a) WAP to Implement single list with following operations: Sort the Linked List, Reverse the Linked List, Concatenation of two Linked Lists.

Pseudocode:

```
Function SortList(head);
    For each node i from head to End:
        For each node j after i:
            IF i.data > j.data :
                swap i.data and j.data
    Return head

Function Reverse List (head):
    prev ← NULL
    current ← head
    While current ≠ NULL:
        next ← current.next
        current.next ← prev
        prev ← current
        current ← next
    Return prev

Function Concat (A,B)
  IF A = NULL:
      Return B
  temp ← A
  While temp-next ≠ NULL :
      temp ← temp.next
  temp.next ← B
  Return A.
```

code:

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

struct node {
   int data ;
   Struct node * next ;
 };

struct node * create (int n){
      struct node * head = NULL, *p , *q = NUL
     for (int i=0 ; i< n; i++ ) {
       p = malloc (size of (struct node));
       scanf ("%d ", &p -> data);
       p -> next = NULL ;
       if (head == NULL)
            head =p;
       else
            q-> next = p;
         q = p;
  }
  return head;

void display (struct node* head){
    while (head) {
       print f(" %d ", head -> data);
         head= head -> next ;
    }
    printf("\n");
}
```

```
struct node       sort                        {
  struct node * i, *j;
  int t;
  for (i= head ; i; i=j->next)
      for (j= i-> next ; j ; j= j->next)
         if (i-> data  > j-> ~~data~~ data){
             t = i-> data;

            i -> data = j-> data;
            j-> data = t;

      }
   return head;

}

struct node * reverse (struct node * head){
  struct node * prev  = NULL, * curr= head,
       * next;
  while (curr) {
   next = curr  -> next;
   curr -> next = prev;
   prev  = curr ;
  curr = next;
   }
   return prev;
  }
 struct node * concat (struct node *a, struct
                                       node *b){

 if (a == NULL) return b;
 struct node * t =a;
 while (t ->next )
   t = t -> next;
```

```
    t->next = b;
    return a;
}
int main () {
    int n, m;
printf ("\n Enter size of List A: ");
scanf ("%d", &n);
printf ("\n Enter %d element(s) of list A : \n", n);
Struct node* A = Create (n);
printf ("\n Sorted List A : ");
A = sort (A);
display (A);
printf ("\n Reversed List A : ");
A = reverse (A);
display (A);
printf ("\n Enter Size of List B : ");
scanf ("%d", &m );
printf ("\nEnter %d elements of list B : \n", m)
 Struct node *B = create (m);
 printf ("\n List A+B : ");
 A = concat (A, B);
display (A);
return 0;
}
```

Output :

```
Enter Size of List A : 4
Enter 4 elements(s) of List A:
 1  2  3  4
Sorted List A : 1 2 3 4
Reversed List A : 4 3 2 1
Enter size of List B : 4
 5 6 7 8
List A + B : 4 3 2 1 5 6 7 8

Process returned 0 .
```

MG
26/11/25