Ali 666

#include<iostream>

using namespace std;

struct node{

int data;

node \* next;

};

class Queue

{

node \* rare ;

node \* front ;

public:

Queue(){

rare = NULL;

front = NULL;

}

bool isEmpty()

{

if(front == NULL)

return true;

else

return false;

}

int getValue()

{

int v;

cout<<"Enter the Value: ";

cin>>v;

return v;

}

void enQueue()

{

node \* news = new node();

int value = getValue();

if(!news)

cout<<"Memory is Full right now\n";

else{

if(rare == NULL)

{

news->data = value;

news->next = NULL;

rare = news;

// front = rare;

front = news;

}

else {

news->data = value;

news->next = NULL;

rare->next = news;

rare = news;

if(isEmpty())

front = news;

}

}

}

void deQueue()

{

if(isEmpty())

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

else{

cout<<"Value "<<front->data<< " is deQueue\n";

front = front->next;

}

}

void display()

{

node \* tempAdress = front;

if(isEmpty()){

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

}

else{

cout<<"Your Output is \n";

while(tempAdress->next != NULL)

{

cout<<tempAdress->data<<endl;

tempAdress = tempAdress->next;

}

cout<<tempAdress->data<<endl;

}

}

};

int main()

{

Queue object;

int n;

while(true)

{

cout<<"1: EnQueue\n2: DeQueue\n3: Display\n4: Exit\n";

cin>>n;

if(n == 1)

object.enQueue();

else if(n == 2)

object.deQueue();

else if(n == 3)

object.display();

else if (n == 4)

break;

}

}

#include<iostream>

using namespace std;

struct node{

int data;

node \* next;

};

node \* head = NULL;

class Queue

{

public:

Queue(){

}

bool isEmpty()

{

if(head == NULL)

return true;

else

return false;

}

int getValue()

{

int v;

cout<<"Enter the Value: ";

cin>>v;

return v;

}

void enQueue()

{

node \* news = new node();

int value = getValue();

if(!news)

cout<<"Memory is Full right now\n";

else{

if(head == NULL)

{

news->data = value;

news->next = NULL;

head = news;

}

else {

node \* tempAdress = head;

while(tempAdress->next != NULL)

{

tempAdress = tempAdress->next;

}

news->data = value;

news->next = NULL;

tempAdress->next = news;

}

}

}

void deQueue()

{

node \* tempAdress = head;

if(isEmpty())

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

else{

// tempAdress = tempAdress->next;

cout<<"Value "<<tempAdress->data<< " is deQueue\n";

head = tempAdress->next;

}

}

void display()

{

node \* tempAdress = head;

if(isEmpty()){

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

}

else{

cout<<"Your Output is \n";

while(tempAdress->next != NULL)

{

cout<<tempAdress->data<<endl;

tempAdress = tempAdress->next;

}

cout<<tempAdress->data<<endl;

}

}

};

int main()

{

Queue object;

int n;

while(true)

{

cout<<"1: EnQueue\n2: DeQueue\n3: Display\n4: Exit\n";

cin>>n;

if(n == 1)

object.enQueue();

else if(n == 2)

object.deQueue();

else if(n == 3)

object.display();

else if (n == 4)

break;

}

}

