Assignment

TOPIC ..... DATE .....

Name: KASAK Roll no - 02715611922 B.tech (AI & DS Section-S11

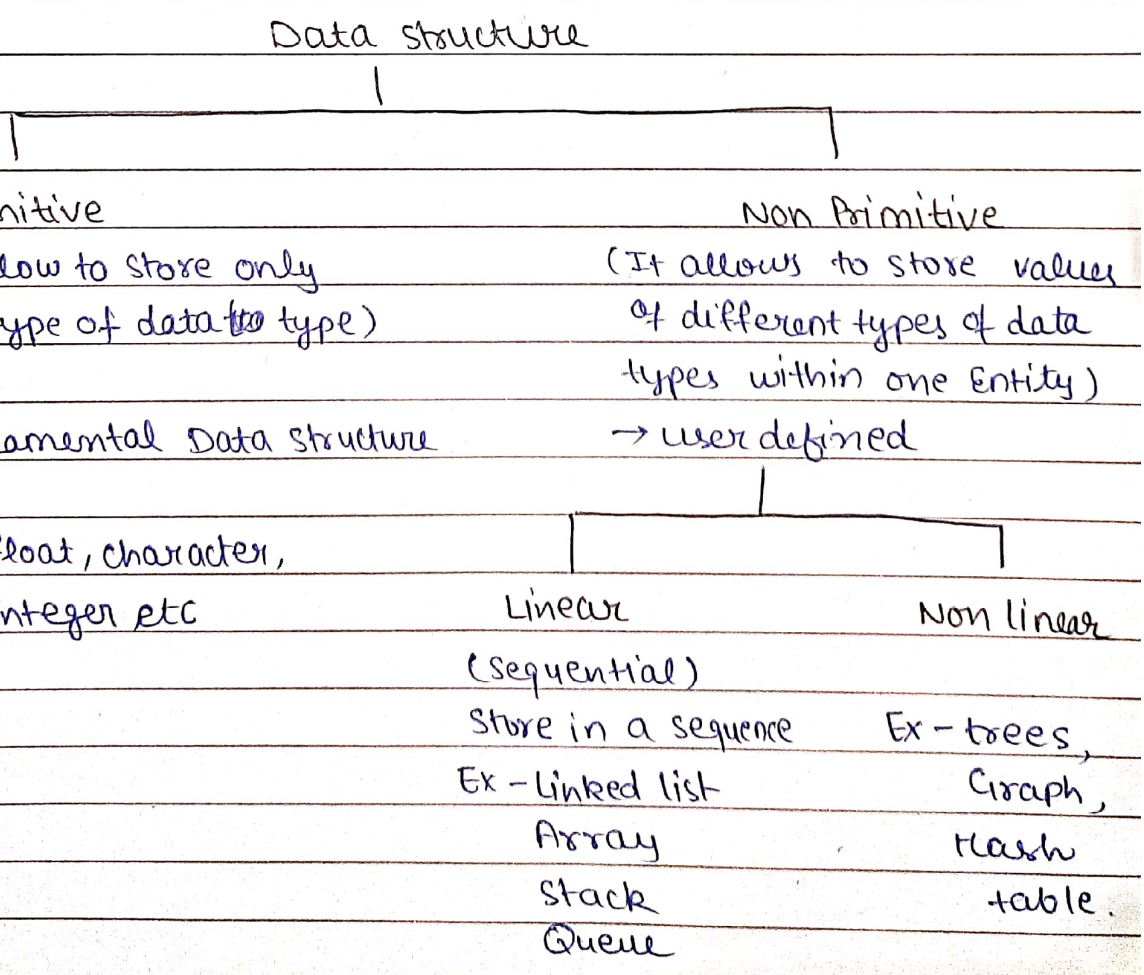
Data Structures

Ques 1^Q Explain the need of Data structure and Explain its types?

Data structure is a concept which shows that How the Data is stored in memory and describes what is the Relation between the Data & other Data.

Need of Data structure -

→ Data structure organize data ⇒ more Efficiently.
→ It has different ways of storing and organizing Data
→ It helps in understanding between Relationship between two Data Element.
→ It helps to store the Data in logical form.
→ It stores a Data that may grow or shrink dynamically.



TOPIC ..... DATE .....

(b) Differentiate between Searching & Sorting?

