

Name:

ID Number:

COMP102: Test 2 A

5 September, 2006

Note: Answers were included for some of the parts of questions 1 and 2 by mistake. Test 2 B contained alternative questions for this part of the test.

Instructions

- Time allowed: **90 minutes** ($1\frac{1}{2}$ hours).
- Answer **all** the questions.
- There are 90 marks in total.
- Write your answers in the boxes in this test paper and hand in all sheets.
- If you think some question is unclear, ask for clarification.
- There is some Java documentation at the end of the test paper.
- This test will contribute 20% of your final grade, if it helps your grade.
- Non-electronic translation dictionaries and calculators without a full set of alphabet keys are permitted.

Questions

Marks

1. Basic Java	[33]	<input type="text"/>
2. Conditionals	[22]	<input type="text"/>
3. Using a DrawingCanvas	[10]	<input type="text"/>
4. Loops with Files	[10]	<input type="text"/>
5. Objects and Fields	[15]	<input type="text"/>
	TOTAL:	<input type="text"/>

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Question 1. Basic Java

[33 marks]

(a) [3 marks] What will the following fragment of Java print out?

```
double size = 4.86;
size = size / 2;
System.out.println("size = " + size);

if ( size > 1 )
    size = size + 1.0;
else
    size = Math.min(-4.86, size);
System.out.printf("size = %.1f\n", size);
```

(b) [4 marks] Consider the following drawShape method.

```
public void drawShape(int n){
    canvas.setForeground(Color.red);
    if (n<=1 || n == 4)
        canvas.setForeground(Color.blue);
    if (n > 2 && n < 5)
        canvas.setForeground(Color.green);
    canvas.fillRect(10,10,100,100);
}
```

(i) [2 marks] What colour rectangle will be drawn if drawShape(1) is called?

(ii) [2 marks] What colour circle will be drawn if drawShape(4) is called?

(Question 1 continued on next page)

(Question 1 continued)

(c) [4 marks] What will the following fragment of Java print?

```
int k = 7;
while (k < 10){
    System.out.println("k = " + k);
    k++;
}
```

(d) [4 marks] Write a fragment of Java using a **for** statement that behaves the same as the fragment in (c) above.

```
for(
}

```

(e) [4 marks] Write a method called `triple` which has one `int` parameter and returns a value of type `int`. `triple` should return the value that is three times the value of its parameter.

```
public
}

```

(Question 1 continued on next page)

Question 2. Conditionals

[22 marks]

The middle method below is intended to return the middle value of three integers — the value that would be second if the values were put in order. For example, `middle(8, 17, 2)` should return 8.

```
/* This version of middle has errors */
public int middle(int a, int b, int c){
    if (a<b && b < c)
        return b;
    else if (a<b && c<b)
        return c;
    else
        return a;
}
```

The middle method above has errors.

(a) [3 marks] What will `middle(8, 17, 2)` actually return?

(b) [3 marks] What will `middle(17, 8, 2)` return?

(c) [6 marks] Show two further calls to `middle` using the arguments 2, 8, 17 in different orders, one of which will return the correct answer and the other will return an incorrect answer.

Correct:

Incorrect:

(Question 2 continued on next page)

(Question 2 continued)

(d) [10 marks] Write a correct version of the middle method.

```
public int middle(int a, int b, int c){
```

