

Family Name:

Other Names:

ID Number:

COMP102: Test 2 | Model Solutions

6 September, 2007

Instructions

- Time allowed: **90 minutes** ($1\frac{1}{2}$ hours).
- There are 90 marks in total.
- Answer **all** the questions.
- 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]

2. Using the DrawingCanvas

[10]

3. Debugging

[20]

4. Loops with Files

[12]

5. Objects and Fields

[15]

TOTAL:

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?

```
String mon = "birthday";
String tue = "lecture";
String wed = "home";
System.out.println(mon + " tue " + wed);
int num = "Saturday".length();
if ( mon.length() > wed.length() )
    num = num + tue.length();
else
    num = mon.length() * wed.length();
System.out.printf("num = %d\n", num);
```

```
birthday tue home
num = 15
```

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

```
public void printMessage(int n){
    String message = "Hello";
    if ( n < 0 && n > 4 )
        message = message + "Tim";
    else if ( n < -2 || (n > 3 && n < 10) )
        message = message + "Tam";

    if ( n == 4 )
        message = message + "Tom";

    message = message + "Tum";
    System.out.println(message);
}
```

(i) [2 marks] What will be printed if printMessage(-4) is called?

```
HelloTamTum
```

(ii) [2 marks] What will be printed if printMessage(4) is called?

```
HelloTamTomTum
```

(Question 1 continued on next page)

(Question 1 continued)

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

```
int k = 5;
while ( k > 1 ){
    k--;
    System.out.println("k = " + k);
}
```

```
k = 4
k = 3
k = 2
k = 1
```

(d) [4 marks] Write a fragment of Java that will print out every odd number from 5 to 555 (including 555), one number per line.

```
for ( int j = 5; j<=555; j=j+2 ){
    System.out.println(j);
} //OR
int j = 5;
while ( j<=555 ){
    System.out.println(j);
    j = j+2;
}
```

(e) [4 marks] Write a method named `repeated` which has one *String* parameter and returns a value of type *String*. `repeated` should return the string that consists of two repetitions of its parameter. For example, `repeated("Allo")` should return the string "AlloAllo".

```
public String repeated (String str){
    return str+str;
}
```

(Question 1 continued on next page)

(Question 1 continued)

(f) [7 marks] The following `guessIt` method has one `String` parameter, `secret`. Complete the method so that it repeatedly prompts the user to enter a word, until the user types a word equal to `secret`. It should then print out the message "You won".

```
public void guessIt(String secret){
    Scanner scan = new Scanner(System.in);
    while (true){
        System.out.print("Enter a word: ");
        if (scan.next().equals(secret)){
            System.out.println("You won");
            break;
        }
    }
//OR
    System.out.print("Enter a word: ");
    while (!scan.next().equals(secret)){
        System.out.print("Enter a word: ");
    }
    System.out.println("You won");
//OR
    do
        System.out.print("Enter a word: ");
    while (!scan.next().equals(secret));
    System.out.println("You won");
```

(Question 1 continued on next page)

(Question 1 continued)

(g) [7 marks] Suppose the file called `data.txt` contains the text:

```
16 March was 82 days ago
```

What will the following fragment of Java print?

```
try{
    Scanner sc = new Scanner(new File("data.txt"));
    int n = 0;
    String msg = "Ans: ";
    while ( sc.hasNext() ){
        if (sc.hasNextInt()) {
            n = n + sc.nextInt( );
        }
        else {
            msg = sc.next( ) + msg;
            n++;
        }
        System.out.println(n + " -- " + msg);
    }
    sc.close( );
}
catch(Exception e){System.out.println("File error: "+ e);}
```

```
16 -- Ans:
17 -- MarchAns:
18 -- wasMarchAns:
100 -- wasMarchAns:
101 -- dayswasMarchAns:
102 -- agodayswasMarchAns:
```