	Searching	Sorting
1.	Searching is used to find a Specific item in a collection of Data.	Sorting is used to organize a collection of data in a Specific order.
2.	Searching is a process that can be done on unsorted data.	Sorting Requires the data to be in Unsorted form.
3.	It doesn't change the order of Elements	It Rearranges the data in specific order.
4.	Ex : linear Search, Binary Search	Ex : Quick sort, Insertion Sort, Selection Sort

(C) Assume that 4 byte of storage is Required to store Each Element of an array of integers A[8][10] what will be the actual address of the Element A[5][9] Assume Base address is 1000

Size of array = A[8][10]
C

Address of Particular Element = Base address + size of data type (r1 * c + c1)

Address of A[5][9] (r1 c1) = 1000 + 4(5X10 + 9)

= 1000 + 4(59)

= 1000 + 245

= 1245

TOPIC ..... DATE.....

(d) write down algorithm for Linear Search?

Ans Step 1 : First, Read the search Element (Target Element) in the array.

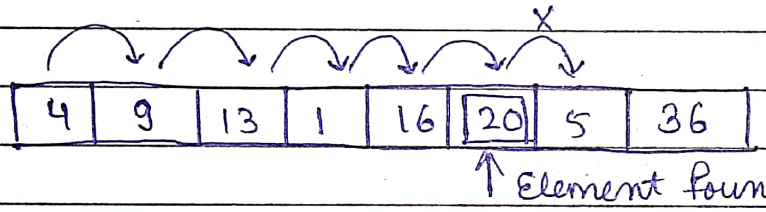
Step 2 : Second, compare the search Element with the first Element in the array.

Step 3: If both are matched, display "Target Element is Found" and terminate the linear search function.

Step 4: If both are not matched, compare the Search Element with the next Element in the array.

Step 5: Repeat 3 & 4 steps until the search (target) Element is compared with the last Element of the array.

Step 6: If last Element doesn't match, the linear search function will be terminated & display "Element is not found in the array".



Search_Element = 20

Question - 2

(a) Consider the queue capable of accommodating maximum five Elements -

Queue : _, C, D, _, _

(a) Add E.

(b) Add F, G, H

(c) Delete one Element.

(d) Add I, J, K.

(e) Delete two Elements

TOPIC ..... DATE.....

		Front	Rear	Queue
	[redacted]	1	2	–, C, D, –, –
(a)	Add "E"	1	3	–, C, D, E, –
(b)	Add "F"	1	4	–, C, D, E, F
	"G"	1	} Queue overflow	
	"H"	1		
(c)	Delete one Element	2	4	–, –, D, E, F
(d)	Add "I"	2	} Queue overflow	
	"J"	2		
	"K"	2		
(e)	Delete two Element	4	4	–, –, –, –, F.

(b) Convert the given infix expression to prefix expression using stack

(A+((B+C)* (D+E)+F/G))

Character	stack	Prefix Expression
)	)	
)	))	
G	))	G
/	))/	·G
F	))/	FG
+	))+	/FG
)	))+)	/FG
E	))+)	E/FG
+	))+)+	E/FG
D	))+)+	DE/FG
(	))+	+DE/FG
*	))+*	+DE/FG
)	))+*)	+DE/FG
C	))+*)	C+DE/FG
+	))+*)+	C+DE/FG

TOPIC ..... DATE .....

Characters	Stack	prefix Expression
B	))+*)+	BC + DE/FG
(	))+*	+BC+DE/FG
(	)	+*+BC+DE/FG
+	)+	+*+BC+DE/FG
A	)+	A+*+BC+DE/FG
(		+A+*+BC+DE/FG

Question-3

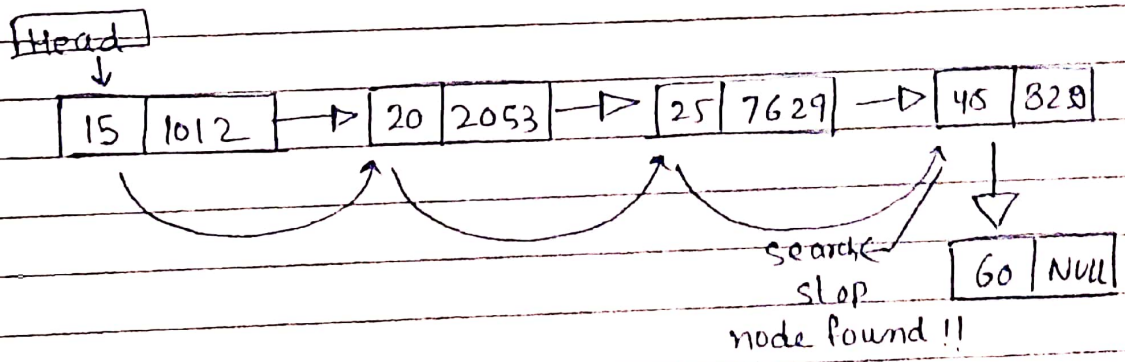
(a) Write an algorithm to search an Element in Linked List. Explain with the help of Example.

