

MARKER'S USE ONLY

QS1.....

QS2.....

QS3.....

Family Name .....

Given Names .....

Signature .....

Student Number .....

Log-in Identifier s.....

THE UNIVERSITY OF QUEENSLAND

Open Book Practical Examination

COMP1500: Introduction to Programming
COMP7901: Software Engineering

PAPER S (Sample Paper)

Time: Two hours for working with the computer
Thirty minutes for perusal before examination begins

You are not permitted to bring floppy disks into the examination room.

This exam comprises three questions each worth 10 marks (making a total of 30 marks for the paper). The code for the three questions will be found respectively in the folders QuestionS1, QuestionS2 and QuestionS3. Each of these three folders is in the folder

C:\genesis\projects\Exams\PaperS

and each contains respectively a file QuestionS1.kawa, QuestionS2.kawa and QuestionS3.kawa that opens a project in Kawa containing all the files needed for that question.

Each question involves the preparation of one or more .java files. During the examination you will need to submit the .java files you have prepared for each question using the online submission system. To do this go to http://www.itee.uq.edu.au/~comp1500 and click on Exam Submission and follow the instructions.

Record here your Submission ID for each question:

Question S1 ..... Question S2 ..... Question S3 .....

This paper must be handed to the invigilator before leaving the examination room.

## Question S1 (10 marks)

Open the project *QuestionS1* in Kawa. This project contains the following (incomplete) code for the class `AverageAndCopy`:

```
import java.io.*;
import genesis.*;

public class AverageAndCopy {

    public static void main (String [] args) throws IOException {
        int [] data = {5, 12, -34, 78, -92, 10};
        double average = averageOfPositives(data);
        Transcript.println(average);
        copyToFile("data.txt", data, 3);
    }

    public static double averageOfPositives (int [] intArray) {
        // ...
        // complete the code for this method (Question S1(a))
        // ...
    }

    public static void copyToFile (String name, int [] xs, int w)
        throws IOException {
        // ...
        // complete the code for this method (Question S1(b))
        // ...
    }
}
```

## Question S1(a) (5 Marks)

Complete the code for the method `averageOfPositives`. This method has one parameter denoting an array of integers. It returns the average of the positive integers (i.e. those integers greater than 0) in the array. If there are no positive integers in the array the method returns 0.

## Question S1(b) (5 Marks)

Complete the code for the method `copyToFile`. This method has three parameters. The first parameter is a string denoting the name of a text file; the second denotes an array of integers; the third is an integer, `w`. The method outputs the integers in the array to the file, with the integers output `w` to a line and with a space after each integer on a line.

*Note:* If you compile and run the class `AverageAndCopy` when you have completed the coding required by this question, first you should get the number **26.25** printed in the Transcript window, and second the text file *data.txt* should be added to the folder containing the code for this project. This file should contain the following two lines of numbers:

```
5 12 -34
78 -92 10
```

When you have completed parts (a) and (b) of Question S1 (or as much as you can complete) submit the entire file *AverageAndCopy.java* using the online submission system.

**Record your Submission ID for Question S1 in the space provided on the cover.**

## Question S2 (10 Marks)

Open the project *QuestionS2* in Kawa. This project contains the class `Employee` whose code is given below:

```
public class Employee {

    private String name;
    private double hoursWorked; // hours worked this week
    private double hourlyRate = 30; // $'s per hour paid

    public Employee (String ident) {
        name = ident;
    }

    public void setHoursWorked (double worked) {
        hoursWorked = worked;
    }

    public double getRate ( ) {
        return hourlyRate;
    }

    public String toString ( ) {
        return name+"\n"
            +"hours worked this week: "+hoursWorked+"\n"
            +"hourly rate: $"+getRate()+"\n"
            +"earnings this week: $"+getRate()*hoursWorked;
    }
}
```

An object of the class `Employee` denotes a basic employee and records the name of the employee, the hours they have worked this week and the hourly rate at which they are paid.

To test this class, we have supplied the class `TestEmployee` whose code is:

```
import genesis.*;

public class TestEmployee {

    public static void main (String [] args) {
        Employee emp = new Employee("Tom Jones");
        emp.setHoursWorked(40);
        Transcript.println(emp);
    }
}
```

Select the class `TestEmployee` as the Main Class and run the application. The following output will appear in the Transcript window:

```
Tom Jones
hours worked this week: 40.0
hourly rate: $30.0
earnings this week: $1200.0
```

Your task in Question S2 is to code a class `Boss` that extends the class `Employee`. That is, the class `Boss` will be a subclass of `Employee`.

