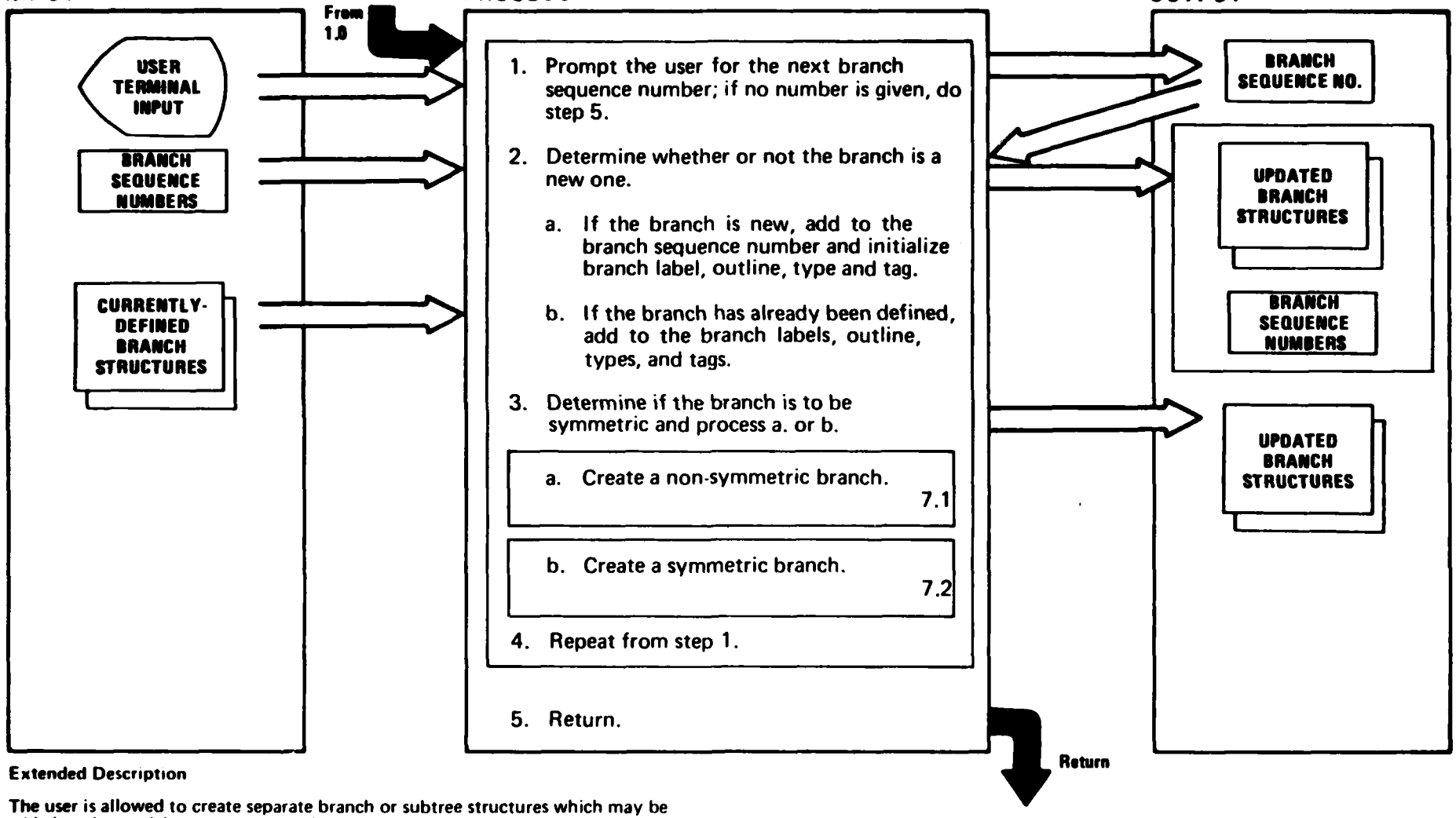
System Program STRUCTURE Name: SAVELIB

Diagram ID 6.3 Description Save the Library Page: ____ of ____



System Program STRUCTURE Name

Diagram ID 7.0 Description Create a Branch to the Structure Page ___ of ___

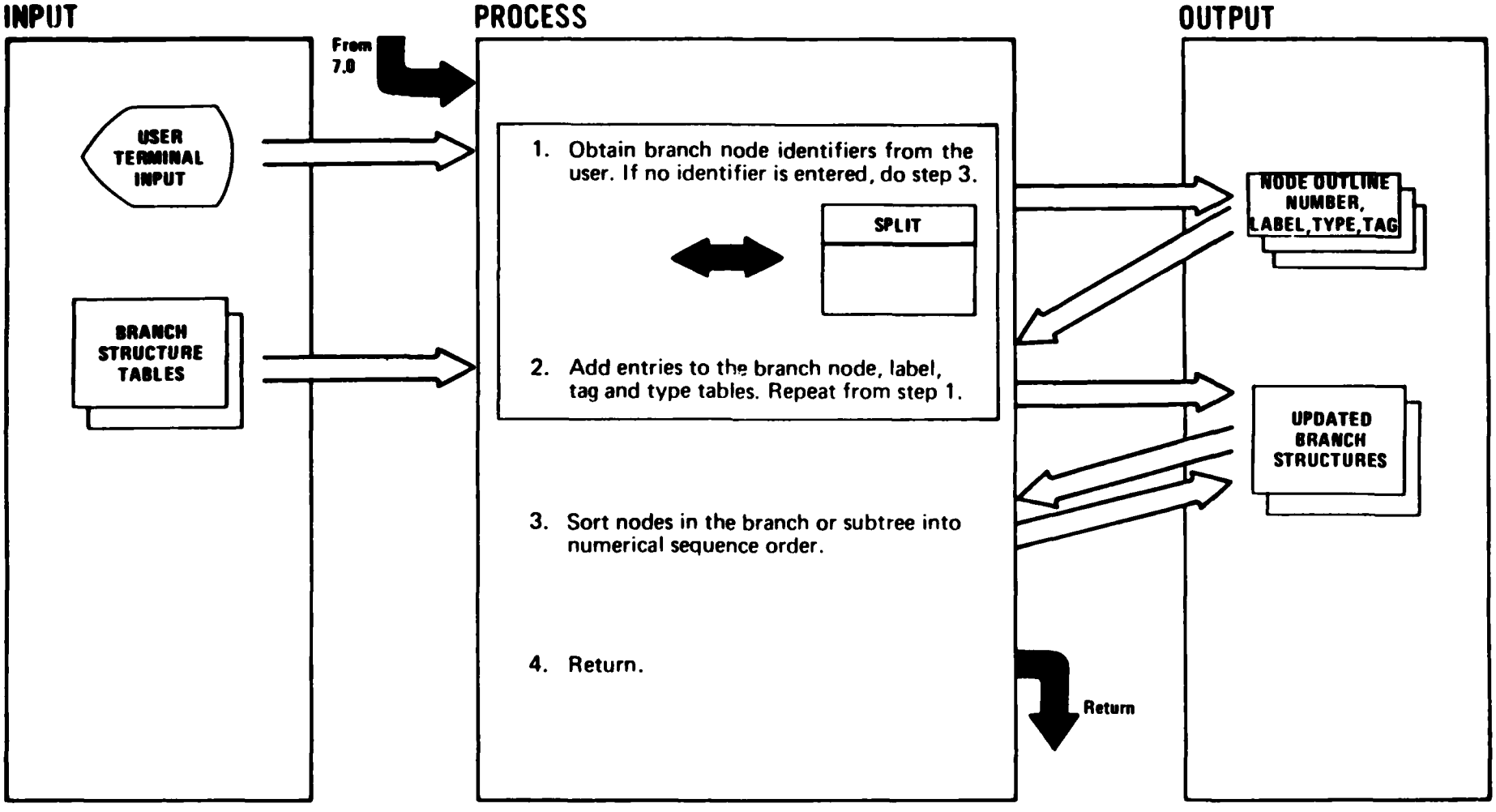


Extended Description

The user is allowed to create separate branch or subtree structures which may be added to the model structure under the "create a structure" process option.

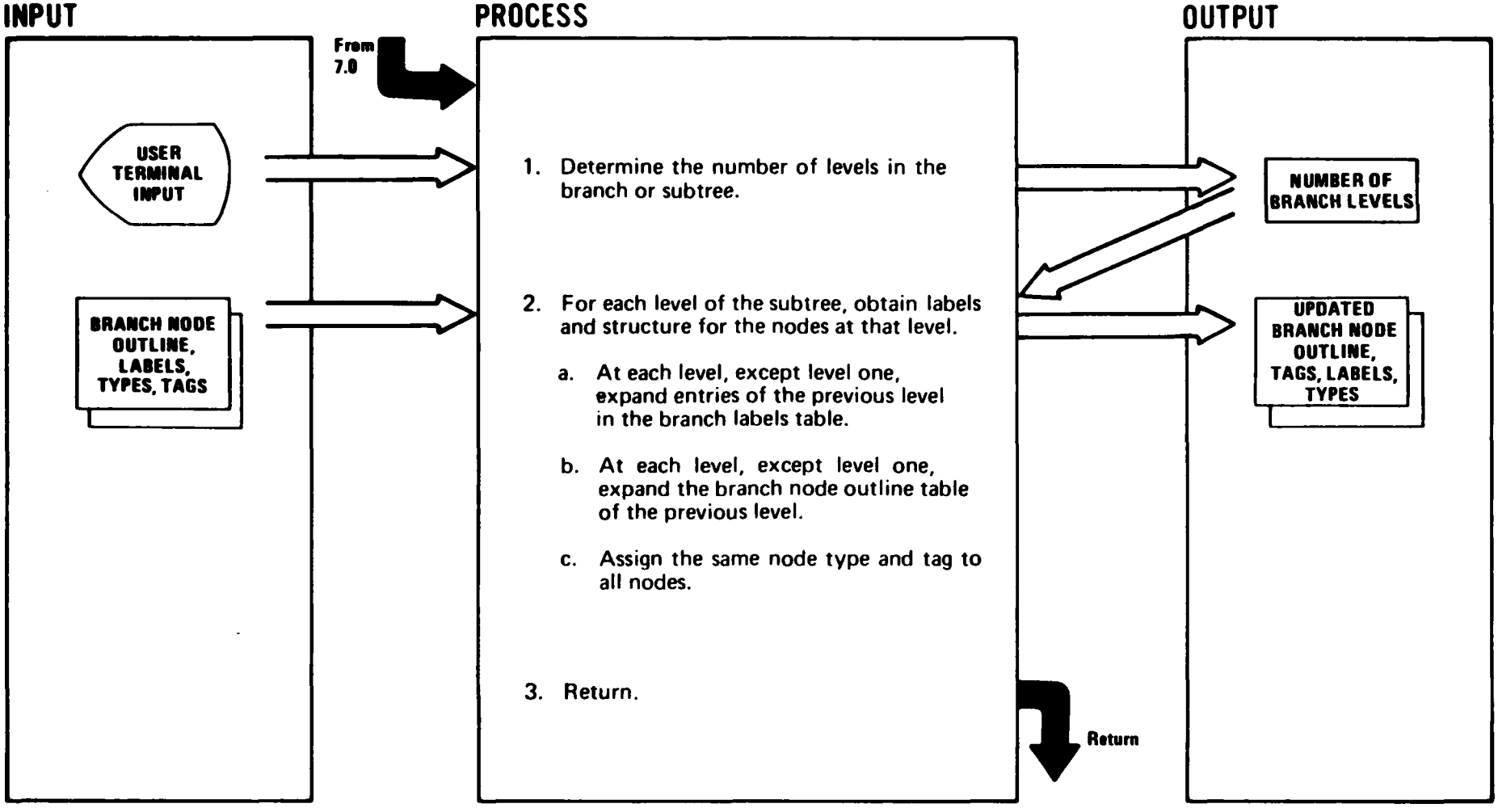
System Program STRUCTURE Name:

Diagram ID 7.1 Description Create a Non-Symmetric Branch Page ___ of ___



System Program STRUCTURE Name

Diagram ID 7.2 Description Create a Symmetric Branch Page ___ of ___

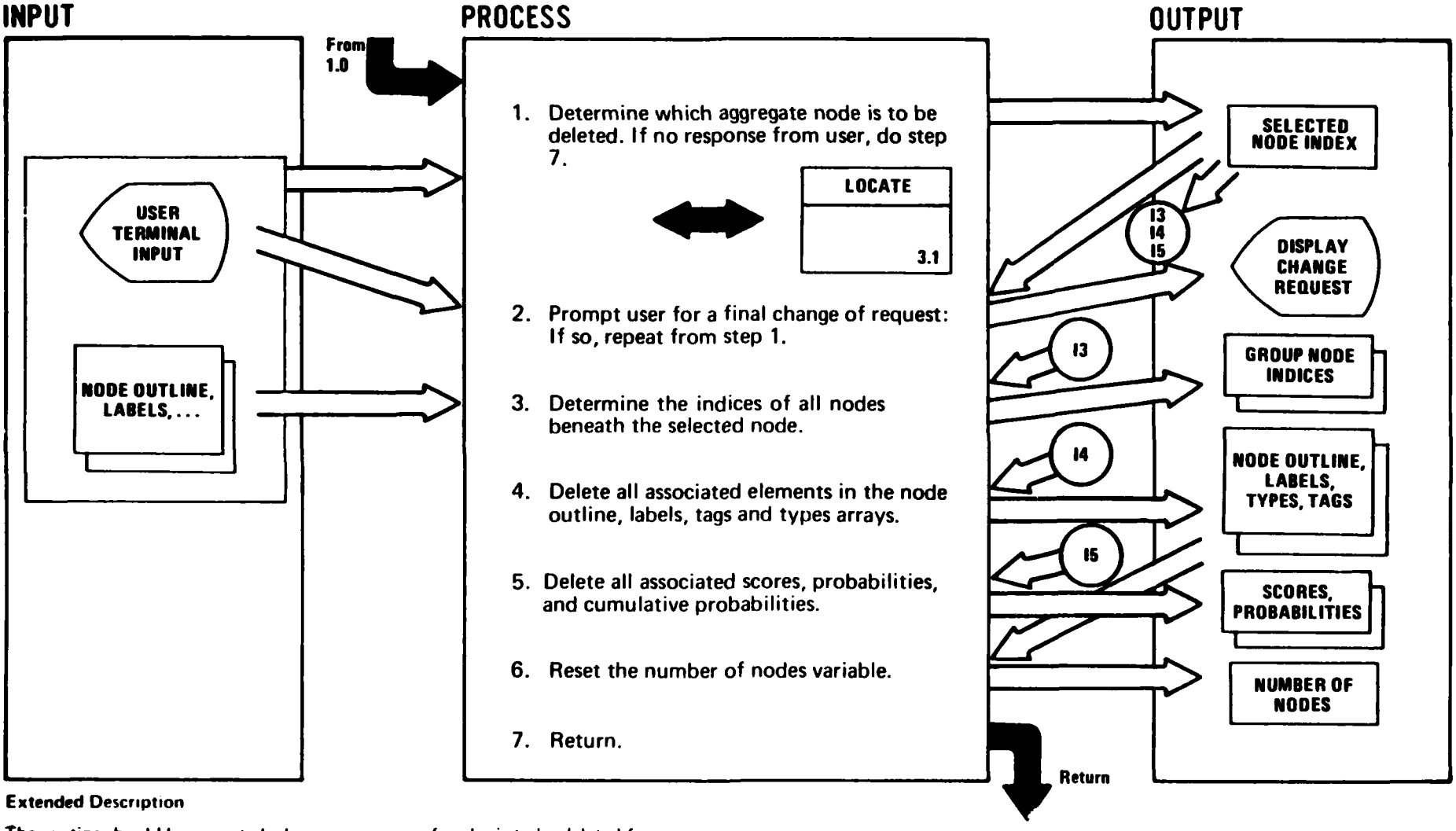


Extended Description

Step 2 processing ensures that for each subsequent level of a multilevel branch structure the outline number, types and labels are all added in the correct numerical sequence to the outline, types, tags and label entries at the previous level. (This is done for every branch node defined at the previous level.)

System Program STRUCTURE Name

Diagram ID 8.0 Description Prune a Structure Section Page ___ of ___

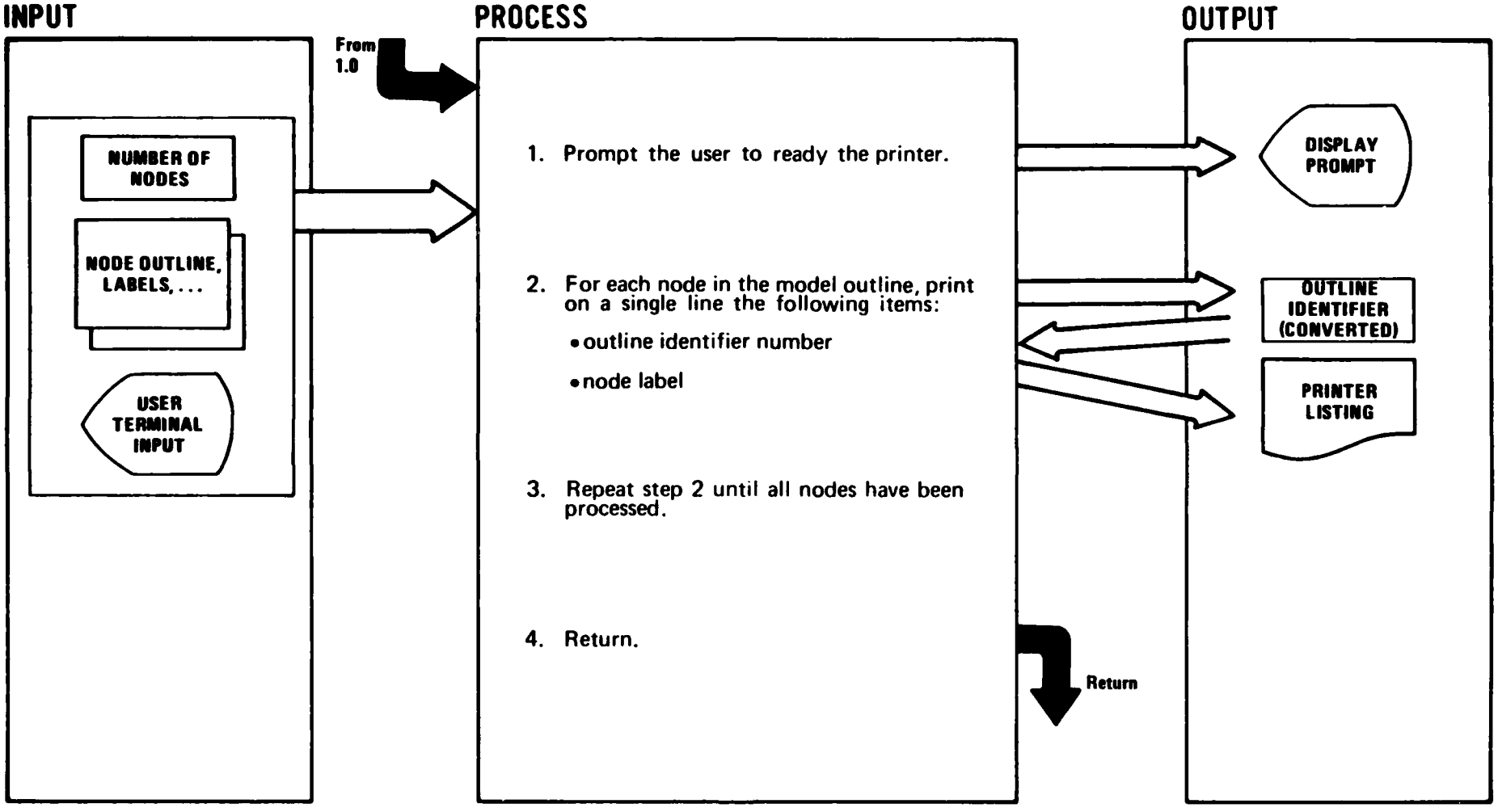


Extended Description

The routine should be executed whenever a group of nodes is to be deleted from an existing node structure. The grouped nodes are all hierarchically placed below a certain aggregate node; hence, a user specification of an aggregate node in step 1 will cause that node and all its subsequent nodes to be deleted.

System Program STRUCTURE Name

Diagram ID: 9.0 Description Print a Review Sheet Page ___ of ___



Extended Description

2. The decoded outline identifier number is formatted for output. The output should be equivalent to the user's original input during the creation of the structure.

System/Program: STRUCTURE Name:

Diagram ID: 10.0 Description General Routines Page: ___ of ___

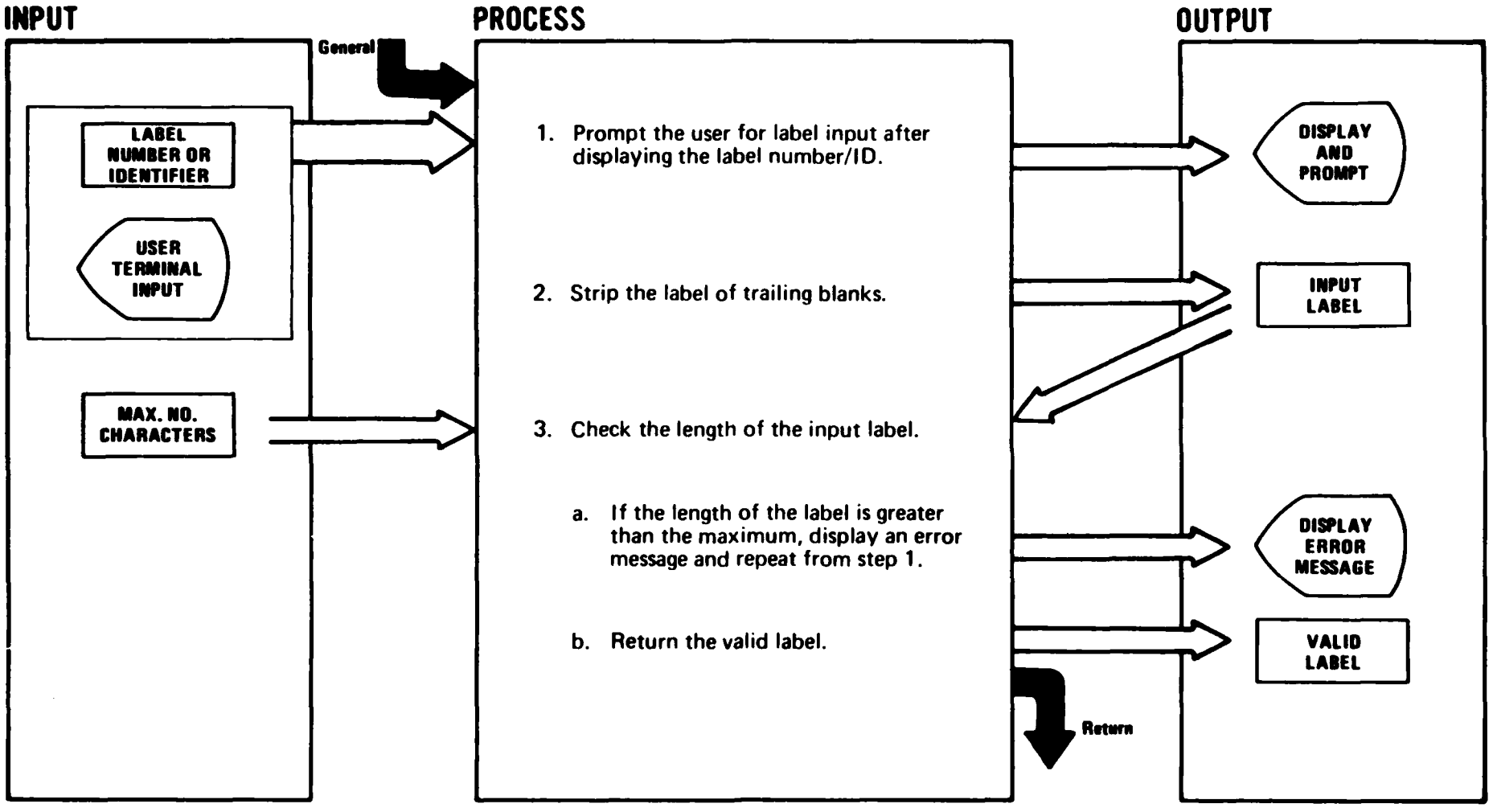
INPUT	PROCESS	OUTPUT

Extended Description

Generalized routines are directly invoked by functional procedures and return to the calling programs.

