Stacks & Queues Competency #6

Stacks	**Mastered** 1 point	**Progressing** 0.001 points	**Novice** 0 points	**Criterion Score**
Can design and implement a stack using an array				/ 1
Can design and implement a stack using a linked list				/ 1

Queues	**Mastered** 1 point	**Progressing** 0.001 points	**Novice** 0 points	**Criterion Score**
Can design and implement a queue using a circular array				/ 1
Can design and implement a queue using a linked list				/ 1
Can design and implement a priority queue using a heap				/ 1

LinkQueue.java

```java
package project;

public class LinkQueue<E> {
    private Node<E> head;
    private Node<E> tail;
    private int manyNodes;

    public LinkQueue() {}

    public void enqueue(E e) {
        if(manyNodes == 0) {
            head = new Node<E>(e, null);
            tail = head;
        } else {
            tail.setLink(head);
            tail = tail.getLink();
        }
        manyNodes++;
    }

    public E dequeue() {
        E element = head.getData();
        head = head.getLink();
        manyNodes--;
        if(manyNodes == 0) {
            tail = null;
        }
        return element;
    }

    public E peek() {
        if(isEmpty()) {
            System.out.println("Is empty!");
        }
        return head.getData();
    }

    public boolean isEmpty() {
        if(head == null) {
            return true;
        } else {
            return false;
        }
    }
}

```

LinkStack.java

```java
package project;

public class LinkStack<E> {
    private Node<E> top;
    private int manyNodes;

    public LinkStack() {
        top = null;
        manyNodes = 0;
    }

    public void push (E e) {
        top = new Node<E>( e, top);
        manyNodes++;
    }

    public E pop() {
        E element = top.getData();
        top = top.getLink();
        manyNodes--;
        return element;
    }

    public E peek() {
        if(isEmpty()) {
            System.out.println("Is empty!");
        }
        return top.getData();
    }

    public boolean isEmpty() {
        if(top == null) {
            return true;
        } else {
            return false;
        }
    }



}

```

ArrayQueue.java

```java
package project;

public class ArrayQueue<E> {
    private Object[] data;
    private int head;
    private int manyItems;

    public ArrayQueue(int num) {
        data = new Object[num];
        manyItems = 0;
    }

    public void enqueue(E e) {
        data[(head+manyItems) % data.length] = e;
        manyItems++;
    }

    public E dequeue() {
        E temp = (E) data[head];
        head = (head + 1) % data.length;
        manyItems--;
        return temp;
    }

    public E peek() {
        if(isEmpty()) {
            System.out.println("Is empty!");
        }
        return (E) data[head];
    }

    public boolean isEmpty() {
        if(data == null) {
            return true;
        } else {
            return false;
        }
    }
}
```

ArrayStack.java

```java
package project;

public class ArrayStack<E> {
    private Object[] data;
    private int top;

    public ArrayStack(int num) {
        data = new Object[num];
        top = 0;
    }

    public void push (E e) {
        data[top++] = e;
    }

    public E pop() {
        if(isEmpty()) {
            System.out.println("Is empty!");
        }
        return (E) data[--top];
    }

    public E peek() {
        if(isEmpty()) {
            System.out.println("Is empty!");
        }
        return (E) data[top];
    }

    public boolean isEmpty() {
        if(data == null) {
            return true;
        } else {
            return false;
        }
    }
}

```

QueueDriver.java

```java
package project;

public class QueueDriver {

    public QueueDriver() {}

    public static void main(String[] args) {
        ArrayQueue<Integer> numbers = new ArrayQueue<>(5);
        numbers.enqueue(34);
        numbers.enqueue(56);
        numbers.dequeue();
        System.out.println(numbers.peek());
        System.out.println(numbers.isEmpty());

        LinkQueue<String> messages = new LinkQueue<>();
        messages.enqueue("Hello");
        messages.enqueue("World");
        messages.dequeue();
        System.out.println(messages.peek());
        System.out.println(messages.isEmpty());
    }

}

```

Console

<terminated> QueueDriver [Java Application] /Users/cindy/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x

```
56
false
Hello
false
```

Run Last Tool

```java
package project;

public class StackDriver {

    public StackDriver() {
        // TODO Auto-generated constructor stub
    }

    public static void main(String[] args) {
        ArrayStack<Integer> numbers = new ArrayStack<Integer>(5);
        numbers.push(1);
        numbers.push(2);
        numbers.pop();
        System.out.println(numbers.peek());
        System.out.println(numbers.isEmpty());

        LinkStack<String> messages = new LinkStack<String>();
        messages.push("Hello");
        messages.push("World");
        messages.pop();
        System.out.println(messages.peek());
        System.out.println(messages.isEmpty());

    }

}

```

Console

<terminated> StackDriver [Java Application] /Users/cindy/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_6

```
2
false
Hello
false
```