

Harsh_Kumar_MAN_106_Assignment_3

- Answer 1:

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  #define MAX 50
5
6  class QUEUE{
7      int q_fornt;
8      int q_rear;
9      int queue[MAX];
10     int size;
11 public:
12     QUEUE();
13     bool underflow();
14     bool overflow();
15     void add(int item);
16     int remove();
17     int count();
18 };
19
20 QUEUE::QUEUE(){
21     queue[MAX] = {0};
22     q_fornt = MAX-1;
23     q_rear = MAX-1;
24     size = 0;
25 }
26
27 bool QUEUE::underflow(){
28     if (size==0) return true;
29     else return false;
30 }
31
32 bool QUEUE::overflow(){
33     if (q_rear==-1) return true;
34     else return false;
35 }
36
```

```

37 void QUEUE::add(int item){
38     if (overflow()) {
39         cout << "Queue is full\nRemove some values first\n";
40         return;
41     }
42     else queue[q_rear--] = item, size++;
43 }
44
45 int QUEUE::remove(){
46     if (underflow()) {
47         cout << "Queue is empty\nAdd some values first\n";
48         return -1;
49     }
50     else {
51         size--;
52         return (queue[q_fornt--]);
53     }
54 }
55
56 int QUEUE::count(){
57     return size;
58 }
59
60 int main(){
61     QUEUE q;
62     for(int i=0; i<10; i++) q.add(i+1);
63
64     cout << q.remove() << endl;
65     for(int i=0; i<9; i++) cout << q.remove() << endl ;
66     cout << q.remove();
67 }

```

- Output :

```

1
2
3
4
5
6
7
8
9
10
Queue is empty
Add some values first
-1

```

- Answer 2_1:

```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  #define MAX 50
5
6  class QUEUE{
7      int q_fornt;
8      int q_rear;
9      char queue[MAX];
10     int size;
11 public:
12     QUEUE();
13     bool underflow();
14     bool overflow();
15     void add(char item);
16     char remove();
17     int count();
18 };
19
20 QUEUE::QUEUE(){
21     queue[MAX] = {0};
22     q_fornt = MAX-1;
23     q_rear = MAX-1;
24     size = 0;
25 }
26
27 bool QUEUE::underflow(){
28     if (size==0) return true;
29     else return false;
30 }
31
32 bool QUEUE::overflow(){
33     if (q_rear==MAX-1) return true;
34     else return false;
35 }
36

```

```

37 void QUEUE::add(char item){
38     if (overflow()) {
39         cout << "Queue is full\nRemove some values first\n";
40         return;
41     }
42     else queue[q_rear++] = item, size++;
43 }
44
45 char QUEUE::remove(){
46     if (underflow()) {
47         cout << "Queue is empty\nAdd some values first\n";
48         return '-';
49     }
50     else {
51         size--;
52         return (queue[q_fornt--]);
53     }
54 }
55
56 int QUEUE::count(){
57     return size;
58 }
59
60 int main(){
61     QUEUE q;
62     for(int i=66; i<76; i++) q.add(char(i+1));
63
64     cout << q.remove() << endl;
65     for(int i=0; i<9; i++) cout << q.remove() << endl ;
66     q.remove();
67 }

```

- Output :

```

C
D
E
F
G
H
I
J
K
L
Queue is empty
Add some values first

```

- Answer 2_2 :

```

1  #include<iostream>
2  #include<string>
3
4  using namespace std;
5
6  #define MAX 50
7
8  class QUEUE{
9      int q_fornt;
10     int q_rear;
11     string queue[MAX];
12     int size;
13 public:
14     QUEUE();
15     bool underflow();
16     bool overflow();
17     void add(string item);
18     string remove();
19     int count();
20 };
21
22 QUEUE::QUEUE(){
23     queue[MAX] = {0};
24     q_fornt = MAX-1;
25     q_rear = MAX-1;
26     size = 0;
27 }
28
29 bool QUEUE::underflow(){
30     if (size==0) return true;
31     else return false;
32 }
33
34 bool QUEUE::overflow(){
35     if (q_rear==MAX-1) return true;
36     else return false;

```

```

39 void QUEUE::add(string item){
40     if (overflow()) {
41         cout << "Queue is full\nRemove some values first\n";
42         return;
43     }
44     else queue[q_rear++] = item, size++;
45 }
46
47 string QUEUE::remove(){
48     if (underflow()) {
49         cout << "Queue is empty\nAdd some values first\n";
50         return "NA";
51     }
52     else {
53         size--;
54         return (queue[q_fornt--]);
55     }
56 }
57
58 int QUEUE::count(){
59     return size;
60 }
61
62 int main(){
63     QUEUE q;
64     string s = "Hello World";
65     for(int i=49; i<58; i++){
66         s.push_back(char(i));
67         q.add(s);
68     }
69     cout << q.remove() << endl;
70
71     for(int i=0; i<9; i++) cout << q.remove() << endl ;
72     q.remove();
73
74 }

```

- Output :

