\*CMP 212\*

DATA

Data is any representation of a fact or input to a computer that can be communicated or manipulated by some process.

DATA STRUCTURES

This is the organisation of data in computer memory in the way in which data can be stored, processed or retrieved.

APPLICATIONS OF DATA STRUCTURES

- Complier design

- Data management system

- Simulation

- Operating System

TYPES OF DATA STRUCTURE

1. Array

2. Ordered array

3. Stack

4. Queue

5. Linked list

6. Binary Trees

7. Hash Table

8. Heap

9. Graph

CLASSIFICATION OF DATA STRUCTURES

1. Simple

2. Compound

3. Linear

4. Non-linear

OPERATIONS FOR LINEAR DATA STRUCTURES

- Add an element

- Delete an element

- Traverse

- Sort the list of elements

- Search for a data element

EXAMPLES: Stack, Queue, Tables, List and linked lists

NON-LNEAR DATA STRUCTURE

This is a collection of a randomly distributed set of data items joined together by using a special pointer.

EXAMPLES: Tree, decision tree, Graph and Forest

OPERATIONS FOR NON-LINEAR DATA STRUCTURES

- Add an element

- Delete an element

- Display the elements

- Sort the list of elements

- Search for a data element

ABSTRACT DATA TYPE

This is the idea of encapsulating the details of the implementation, we are hiding them from the user's view.

Using suitable Data Structures to solve problems such as

- Spreadsheet program - use grid

- Bank-Account database - use array or set

- File-System manager - use tree-like structure

Note: Each problem has a distinct data structure in the solution

DATA TYPE

This is the way to re[resent a articular set of values and to determine what operations can be performed on those values

DATA TYPE CLASSIFICATION

- Integers (simple number eg. 5,-5,9)

- Real Numbers (fractional numbers eg. 2.5, 3.667)

- Characters (readable text and symbols)

- Booleans (values false and true)

PRIMITIVE DATA TYPES

These types may be native (built-in) to the machine on which the programs are run or to the compiler or interpreter that is translating these programs.

FIXED-POINT REPRESENTATION OF REAL NUMBERS

Real numbers should be written with the decimal point fixed in its correct position. eg. 13.75 or 3862.4

This representation becomes cumbersome when the numbers are large dealing e.g. 1375000000, 386240000

FLOATING-POINT NOTATION

This is referred as MANTISSA and EXPONENT. It is a real number times a BASE raised to an integer power

e.g if the base is fixed at 10, the number 387.53 could be represented

as 38,753 times 10 to the -2 power. The mantissa is 38753 and the exponent is -2.

WRITTEN AS:

.38753 X 10^3 and 387.53 \* 10^0

SCIENTIFIC NOTATION

Scientific notation commonly uses the letter E in place of "times 10", as in 5.0E3. meaning 5.0 times 10 to the third power, or 10^3.

Thus 640,000 = 6.4E+05 or 6.4E5.

ARRAY

DEFINITION OF ARRAYS

An array is a collection of variables of the same type lat are referred to through a common name.

HOW TO WRITE ARRAYS

If we call this array a, we can write it as:

a = [1; 4; 17; 3; 90; 79; 4; 6; 81]

SIZE OF AN ARRAY

For instance, since 'a' in the able array has 9 items, then we say that its size is 9.

INDEX (INDICES)

This is the position of an element of an array eg.

a(0) = 1,

all =4.

a[2] = 17, and so on.

PROPERTIES OF ARRAY

1. Each element in an array is of the same data type and carries the same size that is 4 bytes.

2. Elements in the array are stored at contiguous memory locations from which the first element is stored at the smallest memory location.

3. Elements of the array can be randomly accessed since we can calculate the address of each element of the array with the given base address and the size of the data element.

DIMENSION OF AN ARRAY

The dimensionality of an array is how many axes it has. You index into it with one subscript, e.g. array[n] . You index into it with two subscripts, e.g. array[x,y] .