**Data structures**

Program performance is affected by how fast data can be found/retreived when needed.

**Algorithms**

All roads may lead to your destination, but some will get you there faster

**Data Structure Purpose**

is vital role to the design & implementation of efficient algorithms and programming development Used in efficient problem solving field

**Some Basic Concepts**

• Data

Values or set of values

Data Item

Single unit of values

•Group Items

Data Items which are further divided into sub

items

•Elementary Items

Data Items which are not further divided into sub

items

• Information

Processed Data

Examples

•CNIC is an Elementary Item as it is not further divided into sub parts

• Employee's name is a Group Item because it is further divided into sub parts like

First Name, Last Name

**Organization of Data**

Data is organized into a hierarchy of

Fields

Attribute / Characteristic

Records

Collection of related Fields

Files

Collection of related records

**Entity**

Any object which has certain attributes / properties which may be assigned values. Values may be Numeric or Non-Numeric

e.g: Person, Place, Thing, Event etc.

**Field, Record, and File**

All these concepts related to entity

* Field is a single unit of information
* Record is a collection of fields values of a given Entity
* File is a collection or records of the entities

*A* ***File*** *can have fixed length records or variable length records*

**Records**

* Fixed Length Record
* Variable Length Record

•For Example, student records usually have variable lengths, since different students take different number of courses.

•variable length records have a maximum and a minimum length.

**Entity Set**

Entities with similar attributes

Examples

All employee in an organization

All students in a class

**Range**

• Each attribute of an entity set has a range of values.

A range may be defined as

"set of all possible values that could be assigned for a particular attribute"

**Primary Key**

• Each record in a file may contain many field items but the values in certain field may uniquely determine the record in the file. Such a field K is called Primary key and the values k1,k2,... in such a field are called keys or key values.

**Data Structures**

• The study of Data Structures includes 3 steps

1)Logical or Mathematical description of structure

2) Implementation of the structure on a computer

3) Quantitative Analysis of the structure

(Amount of memory to store structure and time required to process the structure)

**Data Structures**

*"an arrangement or organization of data in a computer's memory so that the storing and accessing of data in efficient way"*

