Harsh_Kumar_MAN_106_Assignment_4

- **Answer 1:**

```cpp
#include<bits/stdc++.h>
using namespace std;

struct node{
    int element;
    node *next;
};

class QUEUE{
    node *q_front;
    node *q_rear;
    int size;
public:
    QUEUE();
   ~QUEUE();
    bool underflow();
    void add(int item);
    int remove();
    int count();
};

QUEUE::QUEUE(){
    q_front = NULL;
    q_rear = NULL;
    size = 0;
}

QUEUE::~QUEUE(){
    while (!underflow()) remove();
    delete q_front, q_rear;
}

bool QUEUE::underflow(){
    if (q_front==NULL) return true;
    else return false;
}
```

```
void QUEUE::add(int item){
    node *next_element;
    next_element = new node;
    next_element->element = item;
    next_element->next = NULL;
    if (q_rear == NULL) q_front = next_element;
    else q_rear->next = next_element;
    size++;
    q_rear = next_element;
}

int QUEUE::remove(){
    if (underflow()){
        cout << "Queue is empty. Add some values first\n";
        return -1;
    }
    else{
        node *temp;
        temp = q_front;
        int item = temp->element;
        q_front = temp->next;
        size--;
        delete temp;
        return item;
    }
}

int QUEUE::count(){
    return size;
}
```

```
int main(){
    QUEUE q;
    for (int i=0; i<10; i++) q.add(i+1);
    cout << "There are " << q.count() << " elements in the queue\n";
    for (int i=0; i<11; i++){
        cout << q.remove() << '\n';
        cout << "There are " << q.count() << " elements in the queue\n";
    }
}
```

- Output :

```
There are 10 elements in the queue
1
There are 9 elements in the queue
2
There are 8 elements in the queue
3
There are 7 elements in the queue
4
There are 6 elements in the queue
5
There are 5 elements in the queue
6
There are 4 elements in the queue
7
There are 3 elements in the queue
8
There are 2 elements in the queue
9
There are 1 elements in the queue
10
There are 0 elements in the queue
Queue is empty. Add some values first
-1
There are 0 elements in the queue
```

- **Answer 2_1** :

```cpp
#include<bits/stdc++.h>
using namespace std;

struct node{
    char element;
    node *next;
};

class QUEUE{
    node *q_front;
    node *q_rear;
    int size;
public:
    QUEUE();
   ~QUEUE();
    bool underflow();
    void add(char item);
    char remove();
    int count();
};

QUEUE::QUEUE(){
    q_front = NULL;
    q_rear = NULL;
    size = 0;
}

QUEUE::~QUEUE(){
    while (!underflow()) remove();
    delete q_front, q_rear;
}

bool QUEUE::underflow(){
    if (q_front==NULL) return true;
    else return false;
}
```

```
void QUEUE::add(char item){
    node *next_element;
    next_element = new node;
    next_element->element = item;
    next_element->next = NULL;
    if (q_rear == NULL) q_front = next_element;
    else q_rear->next = next_element;
    size++;
    q_rear = next_element;
}

char QUEUE::remove(){
    if (underflow()){
        cout << "Queue is empty. Add some values first\n";
        return '-';
    }
    else{
        node *temp;
        temp = q_front;
        char item = temp->element;
        q_front = temp->next;
        size--;
        delete temp;
        return item;
    }
}

int QUEUE::count(){
    return size;
}
```

```
int main(){
    QUEUE q;
    for (int i=65; i<75; i++) q.add(char(i));
    cout << "There are " << q.count() << " elements in the queue\n";
    for (int i=0; i<11; i++){
        cout << q.remove() << '\n';
        cout << "There are " << q.count() << " elements in the queue\n";
    }
}
```

- Output :

```
There are 10 elements in the queue
A
There are 9 elements in the queue
B
There are 8 elements in the queue
C
There are 7 elements in the queue
D
There are 6 elements in the queue
E
There are 5 elements in the queue
F
There are 4 elements in the queue
G
There are 3 elements in the queue
H
There are 2 elements in the queue
I
There are 1 elements in the queue
J
There are 0 elements in the queue
Queue is empty. Add some values first
-
There are 0 elements in the queue
```

- **Answer 2_2:**

```cpp
#include<bits/stdc++.h>
using namespace std;

struct node{
    string element;
    node *next;
};

class QUEUE{
    node *q_front;
    node *q_rear;
    int size;
public:
    QUEUE();
   ~QUEUE();
    bool underflow();
    void add(string item);
    string remove();
    int count();
};

QUEUE::QUEUE(){
    q_front = NULL;
    q_rear = NULL;
    size = 0;
}

QUEUE::~QUEUE(){
    while (!underflow()) remove();
    delete q_front, q_rear;
}

bool QUEUE::underflow(){
    if (q_front==NULL) return true;
    else return false;
}
```

```
void QUEUE::add(string item){
    node *next_element;
    next_element = new node;
    next_element->element = item;
    next_element->next = NULL;
    if (q_rear == NULL) q_front = next_element;
    else q_rear->next = next_element;
    size++;
    q_rear = next_element;
}

string QUEUE::remove(){
    if (underflow()){
        cout << "Queue is empty. Add some values first\n";
        return "NA";
    }
    else{
        node *temp;
        temp = q_front;
        string item = temp->element;
        q_front = temp->next;
        size--;
        delete temp;
        return item;
    }
}

int QUEUE::count(){
    return size;
}
```

```
int main(){
    QUEUE q;
    string s = "Hello World";
    for (int i=65; i<70; i++){
        s.push_back(i);
        q.add(s);
    }
    cout << "There are " << q.count() << " elements in the queue\n";
    for (int i=0; i<6; i++){
        cout << q.remove() << '\n';
        cout << "There are " << q.count() << " elements in the queue\n";
    }
}
```

