Date __/__/__
Page _____
Puffin

DATA STRUCTURE AND ALGORITHM

1. Data Structure :-

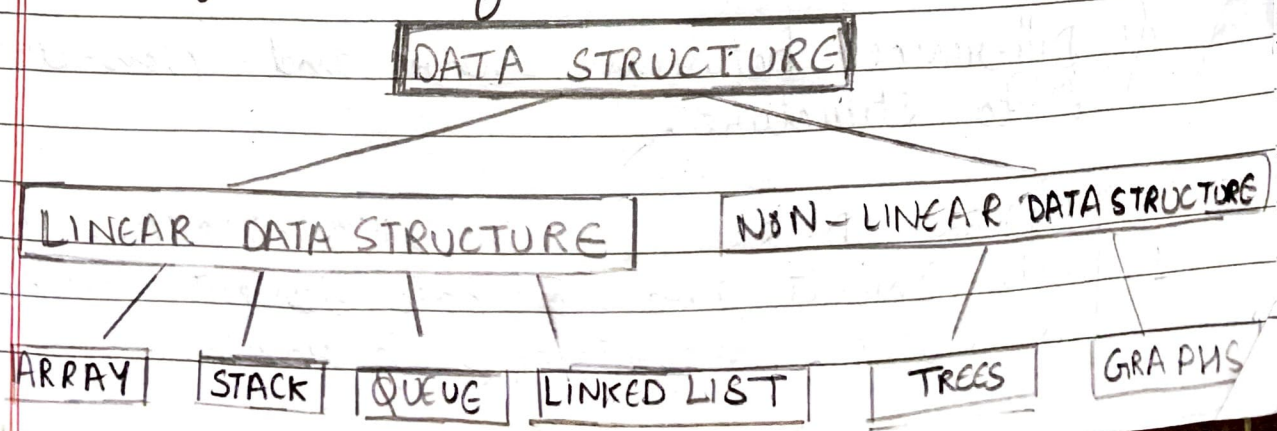
i) Definition ⇒ i) A data structure is a way of storing and organizing data in a computer so that it can be used efficiently.

ii) ALGORITHM + DATA STRUCTURE = PROGRAM

iii) It is basically a scheme for ORGANIZING DATA in COMPUTER'S MEMORY in such a way, that it could make the DATA QUICKLY available to the PROCESSOR for REQUIRED CALCULATIONS.

iv) Data Structure is defined in way of addressing as functions as ADT (Abstract Data Type), it means it is independent of implementation.

ii) Classification of Data Structure ⇒



Date __/__/
Page
Puffin

LINEAR DATA STRUCTURE :-

- Organized in a linear fashion.
- Stored one after another and can be traversed in similar fashion.
- Since, memory organization is in linear fashion, can be easily accessed.
- It has unique predecessor and unique successor.

Example: ARRAY, STACK, QUEUE AND LINKED LIST

NON-LINEAR DATA STRUCTURE :-

- Data not organized in sequential or linear fashion. A data item in a non-linear data structure could be attached to several other data elements.
- Can not be traversed in a single run.
- They have a complex structure.
- These data structure are multilevel structure.

Example; Tree and Graphs

Q Difference between Linear and Non-Linear Data Structure.

	LINEAR D.S	NON-LINEAR D.S
1.	Data elements arranged in a linear order.	Data elements attached in hierarchial manner

2	In linear d.s, single level is involved.	In Non-Linear d.s, multiple levels are involved.
3.	Can only be traversed in single run only.	Here elements can't be traversed in single run.
4.	Memory is not utilized in an efficient way.	Memory is utilized in an efficient way.
5.	Applications of linear d.s are mainly in application software.	Applications of non-linear d.s are in artificial intelligence and image processing.

⇒ OPERATION ON DATA STRUCTURE :-

1. Traversing :-
It is a kind of operation that can be performed on every type of data structure. The meaning of traversing is to visit each and every element of data structure and apply any process required.
Example, 100 employees in a company, printing the salary cheque of each employee is an example of traversing.

2. Insertion :-
Operation used to insert the record in between, starting or end or at any given location in the list of record.
Example, Adding an employee detail in between database when forgotten is an

example of insertion.

3. Deletion :-
It means removing an obsolete records from the data structure.
Example, Deleting the record of the employee who left the organization is an example of deletion.

4. Searching :-
Finding the specified records in a given data structure on the basis of a key value.
Two types of searching are :-
- Linear Search
- Binary Search

5. Selection :-
It accesses the data within the data structure.

6. Sorting :-
Arranging the record on the basis of key value in a specified order.
It is ~~arranged~~ to arrange data in some logical order either in increasing or decreasing order.
Types are :-
Bubble Sort, Quick sort, Merge sort, Heap sort, Insertion Sort etc.

Date __/__/__
Page _____
Puffin

7. Merging :-
This operation is mostly used in every organization where data is placed in different locations. It merges the data of two or more places to a single place or list.

