```
#include <stdio.h>
#include <stdlib.h>
struct node
{
int data;
struct node *next;
};
void create(struct node **,int);
void display(struct node *);
void front_delete(struct node**,int);
void reverse();
void concat(struct node *,struct node *);
void sort(struct node **);
int c=0;
int main(int argc, char **argv)
{
    struct node *head=NULL;
    struct node *head1=NULL;
   struct node *head2=NULL;
       int choice;
       int ele;
       int ch;
       do
      {
     printf("\n1. Create \n2. Display \n3. Delete at front\n4. reverse\n5. Sort the list\n6. Concatenate
two linked lists");
     printf("\nEnter your choice : ");
     scanf("%d",&choice);
     switch(choice)
      {
         case 1: printf("Enter the data : ");
         scanf("%d",&ele);
         create(&head, ele);
         break;

         case 2: display(head);break;

         case 3: front_delete(&head,1);
         break;

         case 4:
         reverse(&head);
         display(head);
         break;

         case 5:
         sort(&head);
         display(head);
         break;

         case 6:
         do
         { printf("Enter the list 1 elements, press 1 to enter and press 0 to exit:");
         scanf("%d",&ch);
            switch(ch)
           {
             case 1:printf("Enter the data : ");
             scanf("%d",&ele);
             create(&head1,ele);
            default:break;
```

```
          }
        }while(ch!=0);
        do
        {
            printf("Enter the list 2 elements, press 1 to enter and press 0 to exit:\n");
            scanf("%d",&ch);
            switch(ch)
            {
             case 1:printf("Enter the data : ");
             scanf("%d",&ele);
            create(&head2,ele);
            default:break;
             }
        }while(ch!=0);
         concat(head1 ,head2);
         printf("\nThe concatenated list is as follows:\n");
         display(head1);
         break;
         default:exit(0);
      }
    } while(choice<=6);
}
void create ( struct node **headptr, int item)
{
    struct node *newnode;
    struct node *temp;
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=item;
    newnode->next=NULL;
    temp=*headptr;
    if(temp==NULL)
        *headptr=newnode;
    else
    {
       while (temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=newnode;
    }
}
void concat (struct node *temp1, struct node *temp2)
{

     while(temp1->next!=NULL)
         temp1=temp1->next;

      temp1->next=temp2;

}
void sort(struct node **h)
{
    int a;

    struct node *temp1;
    struct node *temp2;

    for(temp1=*h; temp1!=NULL; temp1=temp1->next)
      {
        for(temp2=temp1->next; temp2!=NULL; temp2=temp2->next)
```

```
        {
          if(temp2->data < temp1->data)
            {
              a = temp1->data;
              temp1->data = temp2->data;
              temp2->data = a;
            }
         }
      }
}
void front_delete(struct node ** head, int n)
{
    if (*head == NULL)
     {
         printf("Empty List. Can't delete\n");
         return;
     }
    struct node* temp1=*head;
    if(n==1)
    {
        *head=temp1->next;
        free(temp1);
        printf("Front node deleted\n");
        return;
    }
}
void display(struct node *ptr)
{


    if(ptr==NULL)
    {
        printf("Nothing to print\n");
    }
    else
    {
        while(ptr!=NULL)
       {
        printf("%d ",ptr->data);
        ptr=ptr->next;
       }
    }

}
void reverse(struct node ** head)
{
    struct node *prev=NULL,*current=*head, *next=NULL;
    while(current!=NULL)
    {
        next=current->next;
        current->next=prev;
        prev=current;
        current=next;
    }
    *head=prev;
}
```