- Output :

```
There are 5 elements in the queue
Hello WorldA
There are 4 elements in the queue
Hello WorldAB
There are 3 elements in the queue
Hello WorldABC
There are 2 elements in the queue
Hello WorldABCD
There are 1 elements in the queue
Hello WorldABCDE
There are 0 elements in the queue
Queue is empty. Add some values first
NA
There are 0 elements in the queue
```

- Answer 3:

```cpp
#include<bits/stdc++.h>
using namespace std;

struct node{
    int element;
    node *next;
};

class C_QUEUE{
    node *q_front;
    node *q_rear;
    int size;
public:
    C_QUEUE();
   ~C_QUEUE();
    bool underflow();
    void add(int item);
    int remove();
    int count();
};

C_QUEUE::C_QUEUE(){
    q_front = NULL;
    q_rear = NULL;
    size = 0;
}

C_QUEUE::~C_QUEUE(){
    while (!underflow()) remove();
}

bool C_QUEUE::underflow(){
    if (q_front==NULL) return true;
    else return false;
```

```
}

void C_QUEUE::add(int item){
    node *next_element;
    next_element = new node;
    next_element->element = item;
    next_element->next = q_front;
    if (q_rear == NULL) q_front = next_element;
    else q_rear->next = next_element;
    size++;
    q_rear = next_element;
}

int C_QUEUE::remove(){
    if (underflow()){
        cout << "Queue is empty. Add some values first\n";
        return -1;
    }
    else if (q_front==q_rear){
       int item = q_front->element;
       delete(q_rear);
       q_front=NULL;
       size--;
       return item;
    }
    else{
        node *temp;
        temp = q_front;
        int item = temp->element;
        q_front = temp->next;
        size--;
        delete temp;
        return item;
    }
```

```
}

int C_QUEUE::count(){
    return size;
}

int main(){
    C_QUEUE q;
    for (int i=0; i<10; i++) q.add(i+1);
    cout << "There are " << q.count() << " elements in the queue\n";
    for (int i=0; i<11; i++){
        cout << q.remove() << '\n';
        cout << "There are " << q.count() << " elements in the queue\n";
    }
}
```

Output :

```
There are 10 elements in the queue
1
There are 9 elements in the queue
2
There are 8 elements in the queue
3
There are 7 elements in the queue
4
There are 6 elements in the queue
5
There are 5 elements in the queue
6
There are 4 elements in the queue
7
There are 3 elements in the queue
8
There are 2 elements in the queue
9
There are 1 elements in the queue
10
There are 0 elements in the queue
Queue is empty. Add some values first
-1
There are 0 elements in the queue
```

- **Answer 4:**

```cpp
#include<bits/stdc++.h>
using namespace std;

struct node{
    float element;
    node *next;
};

class STACK{
    node *top;
public:
    STACK();
   ~STACK();
    bool underflow();
    void push(float item);
    float pop();
};

STACK::STACK(){
    top = NULL;
}

STACK::~STACK(){
    while (!underflow()) pop();
    delete top;
}

bool STACK::underflow(){
    if (top==NULL) return true;
    else return false;
}
```

```cpp
void STACK::push(float item){
    node *next_element;
    next_element = new node;
    next_element->element = item;
    next_element->next = top;
    top = next_element;
}

float STACK::pop(){
    if (underflow()){
        cout << "Stack underflow. Push some values first\n";
        return -1;
    }
    else{
        node *temp;
        temp = top;
        float item = temp->element;
        top = temp->next;
        delete temp;
        return item;
    }
}

int main(){
    STACK s, t;
    for (int i=0; i<10; i++) s.push(float(i+1)+0.12);
    cout << "Printing initial stack and inverting it:\n";
    for (int i=0; i<10; i++) {
        float last_element = s.pop();
        cout << last_element << '\n';
        t.push(last_element);
    }
    cout << "Printing inverted stack:\n";
    for (int i=0; i<10; i++) cout << t.pop() << '\n';
}
```

- Output :

```
Printing initial stack and inverting it:
10.12
9.12
8.12
7.12
6.12
5.12
4.12
3.12
2.12
1.12
Printing inverted stack:
1.12
2.12
3.12
4.12
5.12
6.12
7.12
8.12
9.12
10.12
```