Requirements Analysis and Specification

Organization of this Lecture

- Introduction
- Requirements analysis
- Requirements specification
- SRS document
- Decision table
- Decision tree
- Summary

Requirements Analysis and Specification

- Goals of requirements analysis and specification phase:
  - fully understand the user requirements
  - remove inconsistencies, anomalies, etc. from requirements
  - document requirements properly in an SRS document
- Consists of two distinct activities:
  - Requirements Gathering and Analysis
  - Specification

Requirements Analysis

- Requirements analysis consists of two main activities:
  - Requirements gathering
  - Analysis of the gathered requirements

- Analyst gathers requirements through:
  - observation of existing systems,
  - studying existing procedures,
  - discussion with the customer and end-users,
  - analysis of what needs to be done, etc.

Inconsistent requirement

- Some part of the requirement:
  - contradicts with some other part.
- Example:
  - One customer says turn off heater and open water shower when temperature > 100 C
  - Another customer says turn off heater and turn ON cooler when temperature > 100 C

Incomplete requirement

- Some requirements have been omitted:
  - due to oversight.
- Example:
  - The analyst has not recorded: when temperature falls below 90 C
    - heater should be turned ON
    - water shower turned OFF.

Analysis of the Gathered Requirements (CONT.)

- **Requirements analysis involves:**
  - obtaining a clear, in-depth understanding of the product to be developed,
  - remove all ambiguities and inconsistencies from the initial customer perception of the problem.

Software Requirements Specification

- **Main aim of requirements specification:**
  - systematically organize the requirements arrived during requirements analysis
  - document requirements properly.

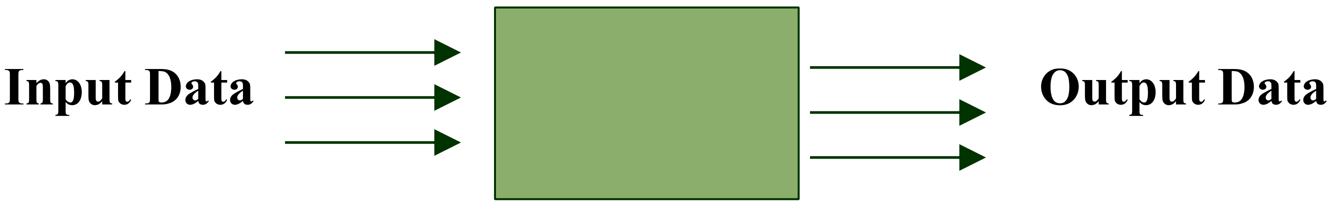
- The SRS document is useful in various contexts:
  - statement of user needs
  - contract document
  - reference document
  - definition for implementation

Software Requirements Specification: A Contract Document

- Requirements document is a reference document.
- SRS document is a contract between the development team and the customer.
  - Once the SRS document is approved by the customer,
    - any subsequent controversies are settled by referring the SRS document.
- Once customer agrees to the SRS document:
  - development team starts to develop the product according to the requirements recorded in the SRS document.
- The final product will be acceptable to the customer:
  - as long as it satisfies all the requirements recorded in the SRS document.

SRS Document (CONT.)

- The SRS document is known as black-box specification:
  - the system is considered as a black box whose internal details are not known.
  - only its visible external (i.e. input/output) behavior is documented.



- SRS document concentrates on:
  - what needs to be done
  - carefully avoids the solution ("how to do") aspects.
- The SRS document serves as a contract
  - between development team and the customer.
  - Should be carefully written

Properties of a good SRS document

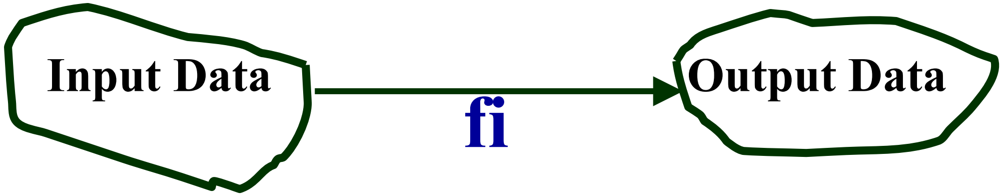
- It should be concise
  - and at the same time should not be ambiguous.
- It should specify what the system must do
  - and not say how to do it.
- Easy to change.,
  - i.e. it should be well-structured.
- It should be consistent.
- It should be complete.
- It should be traceable
  - you should be able to trace which part of the specification corresponds to which part of the design and code, etc and vice versa.
- It should be verifiable
  - e.g. “system should be user friendly” is not verifiable

SRS Document (CONT.)

