香港公開大學
THE OPEN UNIVERSITY OF HONG KONG

COMPS203F
Marker No.:
Total Mark:

Specimen examination paper

INTERMEDIATE JAVA PROGRAMMING AND USER INTERFACE DESIGN (Jan 2013)

dd mmm yyyy **Time Allowed: 2 hours** **14:30–16:30**

Examination Number								
Student Number								

THIS PAPER MUST BE RETURNED

Admissible / Inadmissible materials in this examination:	**Stationery provided to candidates (standard items)**	**Material to be returned to the invigilator at the end of the examination**
1. Calculators are NOT allowed. 2. Dictionaries are NOT allowed. **Violation of the above may lead to disqualification from the examination.**	1 Examination paper	1 Examination paper

Instructions:

1. Answer this examination paper in English.
2. Read the instructions in the examination paper carefully and write your answers in the spaces provided in the examination paper. Answers not recorded in this paper or the appropriate booklet/sheet will not be marked. Begin each question on a new page and write the question number at the top of each page you have worked on.
3. Write any rough work in this examination paper or any booklet you requested, and cross it through afterwards. Rough work will not be marked.
4. Write clearly. It may not be possible to award marks where the writing is very difficult to read.
5. After the invigilator has announced that the examination has started, write your examination number, student number and course code on the cover of the examination paper or other sheet(s) distributed by the invigilator. **Failure to do so will mean that your work cannot be identified**.
6. At the end of the examination, hand over the examination paper to the invigilator.
7. **Do NOT open this examination paper until you are told to do so, otherwise you may be disqualified.**

PART I (60 marks)

(i) *You should attempt* **ALL** *the questions in this part of the examination paper.*
(ii) *There are altogether* ***six*** *questions in this part.*
(iii) *You are advised to spend one hour and ten minutes on this part.*
(iv) *Write all your answers in this examination paper. Answers not recorded on this examination paper will not be marked.*

Question 1

The first two lines of a class `OddNumber`, which displays a button with an odd number on it, are shown below.

```
public class OddNumber extends JFrame implements ActionListener {
private JButton button = new JButton("1");
```

The initial appearance of the GUI is:

(a) Write the constructor `OddNumber()` to generate the GUI and add the object itself as the action listener of the button.

[6 marks]

(b) Complete the action listener below to increase the integer on the button by 2 when the button is clicked.

```
public void actionPerformed(ActionEvent ae) {
```

[4 marks]

Question 2

Tick the "true" box if a statement is true and tick the "false" box otherwise.

(a) A `private` method of an object can be accessed directly by all other objects.

☐ true ☐ false [2 marks]

(b) A "`super();`" statement can be put anywhere within a constructor.

☐ true ☐ false [2 marks]

(c) In Java, if `classB` is a subclass of `classA`, `classB` cannot be a subclass of `classC`.

☐ true ☐ false [2 marks]

(d) The methods of an abstract class cannot be overloaded.

☐ true ☐ false [2 marks]

(e) All the methods of an abstract class must be overridden by its subclasses.

☐ true ☐ false [2 marks]

Question 3

(a) Write a method `upperCase(String inFilename, String outFilename)` of a class `Conversion` which reads a binary byte file with name `inFilename` and converts every lower case English letter to the corresponding upper case letter. Other bytes are unchanged. Handle possible exception(s) by outputting a suitable message using the `getMessage()` method if an exception occurs.

```
public void upperCase(String inFilename, String outFilename){
```

[9 marks]

(b) What exception will occur when the method in part (a) is executed but the file with filename `inFilename` does not exist? Give the most specific one.

[1 mark]

Question 4

(a) A bank has many customers which are modelled by a class `Customer`. This class has two attributes, namely the name and the phone number. Their variable names are `name` and `phone` respectively. Write the class `Customer` with these two attributes. Also write the constructor `Customer(String aName, String phone)` which initializes the two attributes using the parameters `aName` and `phone` respectively.

