// Merge, Sort, Rev, Concat

Name: AMIT. R
USN: 1BM19CS016

lab7:
```
#include <stdlib.h>
#include <stdio.h>
#include <malloc.h>
struct node {
    int data;
    struct node *next;
};
struct node *start = NULL, *start1 = NULL;
struct node *create_ll (struct node *);
struct node *create_ll2 (struct node *);
struct node * display (struct node *);
struct node * reverse (struct node *);
struct node * concat (struct node *, struct node *)
struct node * sort_list (struct node *);
void main () {
    int option;
    do {
        printf ("1. Create 2. Display 3. Reverse 4. Concate 5. Sort
        printf ("\n Enter choice ");
        scanf ("%d", &option);
        switch (option) {
            case 1: start = create_ll (start);
            printf ("\n linked list created");
            break;
            case 2: start = display (start);
```

```
            break;
            case 3 : start = reverse (start);
                break;
            case 4 : start = concat (start, start1);
                break;
            case 5 : start = sort_list (start);
                break;
        } while (option != 15);
        return 0;
}

struct node *create_ll (struct node *start)
{
        struct node *ptr, *new_node;
        int num;
        printf ("\n Enter -1 to end ");
        printf ("\n Enter the data : ");
        scanf ("%d", &num);
        while (num != -1)
{       new_node = (struct node *) malloc (sizeof (struct node));
        new_node -> data = num;
        if (start == NULL)
{       new_node -> next = NULL;
            start = new_node;
        }
        else
{       ptr = start;
        while (ptr -> next != NULL)
            ptr = ptr -> next;
        ptr -> next = new_node;
        new_node -> next = NULL;
        }
        printf ("Enter the data");
```

```
    start1 = create12(start1);
    while (ptr->next != NULL)
        ptr = ptr->next;
        ptr->next = start1;
    return start;
}
struct node *sort_list (struct node *start)
{
    struct node *ptr1, *ptr2;
    int temp;
    ptr1 = start;
    while (ptr1->next != NULL) {
        ptr2 = ptr2->next;
        while (ptr21 = NULL) {
            if (ptr1->data > ptr2->data) {
                temp = ptr1->data;
                ptr1->data = temp;
            }
            ptr2 = ptr2->next;
        }
        ptr1 = ptr1->next;
    }
    return start;
}
```