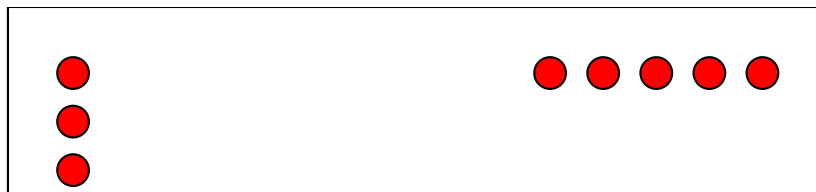
Question 2. Using the DrawingCanvas

[10 marks]

Complete the following `drawCircles` method so that it draws a horizontal or a vertical line of small circles. The first two parameters of the method (`x`, and `y`) specify the position of the first circle. The third parameter (`num`) specifies how many circles to draw, and the fourth parameter (`horizontal`) has the value `true` if the line of circles should go to the right from the first circle, and the value `false` if the line of circles should go down from the first circle. The last parameter (`canvas`) is the `DrawingCanvas` to draw the circles on.

Each circle should be 6 pixels in diameter, and should be red with a black outline. The circles should be 12 pixels apart.

For example, `drawCircles(10, 10, 3, false, canvas)` might draw the circles on the left;
`drawCircles(100, 10, 5, true, canvas)` might draw the circles on the right.



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

```

public void drawCircles(int x, int y, int num, boolean horizontal, DrawingCanvas canvas){
    for( int i = 0; i<num; i++){
        canvas.setColor(Color.red);
        canvas.fillOval(x, y, 6, 6);
        canvas.setColor(Color.black);
        canvas.drawOval(x, y, 6, 6);
        if (horizontal){
            x = x + 12;
        }
        else {
            y = y + 12;
        }
    }
}

```

Question 3. Debugging

[20 marks]

The `printTable` method is intended to print a triangular table of integers of a given size. For example, if the parameter is 5, it should print the table on the left; if the parameter is 4, it should print the table on the right:

<pre>printTable(5) ⇒ row 1: 1 row 2: 2 3 row 3: 4 5 6 row 4: 7 8 9 10 row 5: 11 12 13 14 15</pre>	<pre>printTable(4) ⇒ row 1: 1 row 2: 2 3 row 3: 4 5 6 row 4: 7 8 9 10</pre>
---	--

Note that the last number in the table for n should be $n(n + 1)/2$.

The following version of `printTable` has errors:

```
public void printTable (int n){
    // This version of printTable has errors
    for (int row=1; row<n; row++){
        System.out.printf("row %2d: ", row);
        for (int col=1; col<n; col++){
            if (col <= row)
                System.out.printf(" %2d ", row*row/2 + col);
            else
                System.out.printf("  ");
        }
        System.out.println( );
    }
}
```

(a) [6 marks] What will this version of `printTable(5)` actually print out?

```
row 1:  1
row 2:  3  4
row 3:  5  6  7
row 4:  9 10 11 12
```

(b) [4 marks] On the code for `printTable` above, circle at least four errors in the code.

Identify the errors on the code above!

(Question 3 continued on next page)

(Question 3 continued)

(c) [10 marks] Write a correct version of the printTable method.

```

public void printTable ( int n){
    int num = 1;
    for ( int row=1; row<=n; row++){
        System.out.printf("row %2d: ", row);
        for ( int col=1; col<=n; col++){
            if ( col < (n-row+1))
                System.out.printf("  ");
            else
                System.out.printf(" %2d ", num++);
        }
        System.out.println( );
    }
} // OR
public void printTable ( int n){
    for ( int row=1; row<=n; row++){
        System.out.printf("row %2d: ", row);
        for ( int col=1; col<=n; col++){
            if ( col < (n-row+1))
                System.out.printf("  ");
            else
                System.out.printf(" %2d ", (row*(row-1))/2 +(col+row-n));
        }
        System.out.println( );
    }
} // OR
public void printTable ( int n){
    for ( int row=1; row<=n; row++){
        System.out.printf("row %2d: ", row);
        for ( int col=1; col<=n; col++){
            if ( col < (n-row+1))
                System.out.printf("  ");
            else
                System.out.printf(" %2d ", row*(row+1)/2 + col - n);
        }
        System.out.println( );
    }
}

```

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 4. Loops with files

[12 marks]

The `extractNumbers` method should read a file containing a mixture of words and numbers and write out just the numbers into another file, ignoring all the words. The output file should be formatted with five numbers on each line, separated by single spaces. The method should close the files after reading/writing all the data.