An object of the class `Boss` is a type of employee. However, a boss's hourly rate of pay is variable depending on the level at which the boss is employed. The hourly rate of pay for a boss is calculated at \$50 per hour plus 50 cents per hour for each level. Thus a boss at level 3 gets

$$\$50 + 3 * \$0.50 = \$51.50 \text{ per hour.}$$

In addition, a boss is paid a vehicle allowance at a standard rate of 40 cents for each kilometre driven each week.

To help you code the class `Boss`, we have supplied the following (incomplete) code:

```
public class Boss extends Employee {

    private double allowancePerKilometre = 0.40;
    private int kilometresTravelled; // kilometres travelled this week
    private double bossHourlyRate = 50;
    private int level;

    public Boss (String id, int theLevel) {
        // ...
        // complete the code for this constructor
        // ...
    }

    public double getRate ( ) {
        // ...
        // redefine this method
        // ...
    }

    public void setKilometres (int kilometres) {
        // ...
        // complete the code for this method
        // ...
    }

    public String toString ( ) {
        // ...
        // redefine this method
        // ...
    }
}
```

So that you know what your `Boss` class is meant to do, we have supplied the class `TestBoss` whose code is below:

```
import genesis.*;

public class TestBoss {

    public static void main (String [] args) {
        Boss boss = new Boss("Chief Sitting Bull", 3);
        boss.setHoursWorked(36);
        boss.setKilometres(100);
        Transcript.println(boss);
    }
}
```

After you have completed coding your class `Boss`, select the class `TestBoss` as the Main Class and run the application. If you have coded the class `Boss` correctly the following output should appear in the Transcript window:

```
Chief Sitting Bull
hours worked this week: 36.0
hourly rate: $51.5
earnings this week: $1854.0
kilometres travelled this week: 100
vehicle allowance this week: $40.0
```

Notice in particular that the `toString` method in the class `Boss` first supplies the same information as the `toString` method in the superclass `Employee` but then in addition supplies information about both the distance travelled and the vehicle allowance earned as a consequence.

When you have completed Question S2 (or as much as you can complete) submit the entire file `Boss.java` using the online submission system.

**Record your Submission ID for Question S2 in the space provided on the cover.**

### Question S3 (10 Marks)

Open the project `QuestionS3` in Kawa, select the class `SqGUIDemo` as the Main Class and run the application. A GUI opens that has one text field and two action buttons, 'Square Root' and 'Square'. If a number  $n$ , say, is displayed in the text field (e.g. by selecting the text field with the mouse and typing from the keyboard) and the 'Square Root' button pressed, the square root of  $n$  is displayed in the text field; on the other hand, if the 'Square' button is pressed the square of  $n$  is displayed in the text field.

Your task in this question is to code a class `SqGUI` which behaves the same as this demonstration. To assist you, we have supplied the following `Main` class:

```
public class Main {

    public static void main (String [ ] args) {
        SqGUI win = new SqGUI( );
        win.setVisible(true);
    }
}
```

and the following (incomplete) code for the class SqGUI:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import genesis.*;

public class SqGUI extends JFrame {

    private JTextField number;

    public SqGUI ( ) {
        setTitle("Squaring");
        setBounds(400,150,200,130);
        number = new JTextField("0", 17);
        JButton squareRoot = new JButton("Square Root");
        Container c = getContentPane( );
        JPanel pMid = new JPanel( );
        pMid.add(number);
        c.add(pMid, "Center");
        JPanel pBot = new JPanel( );
        pBot.add(squareRoot);
        c.add(pBot, "South");
    }
}
```

If you select the class `Main` as the Main Class and run the application, a GUI opens with one text field and one action button, 'Square Root'. However, when the button is pressed nothing happens.

Question S3(a) (5 Marks)

Modify the code for the class `SqGUI` so that when a number  $n$ , say, is displayed in the text field and the 'Square Root' button pressed, the square root of  $n$  is displayed in the text field.

Question S3(b) (5 Marks)

Further modify the code for the class `SqGUI` to add an action button 'Square' to the top of the GUI, as in the demonstration. When this button is pressed the number displayed in the text field is replaced by its square.

When you have completed parts (a) and (b) of Question S3 (or as much as you can complete) submit the entire file `SqGUI.java` using the online submission system.

**Record your Submission ID for Question S3 in the space provided on the cover.**