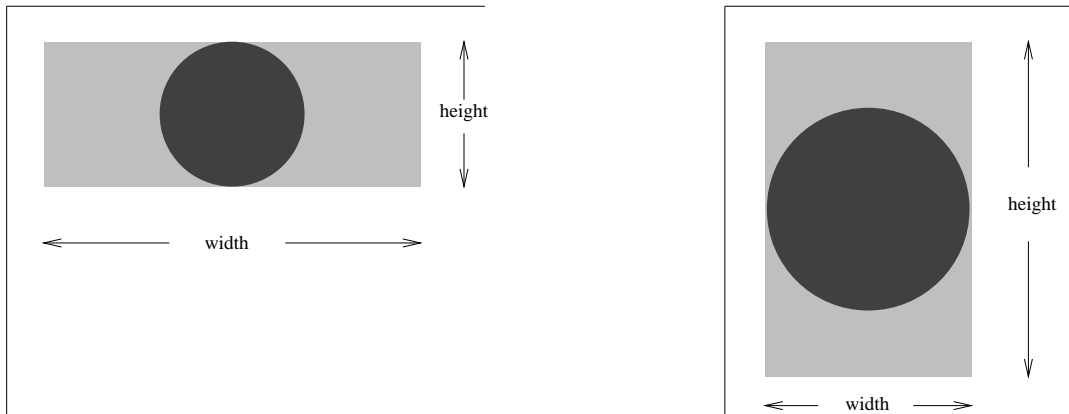
```
}
```

Question 3. Using the DrawingCanvas

[10 marks]

Complete the following `drawIcon` method so that it draws a gray rectangle with a black circle inside it. The `drawIcon` method has two parameters that are the width and the height of the rectangle and a third parameter giving the canvas to draw on.

The top left corner of the rectangle should be placed at (20,20). The circle should be centered in the rectangle and should just touch the long sides of the rectangle. For example, depending on the width and height, the `drawIcon` method might draw one of the following.



There is documentation on the `DrawingCanvas` and `Color` classes at the end of the test paper.

```
public void drawIcon(int width, int height, DrawingCanvas canvas){
```

```
}
```

Question 4. Loops with files

[10 marks]

Complete the `averageWordLength` method below so that it returns the average length of the words in a file. You may assume that that the file contains nothing but words and numbers. The method should only count the words, not the numbers (integers or doubles). The parameter of the method specifies the name of the file. The method should close the file after reading all the words.

There is documentation on the `String` and `Scanner` classes at the end of the test paper.

```
public double averageWordLength(String fileName){
```

```
}
```

Question 5. Objects and Fields

[15 marks]

This question concerns a program for a visual simulation of rocks being dropped to the ground from different heights. The program contains two classes, `RockSimulator` and `Rock`.

Part of `RockSimulator` is shown below. The `simulate` method contains a loop that repeatedly constructs a new `Rock` object, and then simulates it dropping by calling methods on the object.

The `Rock` class on the facing page represents the state of a falling rock, including its height above the ground and its current speed. At each time step after it is dropped, the height of a rock will decrease, and the speed will increase. The amount the height decreases will depend on the current speed.

```

public class RockSimulator{
    private DrawingCanvas canvas;

    public RockSimulator( ){
        :
        :
    }

    /* Keeps dropping rocks from random heights */
    public void simulate( ){
        while (true) { // drop rocks forever
            Rock rock = new Rock(Math.random()*400); // make new rock at a random height
            while ( !rock.atGround( ) ){ // simulate it falling to the ground
                rock.step( );
                this.canvas.clear( );
                rock.draw(this.canvas);

                try{ Thread.sleep(10); }catch(Exception e){} // wait for 0.01 seconds
            }
        }
    }
}

```

(Question 5 continued on next page)

Complete the definition of the Rock class:

- Declare two fields to store the state information.
- Complete the constructor, to initialise the state of the rock to have a speed of 0, and have height above ground specified by the argument to the constructor.
- Complete the atGround method to return true if the rock has hit the ground, and false if the rock is still above the ground.
- Complete the step method to change the state by one 10 millisecond time step: the height should decrease by the speed \times 0.01 and the speed should increase by 0.098.

```

class Rock{
    // fields for the current speed and height above ground.

    /* Constructor: sets initial speed to 0 and initial height to the given value */
    public Rock(double h){

    }

    /* atGround returns true only if the rock has reached ground level . */
    public boolean atGround( ){

    }

    /* Decreases the height by (speed x 0.01), and increases the speed by 0.098 */
    public void step( ){

    }

    /* Draws the rock at the current height */
    public void draw(DrawingCanvas canvas){
        int y = 400-(int)this.height;
        canvas.fillOval(100, y, 10, 10);
    }
}

```

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Brief and partial documentation of some classes and methods

