A program with 9 different operations on SLL (Singly Linked List), input 0 at run time to terminate.

```
#include<stdio.h>
#include<malloc.h>
#include<conio.h>
struct NodeType
{
int info;
struct NodeType *next;
};
void createList(struct NodeType **);
void displayList(struct NodeType *);
void removeList(struct NodeType **);
void insertBeg(struct NodeType **,int);
void delBeg(struct NodeType **);
void insertEnd(struct NodeType **,int);
void delEnd(struct NodeType **);
void count(struct NodeType *);
void search(struct NodeType **,int);
void createList(struct NodeType **list)
{
struct NodeType *toadd;
int no,i,data;
printf("\nHow many node in the list you want to add:-\t");
scanf("%i",&no);
for(i=1;i<=no;i++)
{
toadd=(struct NodeType*)malloc(sizeof(struct NodeType));
if(toadd==(struct NodeType*)NULL)
{
```

```
printf("\nNo node created");
return;
}
printf("\nEnter data for node information:-\t");
scanf("%i",&data);
toadd->info=data;
toadd->next=(struct NodeType*)*list;
(*list)=(struct NodeType*)toadd;
}
}
void displayList(struct NodeType *list)
{
struct NodeType *todisp;
todisp=list;
while(todisp!=(struct NodeType*)NULL)
{
printf("%d\t",todisp->info);
todisp=todisp->next;
}
}
void removeList(struct NodeType **list)
{
struct NodeType *todel;
todel=(*list);
while(todel!=(struct NodeType*)NULL)
{
printf("\n%d is deleted now",todel->info);
(*list)=todel->next;
free(todel);
```

```
todel=(*list);
}
}
void insertBeg(struct NodeType **list,int data)
{
struct NodeType *toadd=(struct NodeType *)malloc(sizeof(struct NodeType));
if(toadd==(struct NodeType *)NULL)
{
printf("\nMemory allocation error!");
return;
}
toadd->info=data;
toadd->next=(*list);
(*list)=toadd;
}
void delBeg(struct NodeType **list)
{
struct NodeType *todel=(*list);
if(todel==(struct NodeType *)NULL)
{
printf("\nList is empty");
return;
}
(*list)=todel->next;
printf("\n%d is being deleted",todel->info);
free(todel);
}
void insertEnd(struct NodeType **list, int data)
{
```

```
struct NodeType *toadd,*foradd=(*list);
toadd=(struct NodeType*)malloc(sizeof(struct NodeType));
if(toadd==(struct NodeType*)NULL)
{
printf("\nMemory allocation error!");
return;
}
toadd->info=data;
if((*list)==(struct NodeType*)NULL)
{
toadd->next=(*list);
(*list)=toadd;
}
else
{
while(foradd->next!=(struct NodeType*)NULL)
foradd=foradd->next;
toadd->next=(struct NodeType*)foradd->next;
foradd->next=toadd;
}
}
void delEnd(struct NodeType **list)
{
struct NodeType *todel,*fordel=(*list);
if(fordel==(struct NodeType *)NULL)
{
printf("\nList is empty already");
return;
}
```

```
while((fordel->next)->next!=(struct NodeType *)NULL)
fordel=fordel->next;
if(fordel->next==(struct NodeType *)NULL)
{
printf("\n%d is being deleted",fordel->info);
(*list)=(struct NodeType *)NULL;
return;
}
todel=fordel->next;
fordel->next=(struct NodeType *)NULL;
printf("\n%d is being deleted",todel->info);
free(todel);
}
void count(struct NodeType *list)
{
int no=0;
while(list!=(struct NodeType *) NULL)
{
no++;
list=list->next;
}
printf("\nTotal number of nodes=%d",no);
}
void search(struct NodeType *list,int elem)
{
int found=0;
while(list!=(struct NodeType*)NULL)
{
if(list->info==elem)
```

```
{
found=1;
break;
}
list=list->next;
}
if(found==1)
printf("\n%d found in list",elem);
else
printf("\n%d not found in list",elem);
}
void main()
{
struct NodeType *SLL;
int ch,data;
clrscr();
SLL=(struct NodeType *)NULL;
do
{
printf("\nFollowing operations can be done on Singly Linked List:\t");
printf("\n1. Create list by adding element at list top");
printf("\n2. Insert element at list top");
printf("\n3. Insert element at list end");
printf("\n4. Delete element from list top");
printf("\n5. Delete element from list end");
printf("\n6. Remove whole list");
printf("\n7. Display entire list");
printf("\n8. Count list element");
printf("\n9. Search an element in list");
```

```
printf("\n0. Exit from program");
printf("\nEnter your choice:\t");
scanf("%i",&ch);
switch(ch)
{
case 1: createList(&SLL);
        break;
case 2: printf("\nEnter the data to insert at top of SLL:\t");
        scanf("%i",&data);
        insertBeg(&SLL,data);
        break;
case 3: printf("\nEnter the data to insert at end of SLL:\t");
        scanf("%i",&data);
        insertEnd(&SLL,data);
        break;
case 4: printf("\nRemoving list element from SLL end");
        delBeg(&SLL);
        break;
case 5: printf("\nRemoving list element from SLL top");
        delEnd(&SLL);
        break;
case 6: removeList(&SLL);
        break;
case 7: displayList(SLL);
        break;
case 8: printf("\nCounting the number of nodes present in the SLL");
        count(SLL);
        break;
case 9: printf("\nEnter the data to search in the list:\t");
```

```
            scanf("%i",&data);
            search(SLL,data);
            break;
case 0: removeList(&SLL);
            return;
default: printf("\nInvalid option input, try again!");
            break;
}
printf("\nPress any key to get menu option");
getch();
clrscr();
}while(ch!=0);
}
```