```c
//Lab Program 7
//sort,reverse,concatenate linked lists(s)

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

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

void insertend(struct node **headptr){
    struct node *newnode,*temp;
    newnode = (struct node*)malloc(sizeof(struct node));
    int value;
    printf("Enter value: ");
    scanf("%d",&value);
    newnode->data = value;
    newnode->next = NULL;
    if((*headptr) == NULL)
        (*headptr) = newnode;
    else{
        temp = (*headptr);
        while(temp->next != NULL)
            temp = temp->next;
        temp->next = newnode;
    }
}
void deleteend(struct node **headptr){
    struct node *temp;
    if((*headptr) == NULL)
        printf("The list is empty\n");
    else{
        temp = (*headptr);
        while((temp->next)->next != NULL)
            temp = temp->next;
        temp->next = NULL;
    }
}
void display(struct node *temp){
    if(temp == NULL)
        printf("The list is empty\n");
    else{
        while(temp != NULL){
```

```
45              printf("%d\t",temp->data);
46              temp = temp->next;
47          }
48          printf("\n");
49      }
50  }
51  void sort(struct node *temp){
52      struct node *p,*q;
53      int a;
54      for(p = temp;p != NULL;p = p->next){
55          for(q=p->next;q != NULL;q = q->next){
56              if(p->data > q->data){
57                  a = p->data;
58                  p->data = q->data;
59                  q->data = a;
60              }
61          }
62      }
63      printf("The sorted list is as follows:\n");
64      while(temp != NULL){
65          printf("%d\t",temp->data);
66          temp = temp->next;
67      }
68      printf("\n");
69  }
70  void reverse(struct node *temp){
71      struct node *first=NULL,*second,*third;
72      second = temp;
73      while(second != NULL){
74          third = second->next;
75          second->next = first;
76          first = second;
77          second = third;
78      }
79      temp = first;
80      printf("The list after reversal is as follows:\n");
81      while(temp != NULL){
82          printf("%d\t",temp->data);
83          temp = temp->next;
84      }
85      printf("\n");
86  }
87  void concatenate(struct node *temp1,struct node *temp2){
88      if(temp1 == NULL && temp2 == NULL){
```

```
        printf("Lists are empty\n");
    }
    else if(temp1 != NULL && temp2 == NULL){
        while(temp1 != NULL){
            printf("%d\t",temp1->data);
            temp1 = temp1->next;
        }
        printf("\n");
    }
    else if(temp1 == NULL && temp2 != NULL){
        while(temp2 != NULL){
            printf("%d\t",temp2->data);
            temp2 = temp2->next;
        }
        printf("\n");
    }
    else{
        struct node *temp;
        temp = temp1;
        while(temp->next != NULL)
            temp = temp->next;
        temp->next = temp2;
        temp = temp1;
        printf("After concatenation:\n");
        while(temp != NULL){
            printf("%d\t",temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
}

int main(int argc,char **argv){
    int choice;
    struct node *head1 = NULL,*head2 = NULL;
    while(choice != 12){
        printf("Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: ");
        scanf("%d",&choice);
        switch(choice){
            case 1:insertend(&head1);break;
            case 2:insertend(&head2);break;
            case 3:deleteend(&head1);break;
            case 4:deleteend(&head2);break;
            case 5:display(head1);break;
```