Lab program (concate, reverse, sort)

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <process.h>
struct node
{
    int info;
    struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
    NODE x;
    x = (NODE) malloc (sizeof(struct node));
    if (x == NULL)
    {
        printf("ee full\n");
        exit(0);
    }
    return x;
}
NODE insert_rear(NODE first, int item)
{
    NODE temp, cur;
    temp = getnode();
    temp->info = item;
    temp->link = NULL;
    if(first == NULL)
        return temp;
    cur = first;
    while(cur->link != NULL)
        cur = cur->link;
    cur->link = temp;
    return first;
}
```

```
NODE delete_front(NODE first)
{
    NODE temp;
    if (first == NULL)
    {
        printf("ee list is empty cannot delete\n");
        return first;
    }
    temp = first;
    temp = temp->link;
    printf("ee item deleted at front end is
        = %d\n", first->info);
    free(first);
    return temp;
}
Void display (NODE first)
{
    NODE temp;
    if (first == NULL)
        printf("ee list empty\n");
    for (temp = first; temp != NULL; temp = temp->link)
    {
        printf("ee%d ", temp->info);
    }
    printf("ee\n");
}
NODE concat (NODE first, NODE second)
{
    NODE cur;
    if (first == NULL)
        return second;
    if (second == NULL)
        return first;
    cur = first;
```

```
while (cur->link != NULL)
        cur = cur->link;
    cur->link = second;
    return first;
}
NODE reverse(NODE first)
{
    NODE cur, temp;
    cur = NULL;
    while (first != NULL)
    {
        temp = first;
        first = first->link;
        temp->link = cur;
        cur = temp;
    }
    return cur;
}
NODE sortlist(NODE first){
    NODE current = first, index = NULL;
    int temp;
    if (first == NULL){
        printf("\n list is empty.");
        return current;
    }
    else{
        while (current != NULL){
            index = current->link;
            while (index != NULL){
                if (current->info > index->info){
                    temp = current->info;
                    current->info = index->info;
                    index->info = temp;
                }
                index = index->link;
            }
            current = current->link;
        }
    return current;
}
```

```
int main ()
{
    int item, choice, pos, i, n;
    NODE first = NULL, a, b;
    for(;;)
    {
    printf("\n 1. insert front  2. Concat  3. reverse
            4. order list  5. display  6. delete front
    printf("\n Enter the choice");          7. exit \n");
    scanf("%d", &choice);
    switch (choice)
    {
      Case 1: printf("\n enter the item: ");
              scanf("\n %d", &item);
              first = insert_rear (first, item);
              break;
      Case 2: printf("\n Enter the no of nodes in
                          the list: ");
              scanf("\n %d", &n);
              a = NULL;
              for (i=0; i<n; i++)
              {
                  printf("\n Enter the item: ");
                  scanf("\n %d", &item);
                  a = insert_rear (a, item);
              }
              first = concat (first, a);
              display (first);
              break;
      Case 3: first = reverse (first);
              display (first);
              break;
      Case 4: sort list (first);
              display (first);
              break;
```

```
        case 5: display(first);
                break;
        case 6: first = delete_front(first);
                break;
        default: exit(0);
    }
}
return 0;
}
```