- SRS document, normally contains three important parts:
  - functional requirements,
  - nonfunctional requirements,
  - constraints on the system.

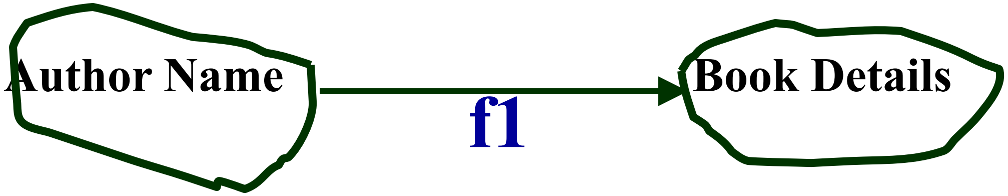
SRS Document (CONT.)

- It is desirable to consider every system:
  - performing a set of functions {fi}.
  - Each function fi considered as:
  - transforming a set of input data to corresponding output data.



Example: Functional Requirement

- F1: Search Book
  - Input:
    - an author's name:
  - Output:
    - details of the author's books and the locations of these books in the library.



Functional Requirements

- Functional requirements describe:
  - A set of high-level requirements
  - Each high-level requirement:
    - takes in some data from the user
    - outputs some data to the user
  - Each high-level requirement:
    - might consist of a set of identifiable functions

- For each high-level requirement:
  - every function is described in terms of
    - input data set
    - output data set
    - processing required to obtain the output data set from the input data set

Nonfunctional Requirements

- ⌘Characteristics of the system which can not be expressed as functions:
  - ☒maintainability,
  - ☒portability,
  - ☒usability, etc.

Nonfunctional Requirements

- Nonfunctional requirements include:
  - reliability issues,
  - performance issues,
  - human-computer interface issues,
  - Interface with other external systems,
  - security, maintainability, etc.

Constraints

- Constraints describe things that the system should or should not do.
  - For example,
    - standards compliance
    - how fast the system can produce results
      - so that it does not overload another system to which it supplies data, etc.

Examples of constraints

- Hardware to be used,
- Operating system
  - or DBMS to be used
- Capabilities of I/O devices
- Standards compliance
- Data representations
  - by the interfaced system

Organization of the SRS Document

- Introduction.
- Functional Requirements
- Nonfunctional Requirements
  - External interface requirements
  - Performance requirements
- Constraints

Example Functional Requirements

- List all functional requirements
  - with proper numbering.

- Req. 1:
  - Once the user selects the “search” option,
    - he is asked to enter the key words.
  - The system should output details of all books
    - whose title or author name matches any of the key words entered.
    - Details include: Title, Author Name, Publisher name, Year of Publication, ISBN Number, Catalog Number, Location in the Library.

Example Functional Requirements

- Req. 2:
  - When the “renew” option is selected,
    - the user is asked to enter his membership number and password.
  - After password validation,
    - the list of the books borrowed by him are displayed.
  - The user can renew any of the books:
    - by clicking in the corresponding renew box.

Req. 1:

- R.1.1:
  - Input: “search” option,
  - Output: user prompted to enter the key words.
- R1.2:
  - Input: key words
  - Output: Details of all books whose title or author name matches any of the key words.
    - Details include: Title, Author Name, Publisher name, Year of Publication, ISBN Number, Catalog Number, Location in the Library.
  - Processing: Search the book list for the keywords

Req. 2:

- R2.1:
  - Input: “renew” option selected,
  - Output: user prompted to enter his membership number and password.
- R2.2:
  - Input: membership number and password
  - Output:
    - list of the books borrowed by user are displayed. User prompted to enter books to be renewed or
    - user informed about bad password
  - Processing: Password validation, search books issued to the user from borrower list and display.

Req. 2:

- R2.3:
  - Input: user choice for renewal of the books issued to him through mouse clicks in the corresponding renew box.
  - Output: Confirmation of the books renewed
  - Processing: Renew the books selected by the in the borrower list.

Examples of Bad SRS Documents

- **Unstructured Specifications:**
  - Narrative essay --- one of the worst types of specification document:
    - Difficult to change,
    - difficult to be precise,
    - difficult to be unambiguous,
    - scope for contradictions, etc.
- Noise:
  - Presence of text containing information irrelevant to the problem.
- Silence:
  - aspects important to proper solution of the problem are omitted.

Examples of Bad SRS Documents

- Overspecification:
  - Addressing "how to" aspects
  - For example, "Library member names should be stored in a sorted descending order"
  - Overspecification restricts the solution space for the designer.
- Contradictions:
  - Contradictions might arise
    - if the same thing described at several places in different ways.
- Ambiguity:
  - Literary expressions
  - Unquantifiable aspects, e.g. "good user interface"
- Forward References:
  - References to aspects of problem
    - defined only later on in the text.
- Wishful Thinking:
  - Descriptions of aspects
    - for which realistic solutions will be hard to find.

Using Diagrams

Graphical representation of the analysis can present the information better using:

- **Decision Tables**
- **Decision Trees**
- Entity-Relationship Diagrams
- Data Flow Diagrams
- State Transition Diagrams
  - event table, action table

