07/12/2020

LAB PROGRAM - 7

NAME : SAIPRAVEEN MARNI
USN : 1BM19CS138

// Sort, Reverse and Concatenation of Linked List.

```
#include<stdio.h>
#include<malloc.h>
struct node {
    int num;
   struct node * next;
};
typedef struct node *NODE;
  NODE getNODE() {
NODE temp = (NODE) malloc (sizeof (struct node));
   if (temp == NULL) {
      return NULL
}
   return temp;
}
   void freeNode (NODE temp) {
      free temp;
}
```

```
NODE insertfront(NODE first){
    NODE temp;
  temp = getNODE();
   int num;
  scanf("%d", &num);
  temp->num = num;
  temp->next = NULL;
   if (first == NULL){
      return temp;
}
  temp->next = first;
     first = temp;
     return first;
}
  NODE deleteFront(NODE first){
        NODE temp;
    if (first == NULL){
       return NULL;
    }
     if (first->next == NULL){
  Printf("Deleted element = %d \n", first->num);
      freeNode(first);
      return NULL;
  }
     temp = first;
    temp = temp->next;
    Printf("Deleted elements = %d \n", first->num);
     freeNode(first);
      return temp;
  }
     NODE sort(NODE first){
      NODE cur, temp;
     if (first == NULL){
        return NULL;
  }
```

```
cur = first ;
while (cur != NULL) {
  temp = cur -> next ;
  while (temp != NULL) {
  if (temp -> num < cur -> num) {
      int num = cur -> num ;
      cur -> num = temp -> num ;
      temp -> num = num ;
  }
      temp = temp -> next ;
  } cur = cur -> next ;
  }
    return first ;
}
  void display (NODE first) {
      NODE cur ;
    if (first == NULL) {
    printf ("List is empty\n");
      return ;
  }
    cur = first ;
  while (cur != NULL) {
    printf ("%d", cur -> num);
    cur = cur -> next ;
  }
    printf ("\n");
  }
      NODE reverse (NODE first) {
      NODE cur = null ;
      NODE temp = getNode() ;
       while(first != NULL) { temp = first ;
        first = first -> next ;
        temp -> next = cur ;
         cur = temp ;
    || printf ("%d ", first -> num) ;
  }
```

```
    return temp;
}
NODE concat (NODE first){
 NODE sec = NULL;
 int chq;
while(1){
Printf("Enter the choice: \n 1-insertFront \t 2- deleteFront \t
 3-display \t 4- concat \n");
 scanf("%d", &chq);
  if(chq==4){
  break;
 }
  switch(chq){
 case 1:
   sec = insertFront (sec);
     break;
 case 2:
   sec = deleteFront (sec);
    break;
 case 3:
    display (sec);
    break;
 }
}
      NODE cur;
      if(first==NULL){
        return sec;
    }
      if(sec==NULL){
        return first;
    }
      cur = first;
    while(cur->next!=NULL){
       cur = cur->next;
    }
      cur->next = sec;
   }    return first;
```

```
int main() {
 int chq;
 NODE first = NULL;
 while(1) {
Printf("Enter the choice :\n 1- inseetFront \t 2- deleteFront \t
    3- display \t 4- sort \t 5- reverse \t 6- concat \t
     7- exit \n");
    scanf("%d", &chq);
    switch (chq) {
    case 1:
    first = inseetFront (first);
        break;
   case 2:
    first = delete Front (first);
        break;
   case 3:
     display (first);
        break;
   case 4:
     first = sort (first);
        break;
   case 5:
     first = reverse (first);
        break;
   case 6:
  Printf("Creating the second list for concat. \n");
        concat (first);
        break;
   case 7:
        return 0;
    }
  }
}
```