[5 marks]

(b) Some customers have their total balances exceed a certain (large) value. Model them using the class `GoodCustomer`, which is a subclass of `Customer`. This new class has an additional attribute to record the bonus interest rate (`bonusRate`), which is added to the normal interest rate to form the higher interest rate given to these customers. Write the class `GoodCustomer` and the constructor `GoodCustomer(String aName, String phone, double bonusRate)` which initializes the three attributes, some of which are achieved using the constructor of `Customer`.

[5 marks]

Question 5

A library stores the information of its books in a Java map `bookMap` (`Map <String, Double>`) in which the book title is the unique key and the price is the value.

(a) Assume there are many books in the map. Write a program segment to find the average price of the books.

[4 marks]

(b) Write a program segment to print each title with the word "computer" in it and the corresponding price. If there is no book with the string "computer" in its title, "No book with 'computer' in the title" should be output.

[6 marks]

Question 6

(a) A table of a database has been created using

```
create table course (code varchar(15), title varchar(50));
```

Write a Java method `findCourse(Connection conn, String courseCode)` to return the course title of the course with course code `courseCode` using `conn`, which is ready to be used. Return `null` if the course code is not found. Use `try-catch` to handle the exception(s) but no `import` statements are needed to be written.

[5 marks]

(b) Write a Java method `createXML()` to return an XML `Document` object which contains the following content :

```
<?xml version="1.0" encoding="UTF-8"?>
<school>
  <course>
    <code>J203</code>
    <title>Java Programming</title>
  </course>
</school>
```

No `import` statements, no formatting, and no `try-catch` are needed to be written.

[5 marks]

[END OF PART I]

PART II (40 marks)

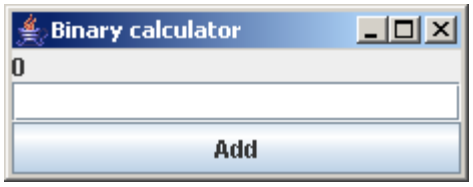
(i) *You should attempt* **ALL** *questions. Each question is worth 20 marks.*
(ii) *Show all your work steps.*
(iii) *You are advised to spend one hour and twenty minutes on this part.*
(iv) *Write all your answers in this examination paper. Answers not recorded on this examination paper will not be marked.*

Question 7

(a) Create the GUI on the right using the constructor of a class `BinaryCalculator`, which is a subclass of `JFrame`. Remember to include the `import` statement(s). There is no need to handle actions yet.

[5 marks]

(b) Using `BorderLayout` manager, state the modifications in part (a) to enhance the GUI to the one on the right. There is also no need to handle actions at this moment.



[4 marks]

[This page is intentionally left blank and should be used for rough work only.

Anything written here will not be marked.]

(c) State the modifications in (b) to handle actions. The action required in this part is to replace the text on the label by the text in the text field when the button is pressed. Remember to include the `import` statement(s).

[4 marks]

(d) State the modifications in (c) to handle the action of binary addition, in which the sum of the binary numbers on the label and in the text field is used to replace the original number on the label when the "Add" button is pressed. You can assume the binary number in the text field (and that on the label) is valid. **Hint**: the following functions of the `Integer` class may be useful:

- `parseInt(String s, int b)`: parses the string `s` using base `b` (instead of base 10) and returns an integer in decimal form (e.g., returns 5 if the string is "101");
- `toBinaryString(int i)`: returns a binary number which equals to integer `i` as a string (e.g., returns "110" if the integer is 6).

[7 marks]

[This page is intentionally left blank and should be used for rough work only.
Anything written here will not be marked.]

Question 8

A simple CPU with a single accumulator A supports the following simple operations:

- load X : load the integer X into the accumulator A (A = X)
- add X : add the integer X to A and put result back to A (A = A + X)
- subtract X : subtract the integer X from A and put result back to A (A = A – X)
- multiply X : multiply A by integer X and put result back to A (A = A * X)

For example, "load 6, then add 3" means "load the value 6 into accumulator A, then add 3 to A". The accumulator A will contain the result 9. A Java class `CPU` is used to simulate the operations of the CPU.

