COMPS203F Specimen Exam Answers

Part I
Question 1

(a)

```
public OddNumber() {
   add(button);
   setDefaultCloseOperation(EXIT_ON_CLOSE);
   pack();
   button.addActionListener(this);
}
```

(b)

```
public void actionPerformed(ActionEvent ae) {
   int oddNumber = Integer.parseInt(button.getText());
   button.setText(oddNumber+2+"");
}
```

Question 2

(a) F:
(b) F:
(c) T:
(d) F:
(e) F:

Question 3

(a)

```
public void upperCase(String inFilename, String outFilename) {
   int byteRead;
   try {
      InputStream in = new FileInputStream(inFilename);
      OutputStream out = new FileOutputStream(outFilename);
      while ((byteRead = in.read()) != -1) {
         if (byteRead >= 'a' && byteRead <= 'z') {
            byteRead -= 'a' - 'A';
         }
         out.write(byteRead);
      }
      in.close();
      out.close();
   } catch (IOException e) {
      System.out.println(e.getMessage());
   }
}
```

(b)

```
FileNotFoundException
```

Question 4

(a)

```
public class Customer {
  private String name;
  private String phone;

  public Customer(String aName, String phone) {
    name       = aName;
    this.phone = phone;
  }
}
```

(b)

```
public class GoodCustomer extends Customer {
  private double bonusRate;

  public GoodCustomer(String aName, String phone, double bonusRate) {
    super(aName, phone);
    this.bonusRate = bonusRate;
  }
}
```

Question 5

(a)

```
double sum = 0;
for (Double price : bookMap.values()) {
    sum += price;
}
double mean = sum / bookMap.size()); // or bookMap.values().size()
```

(b)

```
boolean found = false;
for (String title : bookMap.keySet()) {
    if (title.indexOf("computer") != -1) {
        price = bookMap.get(title);
        System.out.println(title + ": " + price);
        found = true;
    }
}
if (!found) {
    System.out.println("No book with 'computer' in the title");
}
```

Question 6

(a)

```
public String findCourse(Connection conn, String courseCode) {
  try {
    String sql = "select title from course where code=?";
    PreparedStatement pStatement = conn.prepareStatement(sql);
    pStatement.setString(1, courseCode);
    resultSet = pStatement.executeQuery();
    while (resultSet.next()) {
      return resultSet.getString("code");
    }
  } catch (SQLException e) {
    System.out.println("Find Problem: "+e.getMessage());
  }
  return null;
}
```

(b)

```
public Document createXML() {
  Element root = new Element("school");
  Document xmlDoc = new Document(root);
  Element course = new Element("course");
  root.addContent(course);
  course.addContent(new Element("code")
                    .setText("J203"));
  course.addContent(new Element("title")
                    .setText("Java Programming"));
  return xmlDoc;
}
```

Part II

Question 7

(a)

```
import java.awt.*;
import javax.swing.*;
public class BinaryCalculator extends JFrame {
  private JButton button = new JButton("Add");

  public BinaryCalculator() {
    add(button);
    pack();
    setDefaultCloseOperation(EXIT_ON_CLOSE);
  }
```

(b)

```
public class BinaryCalculator extends JFrame {
  private JLabel label = new JLabel("0");
  private JTextField textField = new JTextField(20);
  private JButton button = new JButton("Add");

  public BinaryCalculator() {
    //super("Binary calculator");  // set title, not needed
    add(label, BorderLayout.NORTH);
    add(textField, BorderLayout.CENTER);
    add(button, BorderLayout.SOUTH);
    pack();
    setDefaultCloseOperation(EXIT_ON_CLOSE);
  }
```

(c)

```
import java.awt.event.*;
public class BinaryCalculator extends JFrame implements ActionListener {
  private JLabel label = new JLabel("0");
  private JTextField textField = new JTextField(20);
  private JButton button = new JButton("Add");

  public BinaryCalculator() {
    //super("Binary calculator");
    add(label, BorderLayout.NORTH);
    add(textField, BorderLayout.CENTER);
    add(button, BorderLayout.SOUTH);
    button.addActionListener(this);
    pack();
    setDefaultCloseOperation(EXIT_ON_CLOSE);
  }

  public void actionPerformed(ActionEvent ev) {
    String str = textField.getText();
    label.setText(str);
  }
```

(d)

```
  public void actionPerformed(ActionEvent ev) {
    int operand1 = Integer.parseInt(label.getText(),2);
    int operand2 = Integer.parseInt(textField.getText(),2);
    int result = operand1 + operand2;
    label.setText(Integer.toBinaryString(result));
    pack(); // optional
  }
```

Question 8

(a)

```
private int accumulator; // accumulator A of the CPU
```

getter method:

```
public int getAccumulator() {
   return(accumulator);
}
```

(b)

```
public void load(int anInteger) {
   accumulator = anInteger;
}
public void add(int anInteger) {
   accumulator += anInteger;
}
public void substract(int anInteger) {
   accumulator -= anInteger;
}
```

(c)

```
public void multiply(int anInteger) {
   int temp = accumulator;
   for (int i=0; i<anInteger-1; i++) {
      add(temp);
   }
}
```

(d)

```
public void multiplyRecur(int anInteger) {
   int temp = accumulator;
   if (anInteger >= 2) {
      multiplyRecur(anInteger - 1);
      add(temp);
   }
}
```

**** End ****