Algorithm to Search an Element in Linked list -

1. Initilize head = NULL.
2. Take input from the user for the item user wants to search
3. Linearly transverse the Linked List from head to End until hit the NULL node
4. compare each node data part to the data Entered by user.
5. Return index of node where data found Else Return -1.

Example

Let item to be search = 45



TOPIC ..... DATE.....

```
#include<stdio.h>
#include<stdlib.h>

struct node{
    int data;
    struct node *temp;
};*head;

int main()
search(){
    int s;
    if(head==NULL)
      Printf("List not found");
    Else{
      Print("Enter an Element for search");
      Scanf("%d",&s);
        temp=head;
        while(temp!=NULL){
          if(temp->data==s){
              f=1;
              break;
          }
          temp=temp->add;
        }
      if(f==1){
       Printf("Element found");
      }
      Else{
        Printf("Element not found in list");
    }
}
```

TOPIC ..... DATE .....

(b) Sort the following list using Insertion sort. Show the step by step process.

47, 25, 76, 13, 67, 60, 96, 38, 28, 79

47	25	76	13	67	60	96	38	28	79

swap

Ist	25	47	76	13	67	60	96	38	28	79
IInd	25	47	76	13	67	60	96	38	28	79
IIIrd	13	25	47	76	67	60	96	38	28	79
IVrt	13	25	47	67	76	60	96	38	28	79
V	13	25	47	60	67	76	96	38	28	79
VI	13	25	47	60	67	76	96	38	28	79
VII	13	25	38	47	60	67	76	96	28	79
VIII	13	25	28	38	47	60	67	76	96	79
IX	13	25	28	38	47	60	67	76	79	96

Question - 4

(a) write an algorithm to Merge two Linked list?

TOPIC ..... DATE .....

Question - 4

(a) write an algorithm to merge two linked list?

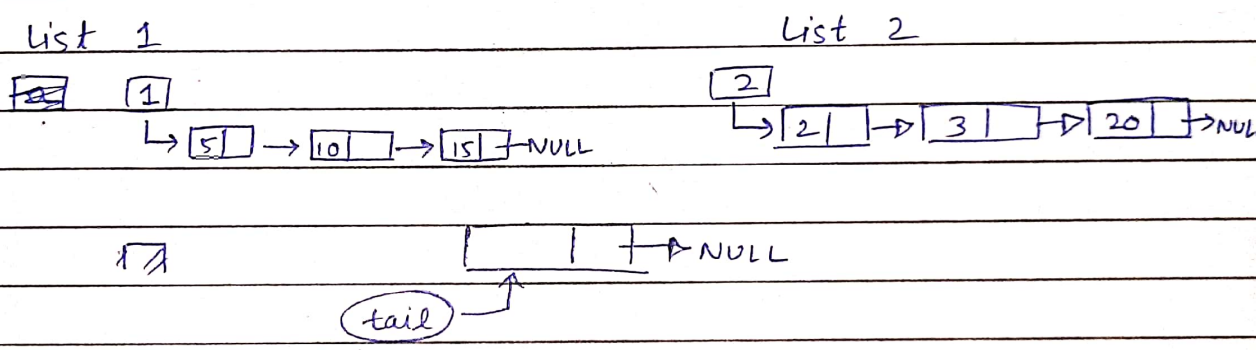
Ans

Step 1: Make a dummy node for the new merged linked list.

