Data Structures Using 'C' – (22317)

UNIT I – Introduction to Data Structures

(Weightage-06marks)

Q1. Define Data structure . (2marks)

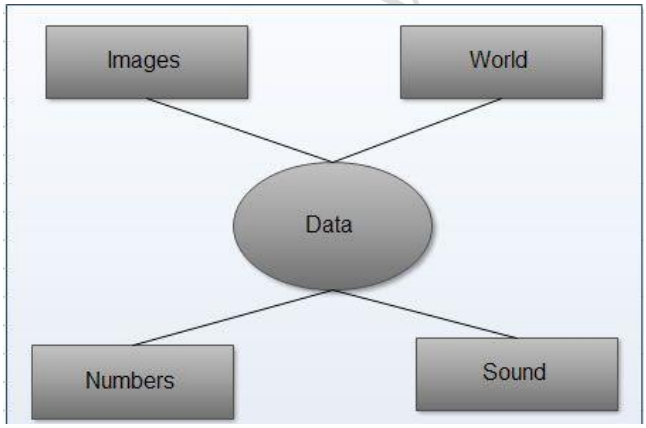
Ans- Data Structure can be defined as the group of data elements which provides an efficient way of storing and organizing data in the computer so that it can be used efficiently.

Some examples of Data Structures are arrays, Linked List, Stack, Queue, etc.

Data Structures are widely used in almost every aspect of Computer Science i.e. operating System, Compiler Design, Artificial intelligence, Graphics and many more.

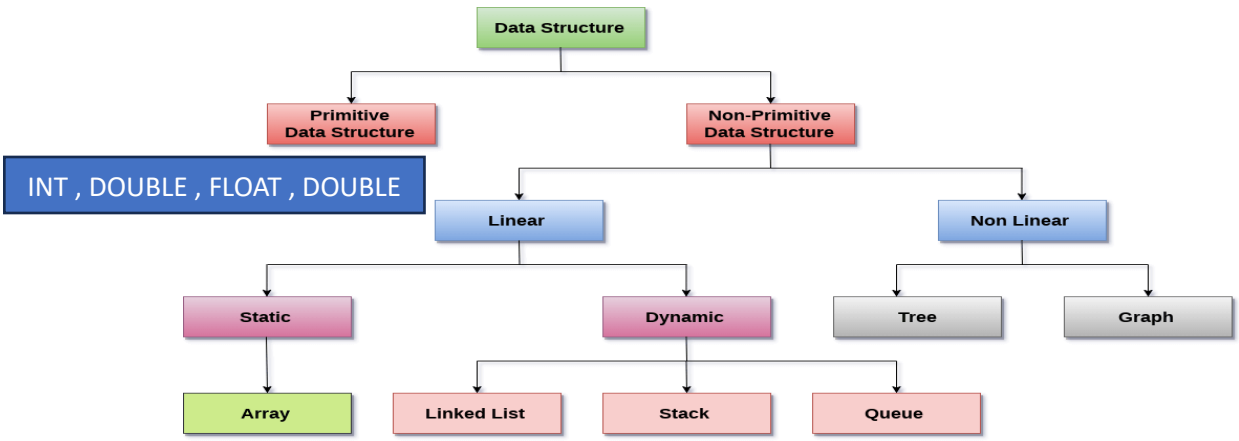
Data Structures are the main part of many computer science algorithms as they enable the programmers to handle the data in an efficient way.

It plays a vital role in enhancing the performance of a software or a program as the main function of the software is to store and retrieve the user's data as fast as possible.



Q2. Give classification of data structure . (2marks)

Ans -



Q3. Define Linear and Non-Linear Data Structure. (2marks)