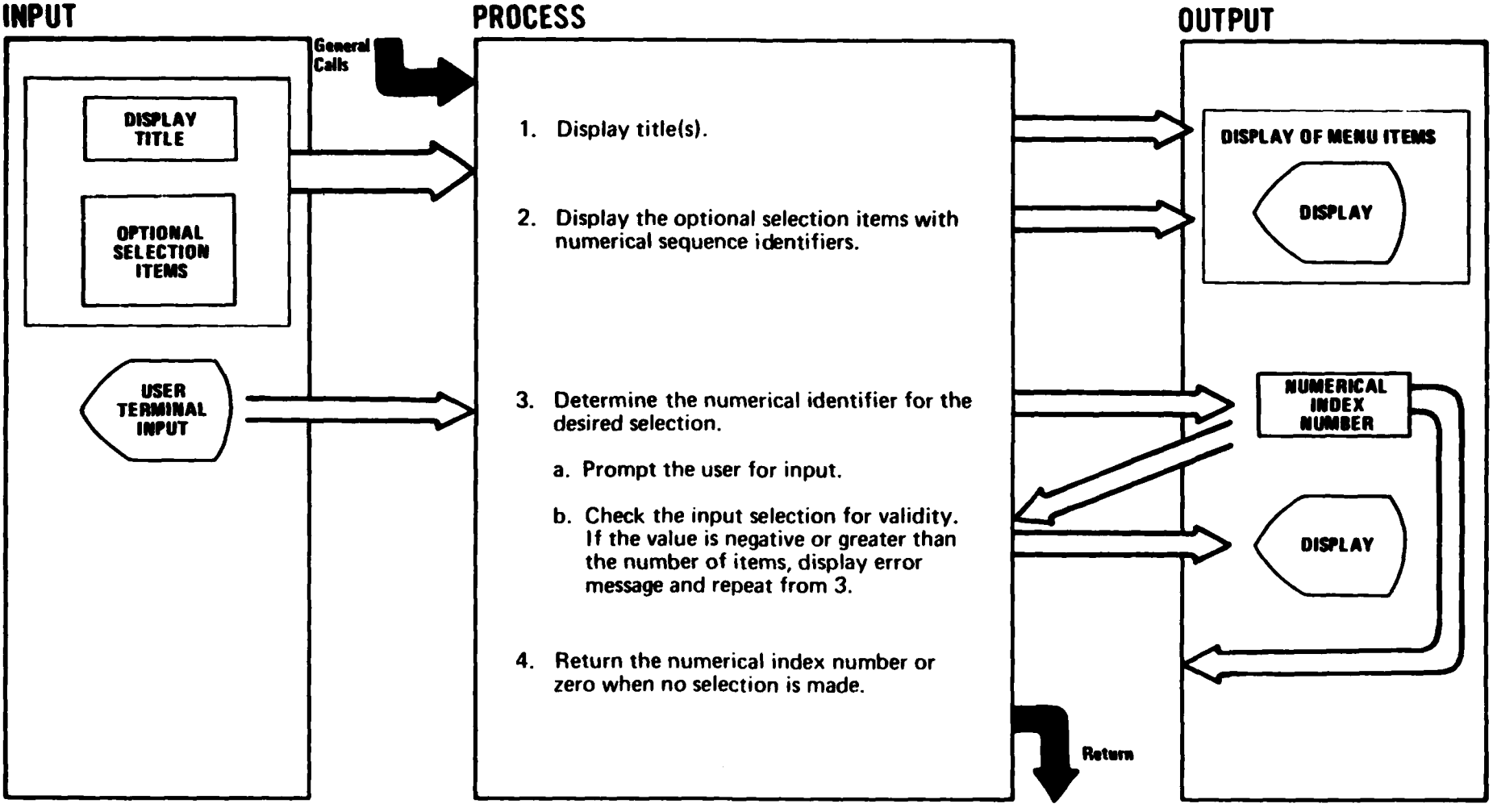
System/Program: STRUCTURE Name: ENTER LABELS

Diagram ID: 10.1 Description: Enter Labels Routine Page: ____ of ____



System/Program: STRUCTURE Name: MENU

Diagram ID: 10.2 Description: Display a Menu Page: ____ of ____



Extended Description

1. The title is passed to this routine so that the display will remain in context with the processing function. For example, a title may be 'DISPLAY RESULTS.'

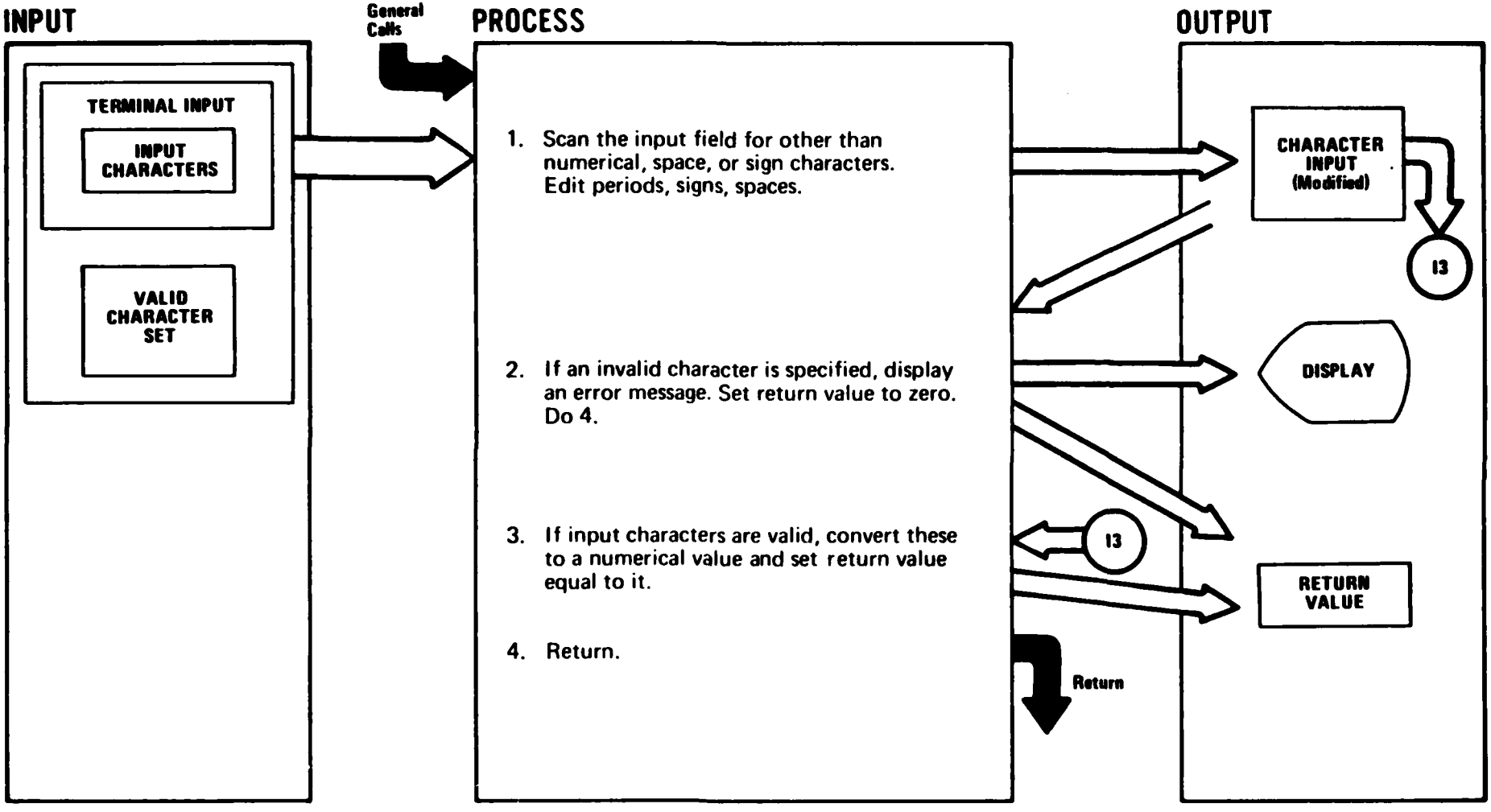
2. The selections that describe what is optimal are passed as input and are displayed in a list or cookbook MENU format along with item sequence numbers.

3. Prompt the user for the item sequence number of the choice selection.

Check the validity of the user input.

System/Program: STRUCTURE Name

Diagram ID 10.3 Description Select Only Numeric Input Page: ___ of ___



Extended Description

This routine will not be required if system error checking routines interface with the standard keyboard display input.

System/Program: RUN Name: RUN

Diagram ID: 1.0 Description: Entrance to Run Program Page: 1 of 2

INPUT

PROCESS

OUTPUT

From Control

1 . Display message telling user to load data tape before proceeding.

2 . Wait for user response.

3 . Have user load a model.

LOADDRIVE 6.0

4 . Present available functions to be performed.

MENU 10.2

5 . Based on user selection, perform one of the following (Repeat from step 4 after any of steps a - h).

a . Display or edit. 2.0

Pg. 2 5.b.

Extended Description

3. *The model variables are all loaded into the current work area at this time, or whenever the user wishes to load a new model. Consequently, this documentation assumes that these variables are "global" and always available for input to procedures, reference, or modification by subroutines.*

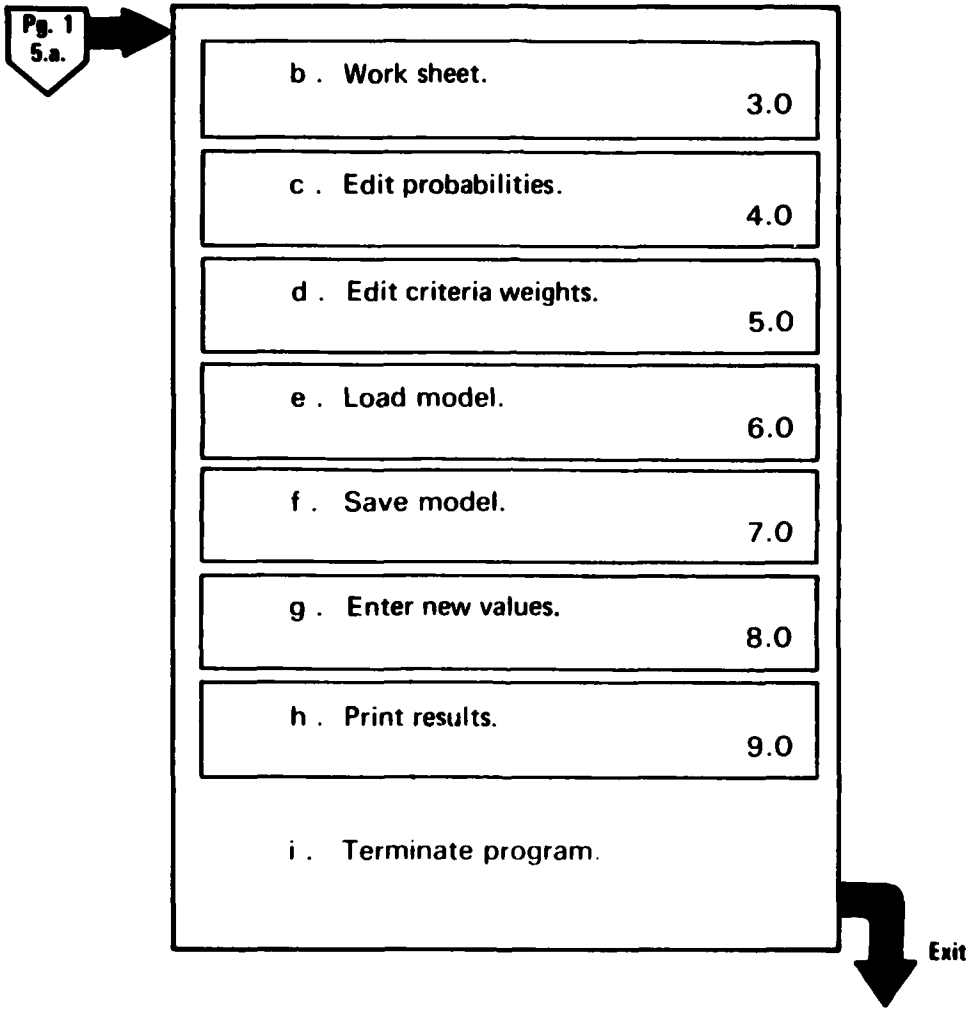
System Program: RUN Name: RUN

Diagram ID: 1.0 Description: Entrance to Run Program Page 2 of 2

INPUT

PROCESS

OUTPUT



System Program RUN Name SELECT

Diagram ID 2.0 Description Display or Edit Page 1 of 2

INPUT

PROCESS

OUTPUT

1.0
4.0

1 . Blank display screen.

2 . Display request for node outline number.

3 . Read a line from the terminal.

4 . Convert input character string to a numeric vector.

NUMBERSONLY
10.3

5 . If numeric vector is not null,

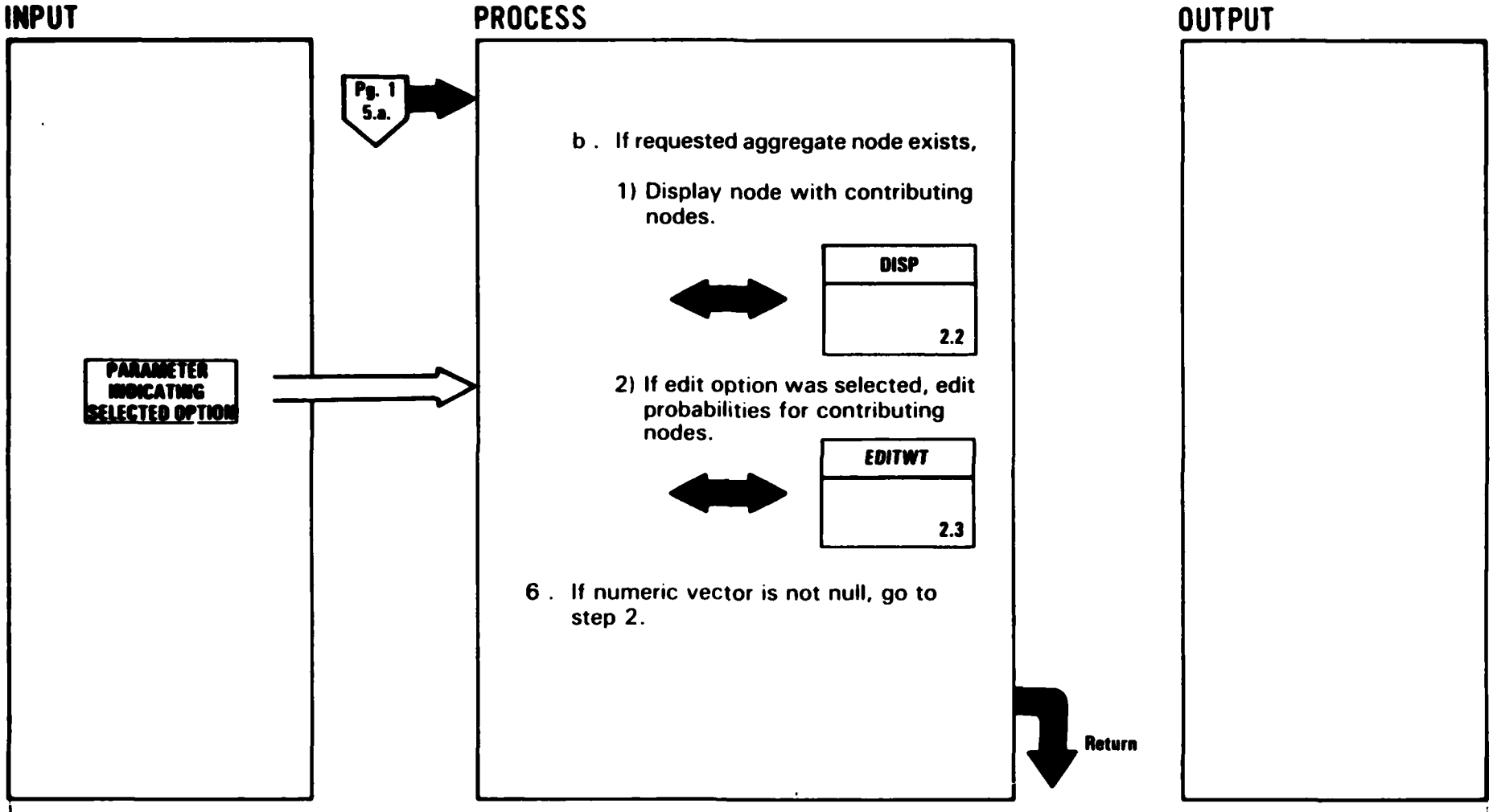
a . Determine node to display/edit.

GETNODE
2.1

Pg. 2
5.b.

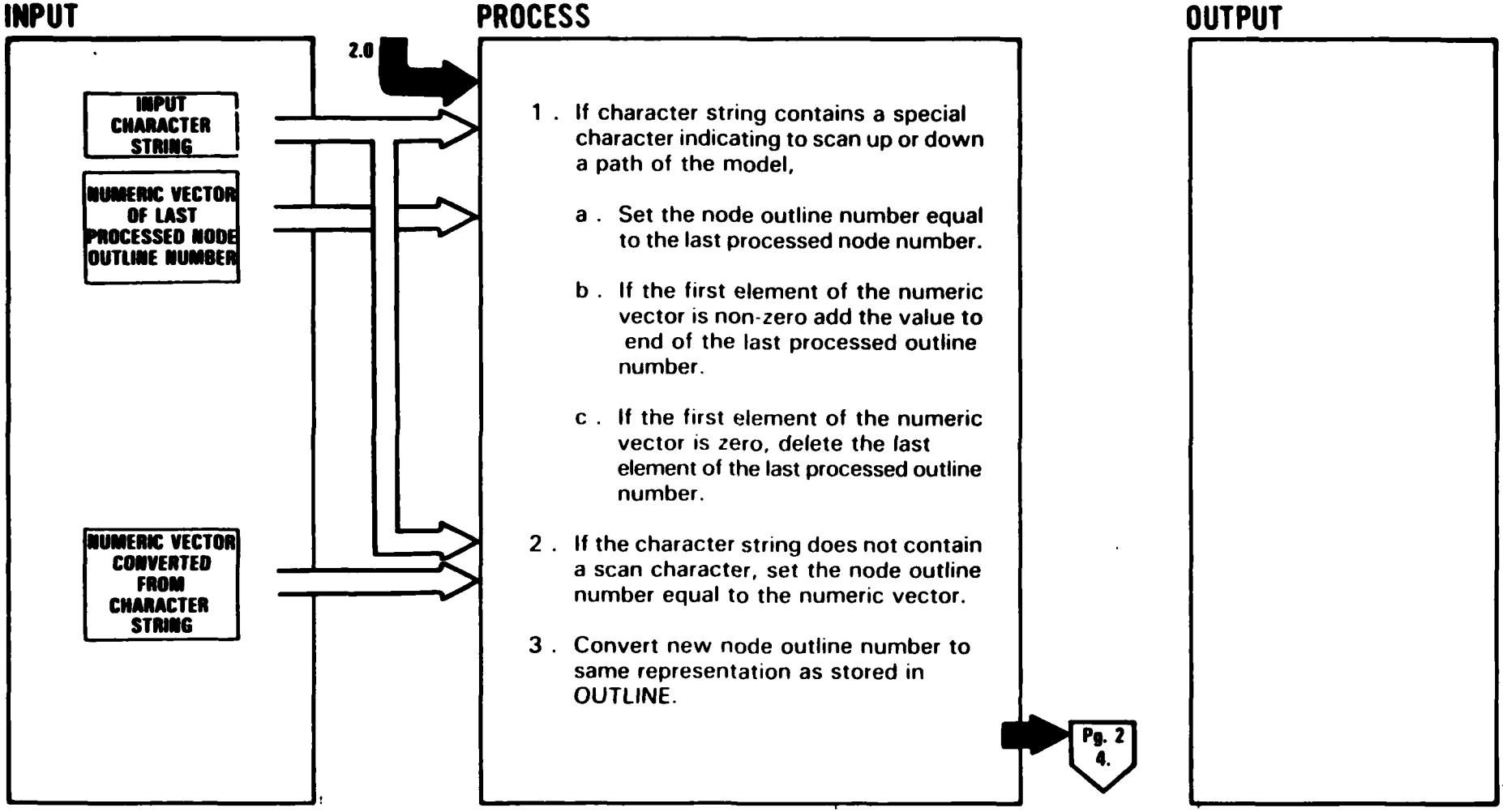
System Program RUN Name SELECT

Diagram ID 2.0 Description Display or Edit Page 2 of 2



System/Program: RUN Name: GETNODE

Diagram ID: 2.1 Description: Determine Node to Display Page: 1 of 2



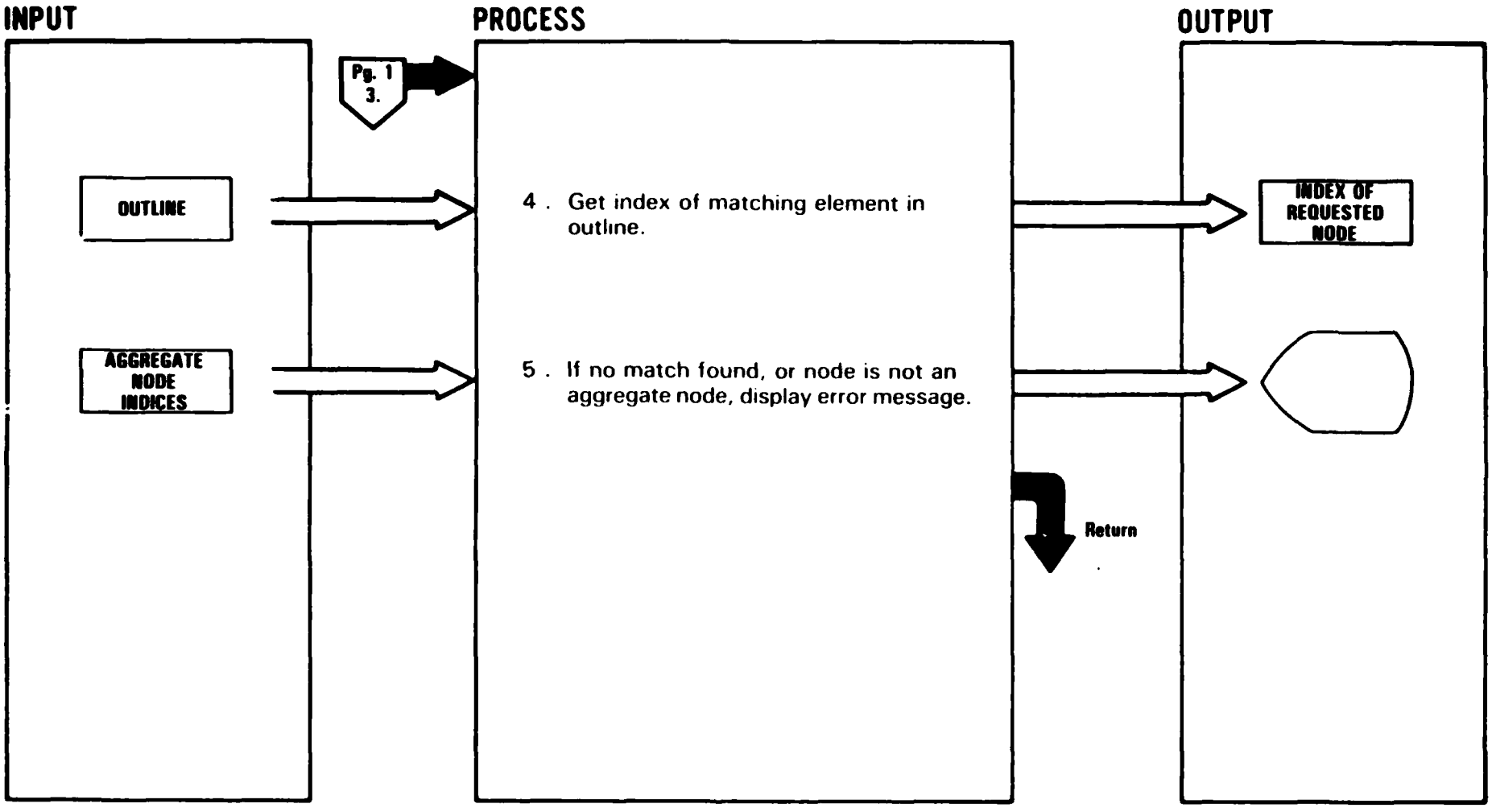
Extended Description

b. This generates a node outline number one level deeper than the previously processed node. For example, if the previously processed number were 3.2.5 and the input '6)' (where the right parenthesis is the scan operator) the new node outline number would be 3.2.5.6.

c. This generates a node outline number one level higher than the previously processed node. For example, if the previously processed number were 3.2.5 and the input zero followed by the right parenthesis scan operator, the new node outline number would be 3.2.

