#include<iostream>

using namespace std;

typedef struct Node

{

int data;

struct Node \*next,\*prev;

}\*nodeptr;

nodeptr head=NULL,tail=NULL;

void insert(int n)

{

int value;

for(int i=0;i<n;i++)

{

cin>>value;

nodeptr newnode= new Node;

newnode->data=value;

newnode->next=NULL;

newnode->prev=NULL;

if(head==NULL&&tail==NULL)

{

head=newnode;

tail=newnode;

}

else

{

tail->next=newnode;

newnode->prev=tail;

}

tail=newnode;

}

}

void insertAtFirst(int value)

{

//write your code here

nodeptr newNode;

if(head == NULL)

{

cout<<"Error! List is Empty!"<<endl;

}

else

{

newNode = new Node;

newNode->data = value;

newNode->next = head;

newNode->prev = NULL;

head->prev = newNode;

head = newNode;

cout<<"Node inserter successfully at beginning"<<endl;

}

}

void insertAtLast(int value)

{

//write your code here

nodeptr newNode;

if(tail == NULL)

{

cout<<"Error! List is emply!"<<endl;

}

else

{

newNode = new Node;

newNode->data = value;

newNode->next = NULL;

newNode->prev = tail;

tail->next = newNode;

tail = newNode;

cout<<"Node is inserted successfully"<<endl;

}

}

void insertAtK(int value,int p)

{

//write your code here

int i;

nodeptr newNode,temp;

if(head == NULL)

{

cout<<"Error! list is empty!"<<endl;

}

else

{

temp = head;

i = 1;

while(i<p-1 && temp!=NULL)

{

temp = temp->next;

i++;

}

if(p==1)

{

insertAtFirst(value);

}

else if(temp==tail)

{

insertAtLast(value);

}

else if(temp!=NULL)

{

newNode = new Node;

newNode->data = value;

newNode->next = temp->next;

newNode->prev = temp;

if(temp->next!=NULL)

{

temp->next->prev = newNode;

}

temp->next = newNode;

cout<<"Node is inserted successfully!"<<endl;

}

else

{

cout<<"Error! "<<endl;

}

}

}

void deletFirst()

{

//write your code here

nodeptr deleteNode;

if(head==NULL)

{

cout<<"Unable to delete! List is empty!"<<endl;

}

else

{

deleteNode = head;

head = head->next;

if(head!=NULL)

{

head->prev =NULL;

}

delete(deleteNode);

cout<<"Successfully deleted node from begining!"<<endl;

}

}

void deletLast()

{

//write your code here

nodeptr deleteNode;

if(tail==NULL)

{

cout<<"Unable to delete! list is empty!"<<endl;

}

else

{

deleteNode = tail;

tail = tail->prev;

if(tail!=NULL)

{

tail->next = NULL;

}

delete(deleteNode);

cout<<"Successfully deleted node form end!"<<endl;

}

}

void deletAtK(int p)

{

//write your code here

nodeptr current;

int i;

current = head;

for(i=1;i<p && current!=NULL;i++)

{

current = current->next;

}

if(p==1)

{

deletFirst();

}

else if(current == tail)

{

deletLast();

}

else if(current!=NULL)

{

current->prev->next = current->next;

current->next->prev = current->prev;

delete(current);

cout<<"Successfully deleted node from "<<p<<" position!"<<endl;

}

else

{

cout<<"Invalid position!"<<endl;

}

}

int Search(int data)

{

//write your code here

int pos = 0;

if(head==NULL)

{

cout<<"Linked List not initialized"<<endl;

return 0;

}

nodeptr current = head;

while(current!=NULL)

{

pos++;

if(current->data == data)

{

return 1;

}

if(current->next != NULL)

current = current->next;

else

break;

}

return 0;

}

void displayF()

{

nodeptr ptr=head;

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

cout<<endl;

}

void displayB()

{

nodeptr ptr=tail;

while(ptr!=NULL)

{

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

ptr=ptr->prev;

}

cout<<endl;

}

int main()

{

int n;

cout<<"Enter number of nodes : ";

cin>>n;

insert(n);

cout<<endl;

cout<<"1. Insert at first "<<endl;

cout<<"2. Insert at last "<<endl;

cout<<"3. Insert at k position "<<endl;

cout<<"4. Delete at first"<<endl;

cout<<"5. Delete at last"<<endl;

cout<<"6. Delete at k position"<<endl;

cout<<"7. Forward Display "<<endl;

cout<<"8. Backward Display "<<endl;

cout<<"9. Search "<<endl;

while(1)

{

int query;

cout<<"Choose a menu: "<<endl;

cin>>query;

if(query==1)

{

int data;

cout<<"Provide a value to insert at first: "<<endl;

cin>>data;

insertAtFirst(data);

}

else if(query==2)

{

int data;

cout<<"Provide a value to insert at last: "<<endl;

cin>>data;

insertAtLast(data);

}

else if(query==3)

{

int data,k;

cout<<"Provide a value to insert at k position: "<<endl;

cin>>data>>k;

insertAtK(data,k);

}

else if(query==4)

{

cout<<"Deleting value at first: "<<endl;

deletFirst();

}

else if(query==5)

{

cout<<"Deleting value at last: "<<endl;

deletLast();

}

else if(query==6)

{

int k;

cout<<"Provide a position to delete at k position: "<<endl;

cin>>k;

deletAtK(k);

}

else if(query==7)

{

cout<<"The list is in forward order: "<<endl;

displayF();

}

else if(query==8)

{

cout<<"The list is in backward order: "<<endl;

displayB();

}

else if(query==9)

{

int data;

cout<<"Provide a value to search: "<<endl;

cin>>data;

int status = Search(data);

if(status)

{

cout<<"Element Found\n"<<endl;

}

else

{

cout<<"Element Not Found\n"<<endl;

}

}

}

return 0;

}