NAME: Akshat Srivastav

slot:B1

Reg number: 19BCE0811

Implementing stacks using linked list:

#include <iostream>

using namespace std;

struct Node

{

int data;

Node\*next;

};

class Stack

{

Node\*top;

public:

Stack()

{

top=NULL;

}

void push(int val)//to add a new number

{

Node\*t=new Node;

if(t==NULL)

{

cout<<"\nOverflow";

return;

}

else

{

t->data=val;

t->next=NULL;

if(top==NULL)

{

top=t;

}

else

{

t->next=top;

top=t ;

}

cout<<"\nInserted\n";

}

}

void pop()//to remove an element

{

Node\*t=new Node;

if(top==NULL)

{

cout<<"\nUnderflow";

return;

}

t=top;

top=top->next;

cout<<"\nDeleted data is "<<t->data;

delete(t);

}

void search(int val)/to search for an element

{

Node\*t=new Node;

t=top;

int count=0;

    int flag=0;

while(t!=NULL)

{

count++;

if(val==t->data)

{

cout<<"\nFound at "<<count;

flag++;

        break;

}

t=t->next;

}

    if(flag==0)

      cout<<"\nThe element is not in the stack";

}

void display()//to display elements

{

cout<<"\nThe stack is: ";

Node\*t=new Node;

t=top;

while(t!=NULL)

{

cout<<"\n"<<t->data;

t=t->next;

}

}

};

int main()

{

Stack s1;

int value,option;

do

{

cout<<"\nEnter the function number for the following"<<endl;

cout<<"\n1.Push";

cout<<"\n2.Pop";

cout<<"\n3.Display";

cout<<"\n4.Search";

cout<<"\n5.Exit";

cout<<endl;

cin>>option;

switch(option)

{

case 1:

{

cout<<"\nEnter the data to be pushed: ";

cin>>value;

s1.push(value);

cout<<endl;

break;

}

case 2:

{

s1.pop();

cout<<endl;

break;

}

case 3:

{

s1.display();

cout<<endl;

break;

}

case 4:

{

cout<<"\nEnter value to search for: ";

cin>>value;

s1.search(value);

break;