PrintStream class:

```
public PrintStream (File f); // Note, System.out is a PrintStream object
public void close (); // Constructor, for printing to a file
public void print (String s); // Close the file (if it is wrapping a File object)
public void print (int i); // Prints s with no newline
public void print (double d); // Prints i with no newline
public void println (); // Prints d with no newline
public void println (String s); // Prints a newline
public void println (int i); // Prints s followed by newline
public void println (double d); // Prints i followed by newline
public void printf (String format, ...); // Prints d followed by newline
// Prints the format string, inserting the remaining
// arguments at the %'s in the format string:
// %s for Strings.
// %d for ints, (%3d: use at least 3 characters),
// %.2f for 2dp doubles,
// (%6.2f: at least 6 characters and 2dp),
// Use \n for newline
```

Scanner class:

```
public Scanner (InputStream i); // Constructor. Note: System.in is an InputStream
public Scanner (File f); // Constructor, for reading from a file
public Scanner (String s); // Constructor, for reading from a string
public boolean hasNext(); // Returns true if there is more to read
public boolean hasNextInt(); // Returns true if the next token is an integer
public boolean hasNextDouble(); // Returns true if the next token is a number
public String next(); // Returns the next token (chars up to a space/line)
public String nextLine(); // Returns the next line
public int nextInt(); // Returns the integer value of the next token
// (throws exception if next token is not an integer)
public double nextDouble(); // Returns the double value of the next token
// (throws exception if next token is not a number)
public void close(); // Closes the file (if it is wrapping a File object)
```

File class:

```
public File (String fname); // Constructor. Creates a File object attached to the
// file with the name fname
```

Integer class:

```
public static final int MAX_VALUE; // The largest possible int:  $2^{(31-1)}$ 
public static final int MIN_VALUE; // The smallest possible int:  $-2^{(31)}$ 
```

Double class:

```
public static final double MAX_VALUE; // The largest possible double: just under  $2^{(1024)}$ 
public static final double MIN_VALUE; // The smallest possible positive nonzero double
public static final double POSITIVE_INFINITY; // positive infinity (greater than any number)
public static final double NEGATIVE_INFINITY; // negative infinity (less than any number)
public static final double NaN; // The Double that is Not a Number ("undefined")
```

(Continued on next page)

String class:

```
public int length (); // Returns the length (number of characters) of the string
public boolean equals(String s); // String has same characters as s
public boolean equalsIgnoreCase(String s); // String has same characters as s, ignoring their case
public boolean startsWith(String s); // First part of string matches s
public boolean contains(String s); // s matches some part of the string
public String substring(int j, int k) // Returns substring from index j to index k-1
public int indexOf(String s); // Returns -1 if it does not contain s anywhere
// otherwise, returns the index of where s first matches
```

Math class:

```
public static double sqrt(double x); // Returns the square root of x
public static double min(double x, double y); // Returns the smaller of x and y
public static double max(double x, double y); // Returns the larger of x and y
public static double abs(double x); // Returns the absolute value of x
public static int min(int x, int y); // Returns the smaller of x and y
public static int max(int x, int y); // Returns the larger of x and y
public static int abs(int x); // Returns the absolute value of x
```

DrawingCanvas class:

```
public void clear (); // Clears the drawing canvas
public void setForeground(Color c); // Change the colour for later commands
public void drawLine(int x, int y, int u, int v); // Draws line from cd{(x, y) to cd{(u, v)
public void drawRect(int x, int y, int wd, int ht); // Draws outline of rectangle
public void fillRect (int x, int y, int wd, int ht); // Draws solid rectangle
public void clearRect(int x, int y, int wd, int ht); // Draws clear rectangle
public void drawOval(int x, int y, int wd, int ht); // Draws outline of oval
public void fillOval (int x, int y, int wd, int ht); // Draws solid oval
```

Color class:

```
public Color(int red, int green, int blue); // Make a colour; arguments must be 0..255
Color.gray, Color.blue, Color.red, // Some of the predefined colours
Color.green, Color.black, Color.white
```