(a) A variable is needed for the class `CPU` to represent the accumulator. Declare the variable with comments describing it and write its getter methods.

[3 marks]

(b) Write the methods `load()`, `add()` and `subtract()` of `CPU` (with suitable parameters) which perform the corresponding load, add and subtract operations respectively as described in part (a).

[7 marks]

[This page is intentionally left blank and should be used for rough work only.
Anything written here will not be marked.]

(c) To save hardware cost, the simple CPU implements multiplication by additions. For example,

7 x 3 = 7 + 7 + 7.

Using a loop and the method `add()`, write a method `multiply()` with suitable parameter(s) to implement the multiply operation in part (a) by additions. (Hint: You may need a local variable in your method.)

[6 marks]

(d) Modify the method `multiply()` in part (c) to use recursion instead of a loop to achieve the same purpose. (Hint: 7 x 3 = (7 x 2) + 7 = ((7 x 1) + 7) + 7) **[recursion not needed in real exam]**

[4 marks]

[This page is intentionally left blank and should be used for rough work only.
Anything written here will not be marked.]

Appendix: Concise Java Statement Examples and Partial Method List

This appendix is provided to reduce the load of memorizing the syntax and methods learnt. This is not a complete reference and total correctness is not guaranteed. Some methods not listed here need to be used.

Statement Examples

```
int a, b, c, e;
int[] f = new int[9];
double d;
TicketCounter tc = new TicketCount();
if (a == b && b != 1 || c <= 0) {
  d = 0;
} else {
  d = 1;
}
switch (e+2) {
  case 2 : case 5:
     f[4] = (int) d; break;
  default :
     tc.increase(); break;
}
```

```
public class Example {
  boolean good;
  int j=0, k=0, m=0, n=0;
  public double loops(String s) {
     for (int i=0; i<n; i++) {
        j += i;
     }
     do {
        k = k + 2;
     } while ( k < 5 );
     while (true || k > 2) {
        m++;
     }
     return (double) k;
  }
}
```

Method List

```
Collection --    add(o), contains(o), isEmpty(), remove(o), size(), toArray()
  List --           add(i,o), get(i), indexOf(o), lastIndexOf(o), remove(i),
                    set(i,o)
  Set --            <see Collection>

Container --     add(co), add(co, i)

File --          exists(), File(s), isFile(), isDirectory(), length()

InputStream --   read(), read(b[])
  FileInputStream -- FileInputStream(f), FileInputStream(s)

JButton --       JButton(s), setText(s)
JCheckBox --     isSelected(), JCheckBox(s), setSelected()
JFrame --        getContentPane(), JFrame(s), pack(), setJMenuBar(mb),
                 setLayout(lm), setSize(i,j), setVisible(bo), show()
JLabel --        JLabel(s)
JMenuBar --      add(m), JMenuBar()
JMenu --         add(mi), addSeparator(), JMenu(s)
JMenuItem --     JMenuItem(s)
JOptionPane --   showMessageDialog(o,s), showInputDialog(o,s),
                 showConfirmDialog(o,s)
JPanel --        add(co), add(co, i)
JRadioButton --  isSelected(), JRadioButton(s), setSelected()
JTextField --    JTextField(i)
JTextArea --     JTextArea(i,j)

Map --           containsKey(k), containsValue(o), get(k), keySet(),
                 put(k,o), remove(k), values()

OutputStream --  write(i), write(b[])
  FileOutputStream -- FileOutputStream(f), FileOutputStream(s)

String --        charAt(i), compareTo(s), equals(o), indexOf(c), indexOf(s),
                 length(), replace(c,c), substring(i,j), toCharArray(),
                 toLowerCase(), toUpperCase()
```

where `b: byte, b[]: byte array, bo: boolean, c:char, co: GUI component, f: File, i,j: int, k: key(Object), lm: layout manager, m: menu, mb: menu bar, mi: menu item, o: Object, s: String`

[END OF EXAMINATION PAPER]