7.
```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct node
 { int usem;
   struct node *next;
 };
struct node * head = NULL;
struct node * head2 =NULL
int c=0;
void insert()
 { struct node *newnode;
   struct node *temp;
   int u;
   printf("Enter data:");
   scanf("%d",&u);
   newnode = (struct node*) malloc (sizeof(
   struct node));
   newnode->usem =u;
   if (head==NULL)
    { newnode->next=NULL;
      head =newnode;
      c++;
    }
   else
    { temp= head
      while (temp->next!=NULL)
       { temp= temp->next;
       }
      temp->next = newnode;
      newnode->next =NULL;
      c++;
```

```
        printf("Node was created\n");
    }
}

void Insert2()
{ struct node *newnode;
  struct node *temp;
  int x, y;
  printf("Enter elements for list 2");
  do {
    printf("Enter data\n");
    scanf("%d", &x);
    newnode = (struct node*) malloc(sizeof
    (struct node));
    newnode->item = x;
    if (head2 == NULL)
      { newnode->next = NULL;
        head2 = newnode;
        c++;
      }
    else
      { temp = head2;
        while (temp->next != NULL)
          { temp = temp->next;
          }
        temp->next = newnode;
        newnode->next = NULL;
        c++;
        printf("Node created");
      }
    printf("Do you want to continue 1 to
      continue 0 to stop\n");
    scanf("%d", &y);
```

```
void bubbleSort()
{ int swapped, i;
  struct node *ptr1;
  struct node *ptr = NULL;
  if (head == NULL)
      return;
  do
  { swapped = 0;
    ptr1 = head;
    while (ptr1->next != ptr)
      {
        if (ptr1->sem > ptr1->next->sem)
        {
          int temp = ptr1->sem;
          ptr1->sem = ptr1->next->sem;
          ptr1->next->sem = temp;
          swapped = 1;
        }
        ptr1 = ptr1->next;
      }
    ptr = ptr1;
  }
  while (swapped);
}

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

```
void concat()
{ struct node *ptr;
  if (head2 == NULL)
    { head2 = head;
    }
  ptr = head;
  while (ptr -> next != NULL)
        ptr -> ptr -> next;
    ptr -> next = head2;
  }
void display1()
 {
   struct node *ptr;
   ptr = head;
   int i = 1;
   if (ptr == NULL)
       {
         printf("The linked list is empty");
       }
   else
     { while (ptr != NULL)
         { printf("%d", ptr -> item);
           i++;
           ptr = ptr -> next;
         }
     }
 }
void display2()
 { struct node *ptr;
   ptr = head2;
   int i = 1;
   if (ptr == NULL)
       { printf("Empty list"); }
```

```
else
{
    while (ptr1 != NULL)
    { printf("%d", ptr->item);
        i++;
        ptr = ptr->next;
    }}}
void main()
{ int choice, pos;
  do {
      printf("\n1. Insert node \n2. Sort \n3. Reverse
        \n4. Concat \n5. Exit\n");
      scanf("%d", &choice);
      switch(choice)
        { case 1:
          Insert();
          break;
          case 2;
          bubbleSort();
          display();
          break;
          case 3:
          Reverse();
          display();
          break;
          case 4:
          Insert2()
          concat();
          display();
          break;
          case 5:
          break;
```

```
default:
    printf("Wrong Choice");
    break;
  }
} while(choice != 5);
}
```