```

Hello World1
Hello World12
Hello World123
Hello World1234
Hello World12345
Hello World123456
Hello World1234567
Hello World12345678
Hello World123456789
Queue is empty
Add some values first
NA

```

- Answer 3 :

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  #define MAX 10
5
6  class QUEUE{
7      int q_fornt;
8      int q_rear;
9      int queue[MAX];
10     int size;
11 public:
12     QUEUE();
13     bool underflow();
14     bool overflow();
15     void add(int item);
16     int remove();
17     int count();
18 };
19
20 QUEUE :: QUEUE(){
21     queue[MAX] = {0};
22     q_fornt = MAX-1;
23     q_rear = MAX-1;
24     size = 0;
25 }
26
27 bool QUEUE :: underflow(){
28     if (size==0) return true;
29     else return false;
30 }
31
32 bool QUEUE :: overflow(){
33     if (size==MAX) return true;
34     else return false;
35 }
36
```

```

37 void QUEUE::add(int item){
38     if (overflow()) {
39         cout << "Queue is full\nRemove some values first\n";
40         return;
41     }
42     else queue[(q_rear--)%MAX] = item, size++;
43 }
44
45 int QUEUE::remove(){
46     if (underflow()) {
47         cout << "Queue is empty\nAdd some values first\n";
48         return -1;
49     }
50     else {
51         size--;
52         return (queue[(q_fornt--)%MAX]);
53     }
54 }
55
56 int QUEUE::count(){
57     return size;
58 }
59
60 int main(){
61     QUEUE q;
62     for(int i=0; i<11; i++) q.add(i+1);
63
64     cout << q.remove() << endl; // As soon as you remove a space is created
65     q.add(42); // Added due to circular queue
66     for(int i=0; i<10; i++) cout << q.remove() << endl ;
67
68 }

```

- Output :

```

Queue is full
Remove some values first
1
2
3
4
5
6
7
8
9
10
42

```

- Answer 4 :


```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  #define MAX 20
5
6  class QUEUE{
7      int q_front;
8      int q_rear;
9      int queue[MAX];
10     int size;
11 public:
12     QUEUE();
13     bool underflow();
14     bool overflow();
15     void add_front(int item);
16     void add_back(int item);
17     int remove_front();
18     int remove_back();
19     int count();
20 };
21
22 QUEUE::QUEUE(){
23     queue[MAX] = {0};
24     q_front = 0;
25     q_rear = 1;
26     size = 0;
27 }
28
29 bool QUEUE::underflow(){
30     if (q_front==0 || q_rear==1) return true;
31     else return false;
32 }
33
34 { bool QUEUE::overflow(){
35     if (q_front==q_rear) return true;
36     else return false;

```

```

38
39 void QUEUE::add_front(int item){
40     if (overflow()) {
41         cout << "Queue is full\nRemove some values first\n";
42         return;
43     }
44     else {
45         q_front--;
46         if (q_front == -1) q_front = MAX-1;
47         queue[(q_front)%MAX] = item, size++;
48     }
49 }
50
51 void QUEUE::add_back(int item){
52     if (overflow()) {
53         cout << "Queue is full\nRemove some values first\n";
54         return;
55     }
56     else {
57         q_rear++;
58         queue[(q_rear)%MAX] = item, size++;
59     }
60 }
61
62 int QUEUE::remove_front(){
63     if (underflow()) {
64         cout << "Queue is empty\nAdd some values first\n";
65         return -1;
66     }
67     else {
68         size--;
69         return (queue[(q_front++)%MAX]);
70     }
71 }
72

```

```

73 int QUEUE::remove_back(){
74     if (underflow()) {
75         cout << "Queue is empty\nAdd some values first\n";
76         return -1;
77     }
78     else {
79         size--;
80         return (queue[(q_rear--)%MAX]);
81     }
82 }
83
84 int QUEUE::count(){
85     return size;
86 }
87
88 int main(){
89     QUEUE q;
90     for(int i=0; i<5; i++) q.add_front(i+1);
91     for(int i=10; i<15; i++) q.add_back(i);
92     cout << "Removing from back:\n";
93     for(int i=0; i<4; i++) cout << q.remove_back() << endl ;
94     cout << "Removing from front:\n";
95     for(int i=0; i<5; i++) cout << q.remove_front() << endl ;
96 }
97

```

- Output :

```
Removing from back:  
14  
13  
12  
11  
Removing from front:  
5  
4  
3  
2  
1
```