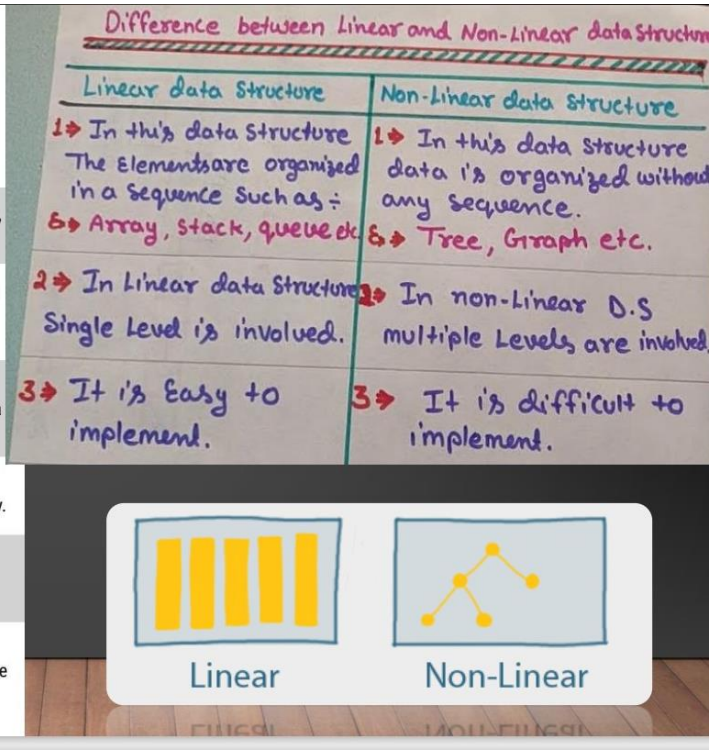
Ans –

Non Linear Data Structures: This data structure does not form a sequence i.e. each item or element is connected with two or more other items in a non-linear arrangement. The data elements are not arranged in sequential structure. Example-Trees and graphs

Linear data structures - Data structures in which data elements are stored in a linear sequence. Example-Array , stack , queues , linked list .

Q4. Compare Linear and Non – Linear Data Structure . (2marks)/(4marks)

Linear Data Structure	Non-linear Data Structure
In a linear data structure, data elements are arranged in a linear order where each and every elements are attached to its previous and next adjacent.	In a non-linear data structure, data elements are attached in hierarchically manner.
In linear data structure, single level is involved.	Whereas in non-linear data structure, multiple levels are involved.
Its implementation is easy in comparison to non-linear data structure.	While its implementation is complex in comparison to linear data structure.
In linear data structure, data elements can be traversed in a single run only.	While in non-linear data structure, data elements can't be traversed in a single run only.
In a linear data structure, memory is not utilized in an efficient way.	While in a non-linear data structure, memory is utilized in an efficient way.
Its examples are: array, stack, queue, linked list, etc.	While its examples are: trees and graphs.
Applications of linear data structures are mainly in application software development.	Applications of non-linear data structures are in Artificial Intelligence and image processing.



Q5. Define Algorithm (2marks)

Algorithm is a stepwise set of instructions written to perform a specific task

It is also called step by step solution of an program.

It refers to logic of program. Algorithms are universal .

The are step by step procedures which perform task of solving logics.

Algorithm should have charaterics such as input , outout .finiteness,effectiveness etc….

Programs becomes easy due to algorithm.

Q6 Define complexity and classify it .(2marks)

Or **Define Algorithm Complexity (2marks)**

Or **Describe Time and Space complexity with Suitable example.** (**4marks)**

ANS-

Algorithm Complexity

Algorithm complexity measures how many steps are required by the algorithm to solve the given problem.

It evaluates the order of count of operations executed by an algorithm as a function of input data size.

The complexity can be found in any form such as constant, logarithmic, linear, n*log(n), quadratic, cubic, exponential, etc.

The efficiency of an algorithm depends on two parameters:

• Time Complexity • Space Complexity

Time Complexity: It is defined as the number of times a particular instruction set is executed rather than the total time is taken.

It is because the total time took also depends on some external factors like the compiler used, processor's speed, etc.

Example,

Instructions	Execution time

i=1
Loop(i<=n)
{
N=n+i;
i=i++;
}
	1 time
	n+1 times
	
	n times
	n times

Total =	3n+2 so, O(n) Linear time complexity

Space Complexity: It is the total memory space required by the program for its execution.

Int=2bytes, float-4bytes , char-1byte, double-8bytes

Example,

Instructions	Space requirement

i=1
Loop(i<=n)
{
N=n+i;
i=i++;
}
	Int i=2bytes
	N=2bytes
	
	Total space complexity – 4bytes

Q7. Abstract Data Type. (2marks)

Ans-

All Programming languages specify some primitive data types but sometime situation occurs that build in data types are not enough to handle complex data structure.

So , programmer creates own data types (User defined) known as Abstract Data structure.

The user of data type does not need to know how that data type is implemented, for example, we have been using Primitive values like int, float, char data types only with the knowledge that these data type can operate and be performed on without any idea of how they are implemented.

It is a special kind of data type , whose behaviour is defined by a set of values and set of operations . The keyword"abstract" is used as we can use these datatypes , we can perform operations.

But operations are working that is totally hidden from user.

Adt is non primitive data types . Operations logics are hidden.

Examples , stacks queue ect…..

Q8. Write any four operations that can be performed on data structure.(2marks)

Inserting: Adding a new data in the data structure is referred as insertion.

Deleting: Removing a data from the data structure is referred as deletion.

Sorting: Arranging the data in some logical order (ascending or descending, numerically or alphabetically).

Searching: Finding the location of data within the data structure which satisfy the searching condition.

Traversing: Accessing each data exactly once in the data structure so that each data item is traversed or visited.

Merging: Combining the data of two different sorted files into a single sorted file.

Creation: To create new Data Structure /Destroy: To delete Data Structure

Updating: To edit or change the data within the data structure.