Implement Queue using array

Basic Accuracy: 47.24% Submissions: 94K+ Points: 1

Implement a Queue using an Array. Queries in the Queue are of the following type:

(i) 1 x (a query of this type means pushing **'x'** into the queue)

(ii) 2 (a query of this type means to pop element from queue and print the poped element)

Example 1:

Input:

Q = 5

Queries = 1 2 1 3 2 1 4 2

Output: 2 3

Explanation:

In the first test case for query

1 2 the queue will be {2}

1 3 the queue will be {2 3}

C++ (g++ 5.4) Average Time: 15m Start Timer

```
// } Driver Code Ends


void MyQueue :: push(int x)
{
        // Your Code
        arr[rear++]=x;
}

//Function to pop an element from queue and return that element.
int MyQueue :: pop()
{
        // Your Code
        if(rear == 0) return -1;

        if(front != rear)
            return arr[front++];
        return -1;
}
```

Custom Input Compile & Run Submit

Implement Queue using Linked List

Basic Accuracy: 45.6% Submissions: 76K+ Points: 1

Implement a Queue using Linked List.

A Query **Q** is of 2 Types

(i) 1 x (a query of this type means pushing **'x'** into the queue)

(ii) 2 (a query of this type means to pop an element from the queue and print the poped element)

Example 1:

Input:

Q = 5

Queries = 1 2 1 3 2 1 4 2

Output: 2 3

Explanation: n the first testcase

1 2 the queue will be {2}

1 3 the queue will be {2 3}

2 poped element will be 2 the


```
// } Driver Code Ends


//Function to push an element into the queue.
void MyQueue:: push(int x)
{
        QueueNode *curr = new QueueNode(x);
        if(front == NULL){
            front = rear= curr;
            return;
        }
        rear->next = curr;
        rear = curr;
}

//Function to pop front element from the queue.
int MyQueue :: pop()
{
        if(front == NULL) return -1;
        QueueNode *curr = front;
        front = front->next;

        return curr->data;
}
```

Queue using stack

Easy Accuracy: 74.83% Submissions: 14K+ Points: 2

Implement a Queue using two stack **s1** and **s2**.

Example 1:

Input:
enqueue(2)
enqueue(3)
dequeue()
enqueue(4)
dequeue()
Output: 2 3
Explanation:
enqueue(2) the queue will be {2}
enqueue(3) the queue will be {3 2}
dequeue() the poped element will be 2
the stack will be {3}

C++ (g++ 5.4) Start Timer

```cpp
// } Driver Code Ends
//User function Template for C++

class Queue {
    stack<int> input, output;
public:

    void enqueue(int x) {
        input.push(x);

    }

    int dequeue() {
        if(input.empty() ) return -1;

        while(!input.empty()){
            output.push(input.top());
            input.pop();
        }
        int res=output.top();
        output.pop();

        while(!output.empty()){
            input.push(output.top());
            output.pop();
        }

        return res;

    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit