**Saad Ahmad 01-134222-130**

Q1 :

#include <iostream>

using namespace std;

struct Node {

int value;

Node\* next;

};

class CirQueue {

Node\* front;

Node\* rear;

int count;

public:

CirQueue() {

front = NULL;

rear = NULL;

count = 0;

}

void Enqueue(int val) {

Node\* temp = new Node();

Node\* temp2 = front;

temp->value = val;

temp->next = NULL;

if (count == 0) {

rear = temp;

front = temp;

temp->next = front;

}

else {

while (temp2->next != front) {

temp2 = temp2->next;

}

temp2->next = temp;

temp->next = front;

rear = temp;

}

count++;

}

void display() {

int counter = count;

while (counter != 0) {

cout << front->value << endl;

front = front->next;

counter--;

}

}

};

int main() {

CirQueue q1;

q1.Enqueue(1);

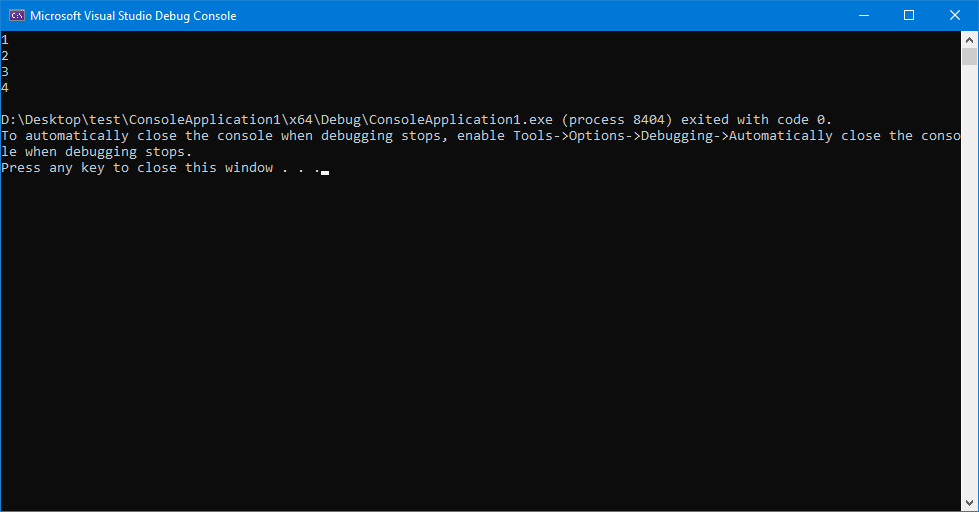
q1.Enqueue(2);

q1.Enqueue(3);

q1.Enqueue(4);

q1.display();

}



Q2 :

#include <iostream>

using namespace std;

struct Node {

int value;

Node\* next;

};

class LinkedList {

Node\* head;

public:

LinkedList() {

head = NULL;

}

void insert(int val) {

Node\* temp = new Node();

temp->value = val;

if (head == NULL) {

temp->next = NULL;

head = temp;

}

else {

Node\* temp2 = head;

while (temp2->next != NULL) {

temp2 = temp2->next;

}

temp2->next = temp;

}

}

int getVal() {

return head->value;

}

void traverse() {

head = head->next;

}

Node\* getHead() {

return head;

}

void copy(LinkedList l1, LinkedList &l2) {

while (l1.getHead()!=NULL) {

l2.insert(l1.getVal());

l1.traverse();

}

}

void display() {

Node\* temp = head;

while (temp!= NULL) {

cout << temp->value << endl;

temp = temp->next;

}

}

};

int main() {

LinkedList l1;

LinkedList l2;

l1.insert(1);

l1.insert(2);

l1.insert(3);

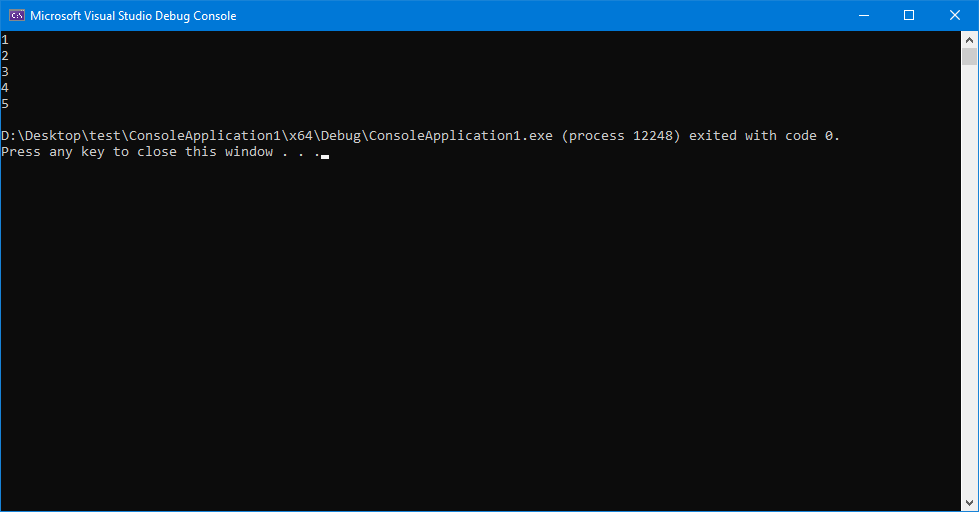
l1.insert(4);

l1.insert(5);

l2.copy(l1, l2);

l2.display();

}



Q4:

#include <iostream>

using namespace std;

struct Node {

int value;

Node\* next;

Node\* prev;

};

class Queue {

Node\* front;

Node\* rear;

public:

Queue() {

front = NULL;

rear = NULL;

}

void Enqueue(int val) {

Node\* temp = new Node();

temp->value = val;

if (front == NULL && rear == NULL) {

temp->next = NULL;

temp->prev = NULL;

rear = front = temp;

}

else {

Node\* temp2 = front;

while (temp2->next!= NULL)

{

temp2 = temp2->next;

}

temp->prev = temp2;

temp2->next = temp;

temp->next = NULL;

rear = temp;

}

}

void display() {

Node\* temp = front;

while (temp != NULL) {

cout << temp->value << endl;

temp = temp->next;

}

}

};

int main() {

Queue q1;

q1.Enqueue(1);

q1.Enqueue(2);

q1.Enqueue(3);

q1.Enqueue(4);

q1.display();

}

