circular\_07.cpp

**Compile:** g++ circular\_07.cpp -o circular\_07

**Run:** ./circular\_07

**Program:**

#include<iostream>

#include<stdlib.h>

using namespace std;

int data2,data1,ch;

struct node

{

int data;

struct node \*next;

};

struct node \*list=NULL,\*p,\*q,\*r,\*s;

class circular

{

public:

void menu()

{

do

{

cout<<endl<<"1. Insert at the begin \n 2.Insert at the end \n 3.Insert before a particular element \n 4. Insert after a particular element \n 5. Display \n 6. Exit 7. Delete from beginning \n 8. Delete at the end \n 9.Delete a particular node \n 10. Search \n 11. Sort \n 12. Count \n 13. Reverse";

cin>>ch;

switch(ch)

{

case 1:

insertb();

break;

case 2:

inserte();

break;

case 3:

insertbp();

break;

case 4:

insertap();

break;

case 5:

display();

break;

case 6:

break;

case 7:

deleteb();

break;

case 8:

deletee();

break;

case 9:

deletep();

break;

case 10:

search();

break;

case 11:

sort();

break;

case 12:

count(); break;

case 13:

reverse();

break;

default:

cout<<endl<<"enter proper choice :( :) :p :D :o ;) :\* B-)";

break;

}

}while(ch!=6);

}

void reverse()

{

if(list==NULL)

{

cout<<endl<<"List is empty";

}

else

{

struct node \*temp=NULL;

q=s=list;

r=q->next;

while(r!=list)

{

temp=q;

q=r;

r=q->next;

q->next=temp;

}

list=q;

r->next=list;

}

}

void sort()

{

if(list==NULL)

{

cout<<endl<<"List is empty";

}

else

{

q=list;

while(q->next!=list)

{

r=q->next;

while(r!=list)

{

if(q->data > r->data)

{

int temp;

temp=q->data;

q->data=r->data;

r->data=temp;

}

r=r->next;

}

q=q->next;

}

}

}

void count()

{

if(list==NULL)

{

cout<<endl<<"list is empty :(";

}

else

{

q=list;

inti=0;

while(q->next!=list)

{

q=q->next;

i++;

}

cout<<endl<<"number of element "<<i+1;

}

}

void search()

{

if(list==NULL)

{

cout<<endl<<"list is empty :(";

}

else

{

cout<<endl<<"Enter key value";

cin>>data1;

q=list;

inti=0;

while(q->data != data1 && q->next!=list)

{

q=q->next;

i++;

}

if(q->data==data1)

{

cout<<endl<<"Element "<<q->data<<" found at location "<<i;

}

else

{

cout<<endl<<"element not found";

}

}

}

voiddeletep()

{

if(list==NULL)

{

cout<<endl<<"List is empty.. Cant delete an element";

}

else

{

q=r=list;

cout<<endl<<"Enter the element to be deleted";

cin>>data1;

if (list->next==list && list->data==data1)

{

cout<<list->data;

list=NULL;

}

else if(list->data==data1)

{

do

{

r=r->next;

}while(r->next!=list);

cout<< list->data;

list=list->next;

r->next=list;

free(q);

}

else

{

do

{

r=q;

q=q->next;

}while(q->data!=data1 && q!=list);

if(q->data==data1)

{

r->next=q->next;

cout<< q->data;

free(q);

}

else

{

cout<<endl<<"Element not found";

}

}

}

}

voiddeletee()

{

if(list==NULL)

{

cout<<endl<<"List is empty.. Cant delete an element";

}

else

{

q=list;

if(list->next==list)

{

cout<<endl<< list->data;

list=NULL;

}

else

{

do

{

r=q;

q=q->next;

}while(q->next!=list);

r->next=list;

cout<< q->data;

free(q);

}

}

}

voiddeleteb()

{

if(list==NULL)

{

cout<<endl<<"List is empty.. Cant delete an element";

}

else

{

q=list;

if(list->next==list)

{

cout<<endl<< list->data;

list=NULL;

}

else

{

do

{

q=q->next;

}while(q->next!=list);

//free(q->next->data);

q->next=list->next;

cout<<endl<< list->data;

//list=list->next;

free(list);

list=q->next;

}

}

}

voidinserte()

{

p=(struct node\*)malloc(sizeof(node));

cout<<endl<<"Enter an element";

cin>>data1;

p->data=data1;

if(list==NULL)

{

list=p;

list->next=list;

}

else

{

q=list;

while(q->next!=list)

{

q=q->next;

}

q->next=p;

p->next=list;

}

}

voidinsertb()

{

p=(struct node\*)malloc(sizeof(node));

cout<<endl<<"Enter an element";

cin>>data1;

p->data=data1;

if(list==NULL)

{

list=p;

list->next=list;

}

else

{

p->next=list;

q=list;

while(q->next!=list)

{

q=q->next;

}

q->next=p;

list=p;

}

}

voidinsertbp()

{

if(list==NULL)

{

cout<<endl<<"List is empty";

}

else

{

cout<<endl<<"Enter the element before which you want to insert new element";

cin>>data2;

cout<<endl<<"Enter the element to be inserted";

cin>>data1;

p=(struct node\*)malloc(sizeof(node));

p->data=data1;

q=list;

if(q->data==data2)

{

p->next=list;

r=list;

while(r->next!=list)

{

r=r->next;

}

r->next=p;

list=p;

}

else

{

do

{

r=q;

q=q->next;

}while(q->data!=data2 && q!=list);

if(q->data == data2)

{

r->next=p;

p->next=q;

}

else

{

cout<<endl<<"Data not found :/ :(";

}

}

}

}

voidinsertap()

{

if(list==NULL)

{

cout<<endl<<"List is empty";

}

else

{

cout<<endl<<"Enter the element after which you want to insert new element";

cin>>data2;

cout<<endl<<"Enter the element to be inserted";

cin>>data1;

p=(struct node\*)malloc(sizeof(node));

p->data=data1;

q=list;

// q=q->next;

do

{

q=q->next;

}while(q->data!=data2 && q!=list);

if(q->data==data2)

{

r=q->next;

p->next=r;

q->next=p;

}

else

{

cout<<endl<<"Data not found";

}

}

}

void display()

{

if(list==NULL)

{

cout<<endl<<"List is empty";

}

else

{

q=list;

do

{

cout<<endl<< q->data<<"====>";

q=q->next;

}while(q!=list);

}

}

};

int main()

{

circular c;

c.menu();

return 0;

}

**Output:**

