lab-6.

4) WAP to implement single list with following operations : sort the linked list , Reverse the linked list , concatenation of two linked lists.

pseudo code:

```
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.
```

arr

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

struct node
{
    int data;
    struct node * next;

    };

struct node * create (int n)
{
struct node *head = NULL, *p, *q = NULL;
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)
   {
    printf ("%d", head->data);
     head = head->next;
   }.
printf("\n");
}.

struct node *sort (struct node *head)
{
  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)
             {
                 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 ("\n Enter %d element(s) 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 element(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.