Name: _______________________

GWU email id: _________________

CS2113 Exam 2

Write your name on the top of this exam and do not open the packet until told to do so by the professor.

45 minutes.

_________ / 41 points

Assume all code compiles unless states otherwise.

1. Presume the following code, which implements a linked-list queue, compiles, but it contains a bug. When two different threads are accessing the same queue, the bug causes the program to crash, but not every single time you run it.
   CIRCLE the line(s) that contain the fault(s), and then fix them while preserving the original behavior of the code. (6pts)

```
public class Queue {

  public void enqueue(String s) {
    if(head == null) {
      head = tail = new Node(s, null);
    } else {
      tail.next = new Node(s, null);
      tail      = tail.next;
    }
  }

  public String dequeue() {
    Node t = head; // we will guarantee this method is never called on an empty queue
    head = head.next;

    if(head == null) {
      tail = null;
    }

    return t.data;
  }

  public boolean empty() {
    return head == null;
  }
}
```

2. **CIRCLE** all true statements; you may justify your answers if you like. (16pts)
   a. In the following code, the only print statements are in `main`, as shown, and in the `thread`'s `run()` method, which prints `yellow` (not shown). True or false, `yellow` may print after `green` is printed:
   ```
   // this code inside main
   thread.start();
   try {
     thread.join();
     System.out.println("green");
   } catch (InterruptedException e) {}
   ```

   b. The `volatile` keyword is applied to a *method* to prevent race conditions.

   c. Race conditions can still happen when two threads are only *reading* the value out of an object shared between them.

   d. It is possible for a single line of code, such as `num++;` to cause a race condition.

e. The requirement, “The GUI must be able to add two integers together” is an example of a non-functional requirement.

f. The `accept()` method of the `ServerSocket` is blocking; it will wait until some client tries to connect on the server’s port before finishing the method call.

g. You cannot catch unchecked exceptions in Java in a try-catch block.

h. Non-functional requirements do not need to be testable.

i. The following code prints `A` and `B`

```
public static void main(String[] args) {
     try {
       int[] x = {1, 2, 3};
       int y = x[5]; // raises out of bounds exception
     } catch (ArrayIndexOutOfBoundsException e) {
       System.out.println("A");
     } catch (Exception e) {
       System.out.println("B");
     }
}
```

3. The following code compiles, but contains a bug; nothing happens when the button is pressed. Fix the code so that when the button is pressed, the text in the label changes: (3 pts)

```
public class HelloGoodbyeEx1 {

    public static void main(String args[]) {
        JFrame f = new JFrame();

        JLabel l = new JLabel("Hello");
        JButton b = new JButton("Click me!");

        f.add(b, BorderLayout.SOUTH);
        f.add(l, BorderLayout.NORTH);

        f.pack();
        f.setVisible(true);
    }
}

public class ButtonClickListenerEx1 implements ActionListener {
    private JLabel label;
```

```
        public ButtonClickListenerEx1(JLabel label) {
            this.label = label; //save the label to modify
        }

        public void actionPerformed(ActionEvent e) {
            if (label.getText().equals("Hello")) {
                label.setText("Goodbye"); //flip text back and forth
            }else{
                label.setText("Hello");
            }
        }
    }
```

4. Complete the code below so that the client writes `hello computer!` to the server at IP address `193.45.234.1` and port `8080`. (7 pts)

```
Socket sock = null;
try{
   sock = new Socket(_________________________________,____________________________);
}catch(Exception e){
    System.err.println("Cannot Connect");
}

try{
    ___________________________________ writer = null;

    writer = new _________________________(_________.______________________________());

    _________________________________________________________________________________;

    // you can omit code for closing buffers and sockets.

}catch(Exception e){
    System.err.print("IOException");
}
// we are done at this point; the socket will not do anything else below here
```

5. Draw a simple sketch of what a GUI window looks like after the following two commands have been run: (2 pts)

```
JFrame f = new JFrame();
f.setVisible(true);
```

6. Imagine each thread below has a print statement inside it's `run` method to print out the object's type once (`ThreadA`, `ThreadB`, or `ThreadC`, which are all subclasses of `Thread`). What are all the possible outputs of the code below? (3 pts)

```
ThreadA a = new ThreadA();
ThreadB b = new ThreadB();
ThreadC c = new ThreadC();

a.start();
b.start();
c.start();
```

7. How is the *Singleton* design pattern related to the `synchronized` keyword? (2 pts)

END OF REQUIRED PORTION OF THE EXAM: the optional midterm coding retake will start at 9:00am. If you wish to complete it, please turn in this exam, leave the room, and come back at 9:00am.

SCRATCH PAPER

SCRATCH PAPER