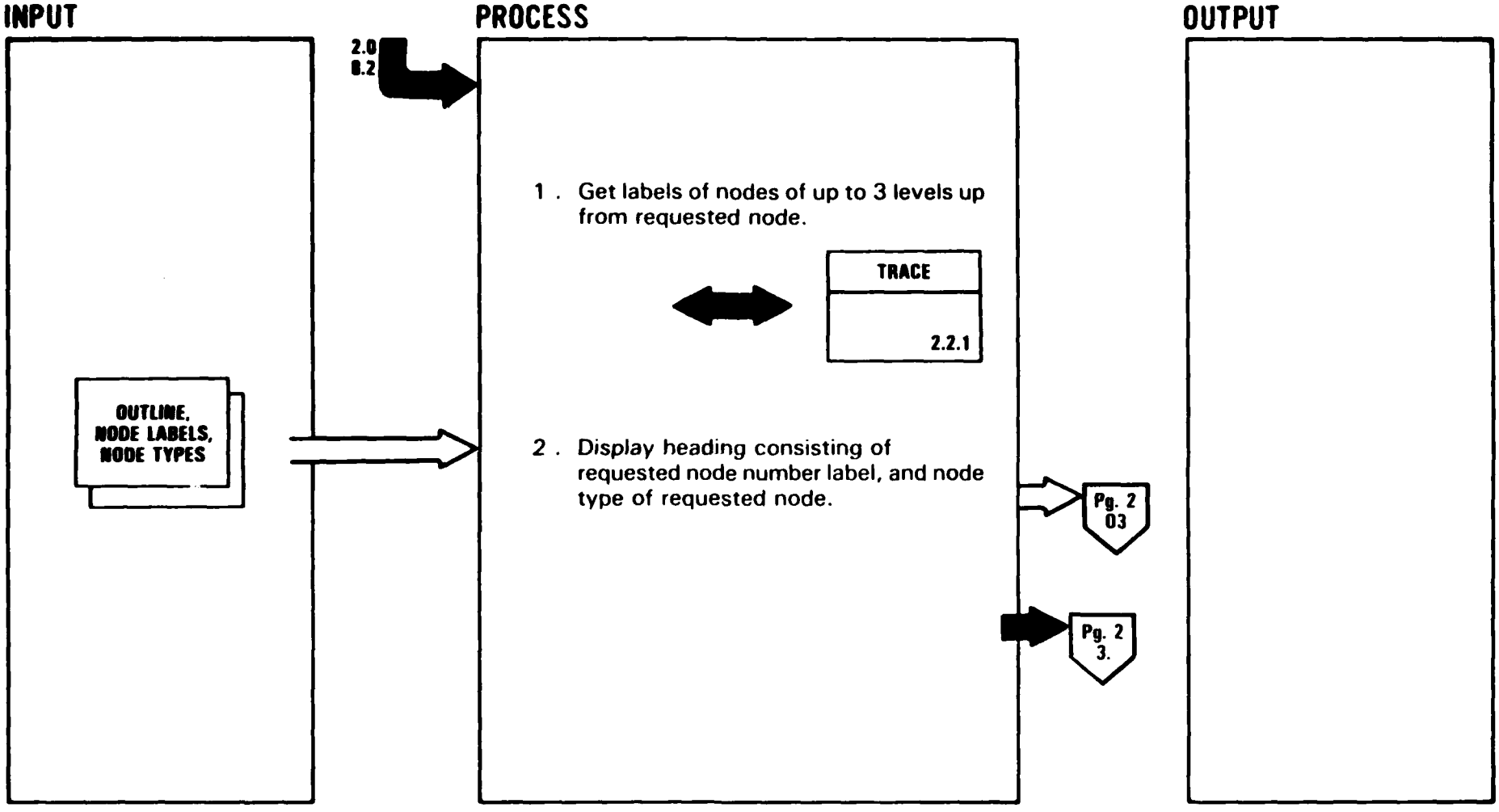
System/Program: RUN Name: GETNODE

Diagram ID: 2.1 Description: Determine Node to Display Page 2 of 2



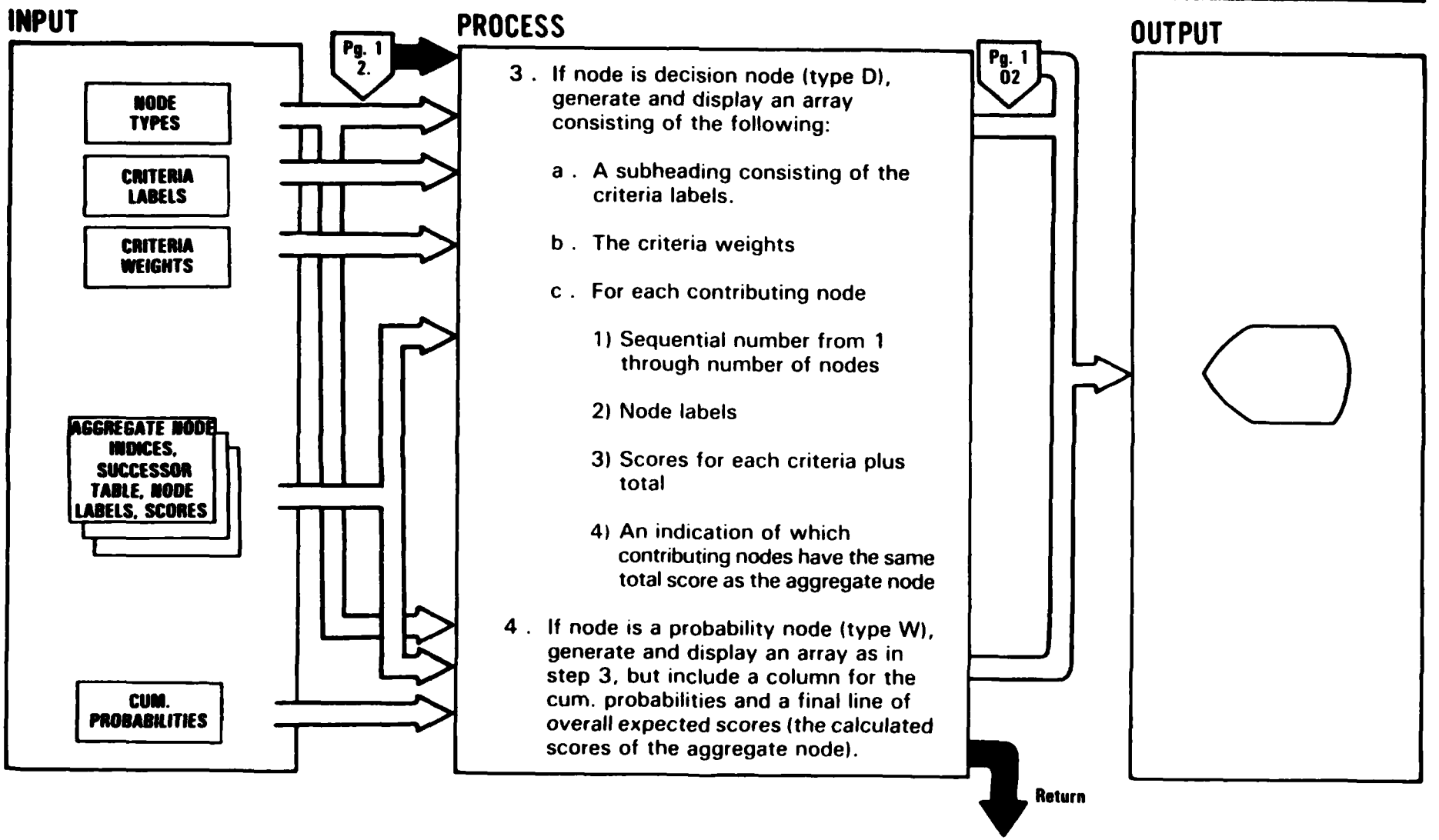
System Program RUN Name DISP

Diagram ID 2.2 Description Display Group Page 1 of 2



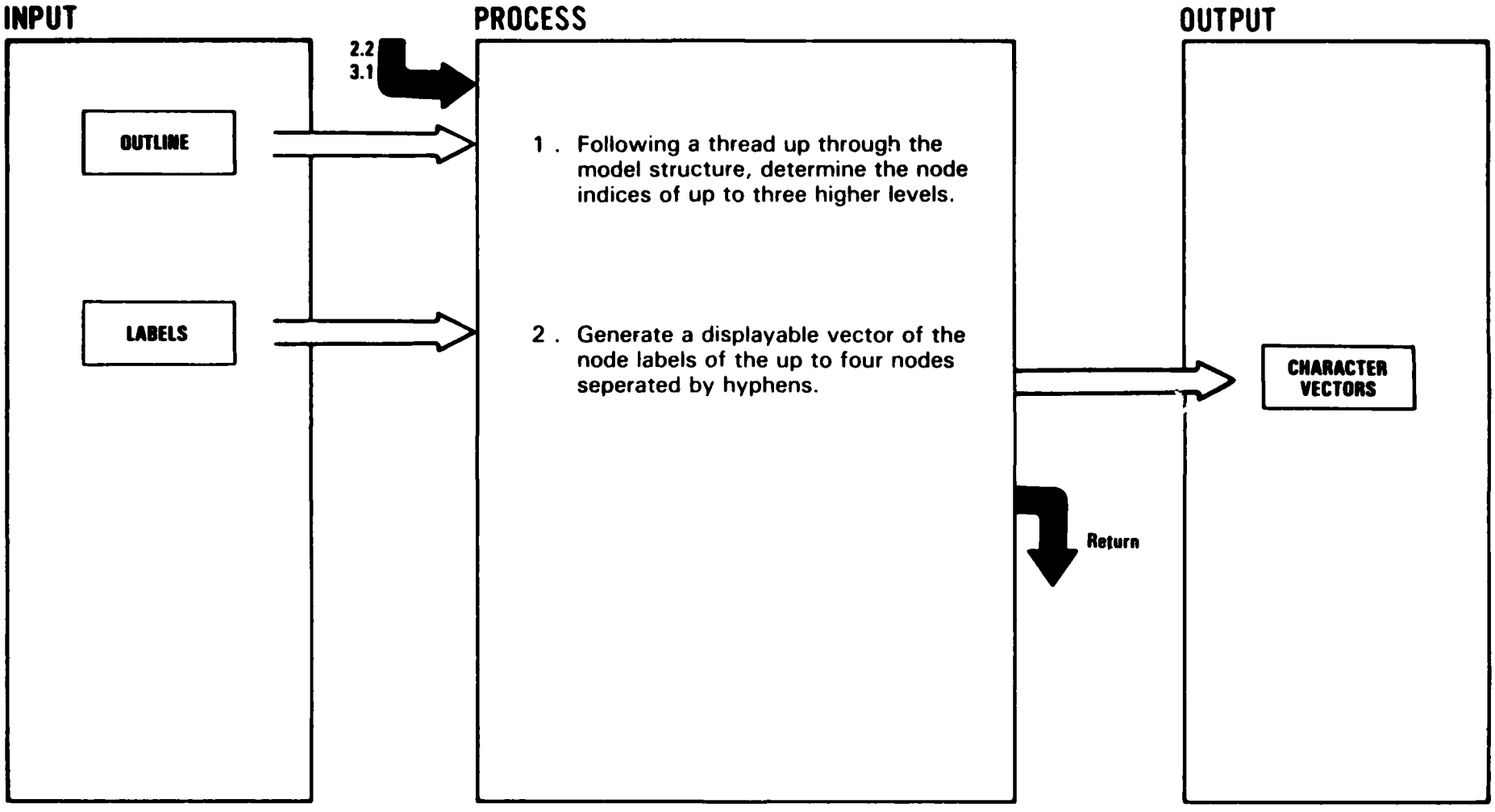
System/Program: RUN Name: DISP

Diagram ID: 2.2 Description: Diplay Group



System Program: RUN Name: TRACE

Diagram ID: 2.2.1 Description: Get Nodes of Thread Page: ___ of ___

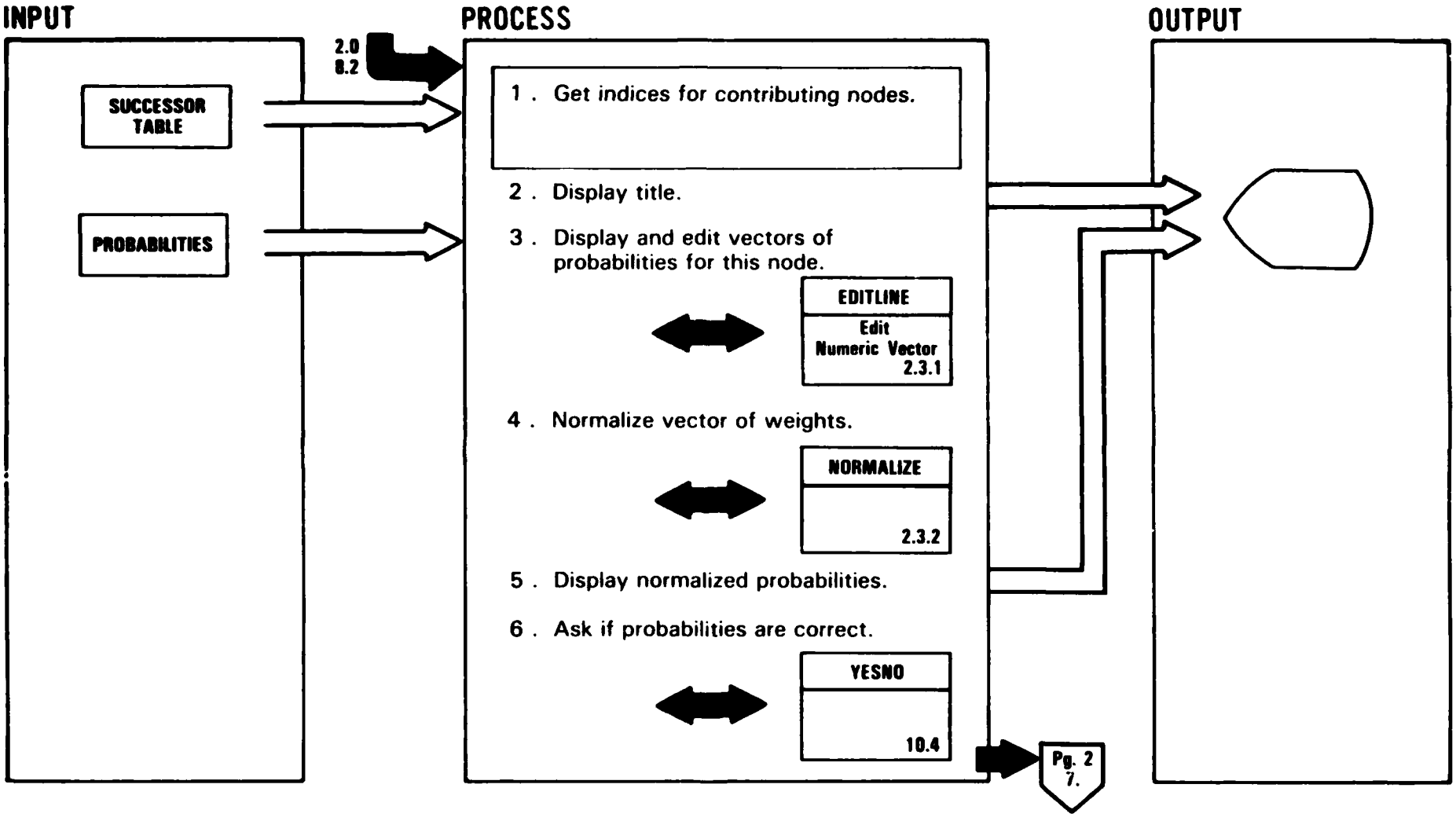


Extended Description

For instance, if the requested node number is 1.4.2.2.6, the next higher level would be 1.4.2.2, the next higher would be 1.4.2, and the fourth (or highest) calculated level would be 1.4.

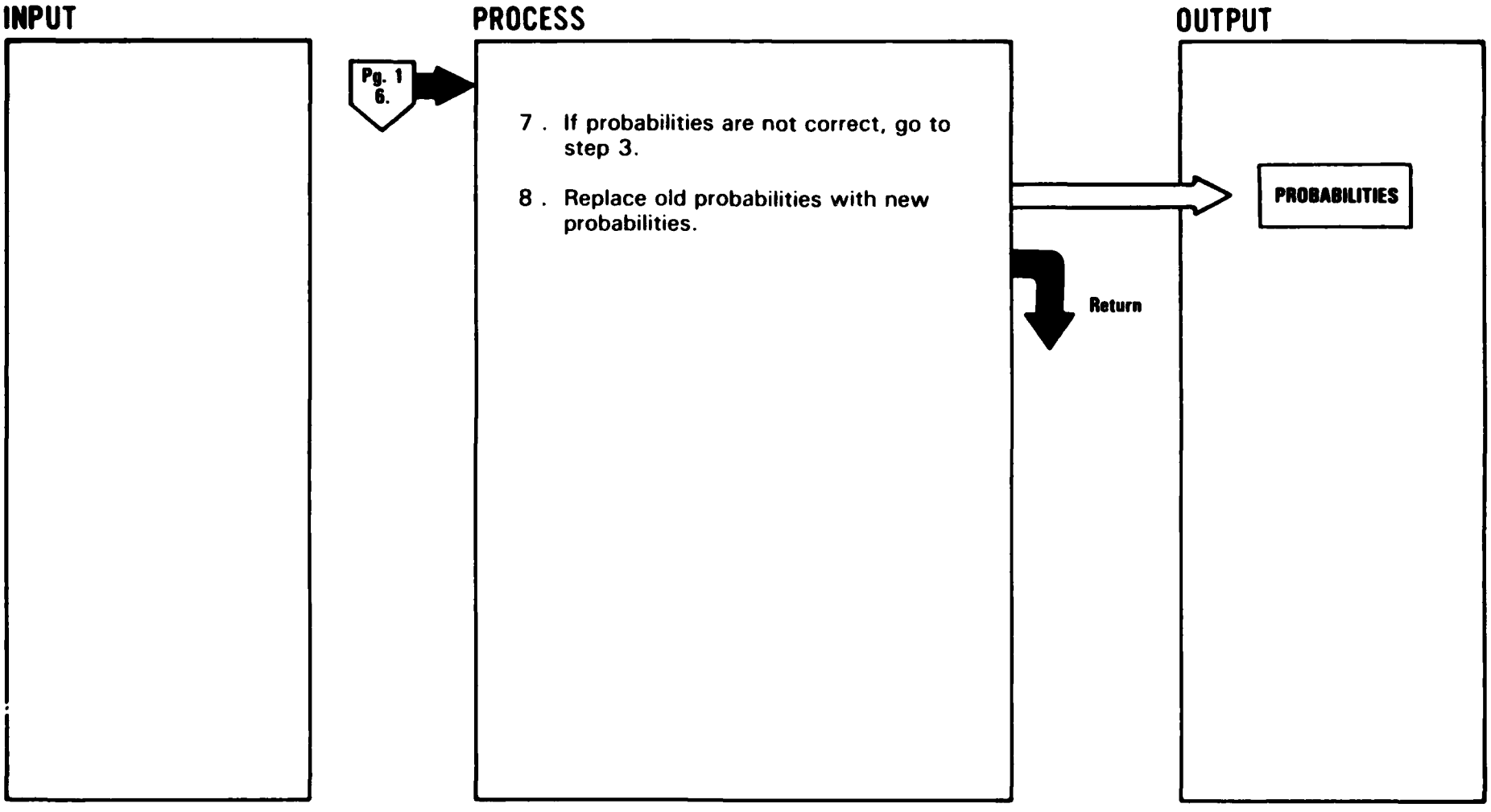
System Program RUN Name EDITWT

Diagram ID 2.3 Description Edit Probabilities Page 1 of 2



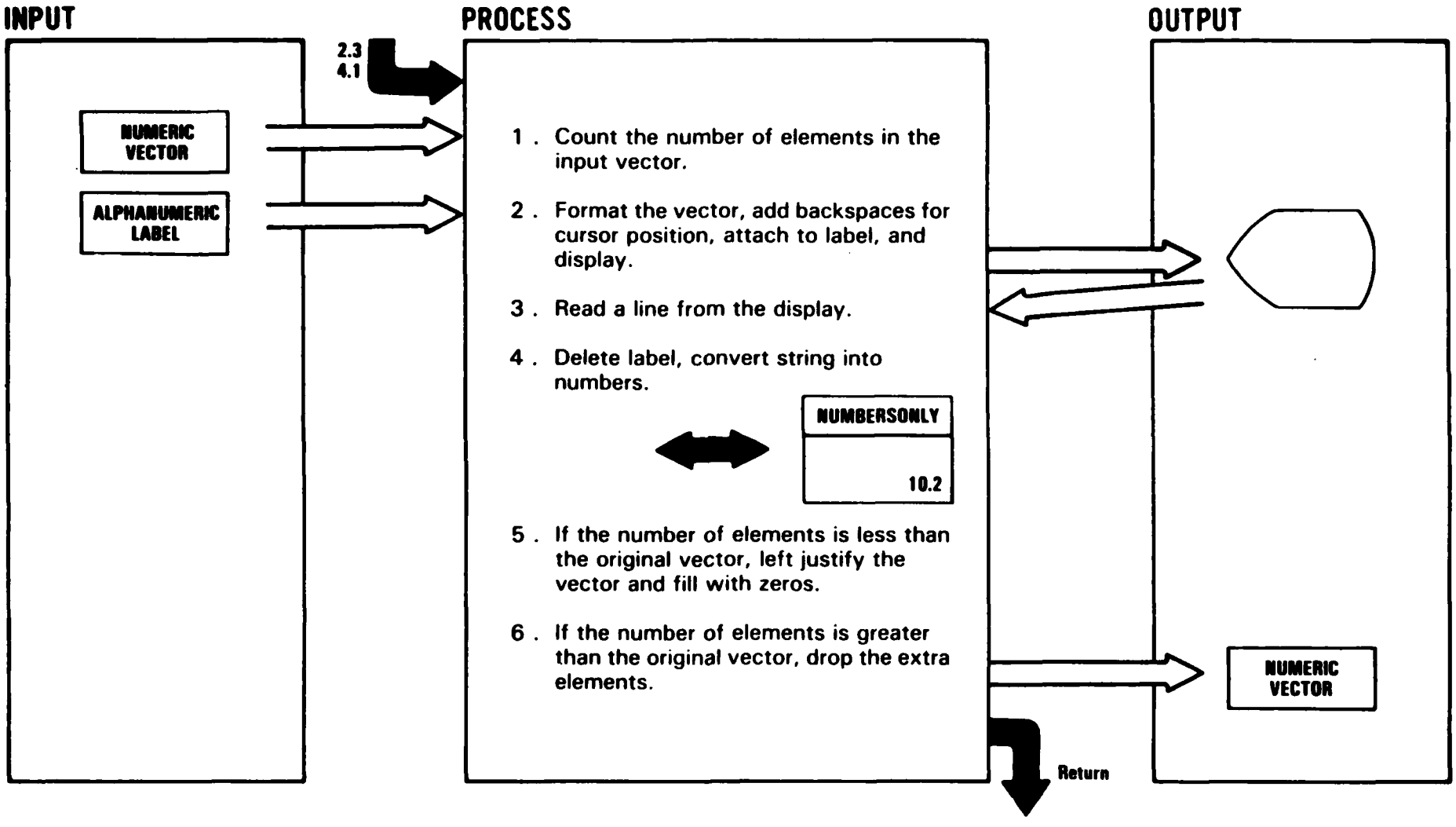
System Program: RUN Name: EDITWT

Diagram ID: 2.3 Description: Edit Probabilities Page 2 of 2



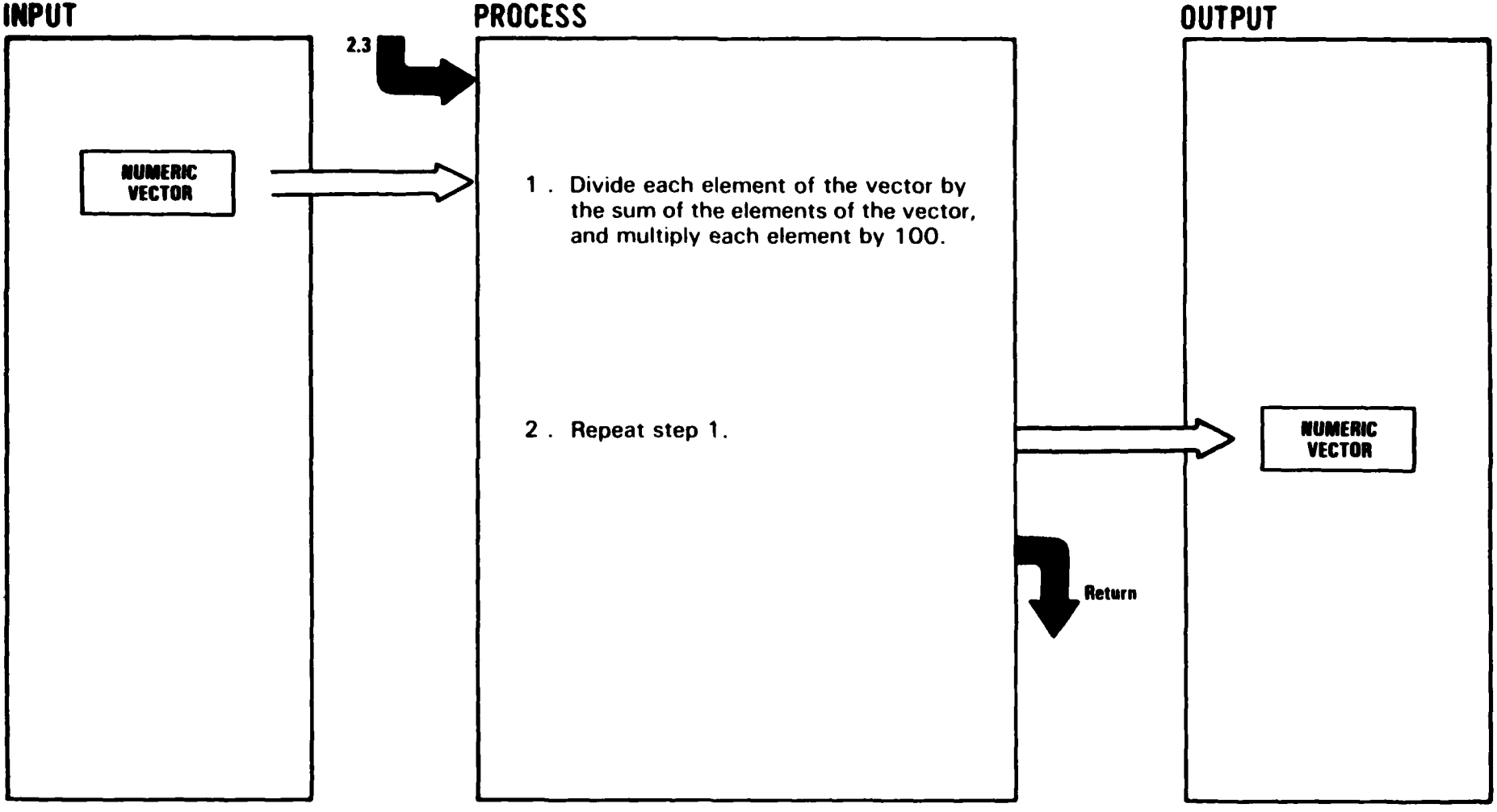
System Program: RUN Name: EDITLINE

Diagram ID: 2.3.1 Description: Edit a Numeric Vector Page ____ of ____



System Program RUN Name NORMALIZE

Diagram ID 2.3.2 Description Normalize a Vector of Numbers Page ___ of ___



Extended Description

1. Performing this operation converts a group of arbitrary values to a group of values that add up to 100. The values all maintain the same relativity.

2. Performing this operation twice allows the case where the original values are all zero. The final result is a group of equal numbers that add up to 100.

System Program RUN Name WORK

Diagram ID 3.0 Description Work Sheet Page ___ of ___

INPUT

PROCESS

1.0

1 . Present user option for type of work sheet.

MENU

10.2

2 . If user selected an option:

a . If user selected probability work sheet,

PROBWORK

3.1

b . If user selected value work sheet,

[illegible]ALWORK

3.2

c . Go to step 1.

