**Assignment No: 1.6**

**Title: Implementation of program based on Single Linked List.**

-------------------------------------------------------------------------

#include<iostream.h>

#include<conio.h>

class NODE

{

public:

int data ,ele;

NODE \*next;

};

class LIST\_

{

private:

NODE \*start;

NODE \*ptr1;

NODE \*ptr2;

public:

void ADD\_FIRST\_(int ele);

int DEL\_FIRST\_();

void ADD\_END\_(int ele);

int DEL\_END\_();

void ADD\_POS\_(int ele,int pos);

int DEL\_POS\_(int pos);

void LIST\_ALL\_();

LIST\_();

};

LIST\_::LIST\_()

{

start=NULL;

}

void LIST\_::ADD\_FIRST\_(int ele)

{

NODE \*NEW=new NODE();

if(NEW==NULL)

{

cout<<"\n List is Full ";

}

else

{

NEW->data=ele;

NEW->next=NULL;

NEW->next=start;

st

art=NEW;

}

}

int LIST\_::DEL\_FIRST\_()

{

if(start==NULL)

{

cout<<"\n List is Empty";

return NULL;

}

else

{

int ele=start->data;

NODE \*TEMP=start;

start=start->next;

delete TEMP;

return ele;

}

}

void LIST\_::ADD\_END\_(int ele)

{

NODE \*NEW=new NODE();

NEW->data=ele;

NEW->next=NULL;

if(NEW==NULL)

{

cout<<"\n List is Full ";

return;

}

if(start==NULL)

{

start=NEW;

}

else

{

NODE \*ptr=start;

while(ptr->next!=NULL)

{

ptr=ptr->next;

}

ptr->next=NEW;

}

}

int LIST\_::DEL\_END\_()

{

if(start==NULL)

{

cout<<"\n List is Empty";

return NULL;

}

else

{

NODE \*ptr1=start;

NODE \*ptr2=NULL;

while(ptr1->next!=NULL)

{

ptr2=ptr1;

ptr1=ptr1->next;

}

int ele=ptr1->data;

NODE \*TEMP=ptr1;

if(ptr2!=NULL)

ptr2->next=NULL;

else

start=NULL;

delete TEMP;

return ele;

}

}

void LIST\_::ADD\_POS\_(int ele,int pos)

{

NODE \*NEW=new NODE();

NEW->data=ele;

NEW->next=NULL;

if(pos==1)

{

NEW->next=start;

start=NEW;

}

else

{

NODE \*ptr1=start;

NODE \*ptr2=NULL;

int count=1;

while(count<pos)

{

ptr2=ptr1;

ptr1=ptr1->next;

count=count+1;

}

NEW->next=ptr1;

ptr2->next=NEW;

}

}

int LIST\_::DEL\_POS\_(int pos)

{

int ele;

NODE \*TEMP;

if(start==NULL)

{

cout<<"\n List is Empty";

return NULL;

}

else

{

if(pos==1)

{

ele=start->data;

TEMP=start;

start=start->next;

}

else

{

NODE \*ptr1=start;

NODE \*ptr2=NULL;

int count=1;

while(count<pos)

{

ptr2=ptr1;

ptr1=ptr1->next;

count=count+1;

}

ele=ptr1->data;

TEMP=ptr1;

ptr2->next=ptr1->next;

}

delete TEMP;

return ele;

}

}

void LIST\_::LIST\_ALL\_()

{

if(start==NULL)

{

cout<<"\n List is Empty";

}

else

{

NODE \*ptr;

ptr=start;

cout<<endl<<" ";

while(ptr!=NULL)

{

cout<<ptr->data<<" ";

ptr=ptr->next;

}

}

}

void MENU()

{

int opt,ele,pos;

LIST\_ obj;

do

{

cout<<"\n 1.ADD\_FIRST ";

cout<<"\n 2.DEL\_FIRST";

cout<<"\n 3.ADD\_END";

cout<<"\n 4.DEL\_END";

cout<<"\n 5.ADD\_POS";

cout<<"\n 6.DEL\_POS";

cout<<"\n 7.LIST\_ALL";

cout<<"\n 8.EXIT ";

cout<<"\n -------------------------";

cout<<"\n Eneter Your Choice =>";

cin>>opt;

switch(opt)

{

case 1:

cout<<"\n ADD Element to List at First => ";

cin>>ele;

obj.ADD\_FIRST\_(ele);

break;

case 2:

ele=obj.DEL\_FIRST\_();

if(ele!=NULL)

cout<<endl<<ele<<" is Deleted at First ";

break;

case 3:

cout<<"\n Add Element to list at Last=> ";

cin>>ele;

obj.ADD\_END\_(ele);

break;

case 4:

ele=obj.DEL\_END\_();

if(ele!=NULL)

cout<<endl<<ele<<"is Deleted at End ";

break;

case 5:

cout<<"\n Enter Element=>";

cin>>ele;

cout<<"\n Enter Position=>";

cin>>pos;

obj.ADD\_POS\_(ele,pos);

break;

case 6:

cout<<"\n Enter Position for deletion ";

cin>>pos;

ele=obj.DEL\_POS\_(pos);

if(ele!=NULL)

cout<<endl<<ele<<"is Deleted at Position";

break;

case 7:

obj.LIST\_ALL\_();

break;

case 8:

return;

default:

cout<<"\n Invalid choice";

break;

}

}while(1);

}

void main()

{

clrscr();

MENU();

getch();

}