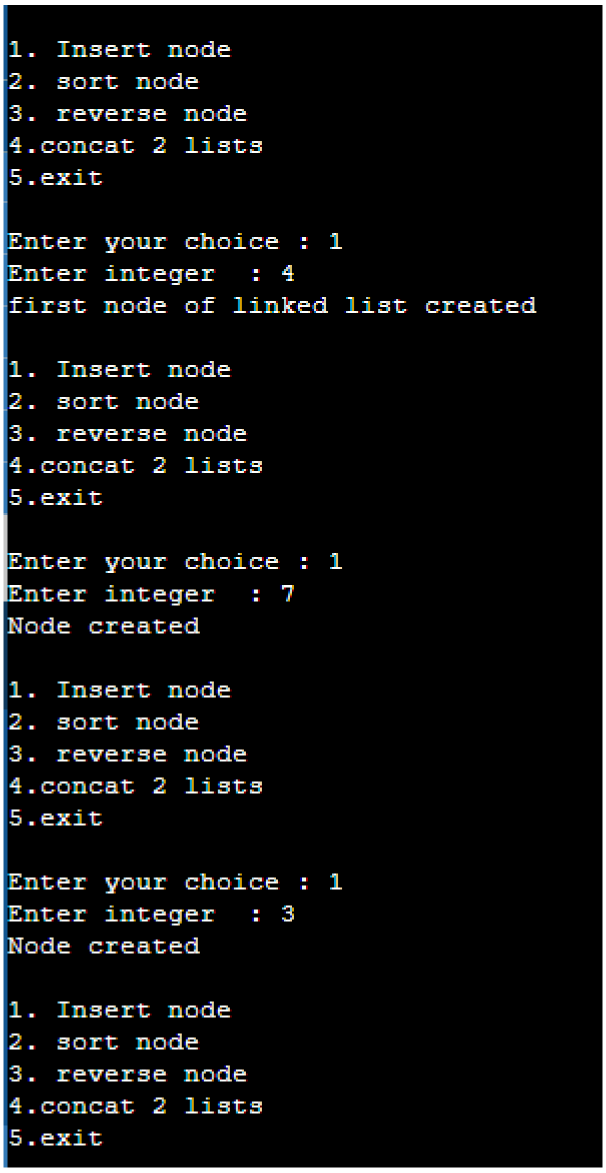
7. Write a program to implement Single Link List with following operations a) a) Sort the linked list. b) Reverse the linked list. c) Concatenation of two linked lists

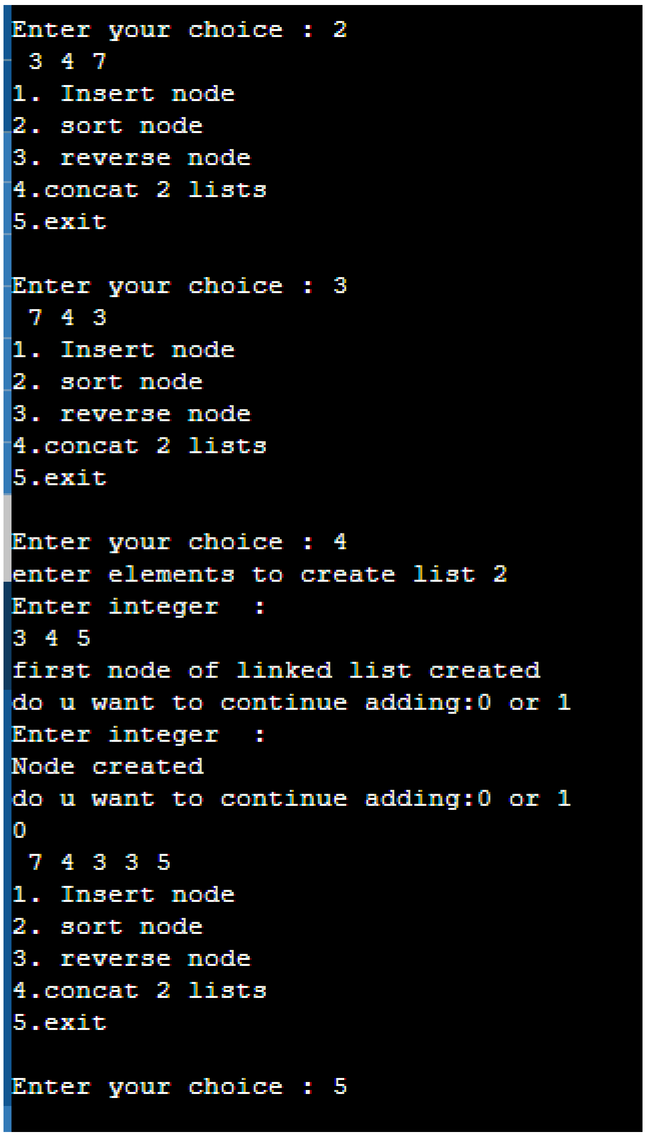
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct node
{
   int sem;
   struct node *next;
};
struct node *head = NULL;
struct node *head2 = NULL;
int c = 0;
void Insert()
{
   struct node *newnode;
   struct node *temp;
   int s;
   printf("Enter integer  : ");
   scanf("%d", &s);
   newnode = (struct node *)malloc(sizeof(struct node));
   newnode->sem = s;
   if (head == NULL)
   {
      newnode->next = NULL;
      head = newnode;
      printf("first node of linked list created\n");
      c++;
   }
   else
   {
      temp = head;
      while (temp->next != NULL)
      {
            temp = temp->next;
        }
        temp->next = newnode;
        newnode->next = NULL;
        c++;
        printf("Node created\n");
    }
}
void Insert2()
{
    struct node *newnode;
    struct node *temp;
    int s, y;
    printf("enter elements to create list 2\n");
    do
    {
        printf("Enter integer  : \n");
        scanf("%d", &s);
        newnode = (struct node *)malloc(sizeof(struct node));
        newnode->sem = s;
        if (head2 == NULL)
        {
            newnode->next = NULL;
            head2 = newnode;
            printf("first node of linked list created\n");
            c++;
        }
        else
        {
            temp = head2;
            while (temp->next != NULL)
            {
                temp = temp->next;
            }
            temp->next = newnode;
            newnode->next = NULL;
            c++;
            printf("Node created\n");
        }
        printf("do u want to continue adding:0 or 1\n");
        scanf("%d", &y);
    } while (y != 0);
}

void bubbleSort()
{
    int swapped, i;
    struct node *ptr1;
    struct node *lptr = NULL;

    if (head == NULL)
        return;

    do
    {
        swapped = 0;
        ptr1 = head;

        while (ptr1->next != lptr)
        {
            if (ptr1->sem > ptr1->next->sem)
            {
                int temp = ptr1->sem;
                ptr1->sem = ptr1->next->sem;
                ptr1->next->sem = temp;
                swapped = 1;
            }
            ptr1 = ptr1->next;
        }
        lptr = 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 (head == NULL)
    {
        head = head2;
    }
    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("Linked list is empty!\n");
   }
   else
   {
      while (ptr != NULL)
      {
         printf(" %d", ptr->sem);
         i++;
         ptr = ptr->next;
      }
   }
}
void display2()
{
   struct node *ptr;
   ptr = head2;
   int i = 1;

   if (ptr == NULL)
   {
      printf("Linked list is empty!\n");
   }
   else
   {
      while (ptr != NULL)
      {

         printf(" %d", ptr->sem);
```

```
            printf("\n");
            i++;
            ptr = ptr->next;
        }
    }
}

int main()
{
    int choice, pos;
    do
    {

        printf("\n1. Insert node \n2. sort node\n3. reverse
node\n4.concat 2 lists \n5.exit\n");
        printf("\nEnter your choice : ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            Insert();
            break;

        case 2:
            bubbleSort();
            display1();
            break;

        case 3:
            reverse();
            display1();
            break;

        case 4:
            Insert2();
            concat();
            display1();
            break;

        case 5:
            break;

        default:
            printf("Wrong choice!\n");
            break;
        }
    } while (choice != 5);
    return 0;
}
```
Output:



```
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 1
Enter integer  : 4
first node of linked list created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 1
Enter integer  : 7
Node created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 1
Enter integer  : 3
Node created

1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit
```



```
Enter your choice : 2
 3 4 7
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 3
 7 4 3
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 4
enter elements to create list 2
Enter integer  :
3 4 5
first node of linked list created
do u want to continue adding:0 or 1
Enter integer  :
Node created
do u want to continue adding:0 or 1
0
 7 4 3 3 5
1. Insert node
2. sort node
3. reverse node
4.concat 2 lists
5.exit

Enter your choice : 5
```