Return

OUTPUT

System Program RUN Name PROBWORK

Diagram ID 3.1 Description Print Probability Work Sheet Page 1 of 2

INPUT

3.0

PROCESS

1 . Request user to turn on printer.

PRINTER

3.1.1

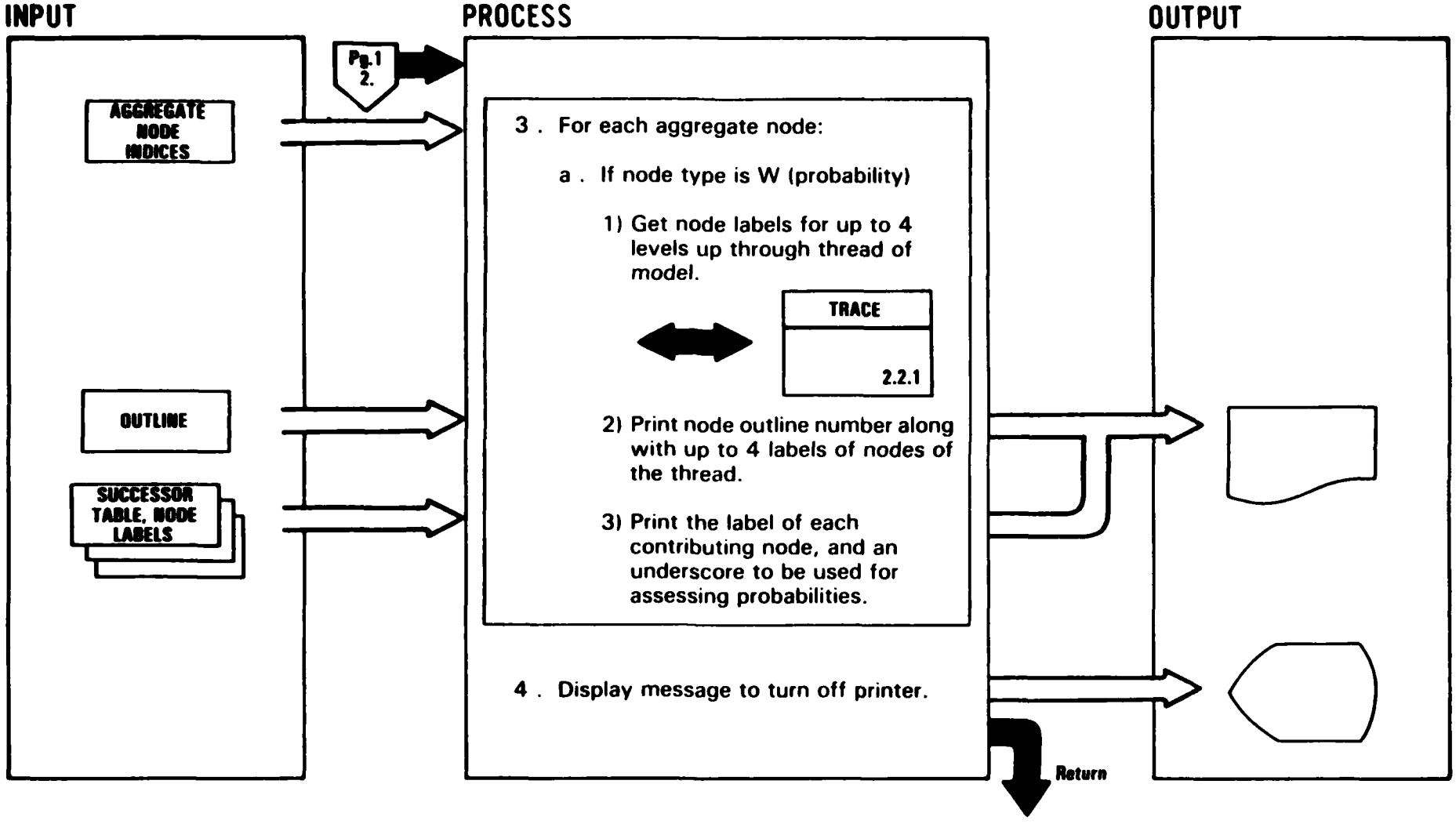
2 . Initialize loop index through aggregate nodes.

Pg. 2
3.

OUTPUT

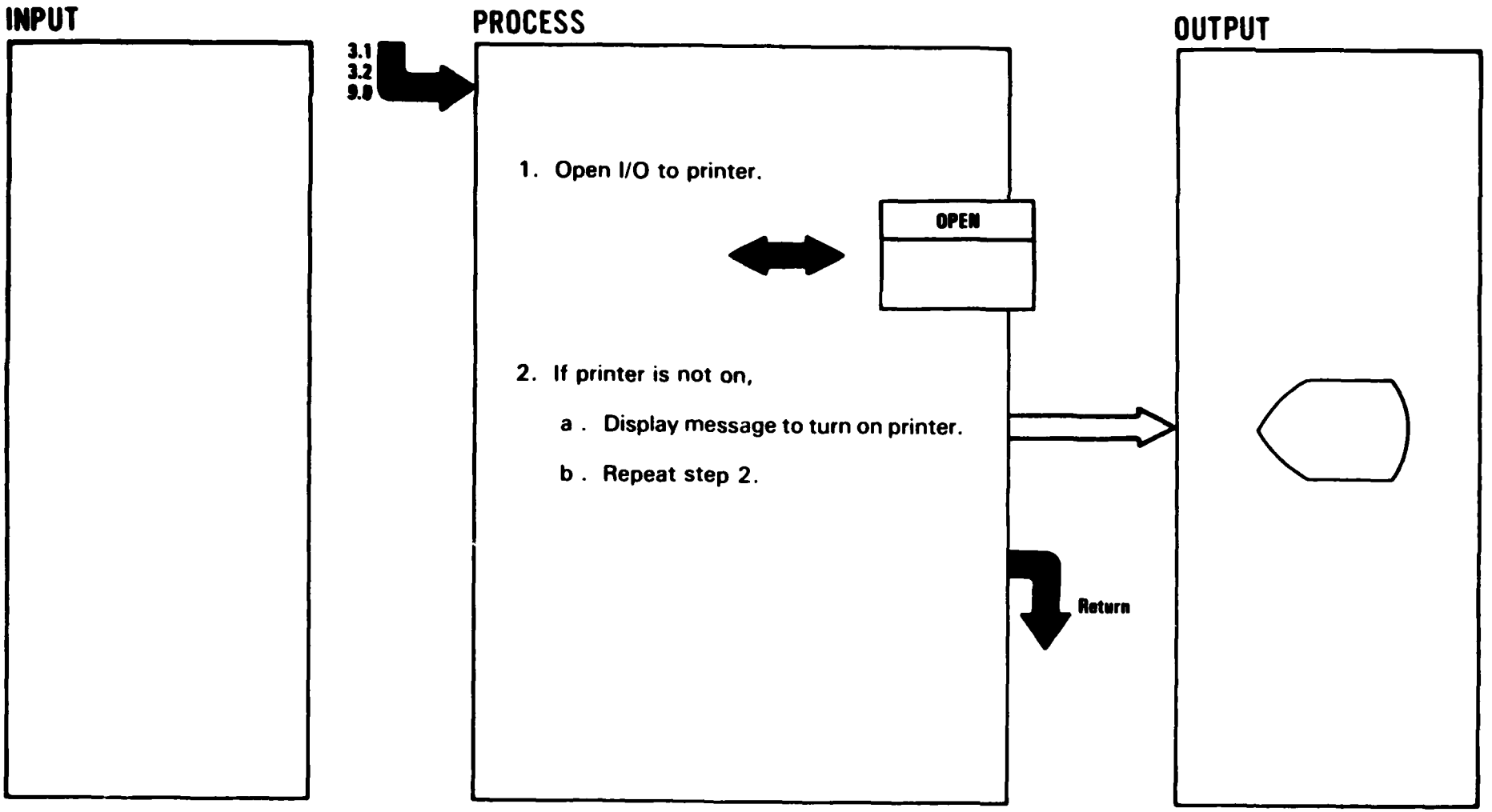
System Program RUN Name PROBWORK

Diagram ID 3.1 Description Print Probability Work Sheet Page 2 of 2



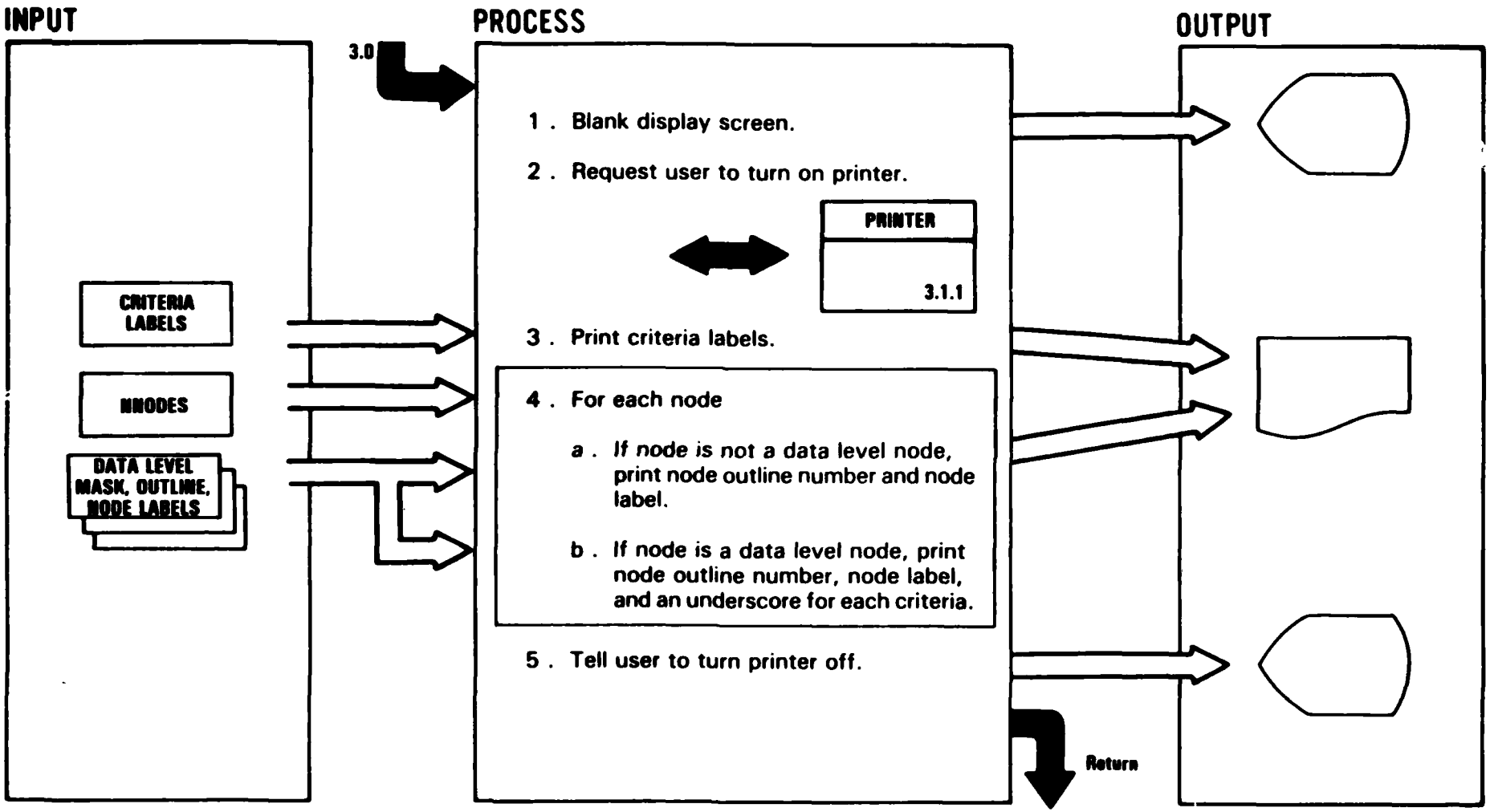
System Program: RUN Name: PRINTER

Diagram ID: 3.1.1 Description: Request Printer Page ___ of ___



System Program RUN Name VALWORK

Diagram ID 3.2 Description Print Value Work Sheet Page ___ of ___



System/Program: RUN Name: EDITOR

Diagram ID: 4.0 Description: Edit Probabilities

INPUT

PROCESS

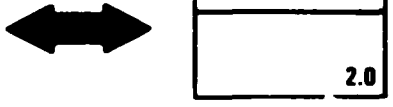
OUTPUT

1.0

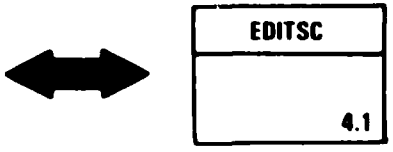
1 . Elicit number of desired option from user.

2 . If option selected,

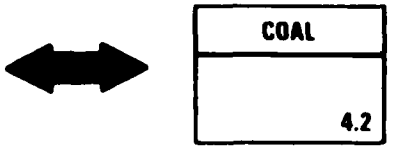
a . If edit probabilities selected,



b . If edit scores selected,



c . If edit independent probabilities selected,



Pg. 2 2.d.

System Program RUN Name EDITOR

Diagram ID 4.0 Description Edit Probabilities Page 2 of 2

INPUT

PROCESS

OUTPUT

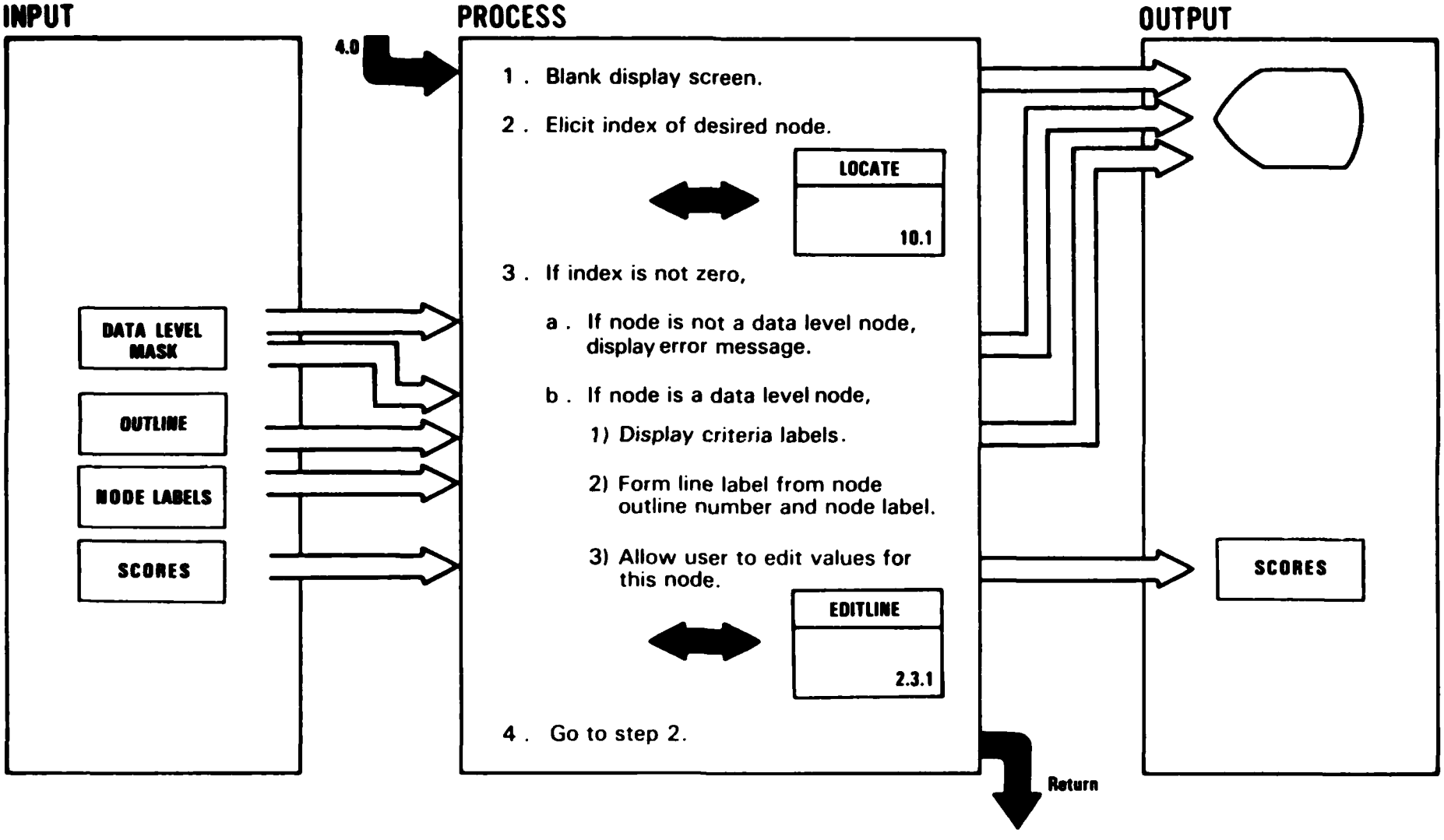
d . Go to step 1.

