22/12/24

DSA

Date __/__/____

Saathi

Q1→ A data structure is a particular way of organizing data in a computer so that it can be used effectively.

	Linear Data structure	Non-linear Data structure
1)	In this Data elements are arranged in a linear order where each & every element are attached to its previous & next adjecent.	In this data elements are attached in hierarchically manner.
2)	In this single level is involved.	In this multiple levels are involved.
3)	Its implementation is easy in comparison to N-L-D-S.	implementation is complex in comparison to L-D-S
4)	In this data elements can be traversed in a single run only	In this data element can't be traversed in a single run.
5)	In this memory is not used in efficient way.	Memory is utilized in efficient way
6)	Eg :- stack, queue	Eg Graphs & trees.

Q1→ A data structure is a particular way of organizing data in a computer so that it can be used effectively.

	Linear Data structure	Non-linear Data structure
1)	In this Data elements are arranged in a linear order where each & every element are attached to its previous & next adjecent.	1) In this data elements are attached in hierarchically manner.
2)	In this single level is involved.	2) In this maltiple levels are involved.
3)	Its implementation is easy in comparison to N-L-D-S.	3) implementation is complex in comparison to L-D-S
4)	In this data elements can be traversed in a single run only	4) In this data element can't be traversed in a single ran.
5)	In this memory is not used in efficient way.	5) Memory is utilized in efficient way
6)	Eg :- stack, queue	6) Eg Graphs & trees.

Date __/__/____

Saathi

Q4→ linear Search (Array A, value x)

Step 1 :- Set i to 1
Step 2 :- If i > 1 then goto step 7
Step 3 :- If A[i] = x then goto step #6
Step 4 :- ~~stet~~ Set i to i+1
Step 5 :- Go to step 2
Step 6 :- Print element x found at index i & goto step 8.
Step 7 :- Print element not found.
Step 8 :- Exit.

Q5→ Asymptotic notations are the mathematical notation used to describe the running time of an algorithm when the input tends toward a particular value or a limiting value.

Big O :- It represents the upper bound of the running time of an algorithm. Thus it gives the worst-case complexity of an algorithm.

Omega (Ω) :- It represents the lower bound of the running time of an algorithm. Thus it provides the best-case complexity of an algo.

Theta (Θ) :- It encloses the function from above & below. It represent upper & lower bound of running time of an algo
Analyzing average-~~time~~ case complexity of an algo.

Page No.

Saathi

Date __/__/____

Q9→ . Characteristics of Algo are as follows.

1) finiteness :- An algo should have finite number of steps & it should end after a finite time.

2) Input :- It may have many input or no inputs at all.

3) output :- It should result at least one output.

4) Definiteness :- Each step must be clear, well defined & precise. There should be no any ~~ambi~~ ambiguity.

5) Effectiveness :- Each step should be finite amount ~~of~~ time & must be simple.

Q10→ (i) The complexity of an algorithm computes the amount of time & spaces required by an algorithm for an input size (n).

(ii) Complexity of an Algo can be divided into two type.

1) Time complexity :-

It is a process of determining a formula for total time required towards the execution of that algo

2) space complexity :-

It is a process of defining a formula for prediction of how much memory space is required for successful execution of algo.

Page No.

Q11. 1. (i) The Data type is basically a type of data that can be used. In diff computer progra
(ii) Eg:- int, float etc
(iii) ADT is a special kind of datatype. whose behaviour is defined by a set of values & set of instruction.
(iv) The keyword 'Abstract' is used as we use this datatypes.
(v) The ADT is Madeup of Primitive Datatype but operation logic is hidden.
Eg) stack, Queue.

Q11. 3 → (i) Recursion is a process where function calls itself indirectly or directly to solve the Problem
(ii) The fuction that performs the process of recursion is called a recursive function.
(iii) There are certain problem than can be solved easily with the help of recursive algo

11.4 → (i) Stack is a data structure that follows the principle of Last in first out (LIFO)
(ii) This means last element inserted inside the stack is removed first.
(iii) Eg:- A pile of plates on one top of another.

11.5 → (i) It is a linear data structure that include series of connected nodes.
(ii) Here each node stores data & address of next node.

Date __/__/____

Saathi

Q16→ Sequential search :-

(i) In this each element of an array is retrieved one by one in a logical order & check weather it is desired element or not.

(ii) A search will be unsuccessful if all the elements are accessed, & the desired element is not found. Its the worst case.

(iii) In the average case we may have to scan half of the size of the array (n/2).

Binary Search :-

(i) It is a extremely efficient algorithm.

(ii) This search technique ~~son~~ consume less time in searching the given item in minimum possible comparison.

(iii) In this we have to find the middle element of the array first.

(iv) & then middle elemen of the array is compared to the element to be searc

(v) If the desired item is less than middle it will search first half

(vi) More 2nd half.

Time complexity of sequensial search is $O(N)$ while binary search has $O(\log_2 N)$.

Date __/__/____

Saathi

The best time of both is. O(1).

Worst case for sequential = N.

Binary = $\log_2 N$.

Q17→ ① The 2-dimentional array can be defined as ~~the~~ an array of arrays.

② The 2-dimentional array are organized as matrices which can be represented as the collection of rows & colums. as array [M] [N]. where M is Number of rows & N is No. of columns.

Eg

2D array	0	1	2
0	a(0,0)	a(0,1)	a[0,2)
1	a(1,0)	a(1,1)	a(1,2)
2	a(2,0)	a(2,1)	a(2,2)

Row Major order :- ~~It assigns~~.

Address of A[I][J] = B+w*((I-LR) * N+(J-LC))

Column Major order.

