Recitation 1

Introduction

TA: Simon Markowski
Office: DH 3112
Office Hours - Mon./Thurs. 3:00 - 5:00, or by appointment
Email: smarkows@andrew.cmu.edu

Administrative Material

- Course Website: http://www.andrew.cmu.edu/course/15-211-kesden/
  - Schedule Tab to find out what subjects we cover
  - Assignments Tab for assignment starter code and documentation
  - Bboard is academic.cs.15-211, use it to ask PUBLIC questions
- Homework 1 theory is due Monday and coding is due Tuesday. Start now!
- Documentation in the labs will state constraints and restrictions, so read it CAREFULLY!
- With all homework there will be a theory part due the Friday before the lab is due on Tuesday night

Handin System

- FTP your solution to /afs/andrew/course/15/211-kesden/handin/lab#/*andrewid*/
- Just submit the .java files we tell you to. I don't need anything else to grade it
- You need SSH Secure Shell (Windows) or Fugu (Mac) to do this
- Demonstration

JUnit Testing

JUnit tests are used to test certain cases in a program to determine its correctness and functionality. JUnit testing is very straight forward if you are using Eclipse, and complex otherwise, so try and use Eclipse. In order to make JUnit tests work, you will need to add the JUnit4 library of Java to your build path in Eclipse (Eclipse should do that for you when it recognizes you are writing JUnit tests) and you need to import several items. A sample JUnit test class is written below:

```
import org.junit.Test; //imports JUnit testing environment
import static org.junit.Assert.assertTrue; //imports assertTrue method
import static org.junit.Assert.assertFalse; //imports assertFalse method
import static org.junit.Assert.assertEquals; //import assertEquals method

public class QueueTest //class for JUnit tests
{
    @Test (timeout=100) //the @Test lets JUnit know to run this code, timeout is optional
    public void simpleTest() //after testing, you know what methods failed/passed by name
    {
        //now set up the problem and compare it to solution at the end
        Queue<String> javas = new LinkedList<String>();
        ArrayQueue<String> mine = new ArrayQueue<String>();
        //now you can insert the same things into both
        for (int i = 0; i < 15; i++)
        {
            javas.offer(“” + i);
            mine.enqueue(“” + i);
        }
        //now you can test that the output from your method
        //assertEquals takes a String that will print out when the
        //test fails (optional), the second argument is the expected
        //answer, and the third argument is the actual solution
        assertEquals(“size check”, javas.size(), mine.size());
        //if the test fails, the method quits here, otherwise it continues
        while (!javas.isEmpty())
            assertEquals(“Strings check”, javas.poll(), mine.dequeue());
        //if this test passes then a green bar appears if this was the only test
        //if this test fails, then a red bar appears, and you know something is wrong
    }
}
```

You probably want more than one test to check for edge cases and stress testing. In the end, you will know what your program excels at and where you need to improve it.

Grading

- Use JUnit Tests
- Uses individual methods and tests one at a time to determine whether correct output results from a method call. We will test edge and corner cases, and they will probably be worth more than the general cases
- One of two things happens at the end of the test
    - I get a green bar, meaning you passed all tests

  - I get a red bar, meaning one or more tests were failed. Each test has a point value, so each failed test will lose a specific number of points.

Suggestions to Help you Succeed

- Start early! You have very little time to start with.
- Once you get the assignment and read it, do NOT touch a computer. Plan out what you want to do on paper first. This will prevent you from "hacking out" a long solution or coding something that doesn't work.
- Email and talk to either Kesden or me. We are here to help you.