```
package queues;
import java.util.*;
public class Queues
{

    public static void main(String[] args)
    {
        Queue<Integer> queue = new ArrayDeque<>();
        queue.add(10);
        queue.add(20);
        queue.add(30);
        System.out.println(queue);
        //[10, 20, 30]
        var front = queue.remove();
        System.out.println(front);
        //10
        System.out.println(queue);
        //[20, 30]
    }
}
```

```
package queueusingarray;

import java.util.Arrays;

class ArrayQueue {

    private int[] items;
    private int rear;
    private int front;
    private int count;

    public ArrayQueue(int capacity) {
        items = new int[capacity];
    }

    public void enqueue(int item) {
        if (isFull()) {
            throw new IllegalStateException();
        }
        items[rear] = item;
        rear++;
        count++;
    }

    public int dequeue() {
        if (isEmpty()) {
            throw new IllegalStateException();
        }

        var item = items[front];
        items[front] = 0;
        front++;
        count--;
        return item;
    }

    public int peek() {
        if (isEmpty()) {
            throw new IllegalStateException();
        }

        return items[front];
    }

    public boolean isEmpty() {
        return count == 0;
    }

    public boolean isFull() {
        return count == items.length;
    }
@Override
    public String toString() {
        return Arrays.toString(items);
    }
}

public class QueueUsingArray {

    public static void main(String[] args) {
        ArrayQueue queue = new ArrayQueue(5);
        queue.enqueue(10);
        queue.enqueue(20);
        queue.enqueue(30);
        System.out.println(queue);
        //[10, 20, 30, 0, 0]
        System.out.println(queue.dequeue());
        //10
        System.out.println(queue);
        //[0, 20, 30, 0, 0]
        System.out.println(queue.dequeue());
        //20
        System.out.println(queue);
        //[0, 0, 30, 0, 0]
        queue.enqueue(40);
        queue.enqueue(50);
        System.out.println(queue);
        //[0, 0, 30, 40, 50]
        queue.enqueue(60);
//        Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for
length 5
//          at queueusingarray.ArrayQueue.enqueue(QueueUsingArray.java:20)
//          at queueusingarray.QueueUsingArray.main(QueueUsingArray.java:80)

    }
}
```

```
package circularqueueusingarray;

import java.util.Arrays;

class ArrayQueue {

    private int[] items;
    private int rear;
    private int front;
    private int count;

    public ArrayQueue(int capacity) {
        items = new int[capacity];
    }

    public void enqueue(int item) {
        if (isFull()) {
            throw new IllegalStateException();
        }

        items[rear] = item;
        rear = (rear + 1) % items.length;
        count++;
    }

    public int dequeue() {
        if (isEmpty()) {
            throw new IllegalStateException();
        }

        var item = items[front];
        items[front] = 0;
        front = (front + 1) % items.length;
        count--;

        return item;
    }

    public int peek() {
        if (isEmpty()) {
            throw new IllegalStateException();
        }

        return items[front];
    }

    public boolean isEmpty() {
        return count == 0;
    }

    public boolean isFull() {
        return count == items.length;
    }

    @Override
    public String toString() {
        return Arrays.toString(items);
    }
}

public class CircularQueueUsingArray {

    public static void main(String[] args) {
        ArrayQueue queue = new ArrayQueue(5);
        queue.enqueue(10);
        queue.enqueue(20);
        queue.enqueue(30);
        System.out.println(queue);
        //[10, 20, 30, 0, 0]
        queue.dequeue();
        queue.dequeue();
        System.out.println(queue);
        //[0, 0, 30, 0, 0]
        queue.enqueue(40);
        queue.enqueue(50);
        System.out.println(queue);
        //[0, 0, 30, 40, 50]
        queue.enqueue(60);
        System.out.println(queue);
        //[60, 0, 30, 40, 50]

    }

}
```

```
package reversequeueusingstack;

import java.util.*;
import java.util.Queue;
import java.util.Stack;

public class ReverseQueueUsingStack {

    public static void main(String[] args) {
        Queue<Integer> queue = new ArrayDeque<>();
        queue.add(10);
        queue.add(20);
        queue.add(30);
        System.out.println(queue);
        reverse(queue);
        System.out.println(queue);
    }

    public static void reverse(Queue<Integer> queue) {
        Stack<Integer> stack = new Stack<>();
        while (!queue.isEmpty()) {
            stack.push(queue.remove());
        }
        while (!stack.isEmpty()) {
            queue.add(stack.pop());
        }
    }

}
```

```
package queueusingtwostacks;

import java.util.*;

public class QueueUsingTwoStacks {

    private Stack<Integer> stack1 = new Stack<>();
    private Stack<Integer> stack2 = new Stack<>();

    // O(1)
    public void enqueue(int item) {
        stack1.push(item);
    }

    // O(n)
    public int dequeue() {
        if (isEmpty()) {
            throw new IllegalStateException();
        }

        moveStack1ToStack2();

        return stack2.pop();
    }

    private void moveStack1ToStack2() {
        if (stack2.isEmpty()) {
            while (!stack1.isEmpty()) {
                stack2.push(stack1.pop());
            }
        }
    }

    public int peek() {
        if (isEmpty()) {
            throw new IllegalStateException();
        }

        moveStack1ToStack2();

        return stack2.peek();
    }

    public boolean isEmpty() {
        return stack1.isEmpty() && stack2.isEmpty();
    }

    public static void main(String[] args) {
        QueueUsingTwoStacks q1 = new QueueUsingTwoStacks();
        q1.enqueue(10);
        q1.enqueue(20);
        q1.enqueue(30);
        q1.enqueue(40);
        q1.enqueue(50);
        System.out.println(q1.stack1);
        //[10, 20, 30, 40, 50]
        System.out.println(q1.dequeue());
        //10
        System.out.println(q1.dequeue());
        //20
        System.out.println(q1.dequeue());
        //30
        System.out.println(q1.dequeue());
        //40
        System.out.println(q1.dequeue());
        //50

    }

}
```