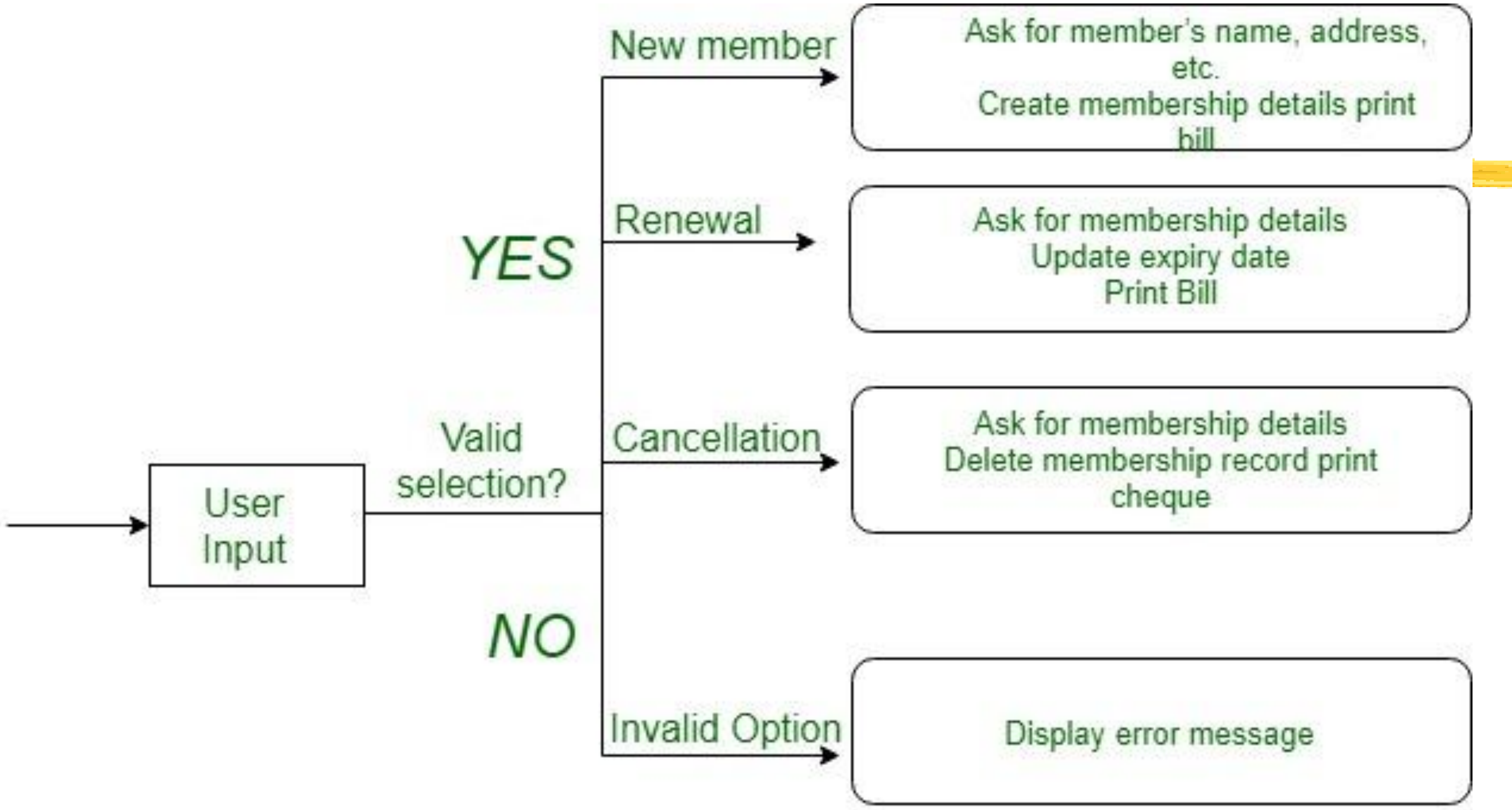
Decision Tree

- A **Decision Tree** offers a graphic read of the logic involved in decision making and therefore the corresponding actions are taken.

- The edges of a decision tree represent conditions **conditions** and therefore **the leaf nodes represent the actions** to be performed looking on the result of testing the condition.
- For example, consider Library Membership Automation Software (LMS) where it ought to support the following three options:
  - New member, Renewal, and Cancel membership.

- **New Member Option:**
  - **Decision:**
    Once the 'new member' possibility is chosen, the software system asks details concerning the member just like the member's name, address, number, etc.
  - **Action:**
    If correct info is entered then a membership record for the member is made and a bill is written for the annual membership charge and the protection deposit collectible.

- **Renewal Option:**
  - **Decision:**
    If the ‘renewal’ possibility is chosen, the LMS asks for the member’s name and his membership range to test whether or not he’s a sound member or not.
  - **Action:**
    If the membership is valid then membership ending date is updated and therefore the annual membership bill is written, otherwise, a slip-up message is displayed.
- **Cancel Membership Option:**
  - **Decision:**
    If the ‘cancel membership’ possibility is chosen, then the software system asks for member’s name and his membership range.
  - **Action:**
    The membership is off, a cheque for the balance quantity because of the member is written and at last the membership record is deleted from the information.


User Input
Valid selection?
YES
NO
New member
Renewal
Cancellation
Invalid Option
Ask for member's name, address, etc.
Create membership details print bill
Ask for membership details
Update expiry date
Print Bill
Ask for membership details
Delete membership record print cheque
Display error message