The parameters of `extractNumbers` specify the names of the input file and the output file.

Note: There is documentation on the `Scanner` and `PrintStream` classes at the end of the test paper.

```
public void extractNumbers(String inFileName, String outFileName){
    try{
        Scanner in = new Scanner(new File(inFileName));
        PrintStream out = new PrintStream(new File(outFileName));
        int counter=0;

        while ( in.hasNext() ){
            if ( in.hasNextDouble() ){
                double num = in.nextDouble();
                if ( counter >= 5 ){
                    counter = 0;
                    out.println ();
                }
                out.print (num + " ");
                counter++;
            }
            else
                in.next ();
        }
        in.close ();
        out.close ();
        System.out.println ("done");
    }
    catch(Exception e){System.out.println("Error while scanning"+e);}
}
```

Question 5. Objects and Fields

[15 marks]

This question concerns a program for simple screensaver that shows random cars appearing on the window, then driving off to the right. The program contains two classes: `CarScreenSaver` and `Car`.

The `Car` class on the facing page represents individual cars.

Part of the `CarScreenSaver` class is shown below. The `simulate` method contains a loop that repeatedly constructs a new `Car` object at a random position, and then drives it off the screen to the right using the `inWindow` and `drive` methods on the `Car` object.

```

public class CarScreenSaver{
    private DrawingCanvas canvas;
        :
    public CarScreenSaver(){
        // sets up the window and drawing canvas
        :
    }
    public void simulate(){
        while (true){
            // make a new Car at a random position
            double randomX = Math.random()*400;           // (a number between 0.0 and 400.0)
            double randomY = Math.random()*400;           // (a number between 0.0 and 400.0)
            Car car = new Car(canvas, randomX, randomY);
            // drive it across the window until it is off the edge
            while (car.inWindow()){
                canvas.clear();
                // move car 2 units to the right
                car.drive (2);
                // wait for 0.01 seconds
                try{Thread.sleep(100);}catch(Exception e){}
            }
        }
    }
}

```

On the facing page, complete the definition of the `Car` class:

- Declare data fields to store the state information, including the `Car`'s position.
- The constructor should initialise the data fields and draw the car at its initial position.
- The `draw` method should draw the car at its current position as a 30×15 rectangle.
- The `inWindow` method should return `true` if the car is visible in the window.
Assume that the window size is 400×400 .
- The `drive` method should move and redraw the car.

Hint: look carefully at how the `Car` object is used in the `CarScreenSaver` class.

(Question 5 continued on next page)

(Question 5 continued)

```
public class Car {
    /* fields */
    private DrawingCanvas canvas;
    private double x;
    private double y;

    /* Constructor: initialises fields and draws car at its initial position */
    public Car(DrawingCanvas c, double xx, double yy){
        this.canvas = c;
        this.x = xx;
        this.y = yy;
        this.draw();
    }

    /* draw: draws the car at its current position as a 30 x 15 rectangle */
    public void draw(){
        canvas.fillRect ((int)this.x, (int)this.y, 30, 15);
    }

    /* inWindow: returns true if and only if the car is still visible . */
    public boolean inWindow(){
        return (this.x <= 400);
    }

    /* drive: moves the car and redraws it */
    public void drive(int dist) {
        this.x = this.x + dist;
        this.draw();
    }
}
```

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:
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 // Prints the format string, inserting the remaining
// arguments at the %'s in the format string: // arguments at the %'s in the format string:
// %s for Strings. // %s for Strings.
// %d for ints, (%3d: use at least 3 characters), // %d for ints, (%3d: use at least 3 characters),
// %.2f for 2dp doubles, // %.2f for 2dp doubles,
// (%6.2f: at least 6 characters and 2dp), // (%6.2f: at least 6 characters and 2dp),
// Use \n for newline // Use \n for newline
```

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

```
Integer
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
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

```
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

```
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

```
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

```
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
```