CS224 Lab# 06 - Fall 2023 > bq08283@st.habib.edu.pk

You can view this report online at : https://www.hackerrank.com/x/tests/1671606/candidates/56677861/report

Full Name:	Breeha Qasim
Email:	bq08283@st.habib.edu.pk
Test Name:	**CS224 Lab# 06 - Fall 2023**
Taken On:	4 Oct 2023 12:20:15 PKT
Time Taken:	122 min 44 sec/ 10000 min
Work Experience:	< 1 years
Invited by:	Shayan
Skills Score:	
Tags Score:	

100%

50/50

scored in **CS224 Lab# 06 - Fall 2023** in 122 min 44 sec on 4 Oct 2023 12:20:15 PKT

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Implementing Queue using Linked List > **Coding**	2 hour 7 sec	50/ 50	✓

QUESTION 1

✓

Correct Answer

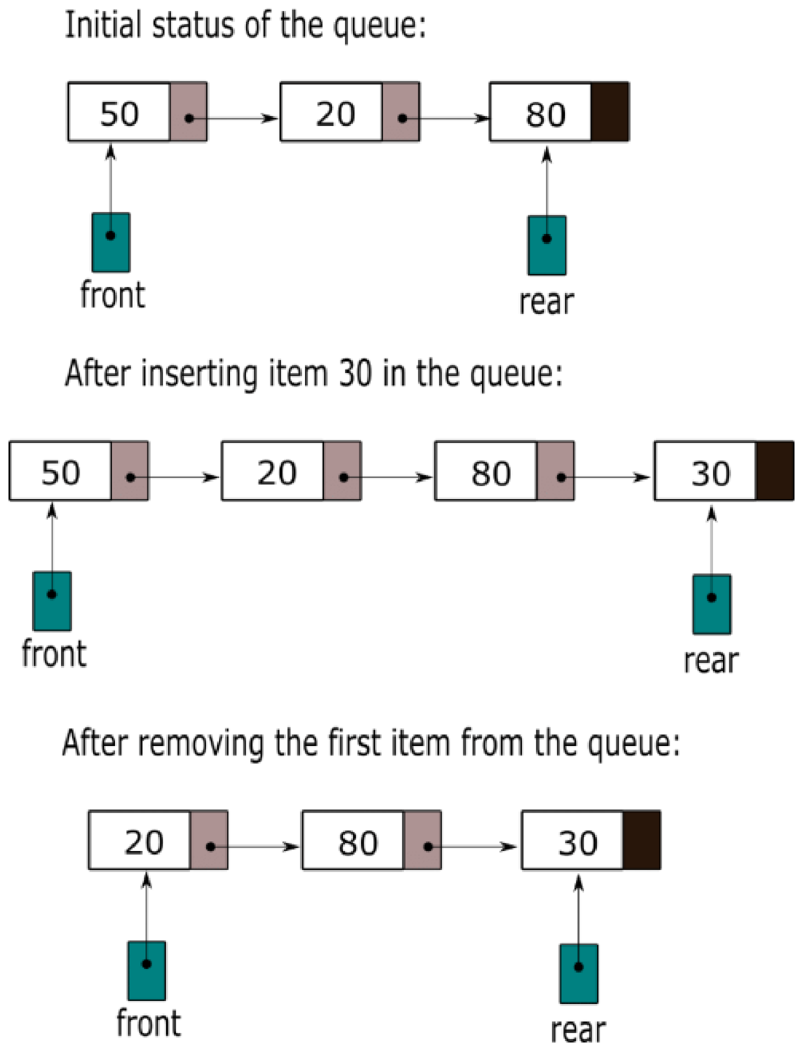
Score 50

Implementing Queue using Linked List > Coding

QUESTION DESCRIPTION

A linked list is made up of many nodes which are connected. Every node is mainly divided into two parts, one part holds the data and the other part is a pointer connecting the next node.
We can implement a Queue using a LinkedList, sample enqueue and dequeue operations are shown in Fig. 1.

Create a class: node which has a variable for storing data and a pointer that points to the next node.
Create a class: Queue, which have private pointers (front, rear), public functions enqueue(int) and int dequeue().
Add a function print_queue() in the Queue class which prints the elements in the queue.



Initial status of the queue:

After inserting item 30 in the queue:

After removing the first item from the queue:

Figure 1: Sample Queue and Dequeue operations

Example input:
5
1 2 3 4 5
3
output:
1 2 3 4 5
4 5

Explanation:
input
Line 1 shows the number of elements
Line 2 are the elements which are to be enqueued
Line 3 shows the number of elements to be dequeued

output
line1 shows the elements after they are enqueued
line2 shows the updated queue after dequeuing the elements

INTERVIEWER GUIDELINES

```
// 1-- Create a class Node, which has one variable for storing data and a
pointer which points to the next node

class Node
{
public:
    int data;
    Node* next;

    Node(int x):data(x),next(nullptr){}

};



// 2 -- Create a class Queue ; which has two public pointers for front and rear nodes
//     and it also has two public functions void Enqueue(int x) and int Dequeue()
//      also add a function print_queue() which prints all the elements in the queue

class Queue
{
    public:
    Node* front;
    Node* rear;

    Queue():front(nullptr), rear(nullptr){} // initialize the pointers to null

    void Enqueue(int x)
    {
        Node* node = new Node(x);

        if(front == nullptr) // no element in the queue
        {
            front = node;    // front and rear points to the same node
            rear = node;
        }

        else
        {
            rear->next = node;
            rear = node;
        }
    }

    int Dequeue()
    {

        if(front == nullptr && rear== nullptr)  // empty queue
            return -1;
        else if(front != nullptr && front == rear) /// only one element in the queue
        {
            int data = front->data;
            delete front;
            front = rear=nullptr;
            return data;
        }
        else // more than one elements in the queue
        {
            int data = front->data;
            Node* n = front;
            front = front->next;
            delete n;
            return data;
        }
    }

    void print_queue()
    {
        if(front == nullptr)
            return;
        else if(front == rear)
        {
            cout << front->data << endl;
        }
        else
        {
            Node* n = front;
            cout << n->data << " " ;

            while(n != rear)
            {
                n = n->next;
                cout << n->data << " ";
            }
            cout << endl;
        }
    }
};
```

CANDIDATE ANSWER

Language used: **C++**

```
// Don't use "using namespace std;"
#include <iostream>
// 1-- Create a class Node, which has one variable for storing data and a
pointer which points to the next node
class Node{
    public:
        int data;
        Node* node_next;
        Node():
            data{0}, node_next{nullptr}
        {
        }
};

// 2 -- Create a class Queue ; which has two public pointers for front and
rear nodes
//      and it also has two public functions void Enqueue(int x) and int
Dequeue()
//       also add a function print_queue() which prints all the elements in
the queue

class Queue{
    private:
        Node* front;
        Node* rear;
    public:
        Queue():
            front{nullptr},rear{nullptr}
        {
        }
        void Enqueue(int val){
            Node* newNode{new Node};
            newNode->data=val;
            newNode->node_next=nullptr;

```

```cpp
            //Two situations
            if (front==nullptr && rear==nullptr){
                front=newNode;
                rear=newNode;
            }
            else{
                rear->node_next = newNode;
                rear = newNode;

            }
        }
        int Dequeue() {
            if (front == nullptr) {
                return -1; // Queue is empty
            }

            Node* temp = front;
            int data = temp->data;

            if (front == rear) {
                front = rear = nullptr; // Queue has only one element
            } else {
                front = front->node_next;
            }
            delete temp;
            return data;
        }

        void print_queue () {
            Node* temp = front;
            while (temp != nullptr) {
                std::cout << temp->data << " ";
                temp = temp->node_next;
            }
            std::cout << std::endl;
        }
        ~Queue() {
            while (front != nullptr) {
                Node* temp = front;
                front = front->node_next;
                delete temp;
        }
}
};
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0512 sec	8.7 KB
Testcase 1	Easy	Hidden case	✓ Success	10	0.0478 sec	8.79 KB
Testcase 2	Easy	Hidden case	✓ Success	10	0.0494 sec	8.91 KB
Testcase 3	Easy	Hidden case	✓ Success	10	0.0285 sec	8.82 KB
Testcase 4	Easy	Sample case	✓ Success	10	0.0459 sec	8.91 KB

No Comments

PDF generated at: 5 Oct 2023 09:49:01 UTC