8. Splitting :-
In this operation, records in a large file are split into smaller files for processing.

9. Copying :-
This data structure is used to copy the contents of one object to another object.

10. Concatenation :-
Selecting the records of two similar data structure and combining them is known as concatenation.

APPLICATION OF DATA STRUCTURE :-

- It describes the physical and logical relationship between the data item.
- It provides the method of representing the data in the memory of computer efficiently.
- Gives considerable help in compiler design.
- Helps in statical analysis.
- Very useful in Stimulation.

- Solving of numerical problems
- Considerable help in managing the operating system.
- Helps in graphic operations.

Q Advantages of Data Structures

- It describes the physical and logical relationship between the data item.
- It helps in statistical design and analysis.
- It is very useful in simulation.
- It helps alot in graphic applications.
- It gives considerable help in compiler design.

ALGORITHM

⇒ An Algorithm is a finite set of instructions or statements that can be used to perform a specific task.

⇒ When the data structure has been chosen for the particular application it is given life by algorithms that MANIPULATE the data elements stored in the structure.

Date __/__/__
Page _____
Puffin

⇒ Algorithm is a language independent, well-structured and detailed description of logic.

⇒ Algorithm must satisfy the following criteria :-

- Input → There should be zero or more values which are to be supplied.
- Output → At least one result is to be produced.
- Definiteness → Each step must be clear and unambiguous.
- Finiteness → If we trace the steps of an algorithm, then for all cases, the algorithm must terminate after a finite number of steps.
- Effectiveness → Each step must be sufficiently basic that a person using only paper and pencil can in principle carry it out.

ALGORITHM BASICS :-

1. Comments :- Statements not executed in program but used to indicate the main purpose of the step.
They are described in the square brackets [ ].
Example, [Initialization of data]

2. Identifying number :- Each and every

algorithm has unique number. Example, Step 1 and Step 2.

3. Assignment Statement :- Assignment statement is used to assign value to a variable.
Example
Set MAX = 10
10 is assigned to variable MAX

4. Operators :- There are two Boolean values "TRUE" and "false". In algorithmic notation logical operators can be used as "AND", "OR" and "NOT".

5. Control Structure :- Every programming language support three types of programming construct and these are sequence control, selection control and iteration construct.

ALGORITHM COMPLEXITY :-

=> Designing an efficient algorithm for a program plays a crucial role in developing large scale computer system.

=> Each algorithm takes some storage and time to execute. These two factors measure the efficiency of

algorithm.

⇒ lesser these resources that an algorithm use more efficient it is.

⇒ Complexity of an algorithm is a function $f(n)$ that gives the running time and memory space required by an algorithm for processing the input data of size n.

⇒ Most important properties of an algorithm is how its execution time increases as the problem is made large. This is called algorithmic complexity of algorithm.

⇒ Complexity of an algorithm depends on the following two factors:-

- Space Complexity ⇒ How much memory cells are required to solve the problem.
- Time Complexity ⇒ How much time algorithm required to run to the completion.

SPACE COMPLEXITY:-

- Number of memory cells an algorithm required to solve a particular problem is called space complexity.
- A good algorithm keeps this number as small as possible.

Date __/__/__
Page
Puffin

- This space not only includes the instruction space but it also includes the data space.

TIME COMPLEXITY

- It is the amount of time it needs to run to completion.

- If input size of algorithm is more the complexity will be more and if input size of algorithm is less, the complexity will be less.

- To calculate the time complexity of an algorithm, the basic approach is to count the number of times a key operation is executed.

- The complexity of an algorithm is dependent upon the input size, still complexity can be divided into three types.

1. Worst Case Complexity :- It is the longest running time of an algorithm for all inputs of the given size. In this type of complexity, the key operation is executed maximum number times.

2. Best Case :- It is the shortest — running

Date __/__/__
Page _____
Puffin

time of an algorithm for all inputs of given size i.e minimum number of steps required to reach at the result.

3. Average Case :- If the running time falls between the best case and worst case then complexity is called average case. It is useful but more difficult to calculate.

TIME AND SPACE TRADE OFF

- Number of ways to solve the problem.
- Prefferable the one which is efficient in terms of time as well as space.
- But, this type of efficiency is not always achieved.
- So we compromise in some cases with time and in some case with space. This concept is basically the TIME - SPACE TRADE OFF.

ASYMPTOTIC NOTATION

- The Asymptotic Notation is used to describe the running time of an algorithm.
- There are three different notations

Date __/__/__
Page
Puffin

1) Big - Oh Notation

It describes the worst - case running time of a program. We compute it by counting no of iterations an algorithm will take in the worst case scenario with an input of N.

2) Big - Ω Notation

Big Ω describes the best running time of a program. We compute the big - Ω by counting how many iterations an algorithm will take in the best - case scenario based on an input of N.

3) Big - Θ Notation

We compute the big Θ of an algorithm by counting the number of iterations the algorithm always takes with an input of n.