Queue

#include"queue.h"

int main(){

//create queue

Queue q(10);

int choice;

do {

cout<<"\n0.Exit\n1.push\n2.pop\n3.display\n";

cout<<"\nEnter the choice:\n";

cin>>choice;

switch(choice){

case 0:{

cout<<"\nExit\n";

break;

}

case 1:{

int ele;

cout<<"Enter the element to push:";

cin>>ele;

if(q.push(ele)){

cout<<"Element added successfully in the queue\n";

}

else{

cout<<"Element is not added";

}

break;

}

case 2:{

int ele;

if(q.pop(ele)){

cout<<ele<<"pop is successfully done\n";

}

else{

cout<<"pop operation is not done\n";

}

break;

}

case 3:{

q.display();

break;

}

default:{

cout<<"\ninvalid Input\n";

break;

}

}

}while(choice!=0);

return 0;

}

#include<iostream>

using namespace std;

class Queue{

int size;

int front;

int rear;

int\* ptr;

public:

Queue(int);

bool isFull();

bool isEmpty();

bool push(int);

bool pop(int&);

void display();

~Queue();

};

#include"Queue.h"

Queue::Queue(int s) {

this->size=s;

this->front =-1;

this->rear=-1;

this->ptr=new int[size];

}

bool Queue::isFull(){

if((front==0 && rear==size-1)||front==rear+1){

return true;

}

else{

return false;

}

}

bool Queue::isEmpty(){

if(front==-1 && rear==-1){

return true;

}

return false;

}

bool Queue::push(int ele){

cout<<"i am in push function\n";

if(isFull()){

cout<<"Queue is full\n";

return false;

}

else{

//set rear then assign value

if(isEmpty()){

front=rear=0;

}

else if(rear==size-1){

rear=0;

}

else{

rear++;

}

//after setting the rear now we can assign the value

this->ptr[rear]=ele;

return true;

}

}

bool Queue::pop(int& ele){

if(isEmpty()){

cout<<"Queue is Empty\n";

return false;

}

else{

//1st assign the value(remove) then set the front

ele=this->ptr[front];

if(front==size-1){

front=0;

}

else if(front ==rear)//only one ele is present

{

front=rear=-1;

}

else{

front++;

}

return true;

}

}

void Queue::display(){

if(isEmpty()){

cout<<"Queue is Empty!!\n";

}else{

int i=front;

cout<<"Queue:\n";

while(i!=rear){

cout<<this->ptr[i]<<"\t";

if(i==size){

i=0;

}

else{

i++;

}

}

//out of loop

//print the rear element

cout<<this->ptr[i];

}

}

Queue::~Queue(){

cout<<"destructor is called\n";

delete[] this->ptr;

}