3. Solve decision tree.

ROLL
4.3

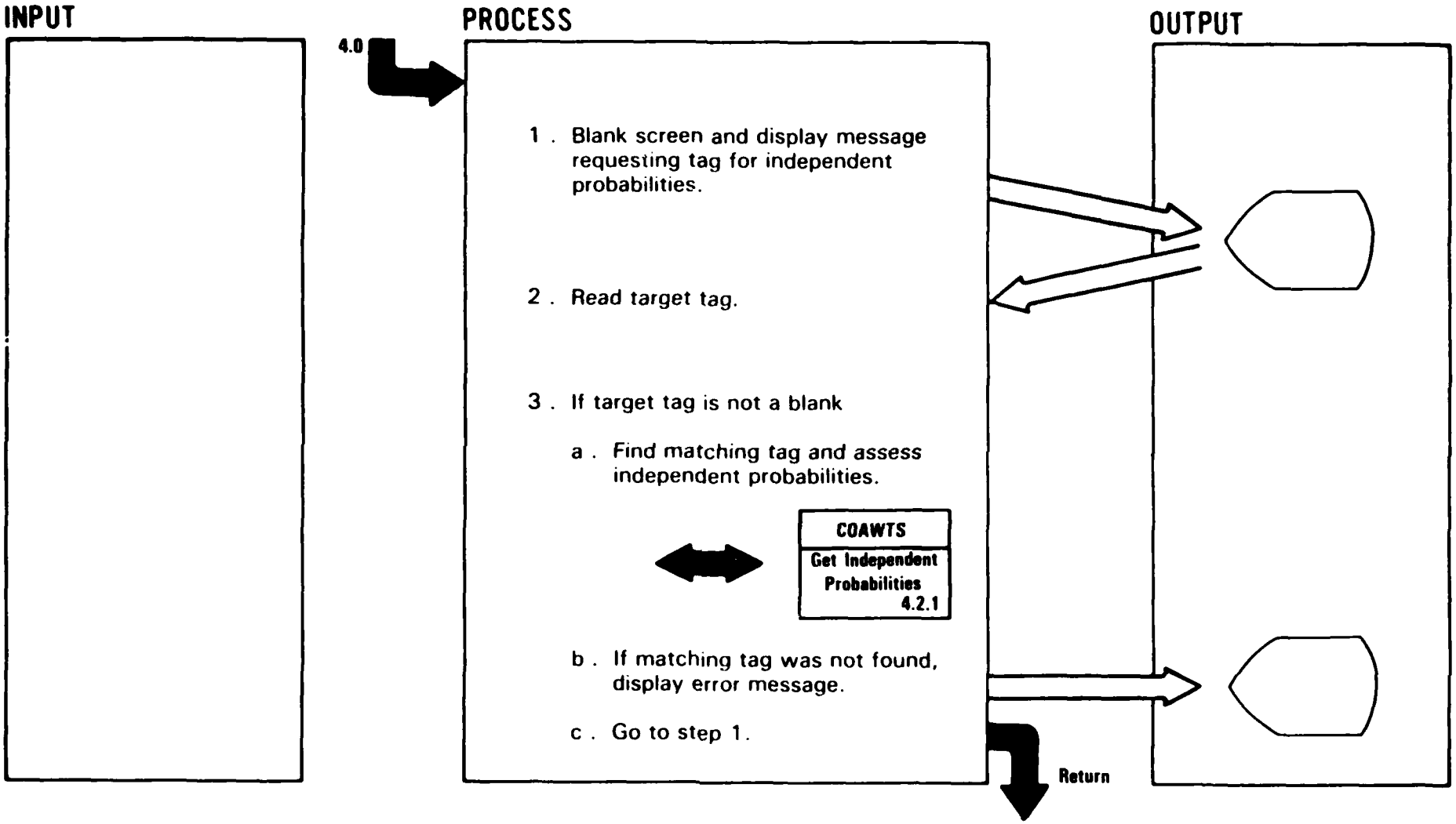
System/Program: RUN Name: EDITSC

Diagram ID: 4.1 Description: Edit Scores Page: ____ of ____



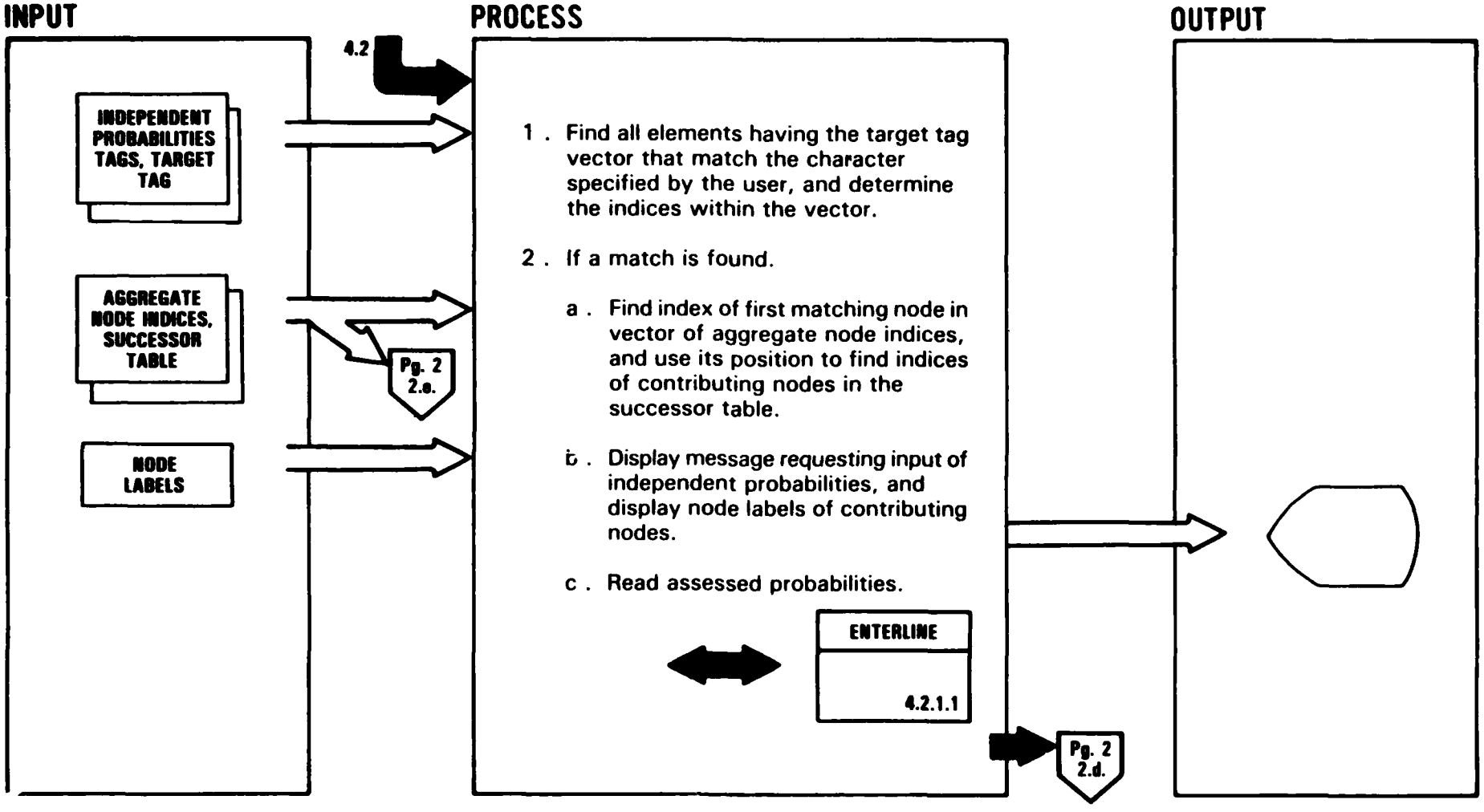
System Program RUN Name COAL

Diagram ID 4.2 Description Edit Independent Probabilities Page ___ of ___



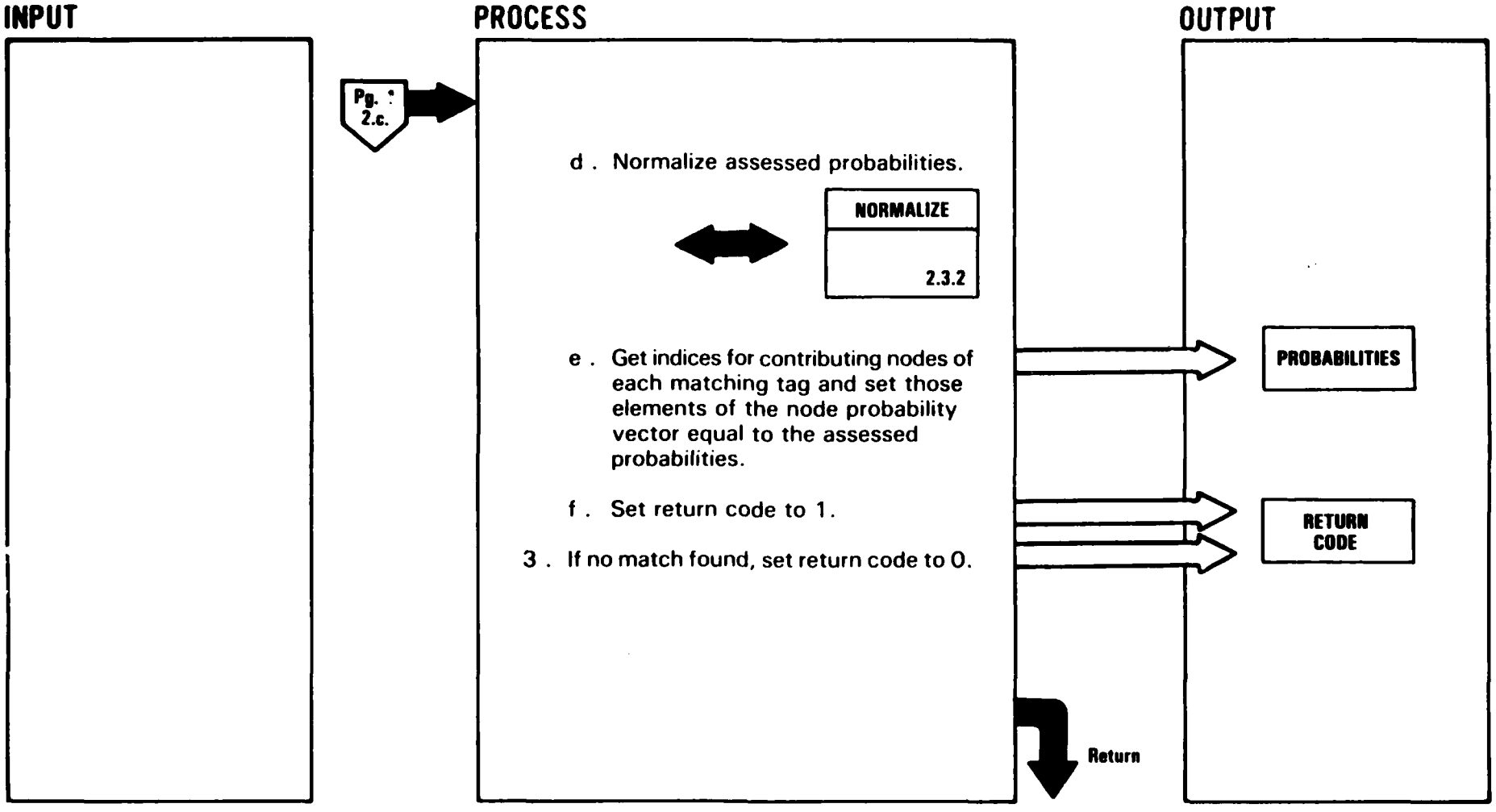
System/Program: RUN Name: COAWTS

Diagram ID: 4.2.1 Description: Get Independent Probabilities Page: 1 of 2



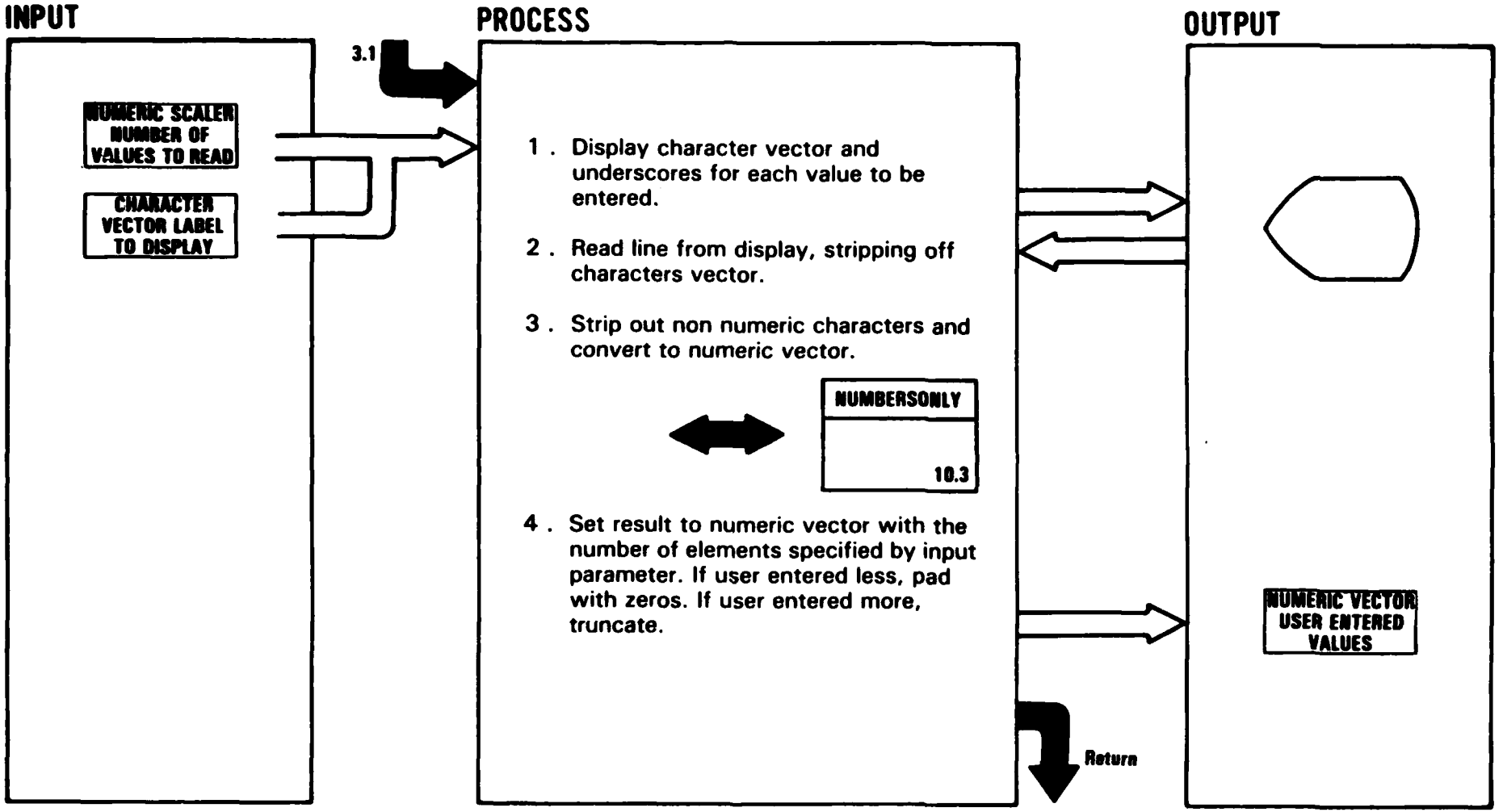
System Program RUN Name COAWTS

Diagram ID: 4.2.1 Description Get Independent Probabilities Page: 2 of 2



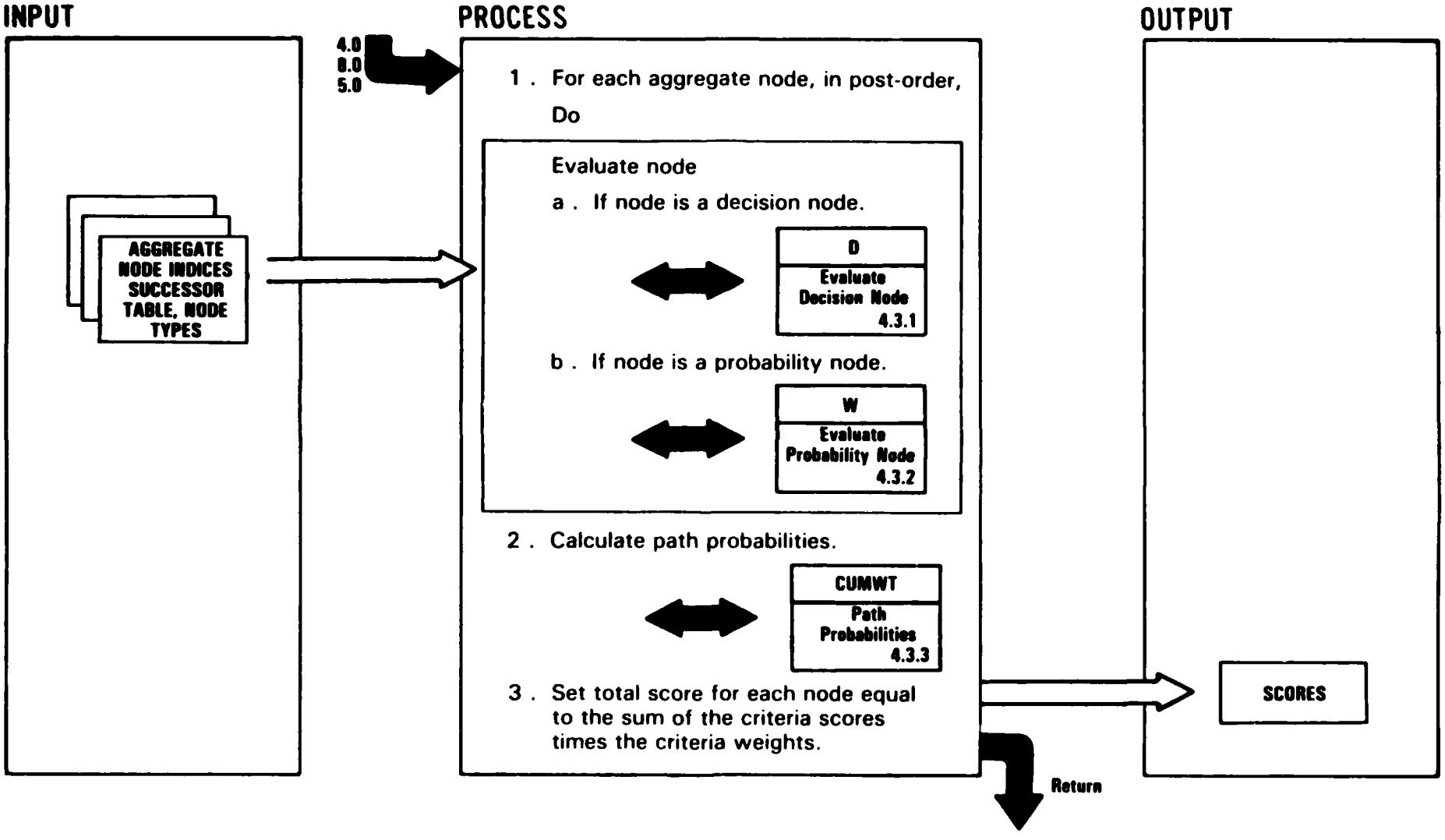
System/Program: RUN Name: ENTERLINE

Diagram ID. 4.2.1.1 Description Enter Numeric Vector Page: ____ of ____



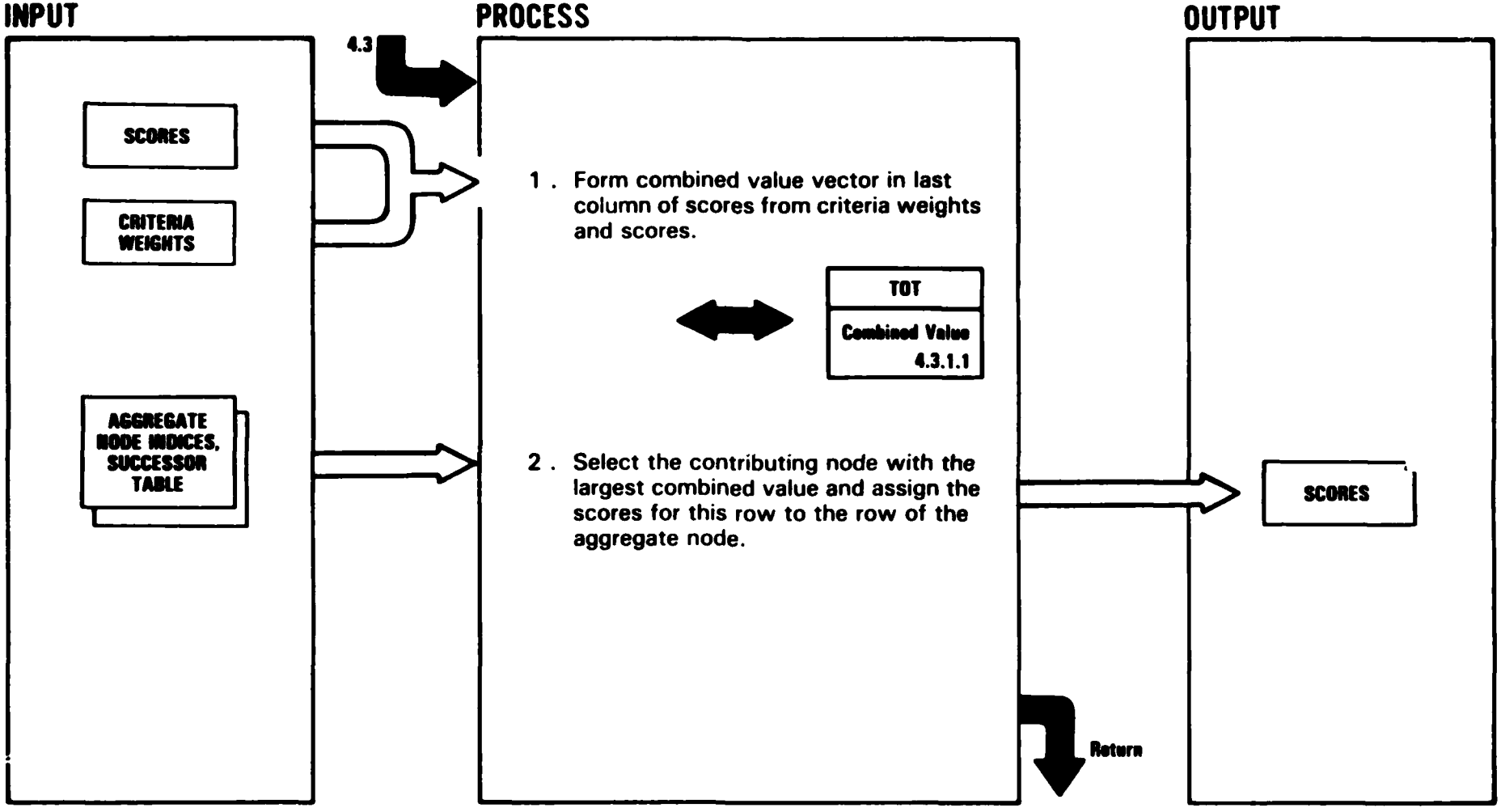
System Program RUN Name ROLL

Diagram ID 4.3 Description Solve Decision Tree Page ___ of ___



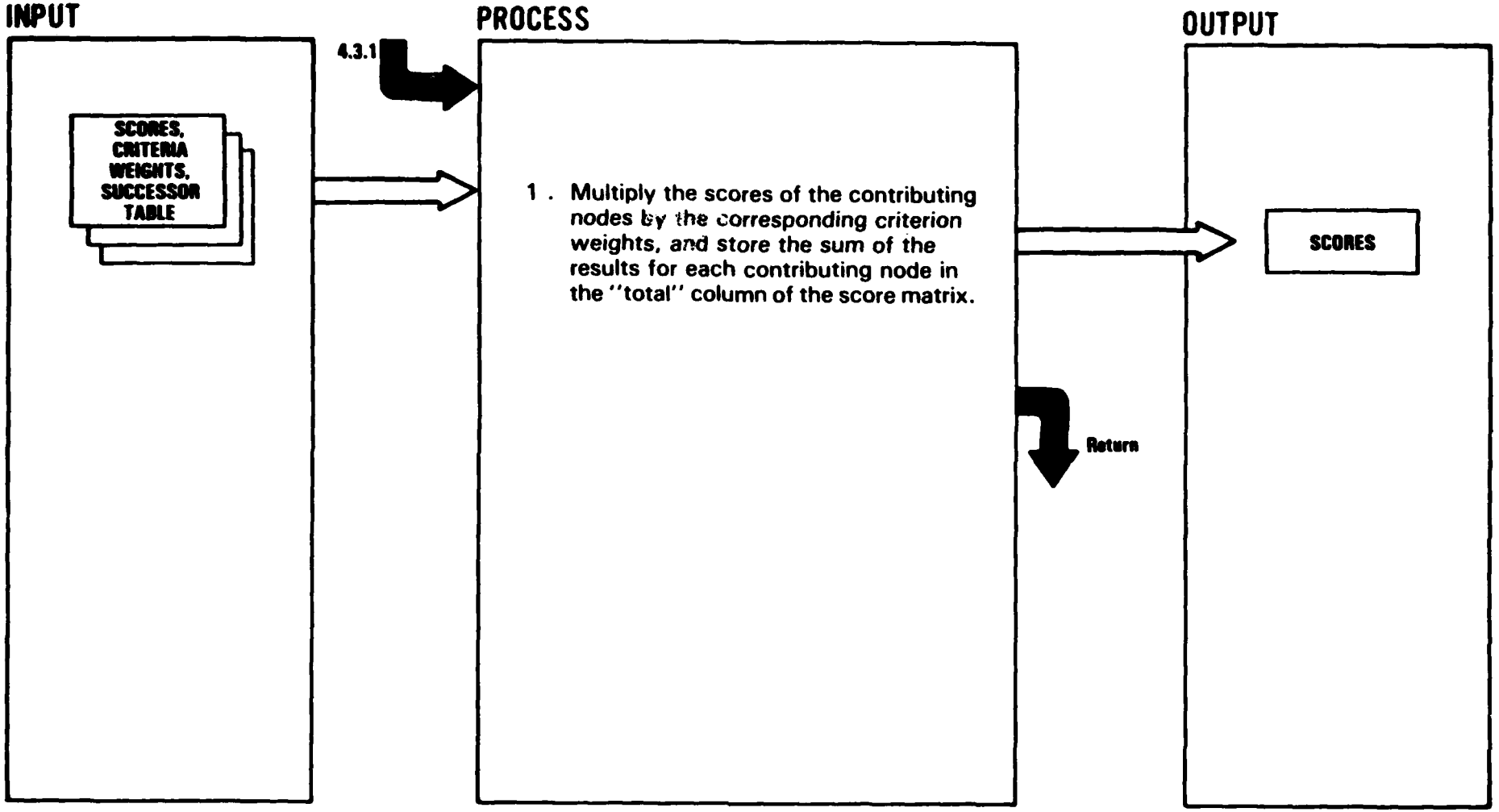
System/Program: RUN Name: D

Diagram ID: 4.3.1 Description: Evaluate Decision Node Page: ___ of ___



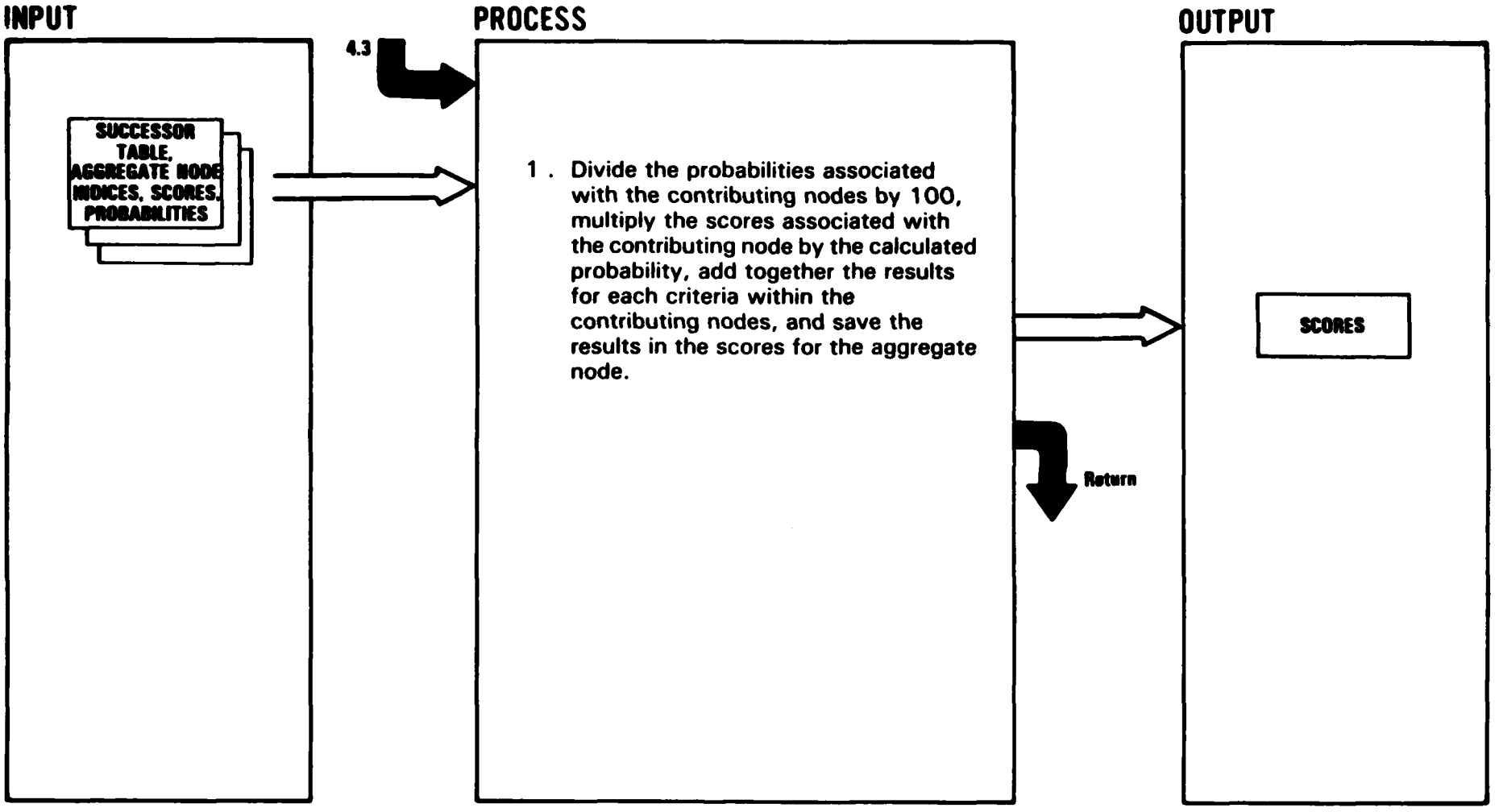
System Program RUN Name: TOT

Diagram ID: 4.3.1.1 Description: Compute Combined Value Page: ___ of ___



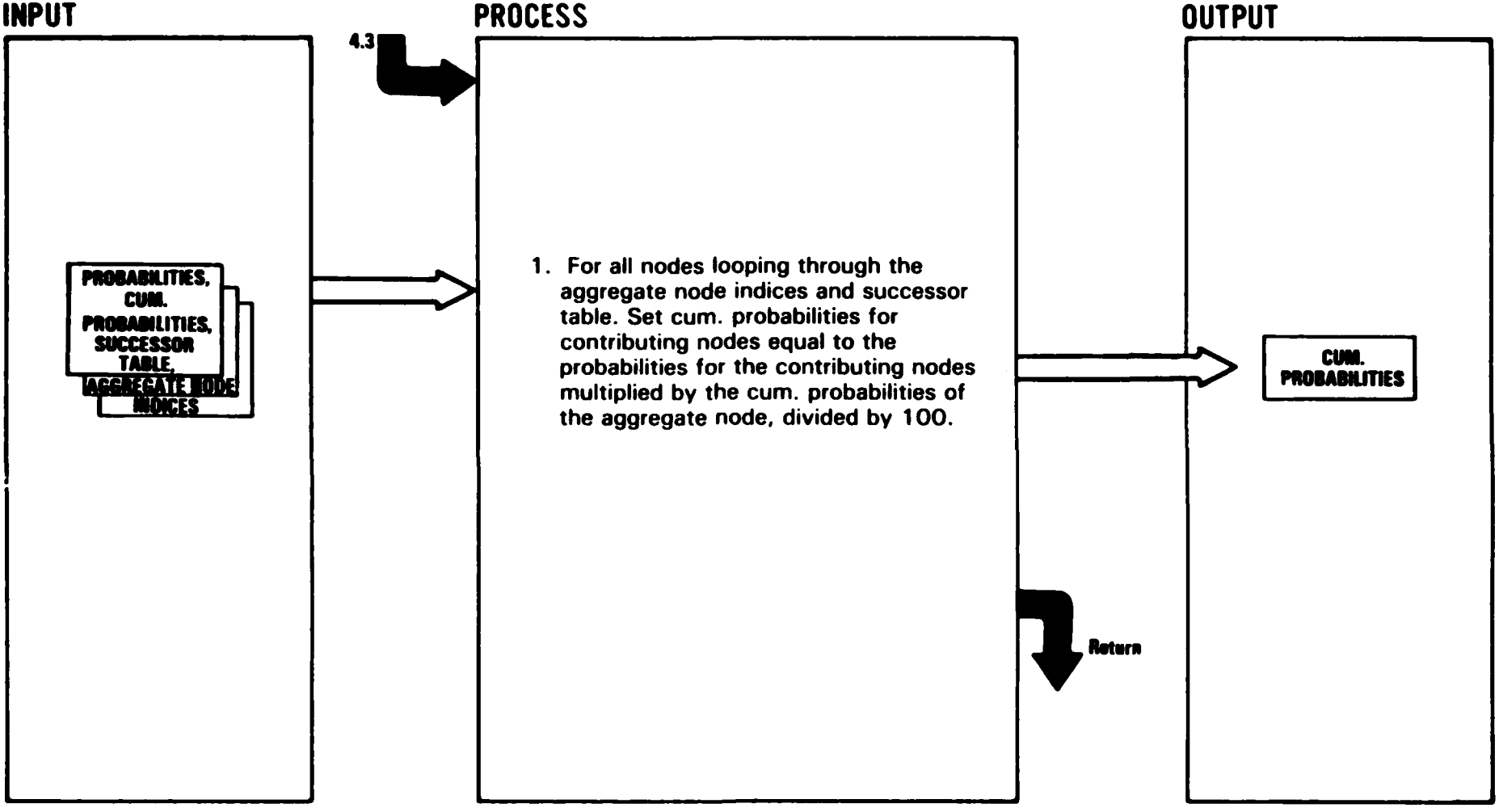
System/Program: RUN Name: W

Diagram ID: 4.3.2 Description Evaluate Probability Node Page: ___ of ___



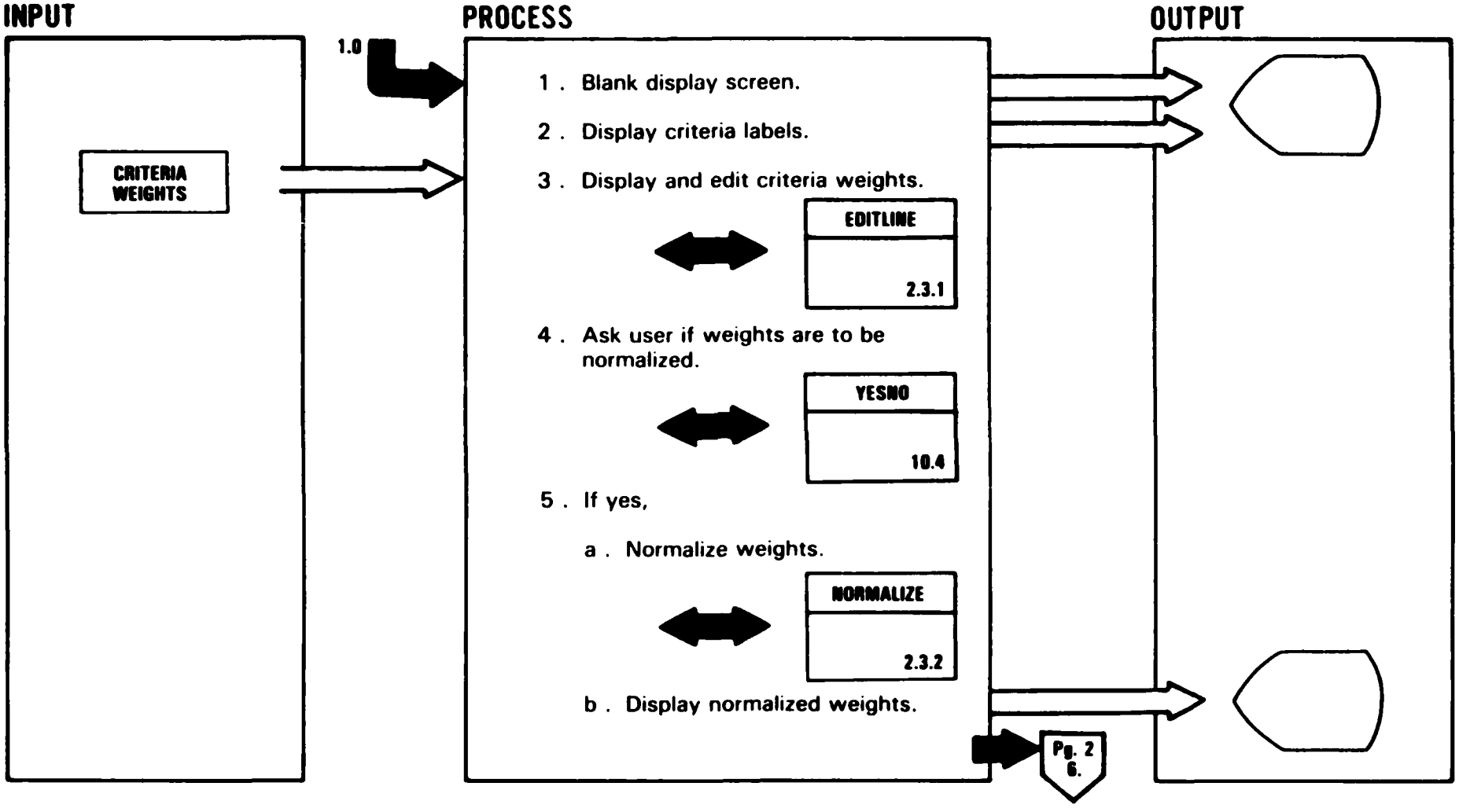
System/Program: RUN Name: CUMWT

Diagram ID: 4.3.3 Description: Calculate Cumulative Probabilities Page: ____ of ____



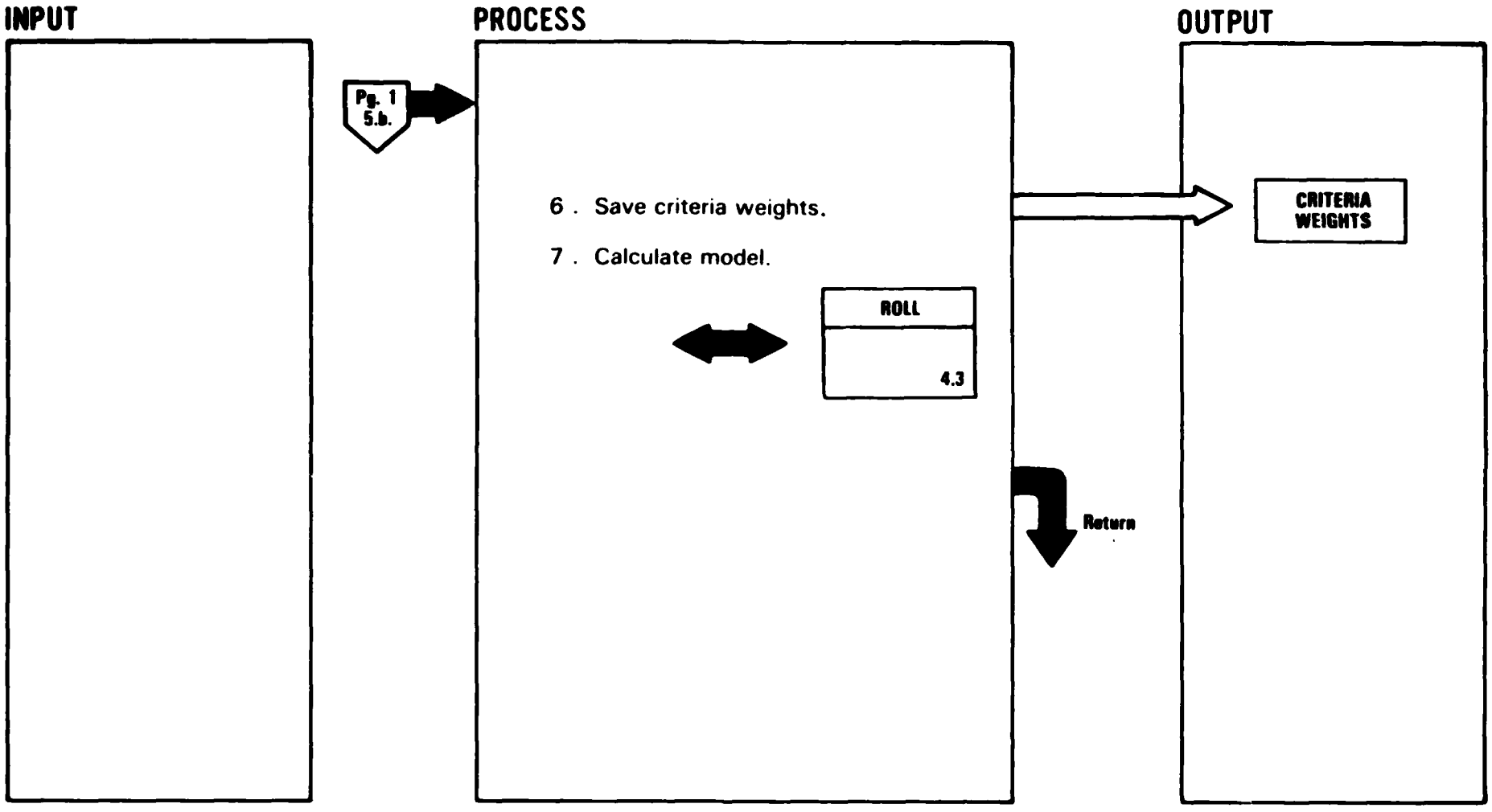
System/Program: RUN Name: EDITW

Diagram ID: 5.0 Description: Edit Criteria Weights Page: 1 of 2



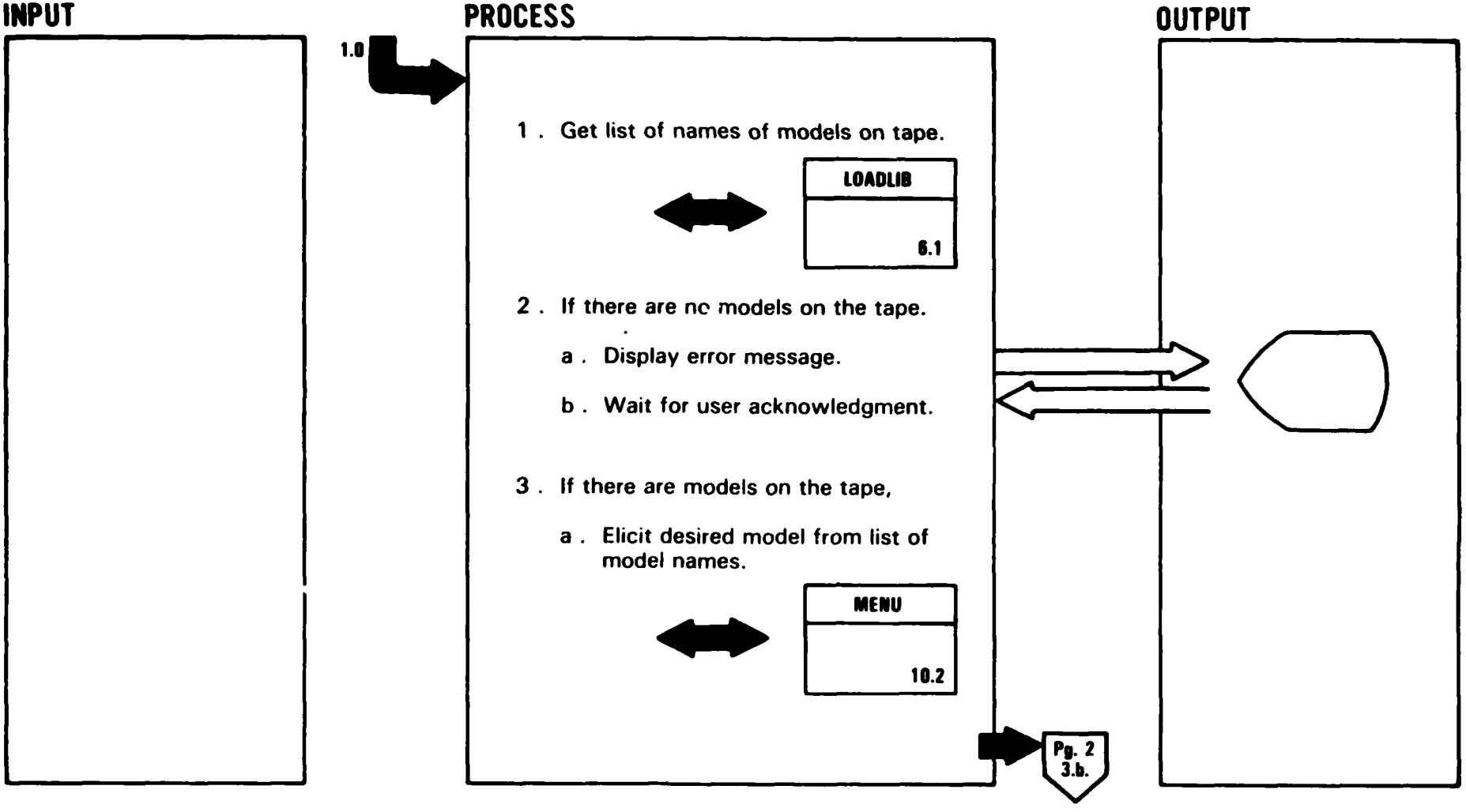
System/Program RUN Name EDITW

Diagram ID 5.0 Description Edit Criteria Weights Page 2 of 2



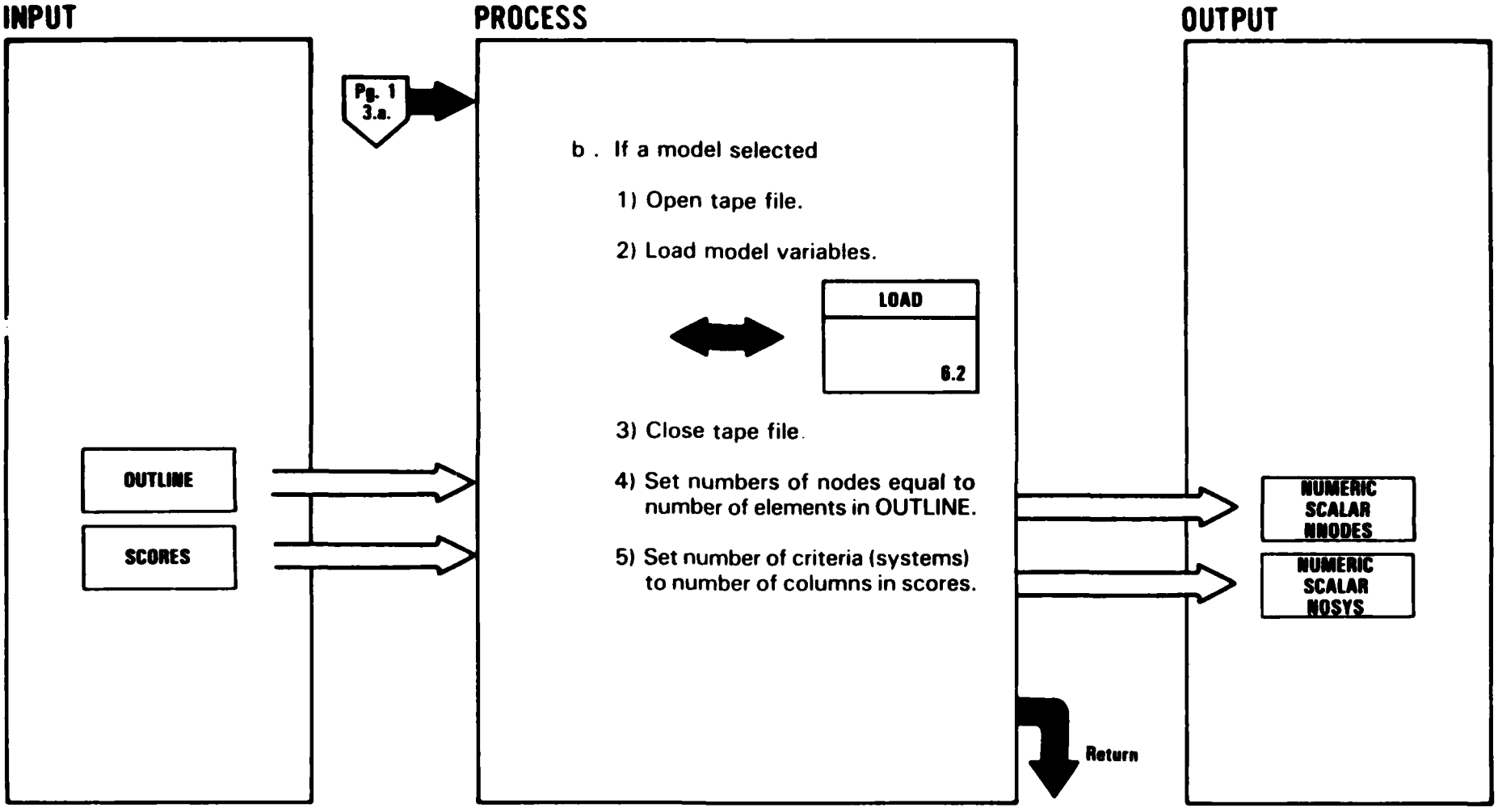
System/Program: RUN Name: LOADDRIVE

Diagram ID: 6.0 Description: Load Model



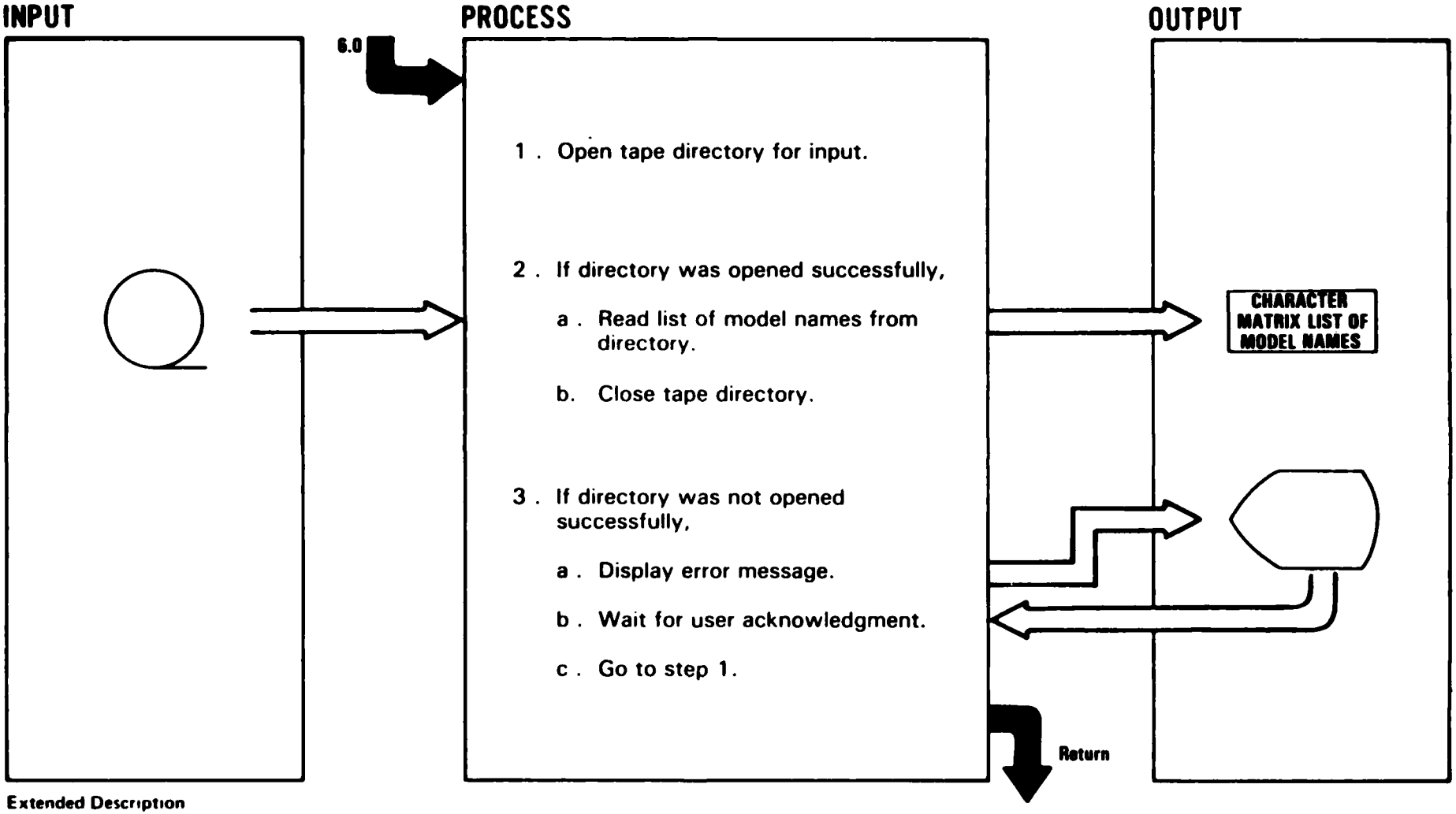
System/Program RUN Name LOADDRIVE

