#include<stdio.h>

#include<stdlib.h>

struct node

{

int value;

struct node \*next,\*prev;

};

typedef struct node node1;

void inb();

void ine();

void inbw();

void delb();

void dele();

void delebw();

void disply();

void displyback();

void search();

node1 \*nptr;

node1 \*start=NULL;

node1 \*create();

void main()

{

int ch;

do

{

printf("\n\n\tlinked list implimentation\n\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\t1.insert at the bignning\n\t2.insert at end\n\t3.insert in between\n\t4.delete from bigginning\n\t5.delete from end\n\t6.delete in between\n\t7.display\n\t8.display backward\n\t9.search\n\t10.exit\n\n\tenter your choice: ");

scanf("%d",&ch) ;

switch(ch)

{

case 1:inb();

break;

case 2:ine();

break;

case 3:inbw();

break;

case 4:delb();

break;

case 5:dele();

break;

case 6:delebw();

break;

case 7:disply();

break;

case 8:displyback();

break;

case 9:search();

break;

case 10:exit(0);

break;

default:printf("\ninvalid choice");

}

}while(ch!=10);

}

node1 \*create()

{

node1\*nptr=(node1\*)malloc(sizeof(node1));

if(nptr==NULL)

{

printf("memory overflow");

return 0;

}

else

return nptr;

}

void inb()

{

int val;

node1\*nptr=create();

if(nptr==NULL)

printf("memory overflow");

else

{

printf("enter element:");

scanf("%d",&val);

nptr->value=val;

if(start==NULL)

{

start=nptr;

nptr->next=NULL;

nptr->prev=NULL;

}

else

{

nptr->next=start;

nptr->prev=NULL;

start->prev=nptr;

start=nptr;

}

}

printf("Value inserted");

}

void ine()

{

node1 \*nptr=create();

node1 \*temp=start;

node1 \*ptr;

int val;

if(nptr==NULL)

printf("memory overflow");

else

{

printf("enter element:");

scanf("%d",&val);

nptr->value=val;

if(temp==NULL)

{

start=nptr;

nptr->next=NULL;

nptr->prev=NULL;

}

else

{

while(temp!=NULL)

{

ptr=temp;

temp=temp->next;

}

ptr->next=nptr;

nptr->next=NULL;

nptr->prev=ptr;

}

}

printf("Value inserted");

}

void disply()

{

node1 \*temp;

temp=start;

if(temp==NULL)

printf("list empty");

else

{

printf("elements are:");

while(temp!=NULL)

{

printf("%d\t",temp->value);

temp=temp->next;

}

}

}

void displyback()

{

node1 \*temp;

temp=start;

if(temp==NULL)

printf("list empty");

else

{

printf("elements are:");

while(temp->next!=NULL)

{

temp=temp->next;

}

while(temp!=NULL)

{

printf("\t%d",temp->value);

temp=temp->prev;

}

}

}

void inbw()

{

int val,pos,i=1;

node1 \*nptr=create();

node1 \*temp=start;

node1 \*ptr;

printf("enter the element to be inserted:");

scanf("%d",&val);

nptr->value=val;

nptr->next=NULL;

printf("enter the position of element to be inserted:");

scanf("%d",&pos);

if(temp==NULL)

{

start=nptr;

nptr->next=NULL;

nptr->prev=NULL;

}

else

{

if(pos==1)

{

nptr->next=start;

start->prev=nptr;

start=nptr;

}

else

{

while(temp!=NULL)

{

if(i==pos)

{

nptr->next=ptr->next;

ptr->next->prev=nptr;

nptr->prev=ptr;

ptr->next=nptr;

}

ptr=temp;

temp=temp->next;

i++;

}

}

}

}

void delb()

{

node1\*nptr;

if(start==NULL)

printf("list empty");

else{

nptr=start;

printf("value deleted is %d",nptr->value);

start=nptr->next;

nptr->next->prev=NULL;

free(nptr);

}

}

void dele()

{

node1\*nptr,\*temp,\*prt;

if(start==NULL){

printf("list empty");

}

else{

nptr=start;

while(nptr->next!=NULL)

{

temp=nptr;

nptr=nptr->next;

}

printf("Value deleted is %d",nptr->value);

temp->next=NULL;

free(nptr);

}

}

void delebw()

{

node1\*temp,\*nptr;

int i=1,pos;

if(start==NULL)

printf("list empty");

else{

printf("enter the position:");

scanf(" %d",&pos);

temp=start;

nptr=start;

if(pos==1)

{

start=temp->next;

printf("element deleted is %d",temp->value);

temp->next->prev=NULL;

free(temp);

}

else

{

while(temp->next!=NULL)

{

if(i==pos)

{

nptr->next=temp->next;

printf("Value deleted is:%d",temp->value);

temp->next->prev=nptr;

free(temp);

}

nptr=temp;

temp=temp->next;

i++;

}

}

}

}

void search(){

int val,i=1;

node1 \*temp;

temp=start;

if(temp==NULL)

{

printf("list is empty");

}

else

{

printf("enter the element to be searched:");

scanf("%d",&val);

}

while(temp!=NULL)

{

if(temp->value==val)

{

printf("element is found at position:%d",i);

}

temp=temp->next;

i++;

}

}