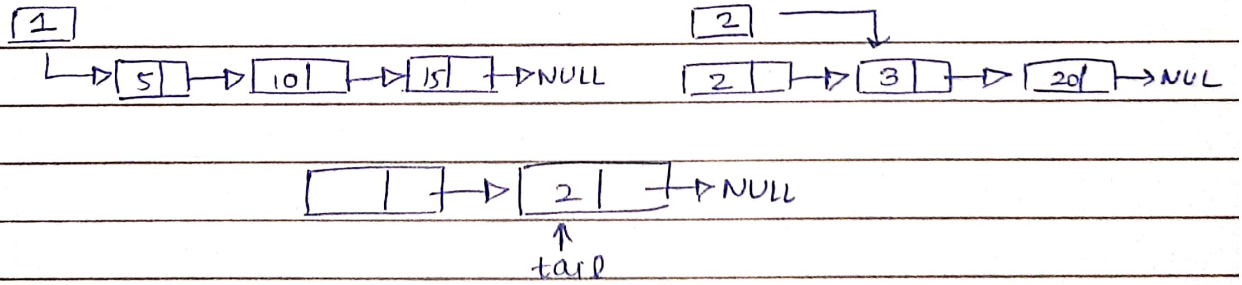
Step 2: make two pointers, one will point to list 1 and another will point to list 2.

Step 3: Transverse the list till one of them get exhausted

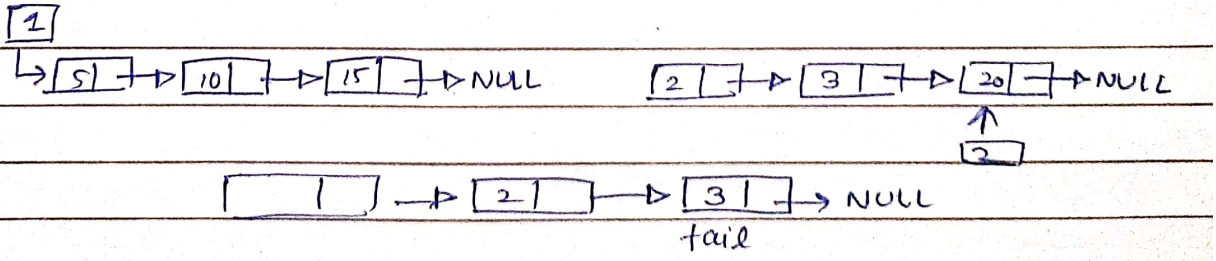
Step 4: If the value of node pointing to either list is smaller than another pointer, add that node to merged list and increment the pointer



Step 1

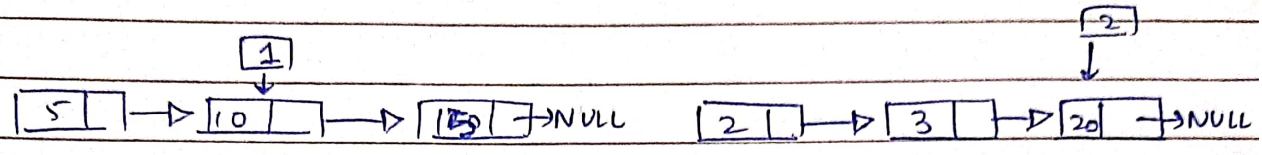


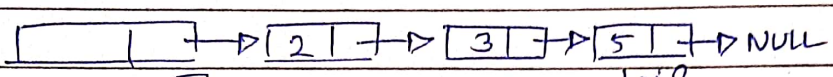
step 2



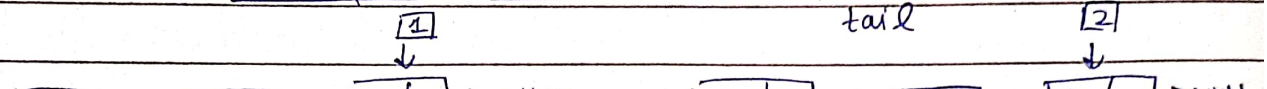
TOPIC ..... DATE .....