Diagram ID 6.0 Description Load Model Page 2 of 2



System Program RUN Name LOADLIB

Diagram ID 6.1 Description Load Model Names Page ____ of ____

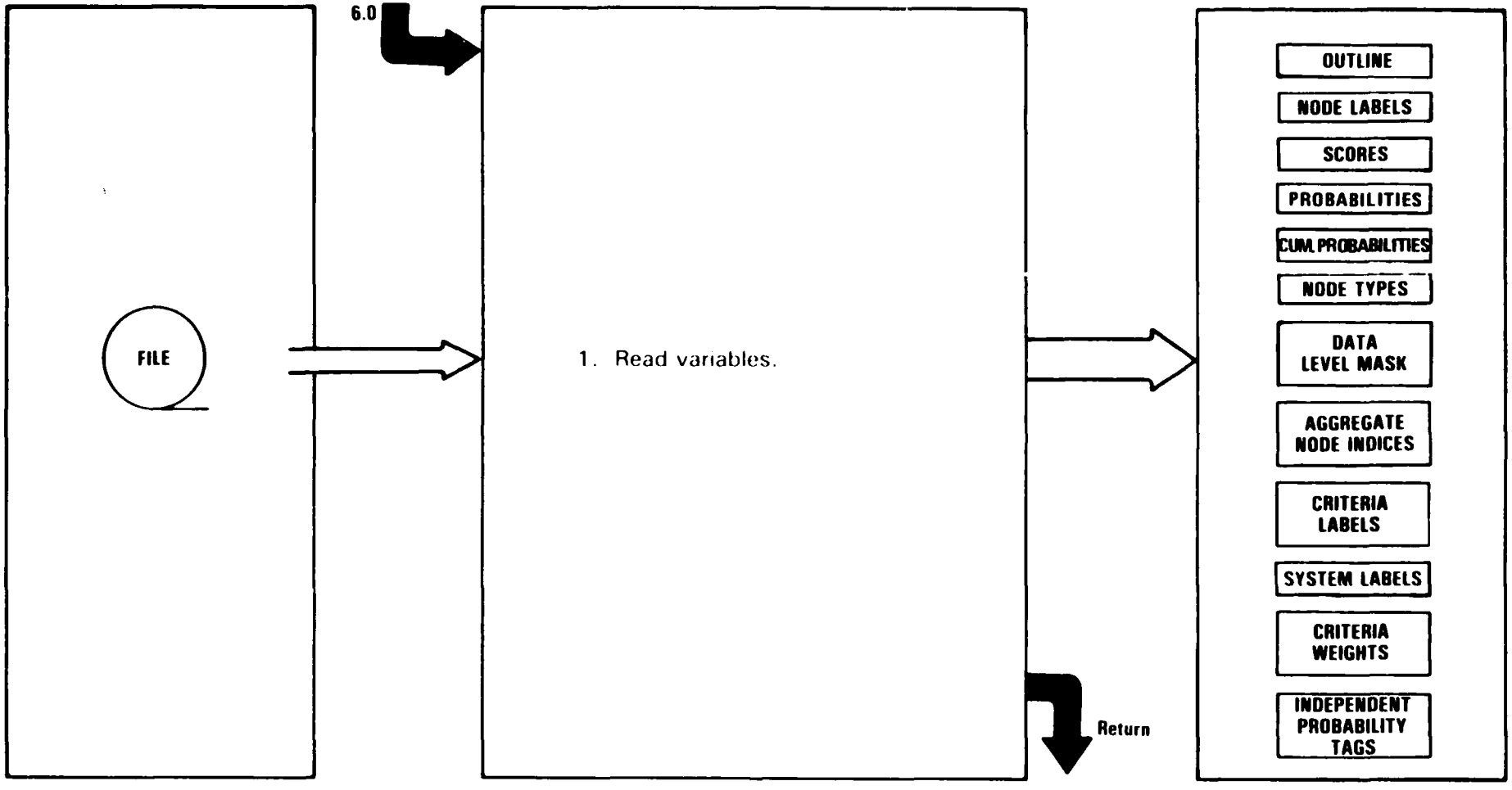


Extended Description

Position of model names within list indicates where models are stored on tape.

System Program RUN Name LOAD

Diagram ID 6.2 Description Load Model Variables Page 1 of 2



Extended Description

1. The OUTLINE Table contains an element for each node in the model, sorted in numerical order. The value is an encoded representation of the node outline number supplied for each node when the model is entered. (See STRUCTURE.)

2. The NODE LABELS contain an element for each node (in the same order as OUTLINE) consisting of the description of each node supplied when the model is entered.

3. SCORES is a numeric matrix containing the values assigned to each criteria plus an extra element for the total score for each node of the model. (The node dimension is in the same order as OUTLINE.)

4. The PROBABILITIES contain the relative importance assigned to each node in the model. The elements are in the same order as OUTLINE.

5. The CUM. PROBABILITIES contain the percentage of the importance of the entire model at each node level.

6. The NODE TYPES contain an indication of the type of calculation to be used in assessing SCORES and WEIGHTS.

7. The DATA LEVEL MASK indicates which nodes are at the data level (bottom level) vs. the nodes that are aggregate nodes.

System/Program RUN Name LOAD

Diagram ID 6.2 Description Load Model Variables Page 2 of 2

INPUT

PROCESS

OUTPUT

Extended Description

8. The AGGREGATE NODE INDICES contain the indices into the model variables that relate to just the aggregate nodes (all nodes that have one or more nodes contributing to them).

9. The SUCCESSOR TABLE is a matrix containing the indices of the nodes that contribute to the aggregate nodes. There is a row for each aggregate node.

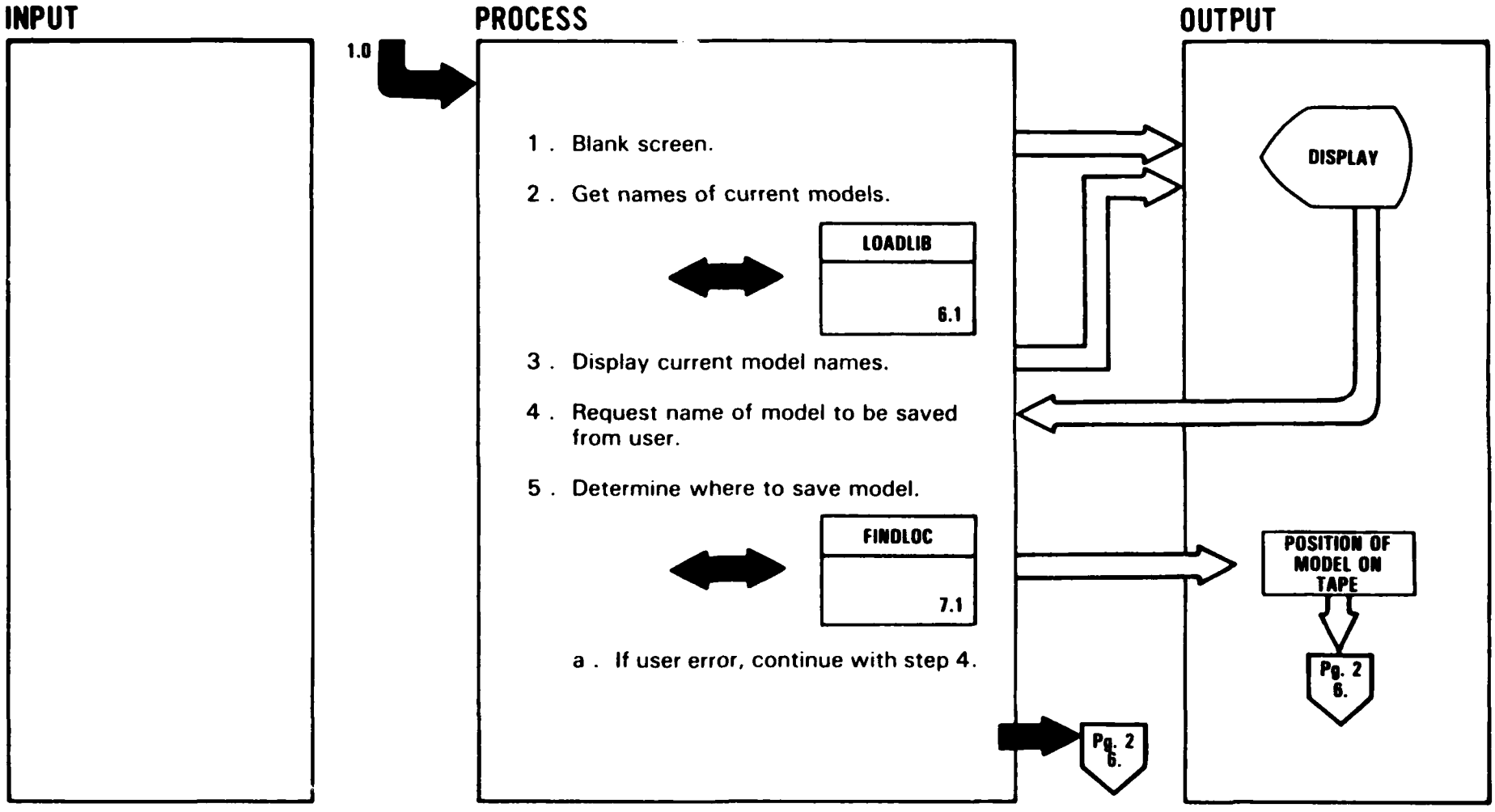
10. The CRITERIA LABELS contain the character discriptions of the criteria being evaluated.

11. The CRITERIA WEIGHTS contain the weights to be applied to each criterion when the decision tree is solved. The number of elements is equal to the number of criteria plus one for the total.

12. The INDEPENDENT PROBABILITY TAGS indicate groups of events that occur more than once in the tree and whose probabilities can be assessed all at once. The number and order of elements is the same as that for OUTLINE.

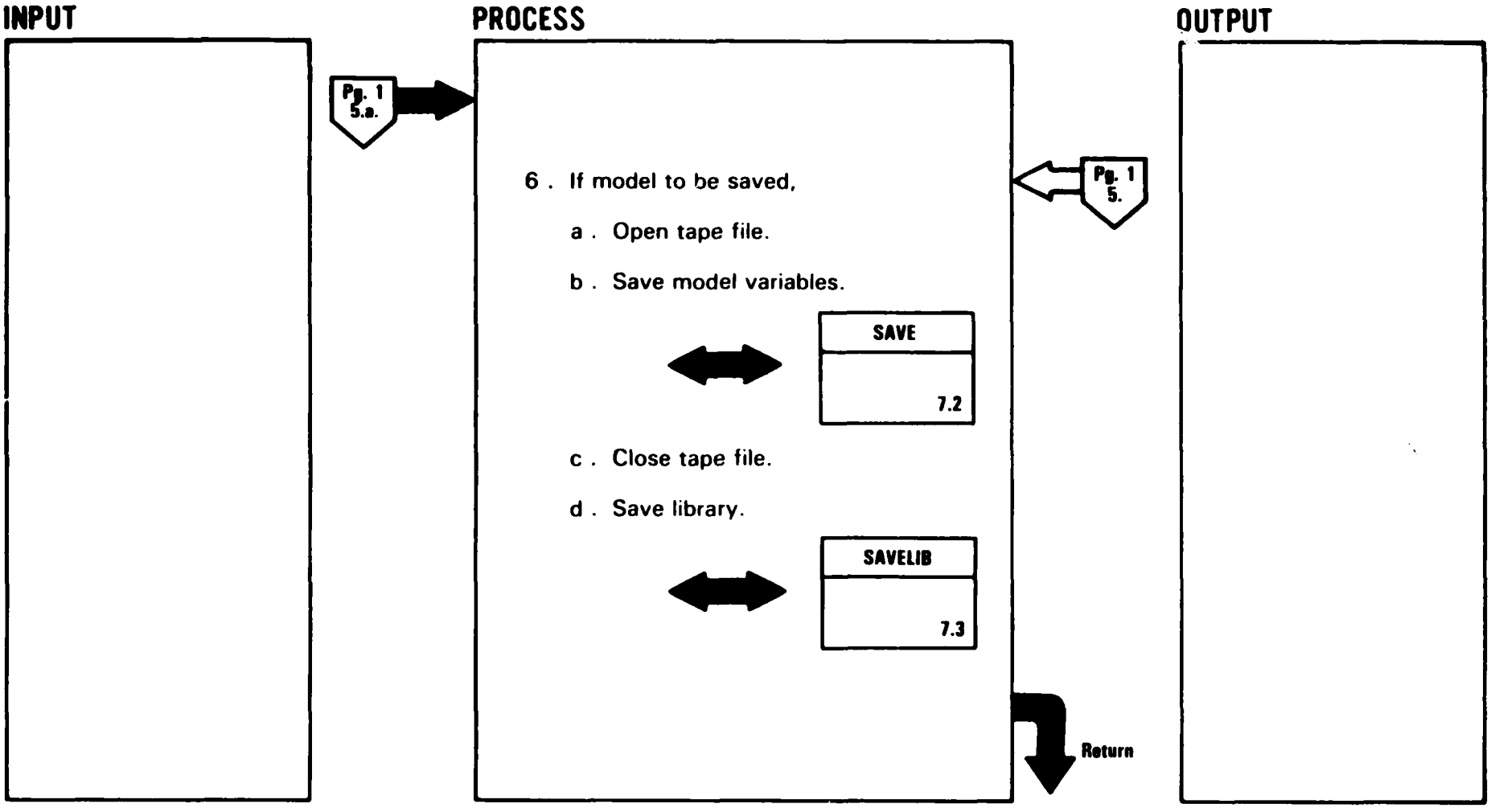
System/Program: RUN Name: SAVEDRIVE

Diagram ID: 7.0 Description: Save Model



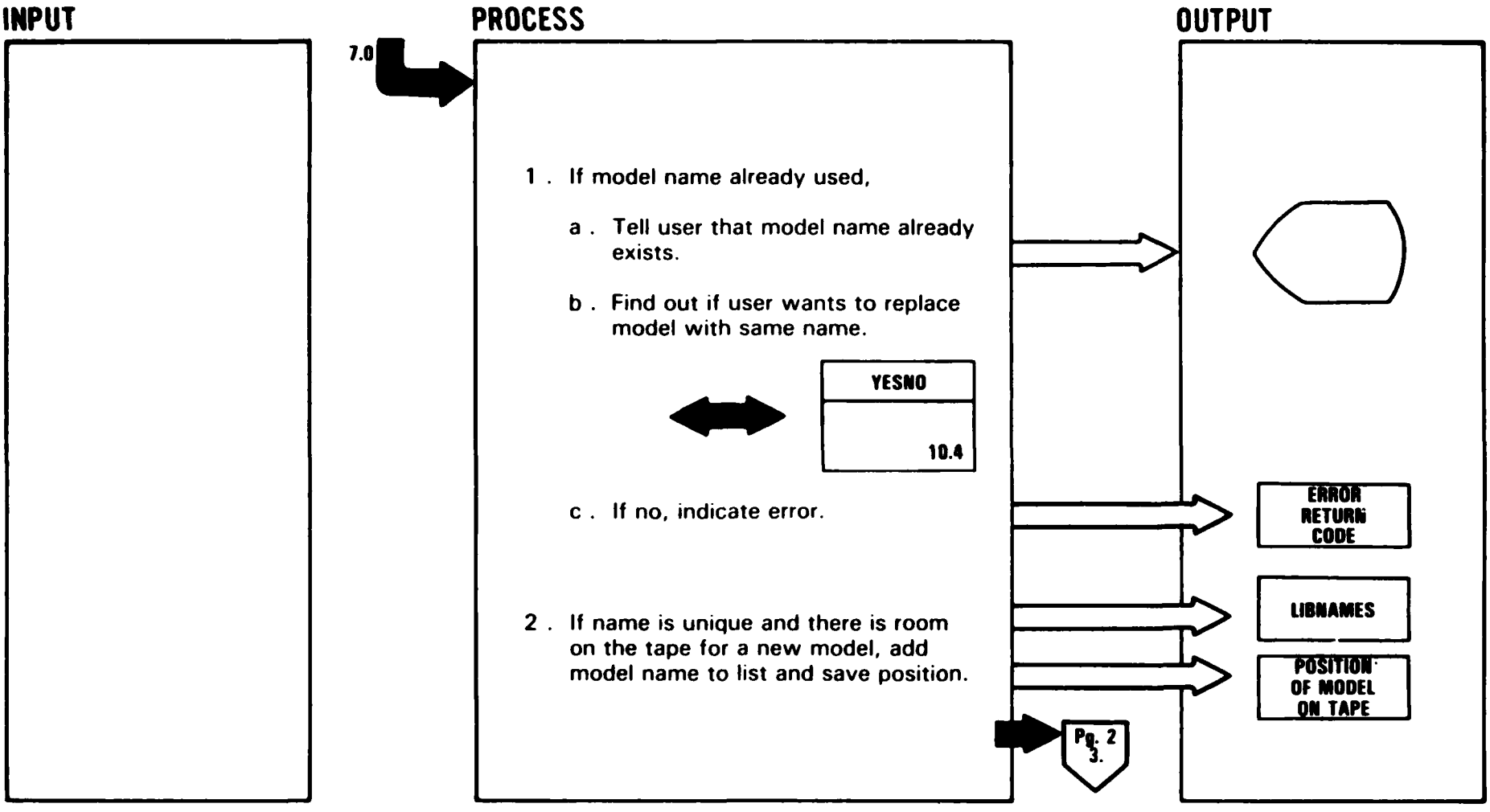
System/Program: RUN Name: SAVEDRIVE

Diagram ID: 7.0 Description: Save Model Page: 2 of 2



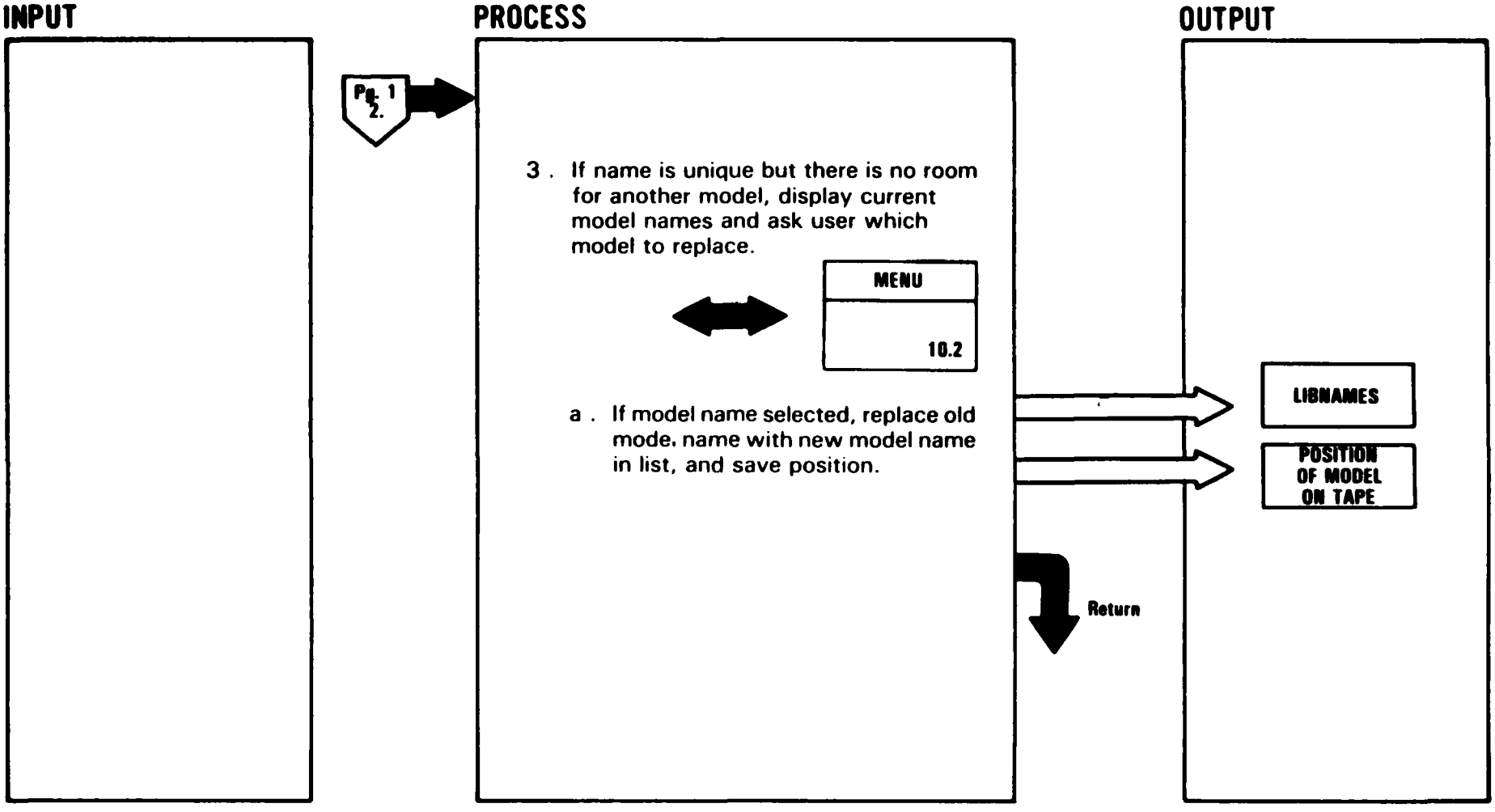
System/Program: RUN Name: FINDLOC

Diagram ID: 7.1 Description: Determine Where to Save Model Page: 1 of 2



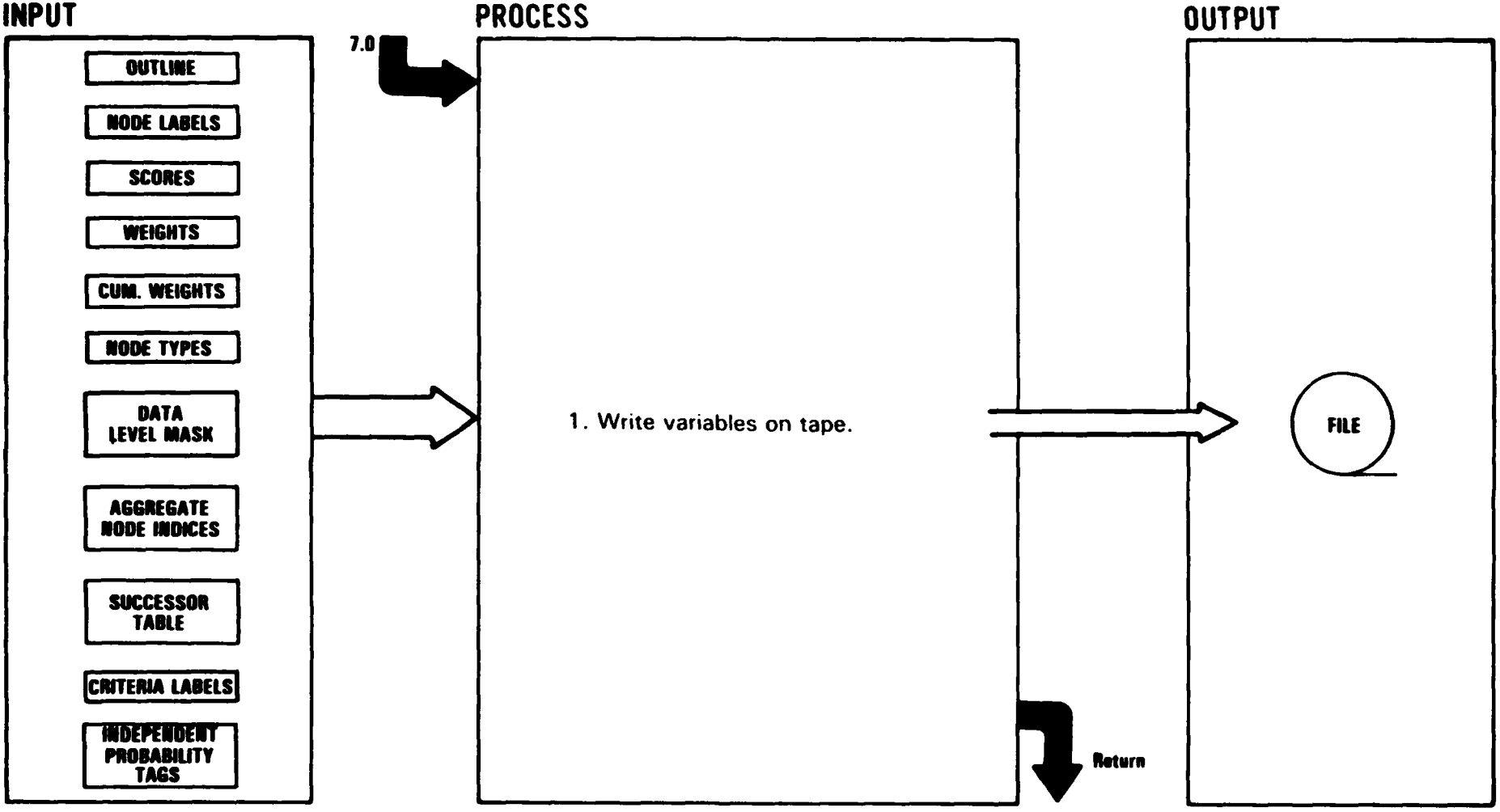
System/Program: RUN Name: FINDLOC

Diagram ID: 7.1 Description: Determine Where to Save Model Page: 2 of 2



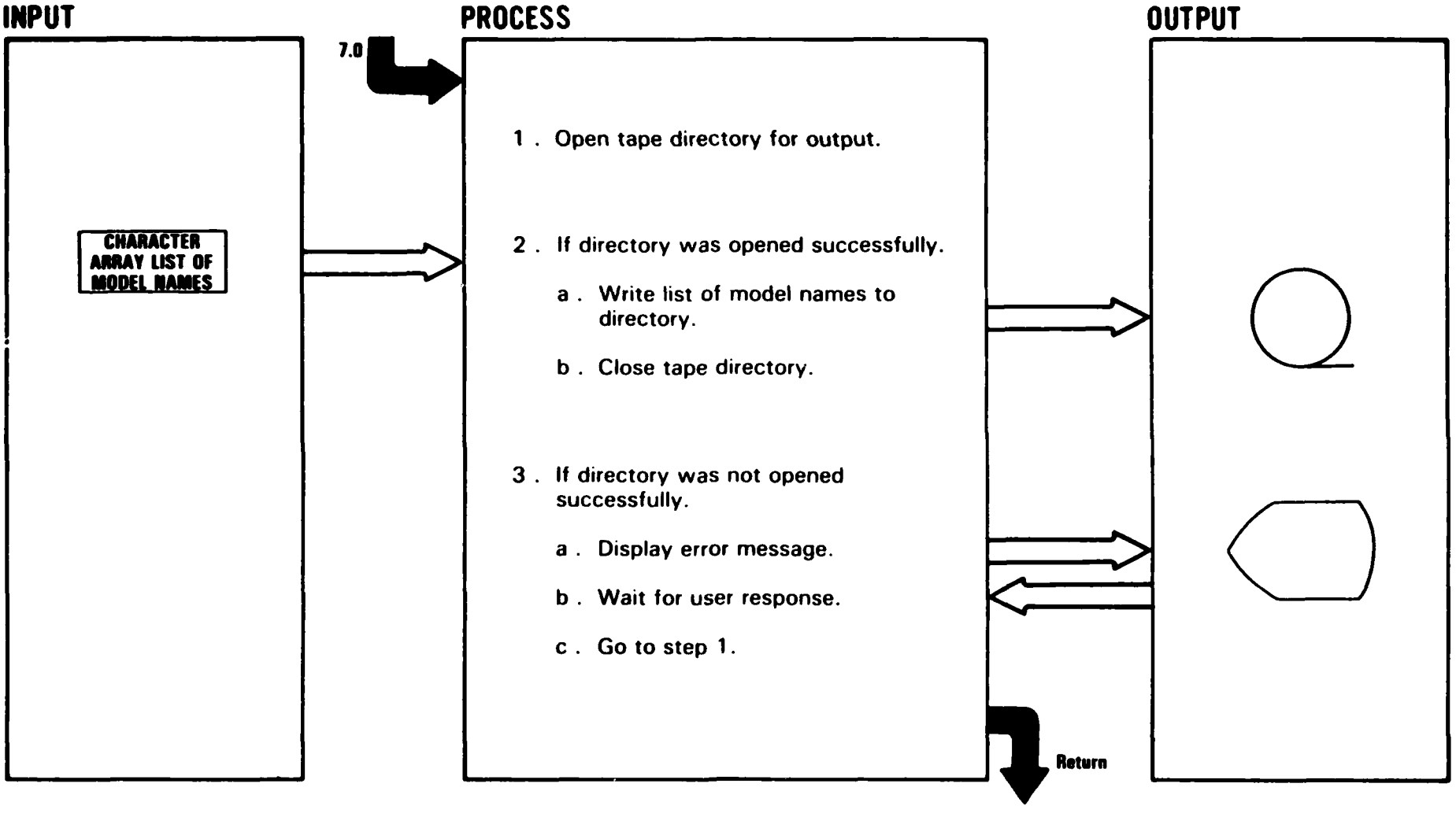
System/Program: RUN Name: SAVE

Diagram ID: 7.2 Description: Save Model Variables Page: ___ of ___



System/Program: RUN Name: SAVLIB

Diagram ID: 7.3 Description: Save Model Names Page: ____ of ____



System/Program: RUN Name: NEWDATA

Diagram ID: 8.0 Description: Enter New Values Page: 1 of 2

INPUT

PROCESS

OUTPUT

1.0

1 . Give the user the option to enter scores or enter probabilities.

MENU

10.2

2 . If enter scores selected.

LOADSCORE

8.1

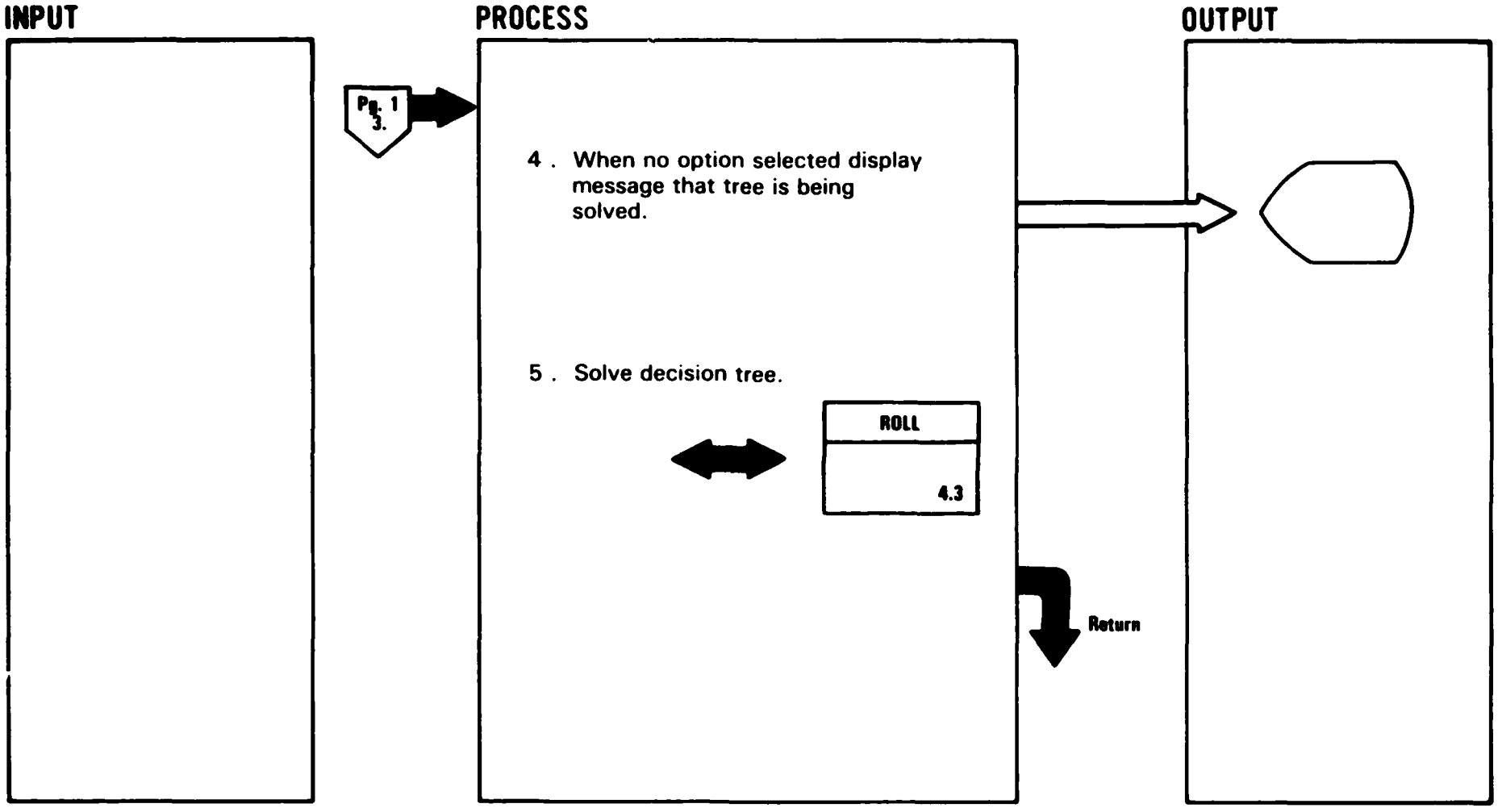
3 . If enter probabilities selected.