Address of A[I][J] = B+w*[(J-LC) * M+(I-LR)]

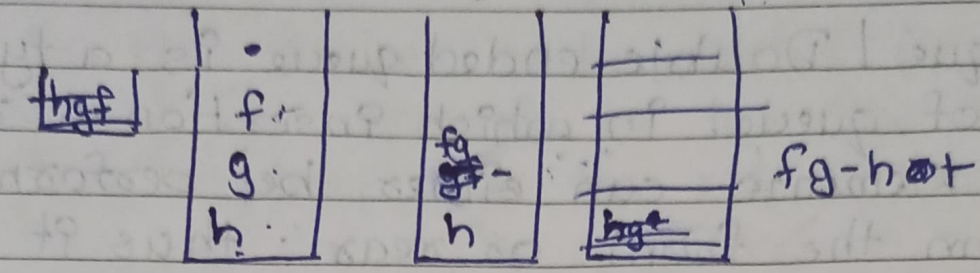
unit 3 & 4

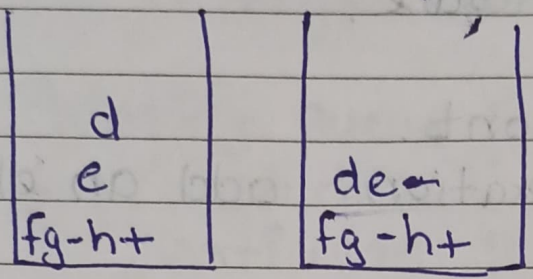
1) → Algorithm

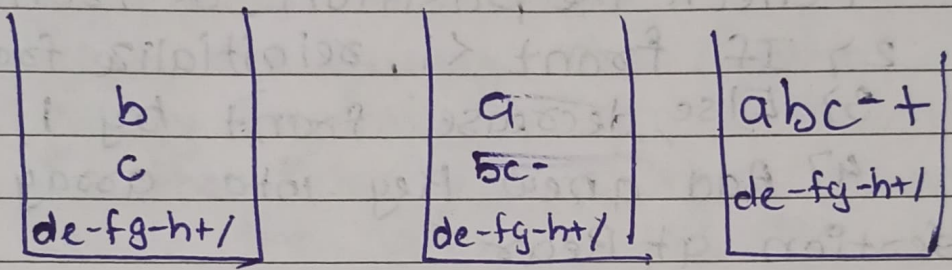
1) Add ')' to postfix expression.
2) Read postfix expression from left to right until) encountered.
3) If operand is encountered push it onto stack. [End if]
4) If operator is encountered pop two elements.
   i) A → top element
   ii) B → Next to top element
   iii) Evaluate: B operator A, Push B operator A onto stack.
5) Set result = pop
6) END.

Q2. Prefix to post fix.

*+a-bc/-de+fgh. {scan from Right to Left}







abc-+de-fg-h+/*.

Q3→ Algo

1) Create a stack.
2) For each character 't' in the input stream
   i) if (t is an operand)
      output t
   ii) else if (t is a right paranthesis)
      {Pop & output tokens until a left paranthesis is popped}.
   ~~iii) else.~~

iii) else.{
pop & output tokens until one of lower priority than t is ~~en~~encountered.
or a left paranthesis is encountered.
or the stack is empty.
PUSH t.
}

3) POP & output tokens until stack is empty.

Q4→ ① circular queue is just a variation of the linear queue in which front & rear end are connected to each other to optimize the space wastage of the linear queue & makes it efficient.

Advantages of circular queue over linear queue are:-

1) Easier for insertion & deletion:-
In circular ~~cur~~ queue, elements can be inserted easily if there are vacant location until its not fully occupied where-as in the case of a linear queue insertion is not possible once the rear reaches the last index even if there empty location present in the queue.

2) Efficient utilization of Memory:-
In the circular queue there is no wastage of memory as it uses the unoccupied ~~ca~~ space, & memory is used properly in a valuable & efficient manner as compared to a linear queue.

3) Ease of performing operation:-
In the linear queue., FIFO is followed, so the element inserted first is the element to be deleted first. but this is not the scenario in the case of the circular queue. as the rear & front are not fixed. so the order of insertion - deletion can be changed, which is very useful.

05→ (i) Priority Queue. is an extension of the Queue data structure: where each element has a particular priority associated with it.
(ii) Based on the priority value. the elements from the queue are deleted.
(iii) Array implementation of priority queue. is the idea to creat a structure to store the value & priority of the element & then creat an array of that structure to store elements.
Its basic operations are as follows.
1) enqueue :- used to insert element at the end of the queue
2) Peek() :- (i) Traverse accross the priority queue, & find the element with higest priority & return its index.
(ii) In the case of maltiple element with same priority, find the element with higest value & higest priority.

Date __/__/____

Saathi

dequeue = Decrease the size by one
~~& calls the peek.~~

Q11) → Deque / Double ended queue is a type of queue. in which insertion & removal of elements can either be performed from the front or rear. thus it does not follow FIFO rule.

operation on Deque

1) insert at the front.

This operation add an element at the front.

1) check the position of front
2) If front < 1, reinitialize front = n-1
3) Else, decrease front by 1
4) Add a new key into array [front].

2) Insertion at Rear.

This operation add an element to the rear.

1) Check if array is full.
2) If the deque is full, ~~recentralize~~ reinitialize rear = 0
3) Else rear by 1
4) add a new key into array [rear]

3) Delete from front.

operation deletes an element from the front.

1) check if deque is empty
2) if deque is empty ~~de~~ deletion cannot be performed.