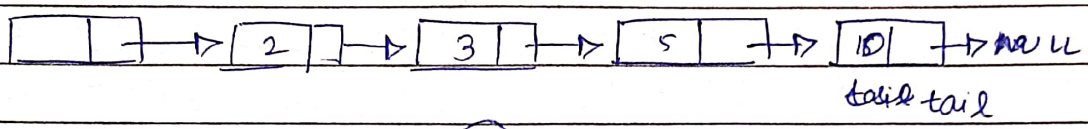
Step3



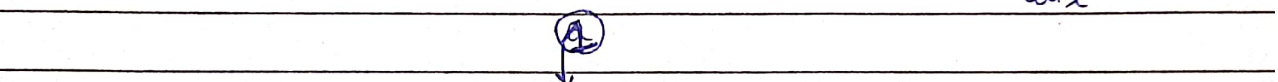


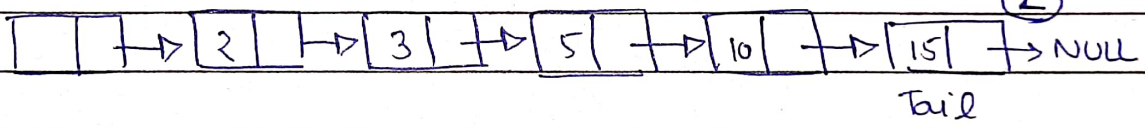
Step4



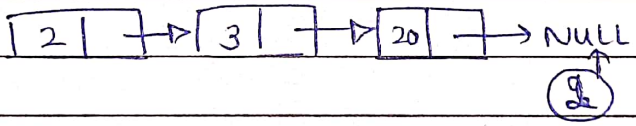


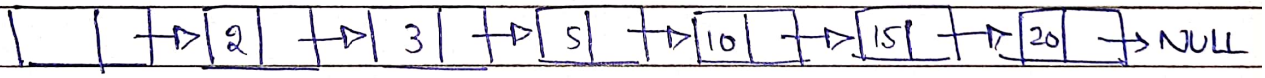
Step5





Step6





(b) write an algorithm to Search an Element in an array using Binary search. Explain with the help of an Example.

Algorithm for Binary Search -

1. Initilize three variable "left" to "0", "right" to the length of array minus 1, and "target" to the Element you want to find.
2. while "left" is less than Equal to "right" repeat -
   a. Calculate the middle index as "mid" using mid = (left + Right)/2.
   b. Compare the Element at index "mid" with "target" Element.
   → If Element at index mid == target Return mid.
   → If Element at index mid > target set Right to mid-1
   → If Element at index mid < target set left to mid+1

TOPIC ..... DATE.....

3. If ~~user Reach~~ Right = left then return -1. i.e., Element haven't found in array.

Example

arr = 2 4 6 8 10 12 14 16 18 20

Let Element user want to search = 14

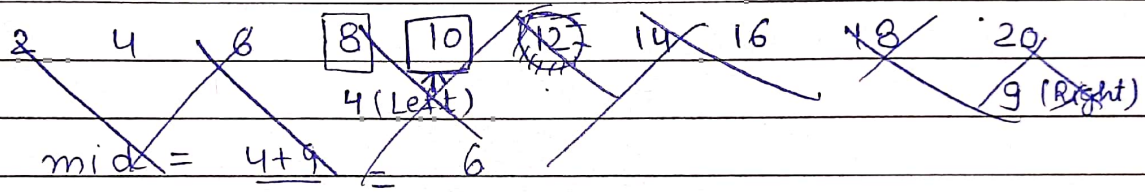
I^st

2 4 6 8 10 12 14 16 18 20

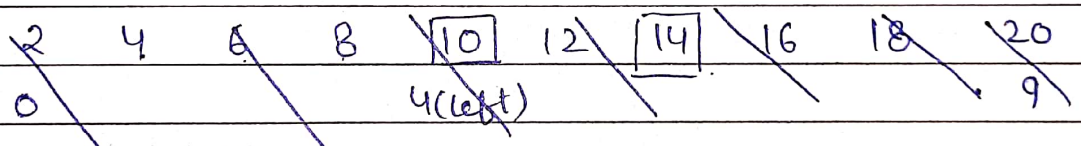
0 mid 9

$mid = \frac{9+0}{2} = 4$

14 > 10



$mid = \frac{4+9}{2} = 6$



$mid = \frac{4+9}{2} = 6$

14 = 14

II^nd iteration 2 4 6 8 10 12 14 16 18 20

mid, left (mid+1) = 5, Right = 9

$mid = \frac{5+9}{2} = 7$

16 ≠ 14

III^rd

2 4 6 8 10 12 14 16 18 20

left = 5, Right = mid - 1 = 6, mid

$mid = \frac{5+6}{2} = 5$

2 4 6 8 10 12 14 16 18 20

Now, Left & Right is 6 index and 14 is also present at index 6.