NEWWEIGHT

8.2

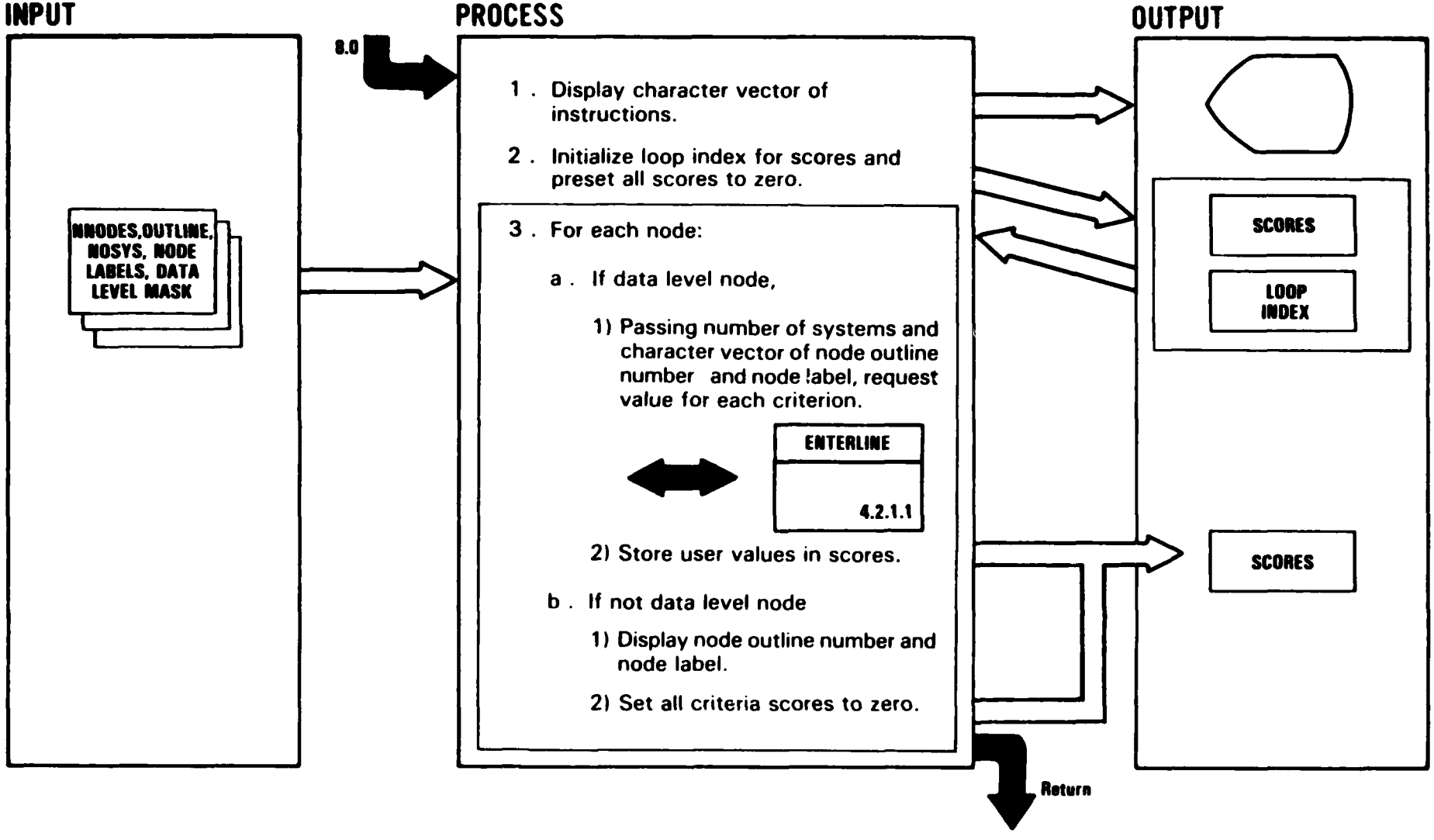
System/Program: RUN Name: NEWDATA

Diagram ID: 8.0 Description: Enter New Values Page: 2 of 2



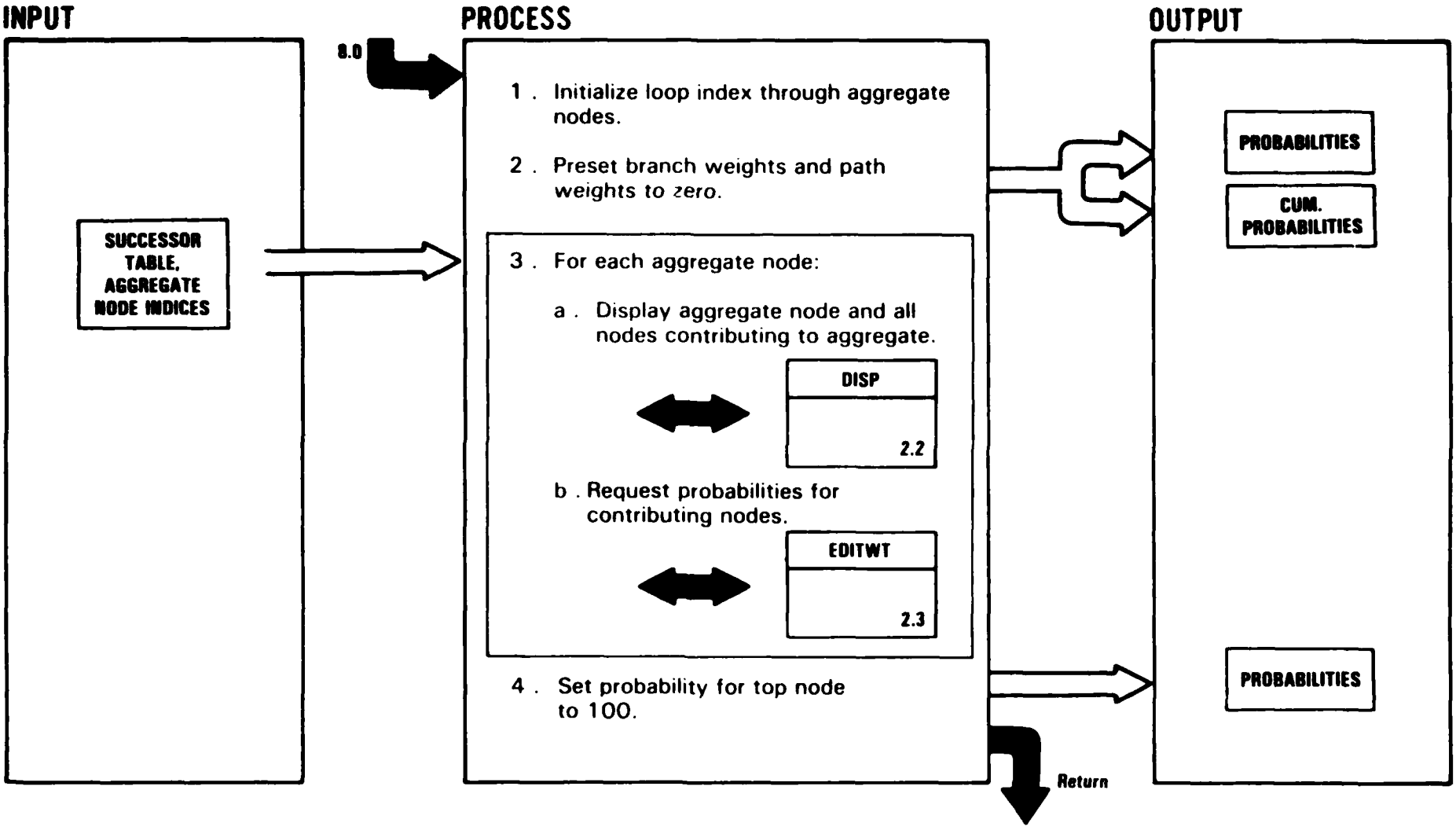
System/Program: RUN Name: LOADSCORE

Diagram ID: 8.1 Description: Enter All Scores Page: ___ of ___



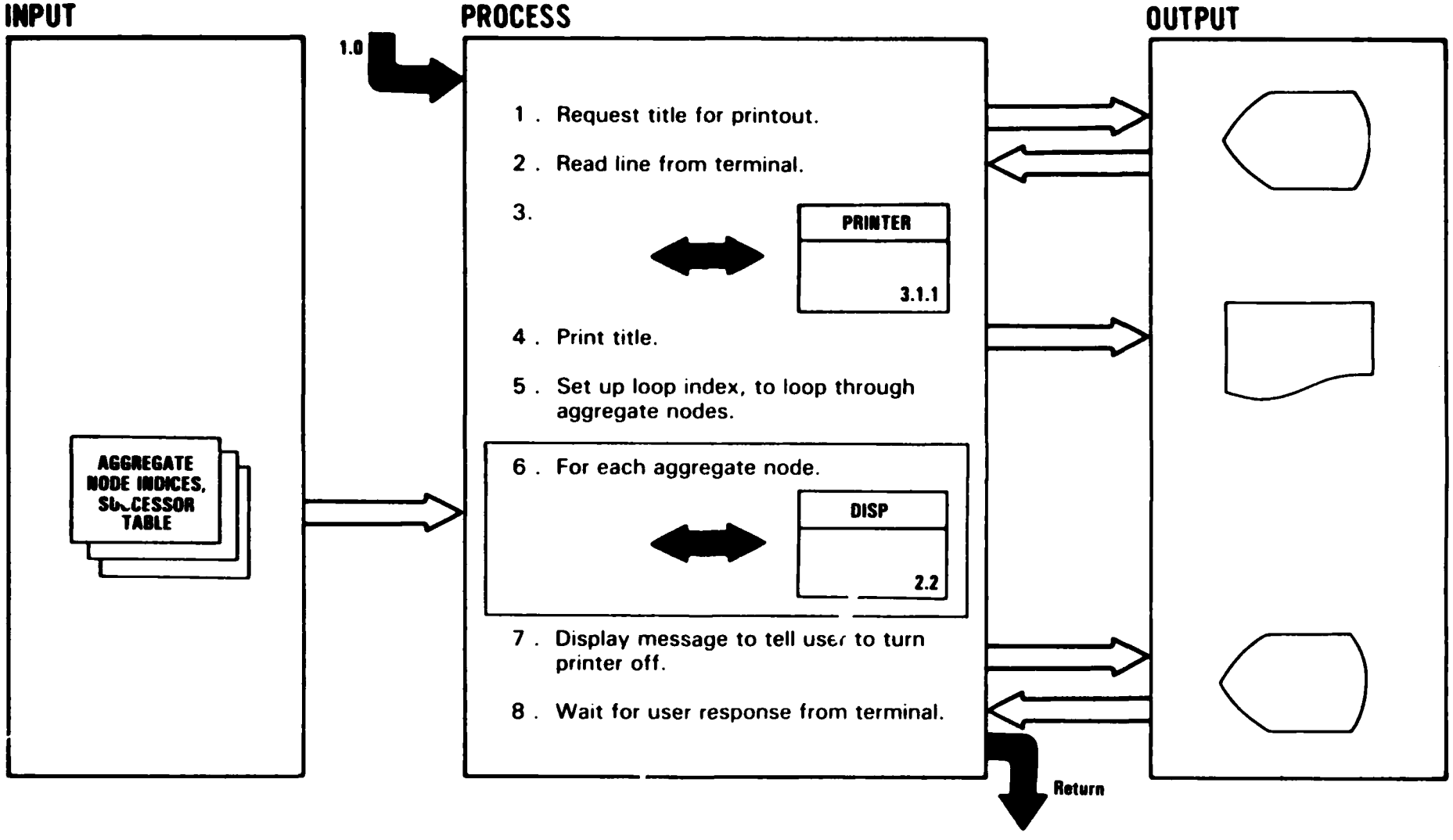
System/Program: RUN Name: NEWWEIGHT

Diagram ID: 8.2 Description: Enter All Probabilities Page: ___ of ___



System/Program: RUN Name: SHOWALL

Diagram ID: 9.0 Description: Print Results Page: ___ of ___



System Program RUN Name

Diagram ID 10.0 Description General Routines Page ____ of ____

INPUT

PROCESS

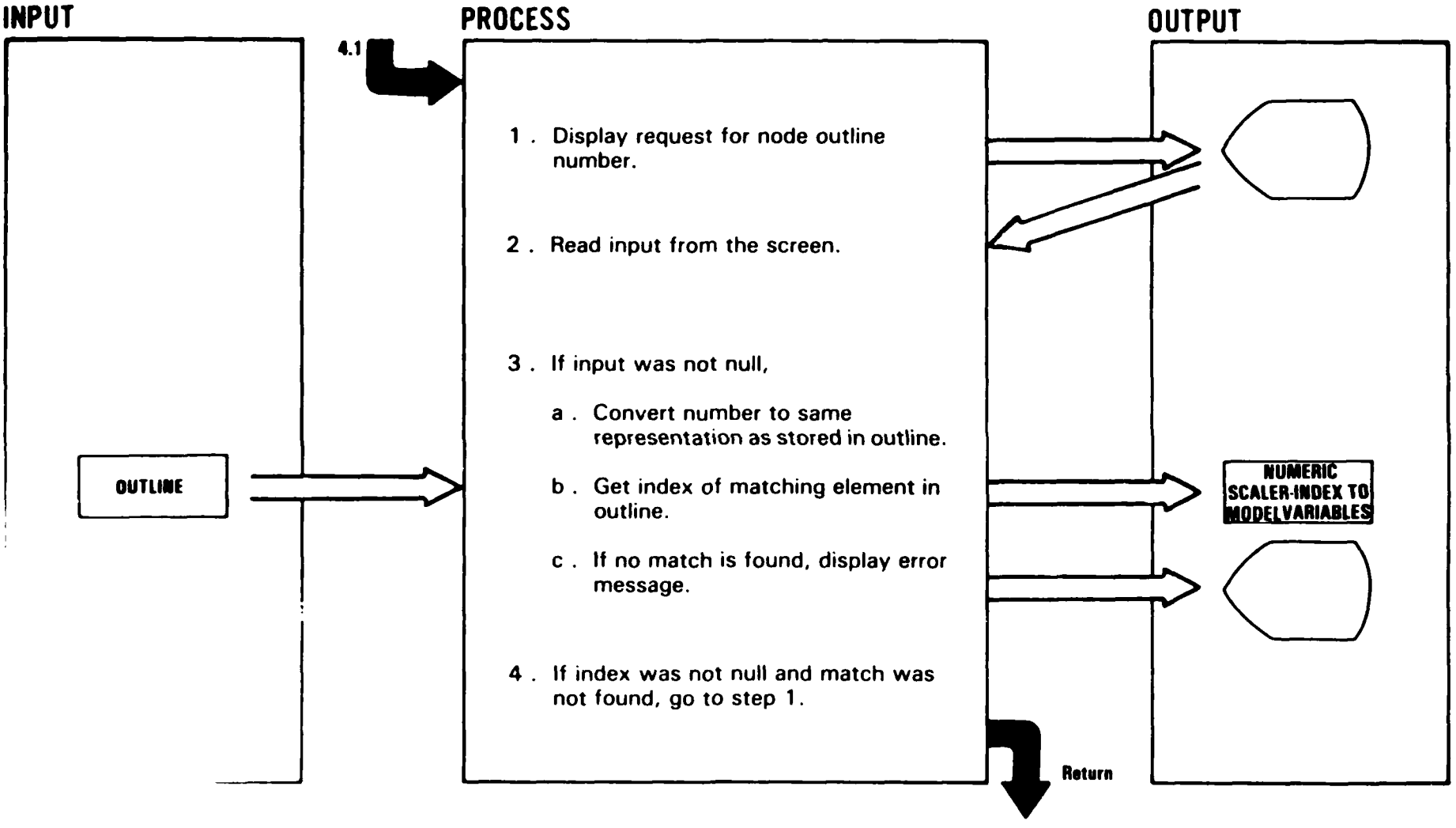
OUTPUT

Extended Description

Generalized routines are directly invoked by functional procedures and return to the calling programs.

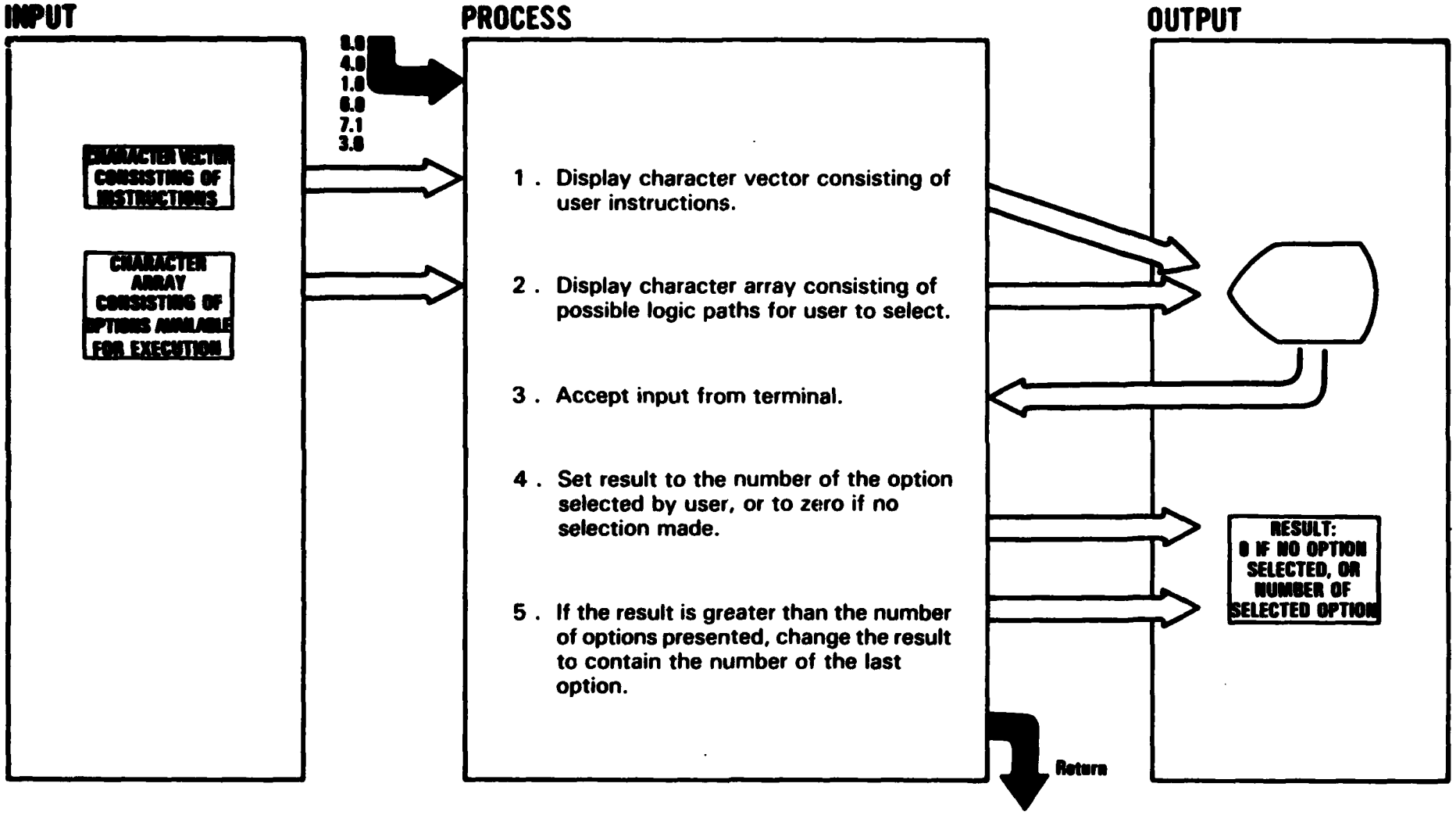
System/Program: RUN Name: LOCATE

Diagram ID: 10.1 Description: Elicit Node Number Page: ____ of ____



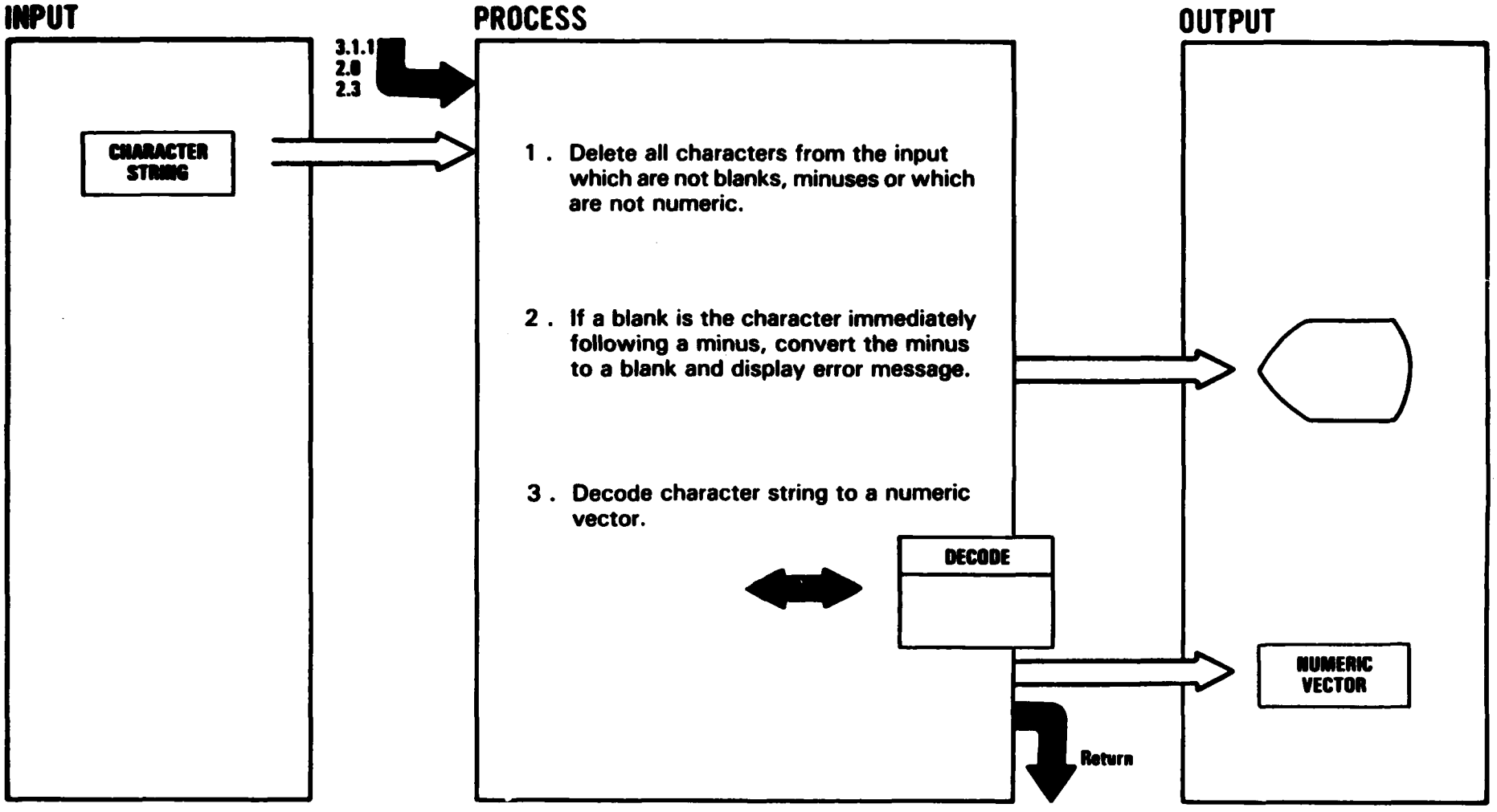
System/Program: RUN Name: MENU

Diagram ID: 10.2 Description: Present User Options Page: ____ of ____



System/Program: RUN Name: NUMBERSONLY

Diagram ID: 10.3 Description: Convert Alpha Numbers to Numeric Page: ___ of ___



System/Program: RUN Name: YESNO

Diagram ID: 10.4 Description: Get Yes or No Answer Page: ____ of ____