Examples of several common data structures are

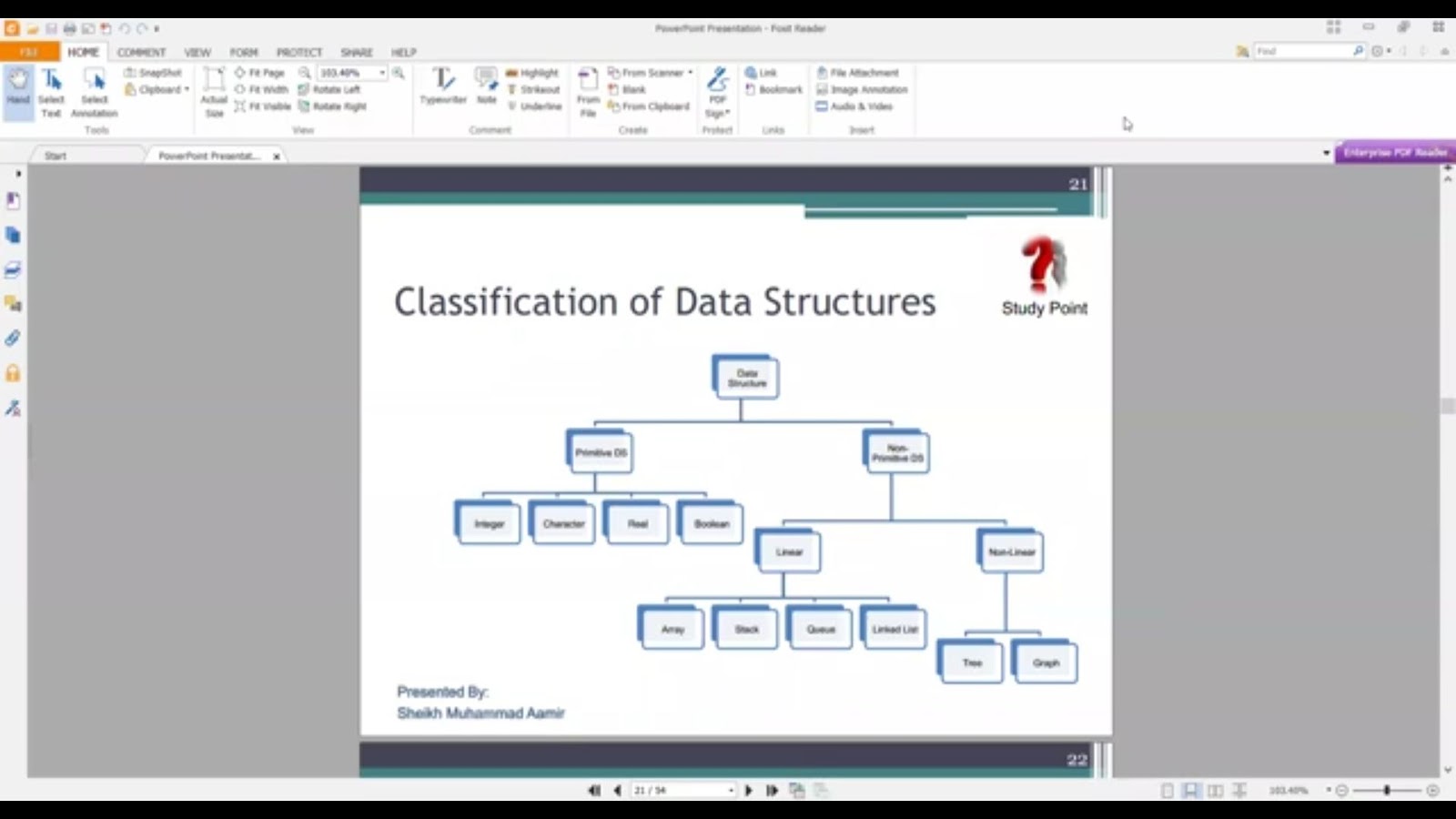
* Arrays
* linked lists
* Queues
* Stacks Binary
* trees
* Graph

**Algorithm**

are used to manipulate the data contained in these data structures as in “Searching and Sorting”

"Step by step procedures to solve the problem efficiently using data structures"

Classification of Data Structures



Arrays

•Array is a linear data structures (simplest type of DS)

• List of finite numbers of similar data elements referenced respectively by a set of n consecutive numbers like 1,2,...n

•Array has a name like A, ARRAY etc.

a1,a2,a3,....an (elements)

A[1], A[2]...[AN]

K in A[K]-------- Subscript/index

A[K]---- Subscripted variable

*Linear arrays are called* ***one-dimensional arrays*** *because each element in such an array is referenced by one subscript.*

***A two-dimensional array*** *is a collection of similar data elements where each element is referenced by two subscripts. Such arrays are called matrices in mathematics, and tables in business applications.*