Decision tree for LMS

Decision Table

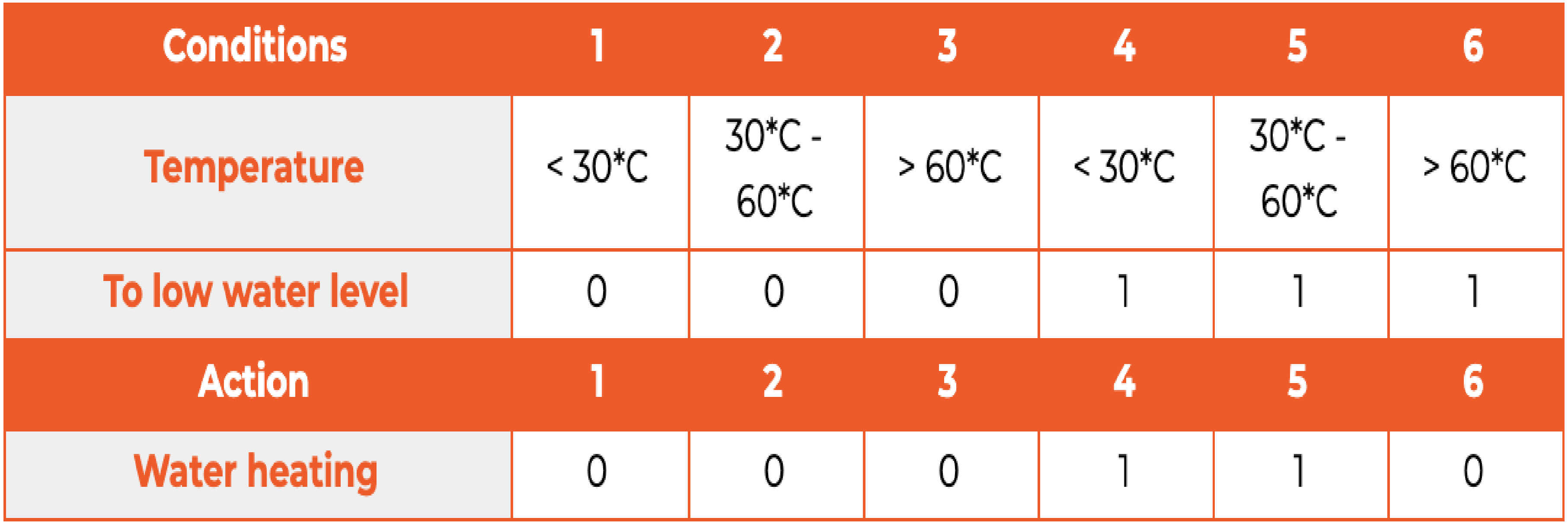
- A decision table shows in a tabular form:
  - processing logic and corresponding actions
- Upper rows of the table specify:
  - the variables or conditions to be evaluated
- Lower rows specify:
  - the actions to be taken when the corresponding conditions are satisfied.
- In technical terminology,
  - a column of the table is called a rule:
  - A rule implies:
    - if a condition is true, then execute the corresponding action.

Structure of Decision Table

	Decision rules			
	Rule 1	Rule 2	Rule 3	Rule 4
(condition stub)			(Condition entries)	
(action stub)			(Action entries)	

To present how a decision board can help in finding missing signals, we will use the example of simplified water heating, operating on the following principles:

-----Water in the tank should have a temperature between 30*C and 60*C.
-----The heater turns on when the temperature drops below 30*C.
-----The heater turns off when the temperature rises to 60*C.
-----The heater will also turn off when the water in the tank drops below the minimum level.



Conditions	1	2	3	4	5	6
Temperature	< 30*C	30*C - 60*C	> 60*C	< 30*C	30*C - 60*C	> 60*C
To low water level	0	0	0	1	1	1
Action	**1**	**2**	**3**	**4**	**5**	**6**
Water heating	0	0	0	1	1	0

Constructing Decision Table

- Name the conditions and the values each condition can assume
- Name all possible actions that can occur
- List all possible rules
- Define the actions for each rule
- Simplify the decision table

A Garment House announces its trade discount policy as follows:

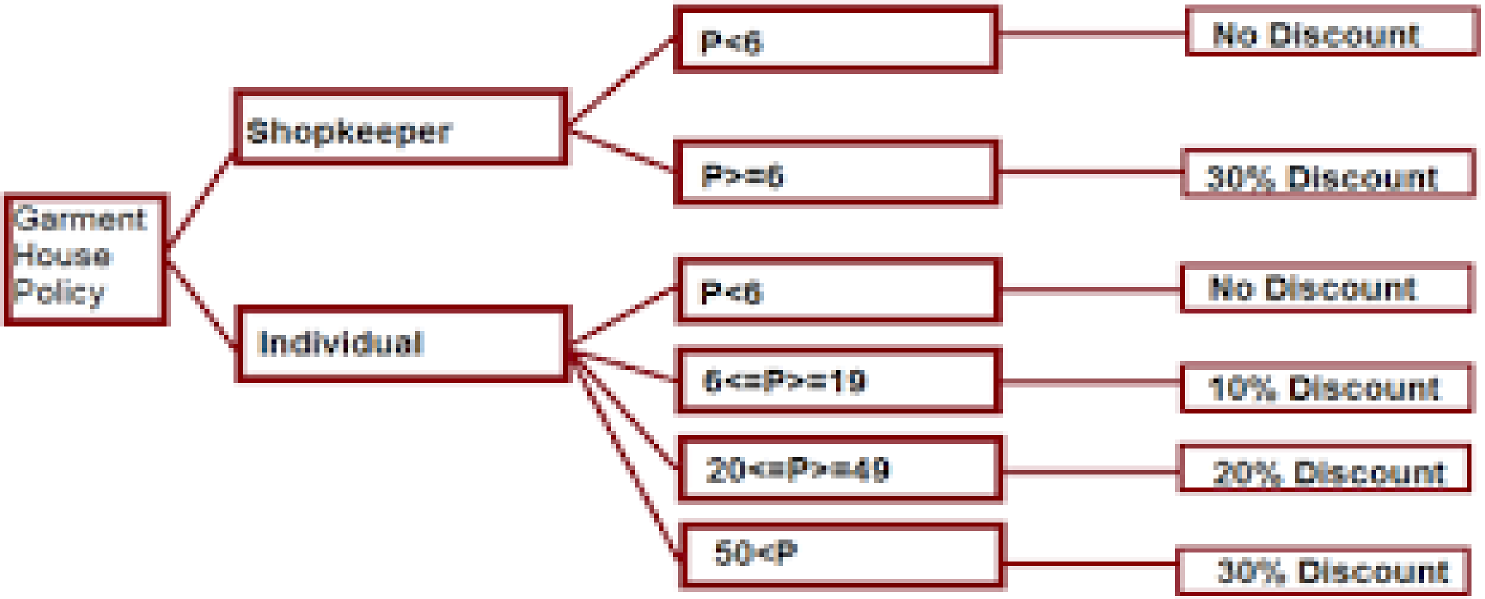
If a customer is from shop and does purchase garments more that 6 pair a flat discount of 30% would be provided.

If a customer is individual and does purchase garment of 6-19 pair 10% discount would be provided, 20% on discount for 20-49 pair and 30% discount on more than equal to 50 pair.

In no other case, a discount would be provided. Draw a decision tree and table table for above policies.

- A Garment House announces its trade discount policy as follows:-
  - if a customer is from shop and does purchase garments more that 6 pair a flat discount of 30% would be provided.
  - If a customer is individual and does purchase garment of 6-19 pair 10% discount would be provided, 20% on discount for 20-49 pair and 30% discount on more than equal to 50 pair. In no other case, a discount would be provided. Draw a decision table for above policies.

Dicision Tree



Condition	R-1	R-2	R-3	R-4	Else
Shopkeeper	Y	N	N	N	
p>=6	Y				
6<=p>=19		Y			
20<=p<=49			Y		
p>=50				Y	
ACTION:					
30%	X			X	
10%		X			
20%			X		
No Discount					X

Example: LMS

- A Library Membership automation Software (LMS) should support the following three options:
  - new member,
  - renewal,
  - cancel membership.
- When the new member option is selected,
  - the software asks details about the member:
    - name,
    - address,
    - phone number, etc.
- If proper information is entered,
  - a membership record for the member is created
  - a bill is printed for the annual membership charge plus the security deposit payable.

- If the renewal option is chosen,
  - LMS asks the member's name and his membership number
    - checks whether he is a valid member.
  - If the name represents a valid member,
    - the membership expiry date is updated and the annual membership bill is printed,
    - otherwise an error message is displayed.

- If the cancel membership option is selected and the name of a valid member is entered,
  - the membership is cancelled,
  - a cheque for the balance amount due to the member is printed
  - the membership record is deleted.

Comparison

- Both decision tables and decision trees
  - can represent complex program logic.
- Decision trees are easier to read and understand
  - when the number of conditions are small.
- Decision tables help to look at every possible combination of conditions.

Decision Trees

- Decision trees:
  - edges of a decision tree represent conditions
  - leaf nodes represent actions to be performed.
- A decision tree gives a graphic view of:
  - logic involved in decision making
  - corresponding actions taken.

  Example: LMS
- A Library Membership automation Software (LMS) should support the following three options:
  - new member,
  - renewal,
  - cancel membership.