4.3. A menu driven program that implements Singly circular linked list for the following operations: - Insert, Display, Count, Delete, Search.

#include<iostream> #include<conio.h> using namespace std; class node

{

private:

int data;

node \*address; public:

void add(int); void display(void); int count(void);

void sort(void);

void insert(int,int); void remove(int); void search(int);

};

node \*p;

void node::add(int num)

{

node \*q=p;

if(p==NULL)

{

}

else

{

}

}

p=new node; p->data=num; p->address=p;

while(q->address != p)

{

q=q->address;

}

q->address=new node; q->address->data=num; q->address->address=p;

void node::display(void)

{

node \*q=p; if(p==NULL)

{

cout<<"No Circular link list/n";

}

else

{

do{

}

cout<<q->data; q=q->address;

while(q!= p);

}

}

int node::count(void)

{

node \*q=p; int i=0; if(p==NULL)

{

}

else

{

}

}

do{

cout<<"NO CIRCULAR LINKEDLIST EXISTS";

i=i++;

q=q->address;

}while(q!=p); return i;

void node:: sort(void)

{

node \*i; node \*j; int temp;

if (p == NULL)

{

}

else

{

cout << "NO CIRCULAR LINKEDLIST EXISTS";

i = p; do

{

j = i -> address; while ( j !=p)

{

if(i->data>j->data)

{

temp=i->data;

i->data=j->data; j->data=temp;

}

j = j -> address;

}

i = i -> address;

}

while (i -> address != p);

}

}

void node:: insert(int pos,int num)

{

node\*temp; node \*q=p; int i; if(pos==1)

{

p=new node; p->data=num; p->address=q; return;

}

if(pos==1+count())

{

add(num); return;

}

for(i=1;i<=(pos-2);i++)

{

q=q->address;

}

temp=q->address;

q->address=new node; q->address->data=num;

q->address->address=temp;

}

void node::remove(int pos)

{

int i; node\*q = p; node\*temp;

if(pos==1)

{

p=q->address; delete (q); return;

}

for(i=1;i<=(pos-2);i++)

{

q=q->address;

}

temp=q->address;

q->address=q->address->address; delete(temp);

}

void node:: search(int num)

{

node\*q= p; int flag =0;

int pos =0;

do{

}

if(q->data == num){ flag =1; break;

pos++;

q=q->address;

}while(q!=p);

if(flag ==1)

{

}

else

{

}

}

cout<<"Number found at position"<<pos;

cout<<"number not found";

int main(void)

{

int num,pos,option; node n;

p=NULL;

char ch='y'; while(ch=='y')

{

cout<<"\n 1.Add"; cout<<"\n 2.Display"; cout<<"\n 3.Count"; cout<<"\n 4.Sort"; cout<<"\n 5.Insert"; cout<<"\n 6.Remove"; cout<<"\n 7.Search"; cout<<"\nEnter an option: ";

cin>>option; switch(option)

{

case 1:

{

}

case 2:

{

}

case 3:

{

cout<<"\nEnter Number:"; cin>>num;

n.add(num);

cout<<"\nDo you want to Continue"; break;

n.display();

cout<<"\nDo you want to Continue"; break;

num=n.count(); cout<<"\ncount :"<<num;

cout<<"\nDo you want to continue"; break;

}

case 4:

{

n.sort(); cout<<"\nSorted"; n.display();

cout<<"\n Do you want to continue "; break;

}

case 5:

{

}

case 6:

{

}

cout<<"\nENter position: "; cin>>pos;

cout<<"\nEnter the Number: "; cin>>num;

n.insert(pos,num);

cout<<"\n Do you want to continue "; break;

cout<<"\nEnter Position: "; cin>>pos;

n.remove(pos);

cout<<"Do You Want to continue "; break;

case 7:

{

cout<<"\nEnter Number to be searched: "; cin>>num;

n.search(num);

cout<<"\nDo you wantto continue"; break;

}

default: cout<<"Not valid";

}

ch=getch();

}

}