**Arrays ...**

One dimensional array

Each element is referenced by one subscripted

e.g. Student

|  |
| --- |
| **Ali**  **Umar**  **Ahmad**  **Jamil** |

|  |
| --- |
| ALI UMAR AHMAD JAMIL |

**Link List**

•Linear data structure with dynamic memory allocation

Linear by means of Pointers

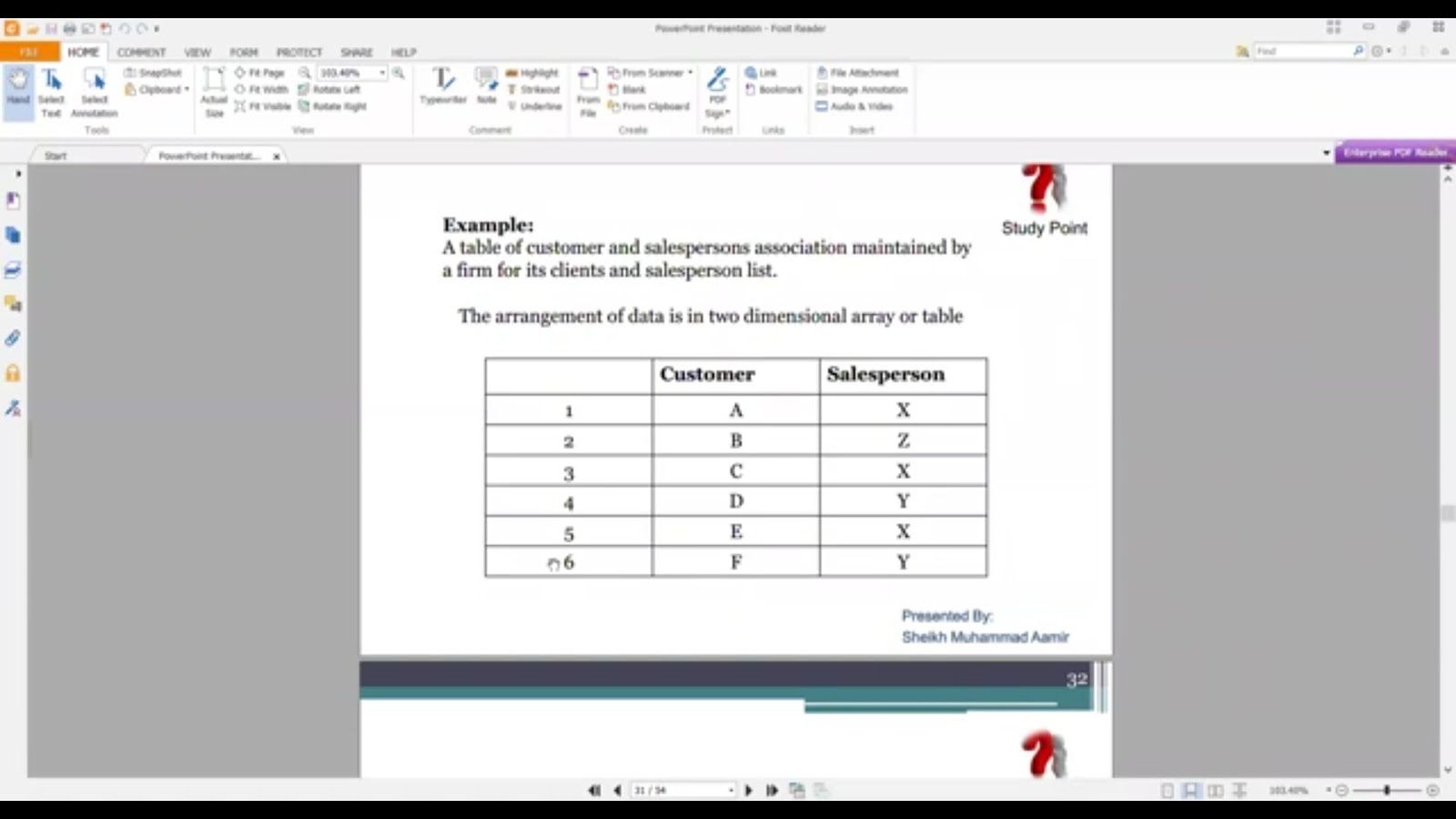
*A linked list, or* ***one-way list****, is a linear collection of data elements, called nodes, where the linear order is given by means of pointers. That is, each node is divided into two parts:*

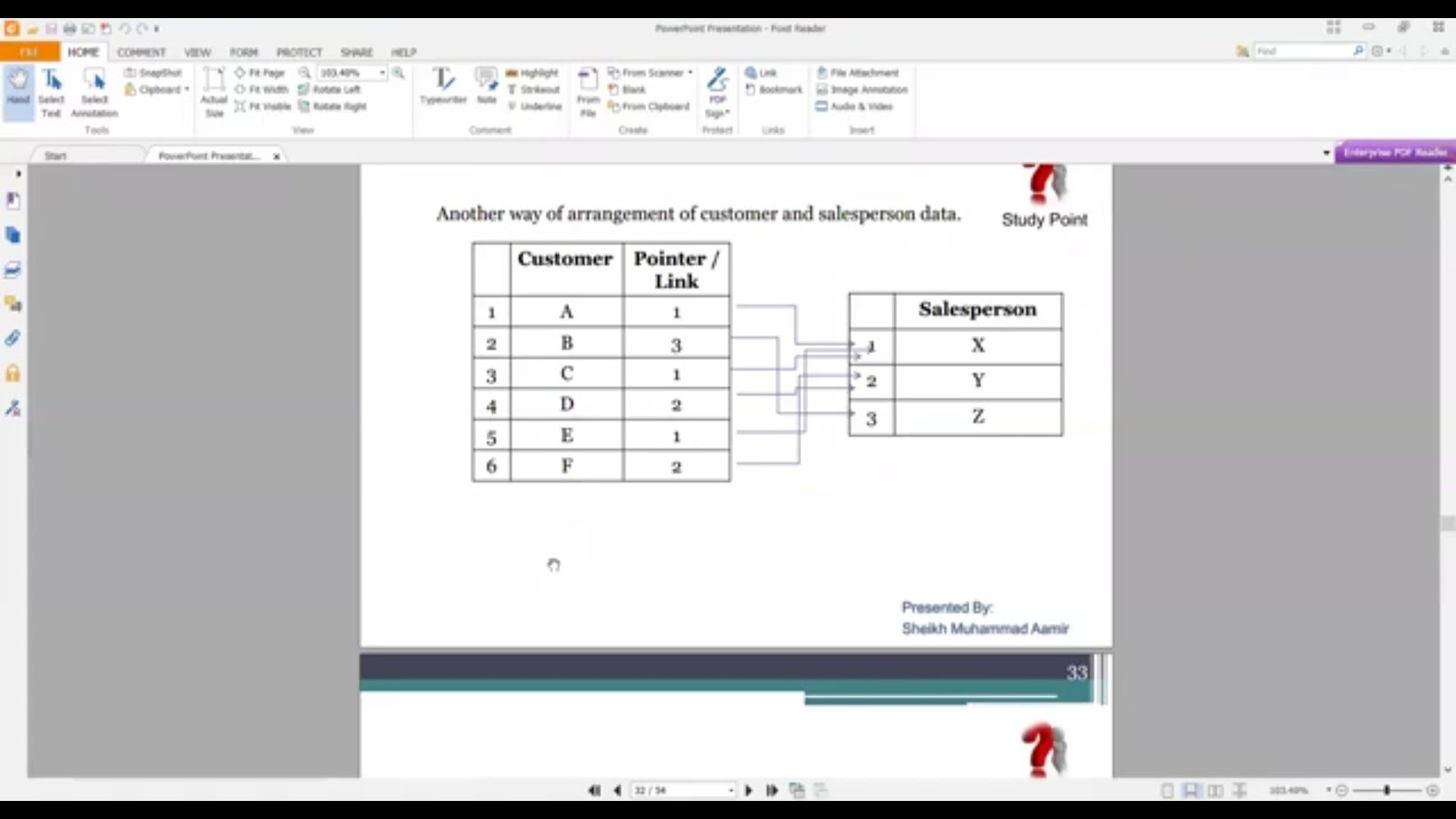
*a) The first part contains the data/value/information of the element*

*b) The second part, called the link field or next pointer field, contains the address of the next node in the list.*

Example: A table of customer and salespersons association maintained by a firm for its clients and salesperson list.

The arrangement of data is in two dimensional array or table





**Stack**

Linear Non-Primitive Data Structure

LIFO (Last in First Out) structure

*"Push-down list"*

• Insertion and Deletion can take place at one end (top).

E.g. Stack of dishes

**Queue**

Linear Non-Primitive Data Structure

•FIFO (First in First Out) data structure

•Insertion of element is done at one end (Rear)

Deletion of an element is done at other end (Front)

The features of a Queue are like the features of any queue of customers at a counter, at a bus stop, at railway reservation counter etc.

**Trees**

* Non-Linear Non-Primitive Data Structure
* Data frequently contain a hierarchical relationship b/w various elements.
* The data structure which reflects this relationship is called Rooted Tree Graph or simply Tree.

**Graph**

* Graphs are classified in the non-linear category of data structures.
* Sometimes data contain a relationship b/w pairs of elements which is not necessarily hierarchical in nature
* A graph G may be defined as a finite set V of vertices and a set E of edges (pair of connected vertices). The notation used is as follows:
* Graph G = (VE)

Graph ...

* Study Point Let we consider graph of figure, we may observe that the set of vertices for the graph is
* V = {1,2,3,4,5) and the set of edges for the graph is E = {(1,2), (1,5), (1,3), (5,4),(4,3),(2,3)} The elements of E are always a pair of elements. The relationship between Pairs of these elements is not necessarily hierarchical in nature.

Graph ...

Study Point A graph is like a road map. Cities are vertices. Roads from city to city are edges. (How about junctions or branches in a road? You could consider junctions to be vertices, too. If you don't want to count them as vertices, a road may connect more than two cities. So strictly speaking you have huperedaes in a hupergraph. If you want to allow more than one road between each pair of cities, you have a multigraph, instead. It all depends on how you want to define it.)

Data Structure Operations Study Point • Traversing

Accessing each record exactly once so that certain items in records may be processed Searching

Finding the location of record with a given key value Inserting

-Adding new record • Deletion

Removing a record

Data Structure Operations...

Study Point

Study Point

•Sorting

Arranging the elements in ascending / descending

order •Merging

Combining two lists into a single one

Abstract Data Types (ADT)

Study Point Collection of values(data) and a set of operations supported on these values. Example int Data:

integer values Operation: addition, subtraction,

multiplication, division etc, -An abstract data type can be further defined as a data declaration packaged together with the operations that are meaningful for the data type. In other words, we encapsulate the data and the operations on the data, and then we hide them from the user.

ADTS vs. Data Structures

Study Point An ADT is a description of some type of data (or a collection of data) and the operations on that data

Example: A Bank . It stores money . You can deposit, withdraw, write checks, check balance

A data structure is a way of structuring some collection of data Example: A pile of money, a safe full of money, etc.

ADTs have clean interfaces, and the implementation details are hidden

Data structures are often used to implement ADTS

Example: A Phone Book

Study Point Data: • Phone records Operations: • Add an entry • Remove an entry • Look up someone's phone number (by name)

View all entries • View one entry at a time, in order

Abstract Data Type Model

Two different parts of the ADT modelfunctions (public and private) and data structures. Data structures are available to all of the ADTS functions as required

A function may call on any other function to accomplish its task.

Dynamke Memory

ADTM

Abstract Data Type Model... Data are entered, accessed, modified and deleted through the external application programming interface.

Study Point

\*This interface can only access the public functions

•For each ADT operation, there is an algorithm that performs its specific task.

The operation name and parameters are available to the application, and they provide the only interface to the application.

Abstract Data Type Model...

Study Point

Data are entered, accessed, modified and deleted through the external application programming interface.

\*This interface can only access the public functions

•For each ADT operation, there is an algorithm that performs its specific task.

The operation name and parameters are available to the application, and they provide the only interface to the application.

Linear Arrays

Study Point

A linear array is a list of a finite number n of inhomogeneous data elements

(i.e., data elements of the same type) such that:

.(a) The elements of the array are referenced respectively by an index set consisting of n consecutive numbers,

-(b) The elements of the array are stored respectively in successive memory locations.

The number n of elements is called the length or size of the array. If not explicitly stated, we may assume the index set consists of the integers 1, 2, 3........

Data Structures & Algorithms "Algorithm, Notations, & Complexity"

(Chapter No. 2)

Algorithm

Study Point

• An algorithm is a step-by-step method of solving

computational tasks.

• Algorithm = A Precise Sequence of Actions for performing a computational task

(independent from computer languages, i.e. pseudo code).

Algorithmic Presentation

Algorithmic Presentation

Paragraph • Algorithm Name • Purpose of Algorithm • List of variables • List the input data

Steps

List of steps to be executed

Algorithmic Notation ...

Study Point Name of Algorithm • Every algorithm is given a name that represents

the purpose of the algorithm like "sum" etc. Steps, Control, Exit • The algorithm consists of a sequence of

numbered steps. • Each step describes one task to be performed. • The instructions in a computer program are written according to these steps.

Algorithmic notation ...

Study Point Variable Name • Variable name is usually written in capital

letters. • A variable name consists of letters, numeric

digits and some special characters. • Variable name is always beginning with a letter • Spaces are not allowed within a variable name

Algorithmic Notation ...

Study Point

Comments • Comments are used to explain the purpose of

step or statement. These may be given at the end of each statement or at the beginning of the step.

• Comments are written in [ ] for an algorithm.

Algorithmic Notation ...

Study Point

Assignment Statement • The assignment statement is used to evaluate an

expression and assign the calculated value to the

variable • ":=" operator is used to assign a value to the

variable with Set keyword • E.g. Set A:=5

Algorithmic Notation ...

Study Point

Input and Output • Data may be read / input by using Read statement

e.g. Read : variable names

• Similarly, messages placed in double quotes and data in variables may be output by means of a Write / Print statement as

e.g. Write : message / variable names

Algorithmic Notation ...

Study Point

Procedures • The term procedure will be used for independent module/ sub program which solves a particular problem.

• The word "Algorithm" will be reserved for the

solution of general problems.

Example-1

Study Point (ADDITION) This algorithm will find the sum to two

variables A, B in C. Steps: 1. Write: "Enter first value" 2. Read: A

Write: "Enter second value" Read: B

SET C:= A + B 6. Write: "The sum is". C

Exit

Algorithmic Notation ...

Study Point

Input and Output • Data may be read / input by using Read statement

e.g. Read : variable names

• Similarly, messages placed in double quotes and data in variables may be output by means of a Write / Print statement as

e.g. Write : message / variable names

Example-2

Study Point (Temperature) This algorithm will convert the

temperature from C (Celsius) to F (Fahrenheit) after reading from user. Steps: 1. Write: "Enter temperature in Celsius" 2. Read: C

SET FI=C\*9 / 5+ 32

Write: "The temperature in Fahrenheit is ", F 5. Exit

Sequential Logic (Sequential Flow)

Study Point

• The modules are executed in sequence

Control Structures

 1) Sequence Logic, or sequential flow

2) Selection Logic, or conditional flow

3) Iteration Logic, or repetition flow

Selection Logic (Conditional Flow) • The structure which implement the selection

logic are called if structures

• The end of such a structure is presented by the

statement [end of if structure]

Selection Logic (Conditional Flow)...

• Single Alternative

If condition, then:

 · [Module A]

[End of If structure]

Selection Logic (Conditional Flow)...

• Double Alternative

• If condition, then:

[Module A] Else:

[Module B]

[End of If structure]

Selection Logic (Conditional Flow)... • Multiple Alternative • If condition(1), then:

[Module A1] Else if condition (2), then:

[Module A2] Else:

[Module B] [End of If structure]

Example • The solution for quadratic equation ax2+bx+c=0 where a = 0, are given by quadratic formula

-b1b-4c

Iteration Logic (Repetitive Flow) • Repetitive flow begins with Repeat statement and is followed by a

modale called the body of the loop) • End of this structure is presented by

[End of loop) • The rear for loops the index variable, wachas K, to control the

loop

• The format of a loop is as

Repeat for kRtes by

Module) [End of loopi Here is an initial value in last value and is increment. Loop will Berminate when KS

Iteration Logic (Repetitive Flow)... • The repeat-while loop uses the condition to

control the loop.

• The format of this loop is as

Repeat while condition:

[Module] [End of loop]