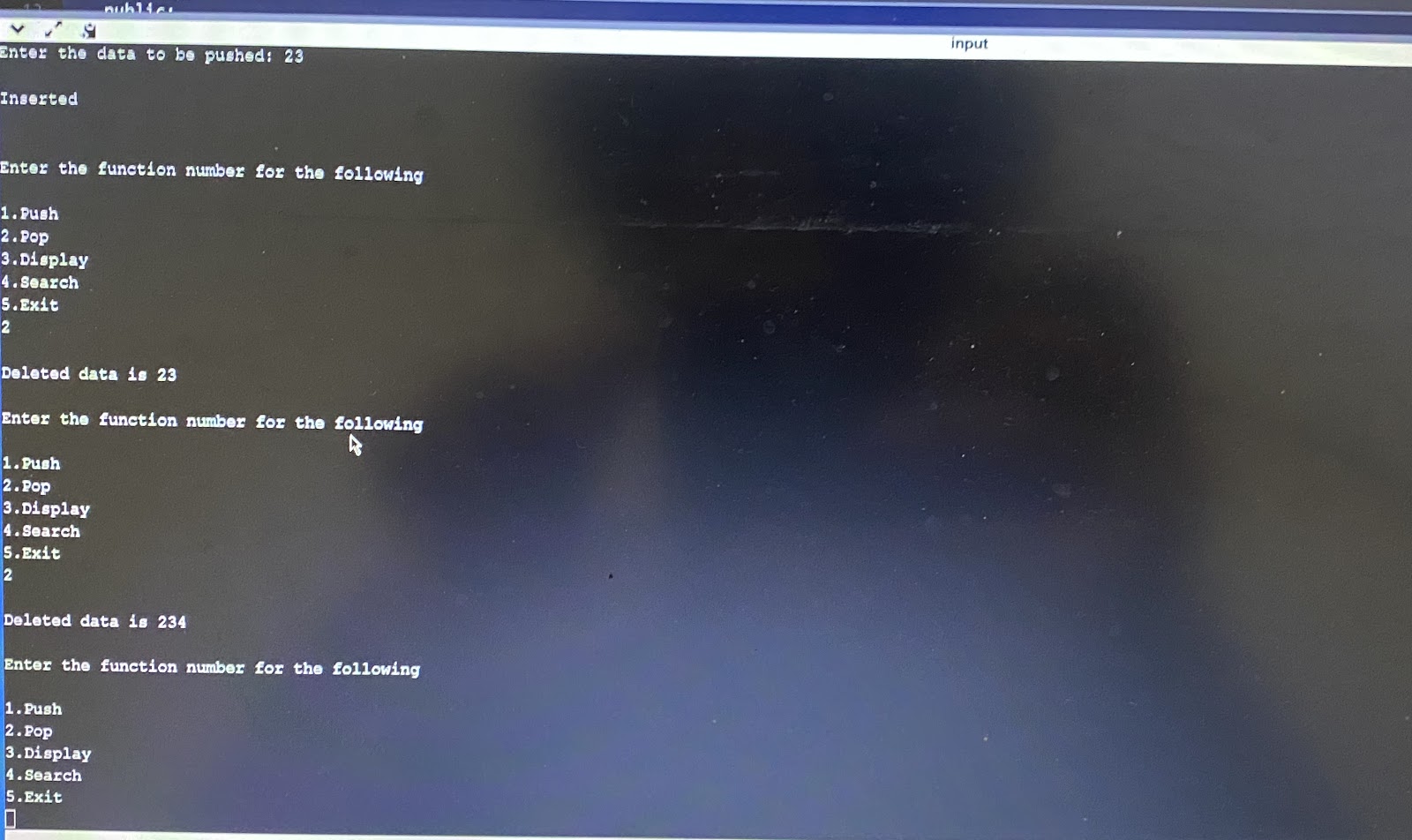
}

}}while(option!=5);

return 0;

}

output :



Implementing queue using linked list:

#include <iostream>

using namespace std;

struct Node

{

int data;

Node\*next;

};

class Queue

{

Node\*front;

Node\*rear;

public:

Queue()

{

front=NULL;

rear=NULL;

}

void insert(int val)//inserting elements

{

Node\*t=new Node;

if(t==NULL)

{

cout<<"\nQueue is full";

return;

}

else

{

t->data=val;

t->next=NULL;

if(front==NULL)

{

front=rear=t;

}

else

{

rear->next=t;

rear=t;

}

}

}

void delete1()//deleting elements

{

Node\*t=new Node;

if(front==NULL)

{

cout<<"\nQueue is empty";

return;

}

else

{

t=front;

cout<<"\nDelete data is: "<<t->data;

front=front->next;

delete(t);

}

}

void search(int val)//search for elements

{

Node\*t=new Node;

t=front;

int count=0;

      int flag=0;

while(t!=NULL)

{

count++;

if(val==t->data)

{

cout<<"\nFound at "<<count;

          flag++;

break;

}

t=t->next;

}

      if(flag==0)

        cout<<"\nThe element is not present in the queue";

}

void display()//display elements

{

Node\*t=new Node;

t=front;

cout<<"\nThe Queue is: ";

while(t!=NULL)

{

cout<<endl<<t->data;

t=t->next;

}

}

};

int main()

{

Queue q1;

int option,value;

do

{

cout<<"\nfunction of queues using linked list: ";

cout<<"\n1.Insert";

cout<<"\n2.Delete";

cout<<"\n3.Display";

cout<<"\n4.Search";

cout<<"\n5.Exit\n";

cin>>option;

switch(option)

{

case 1:

{

cout<<"\nEnter data: ";

cin>>value;

q1.insert(value);

break;

}

case 2:

{

q1.delete1();

break;

}

case 3:

{

q1.display();

break;

}

case 4:

{

cout<<"\nEnter the value to search: ";

cin>>value;

q1.search(value);

break;

}

}

}while(option!=5);

return 0;

}

OUTPUT:

