UNIVERSITY OF VIRGINIA, DEPARTMENT OF COMPUTER SCIENCE

Stacks and Queues - Stack / Queue Implementations

Nada Basit and Mark Floryan

January 11, 2022

1 SUMMARY

For this homework, you will be implementing an **array based** stack and a **linked-list based** queue.

1. Download the starter code and import the project into Eclipse
2. Implement the Stack.java class
3. Implement the Queue.java class
4. Verify your implementation using the provided tester class
5. **FILES TO DOWNLOAD:** StacksAndQueues.zip
6. **FILES TO SUBMIT:** StacksAndQueues.zip

1.1 STACK.JAVA

You will implement a basic stack in java. This stack **must be an array-based stack**. Thus, your stack must support the resizing of the capacity of the underlying array. You **may not use an instance of Vector** in order to accomplish this. Instead, your class must have a field that is the underlying array. We are doing this so that you are implementing both a stack, and also learning how vector is implemented at the same time.

The methods you are responsible for are shown below:

```
1   public class Stack<T>{
        // The fields of this stack are given to you as follows
3       private Object[] data;
        private int size = 0;
5       private static final int INITIAL_CAPACITY = 100;

7       /* Constructors not shown. Are implemented for you. */

9       public void push(T data);

11      @SuppressWarnings("unchecked")
        public T pop();
13
        public int size();
15
        public int capacity();
17
        public void resize(int newCapacity);
19  }
```

1.2 TESTING AND SUBMITTING

We are providing a tester class with a main function. This class will add and remove many random elements to your data structure and check that it works correctly. The tester will output an error message if anything goes wrong.

1.3 QUEUE.JAVA

Implement the *Queue* class inside the Queue.java file. The methods you are responsible for are listed below. This Queue **must be a linked-list based queue**. You may use Java's built-in

Linked List (import java.util.LinkedList) or you may use your own implementation from the previous labs.

```
public class Queue<T>{
    public Queue();

    public int size();

    public void enqueue(T data);

    public T dequeue();
}
```

1.4 Testing and submitting

We are providing a tester class with a main function. This class will add and remove many random elements to your data structure and check that it works correctly. The tester will output an error message if anything goes wrong.

To submit, please submit your entire project as a zip.