

ID Number:

Signature:

COMP102: Test — Model Solutions

30 April, 2001

Instructions

- Time allowed: **2 hours**.
- Answer **all** the questions.
- There are 120 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.
- This test will contribute 25% of your final grade.
- Numeric keypad calculators and non-electronic dictionaries are permitted.

Questions

Marks

1. Understanding Java I	[21]	<input type="checkbox"/>
2. Understanding Java II	[24]	<input type="checkbox"/>
3. Understanding and writing Java I	[25]	<input type="checkbox"/>
4. Understanding and writing Java II	[30]	<input type="checkbox"/>
5. Files	[20]	<input type="checkbox"/>
	TOTAL:	<input type="checkbox"/>

Question 1. Understanding Java I

[21 marks]

Consider the following Java program:

```
1 // Budget calculator.
2
3 import mcs.compl00.*;
4
5 public class BudgetSheet {
6     public static void main(String[] args) {
7         UI w = new UI("Budget Sheet");
8         Budget bp = new Budget(w);
9     }
10 }
11
12 class Budget implements ButtonListener {
13
14     private UI win;
15     int rent, elec, food;
16
17     public Budget(UI window) {
18         win = window;
19         rent = 0;
20         elec = 0;
21         food = 0;
22         win.addButton("Set Rent", this);
23         win.addButton("Set Electricity", this);
24         win.addButton("Set Food", this);
25     }
26
27     public void buttonPerformed(String button) {
28         if (button.equals("Set Rent"))
29             rent = getVal("Rent");
30         else if (button.equals("Set Electricity"))
31             elec = getVal("Electricity");
32         else if (button.equals("Set Food"))
33             food = getVal("Food");
34         win.println("Rent : " + rent);
35         win.println("Elec : " + elec);
36         win.println("Food : " + food);
37         int total = rent + elec + food;
38         win.println("Total Budget: " + total);
39     }
40
41     private int getVal(String name) {
42         int amount = win.getInt("Budget for " + name + ":");
43         return amount;
44     }
45
46 }
```

(a) [4 marks] Name the methods (including constructors) declared in the program.

main (line 6)
Budget (constructor, line 17)
buttonPerformed (line 27)
getVal (line 41)
(You were not required to give the line numbers.)

(b) [4 marks] Name the UI methods (including constructors) called in the program.

UI (constructor, line 7)
addButton (lines 22–24)
println (lines 34–36 and 38)
getInt (line 42)
(You were not required to give the line numbers.)

(c) [4 marks] Name the fields declared in the program.

win (line 14)
rent (line 15)
elec (line 15)
food (line 15)
(You were not required to give the line numbers.)

(d) [4 marks] Name the local variables declared in the program.

w (line 7)
bp (line 8)
total (line 37)
amount (line 42)
(You were not required to give the line numbers.)

(e) [5 marks] Write the line numbers of the statements that will be executed, in the order they are executed, when the button labelled “Set Electricity” is pressed.

28, 30, 31, 42, 43, 34, 35, 36, 37, 38

Question 2. Understanding Java II

[24 marks]

For each of the following programs, show the output produced when the program is run.

(a) [4 marks]

```
import mcs.compl00.*;
class Pop1 {
    public static void main(String[] args) {
        UI win = new UI("Pop1");
        int a, b, c;
        a = 3;
        b = 4;
        c = 5;
        c = a + b + c;
        win.println(c);
    }
}
```

12

(b) [4 marks]

```
import mcs.compl00.*;
class Pop2 {
    public static void main(String[] args) {
        UI win = new UI("Pop2");
        int a, b;
        a = 7;
        b = 13;
        if (b < 2*a)
            win.print("red ");
        else
            win.print("blue ");
        win.println("green");
    }
}
```

red green

(c) [8 marks]

```
import mcs.comp100.*;
class Pop3 {
    public static void main(String[] args) {
        UI win = new UI("Pop3");
        int a, b;
        a = 10;
        b = 30;
        while (a < b) {
            win.println(a + " " + b);
            a = a+3;
            b = b-2;
        }
    }
}
```

```
10 30
13 28
16 26
19 24
```

(d) [8 marks]

```
import mcs.comp100.*;
class Pop4 {
    public static void main(String[] args) {
        UI win = new UI("Pop3");
        int max = 8;
        for (int a = 1; 2*a <= max; a = a+1) {
            for (int b = a; b <= max; b = b+a) {
                win.print(b + " ");
            }
            win.println();
        }
    }
}
```

```
1 2 3 4 5 6 7 8
2 4 6 8
3 6
4 8
```

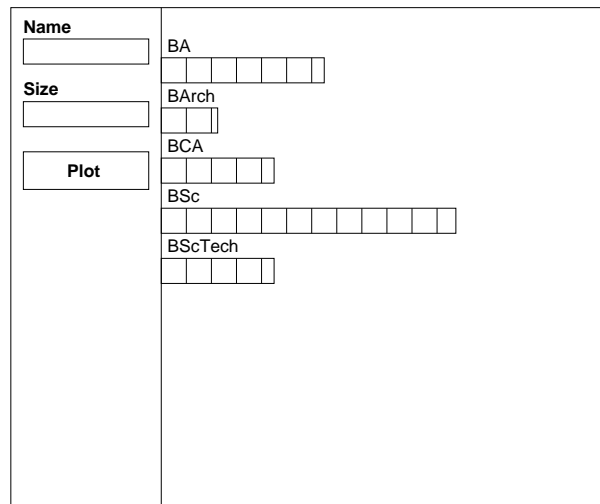
Question 3. Understanding and writing Java I

[25 marks]

This question involves a program that draws a bar-chart using values entered by the user. The user enters a label for each bar of the bar-chart (using a text field) and the size of the bar (using a number field). The program provides a button to add a bar to the bar-chart using the current contents of the text and number fields.

The bar-chart is drawn with each bar extending horizontally from the left of the graphics pane, with the label drawn just above it. The bar is graduated with lines showing multiples of 20.

For example, suppose the user entered the following sequence of text and number pairs: BA 128, BArch 46, BCA 94, BSc 235, BScTech 87. The resulting screen would look something like this:



The following is an outline of the BarChart program. You should complete the program outline, declaring any fields required and providing bodies for the three methods.

Write your answers in the boxes provided. Do not change the given code.

```
// Bar Chart Plotter:

import mcs.compl00.*;

public class BarChart {

    public static void main(String[] args) {
        UI win = new UI("Bar Chart Plotter");
        Plotter pp = new Plotter(win);
    }
}
```

```
class Plotter
    implements NumberFieldListener, TextFieldListener, ButtonListener {
```

```
// (a) Declare fields (6 marks)
private UI w;

    String name;
    double size;
    int nextpos = 50;
```

```
public Plotter(UI window) {
    w = window;
    w.addTextField("Name", this);
    w.addNumberField("Size", this);
    w.addButton("Plot", this);
}
```

```
// (b) Handle text field events (4 marks)
public void textFieldPerformed(String label, String text) {
    if (label.equals("Name")) {

        name = text;

    }
}
```

```
// (c) Handle number field events (4 marks)
public void numberFieldPerformed(String label, double val) {
    if (label.equals("Size")) {

        size = val;

    }
}
```

```
// (d) Handle button events (11 marks)
public void buttonPerformed(String label) {
    if (label.equals("Plot")) {

        w.drawString(name, 0, nextpos-2);
        w.drawRect(0, nextpos, size, 25);
        for (int k = 20; k < size; k = k+20)
            w.drawLine(k, nextpos, k, nextpos+25);
        nextpos = nextpos + 50;

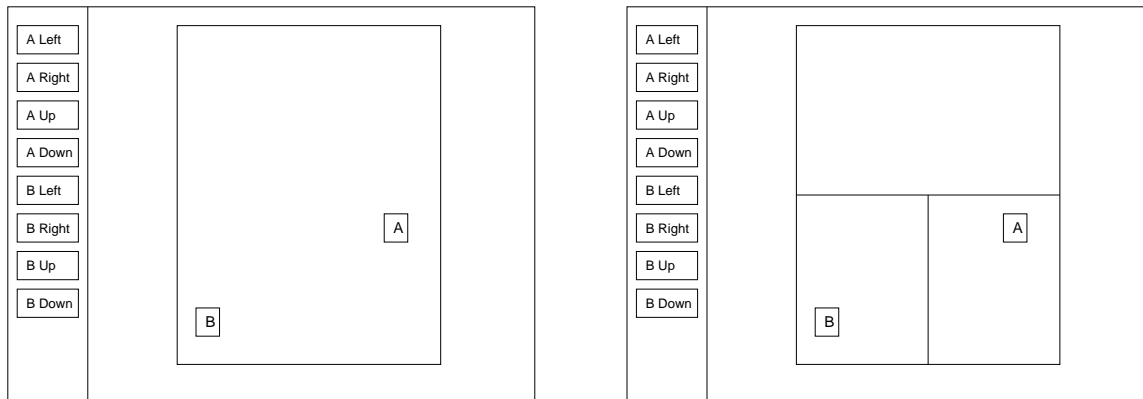
    }
}
```

```
}
```

Question 4. Understanding and writing Java II

[30 marks]

The Java program on the following page is intended to assist a squash coach in demonstrating where players should position themselves on the squash court. The program draws an outline of the court and labelled boxes representing two squash players, and provides buttons allowing each of the players to be moved in each of four directions. The initial screen layout looks something like the diagram on the left:



The program creates three objects: one for the court and one for each of the players. The `Court` object draws a rectangle representing the walls of the court. The two `Player` objects each add the four buttons for that player, draw the player in its initial position, and handle button events to move the player.

You are asked to make three changes to the program. Note that you do not need to know anything about the rules of squash, except that both players can move anywhere on the squash court.

Write your answers in the boxes provided. Do not change the given code.

(a) [10 marks] The given program allows players to move through the walls of the court! Modify the program so that trying to move a player through a wall leaves the player's position unchanged.

(b) [10 marks] The given program does not draw the lines marked on the floor of the squash court. You are to add the two extra lines shown in the diagram above on the right.

The horizontal line is half way up the court, and the vertical line is half way across the court.

(c) [10 marks] Modify the program so that it displays a message below the court area indicating the relative positions of the two players. The message should indicate whether player A is behind, ahead of or beside player B, and whether player A is to left of, to right of or in line with player B. For example, in their initial positions, as shown in the diagram above, the message should say "Player A is ahead of and to right of player B". A special message should also be displayed if the players collide.

```

import mcs.compl00.*;
public class Coach {
    public static void main(String args[]) {
        UI w = new UI("Squash Coach");
        Court c = new Court(w);
    }
}
class Court {
    private UI win;                // Window to draw on
    private int top = 50;          // Top of court
    private int left = 100;       // Left of court
    private int width = 300;      // Width of court
    private int length = 450;     // Length of court
    Player a, b;                  // The two players

    public Court(UI w) {
        win = w;
        a = new Player(w, "A", this, 340, 275);
        b = new Player(w, "B", this, 130, 455);
        drawCourt();
    }

    public boolean inCourt(int x, int y, int w, int h) {
        return (left < x && x+w < left+width && top < y && y+h < top+length);
    }

    public void drawCourt() {
        win.drawRect(left, top, width, length);

```

```
// (b) (10 marks)
```

```

win.drawLine(left, top+length/2, left+width, top+length/2);
win.drawLine(left+width/2, top+length/2, left+width/2, top+length);

```

```
// (c) (10 marks)
```

```

String pos1, pos2;
if (a.gety() > b.gety()) pos1 = "behind";
else if (a.gety() < b.gety()) pos1 = "ahead of";
else pos1 = "beside";
if (a.getx() < b.getx()) pos2 = "to left of";
else if (a.getx() > b.getx()) pos2 = "to right of";
else pos2 = "in line with";
win.eraseRect(left, top+length+2, 300, 40);
win.drawString("Player A is " + pos1 + " and " + pos2 + " player B", left+40, top+length+20);
if (Math.abs(a.getx()-b.getx()) <= a.pw && Math.abs(a.gety()-b.gety()) <= a.ph )
    win.drawString("The players have collided!", left+40, top+length+40);

```

```

}
}

```

```

class Player implements ButtonListener {
    private UI win;                // Window to draw on
    private Court crt;            // The court being played on
    private String name;         // Player's name
    private int posx, posy;      // Player's current position
    public int pw = 25, ph = 30; // Width and height of player
    public Player(UI w, String nam, Court c, int x, int y) {
        win = w; crt = c; name = nam;
        posx = x; posy = y;
        drawPlayer();
        win.addButton(name + " Left", this);
        win.addButton(name + " Right", this);
        win.addButton(name + " Up", this);
        win.addButton(name + " Down", this);
    }
    public void buttonPerformed(String button) {
        erasePlayer();
        if (button.equals(name + " Left"))
            moveHoriz(-10);
        else if (button.equals(name + " Right"))
            moveHoriz(10);
        else if (button.equals(name + " Up"))
            moveVert(-10);
        else if (button.equals(name + " Down"))
            moveVert(10);
        drawPlayer();
        crt.drawCourt();
    }
    public int getx() { return posx; }
    public int gety() { return posy; }
    private void erasePlayer() {
        win.eraseRect(posx, posy, pw, ph);
    }
    private void drawPlayer() {
        win.drawRect(posx, posy, pw, ph);
        win.drawString(name, posx+pw*.4, posy+ph*.6);
    }
    // (a) (10 marks)
    private void moveHoriz(int step) {
        if ( crt.inCourt(posx+step, posy, pw, ph) )
            posx = posx + step;
    }
    private void moveVert(int step) {
        if ( crt.inCourt(posx, posy+step, pw, ph) )
            posy = posy + step;
    }
}

```

Question 5. Files

[20 marks]

(a) Reading and writing [6]

Suppose we have a file of numbers, one number on each line. Complete the following method so that it will read all the numbers, and for each number, display the number on the screen and write the square of the number to a new file (one number on each line).

Note: We assume `win` is a UI object.

```
public void readAndWrite() {
    // open files
    DataFileReader inFile = new DataFileReader();
    DataFileWriter outFile = new DataFileWriter();

    // declare a variable called num
    double num;

    // loop to read a number, display it, write the square of it to the output file
    while(!inFile.endOfFile()){

        // read a number to the variable called num

        num = inFile.readNumber();
        .....

        // display the number on screen
        win.println(num);

        // calculate the square of the number
        double numSquare = num * num;

        // write the square of the number to the output file

        outFile.println(numSquare);
        .....
    }

    // close the files

    inFile.close();
    .....

    outFile.close();
    .....
}
```

(b) Drawing shapes from a file [14]

This question asks you to write a program that will read data from a file and use the data to draw pictures (shapes). The file contains descriptions of two types of shapes: rectangles and circles. Each shape is specified in one line of the file, organised as follows:

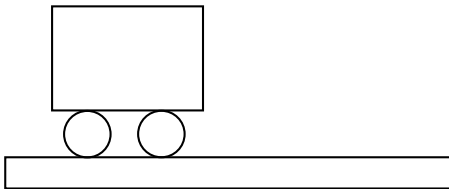
- The shape: either `rectangle` or `circle`.
- The arguments:
 - 4 numbers for a rectangle: `x`, `y`, `width`, `height`
 - 3 numbers for a circle: `x`, `y`, `diameter`

Note: We assume that all data will be in the correct format.

Here is an example of a file:

```
rectangle 50 50 90 60
circle 60 110 30
circle 100 110 30
rectangle 20 140 260 20
```

Given the file above, the program should draw a figure like this:



Your task is to complete the `readAndDraw` method on the following page so that it asks the user to select a file and draws all the shapes in the file.

```
import mcs.comp100.*;

public class DrawFromFileProgram {
    public static void main (String[] args) {
        UI window = new UI("Read a file and draw shapes");
        DrawFromFile f = new DrawFromFile(window);
    }
}
```

```

class DrawFromFile implements ButtonListener {
    private UI win;
    public DrawFromFile(UI w){
        win = w;
        win.addButton("Read and Draw", this);
        win.addButton("Clear", this);
    }
    public void buttonPerformed(String but){
        if (but.equals("Read and Draw"))
            readAndDraw();
        else if (but.equals("Clear"))
            win.clear();
    }
    public void readAndDraw() {
        // Open a file
        DataFileReader file = new DataFileReader();
        // Read data and draw shapes
        while (!file EOF()) {
            String shape = file.readToken();
            if (shape.equals("rectangle") ) {
                double x = file.readNumber();
                double y = file.readNumber();
                double w = file.readNumber();
                double h = file.readNumber();
                win.drawRect(x, y, w, h);
            }
            else if ( shape.equals("circle") ) {
                double xx = file.readNumber();
                double yy = file.readNumber();
                double d = file.readNumber();
                win.drawCircle(xx, yy, d);
            }
        }
        // Close the file
        file.close();
    }
}

```

Another version is given on the following page

```
public void readAndDraw() {  
    // Open a file  
    DataFileReader file = new DataFileReader();  
    // Read data and draw shapes  
    while (!file.endOfFile()){  
        String shape = file.readToken();  
        if (shape.equals("rectangle") ){  
            win.drawRect(file.readNumber(), file.readNumber(), file.readNumber(), file.readNumber());  
        }  
        else if ( shape.equals("circle") ){  
            win.drawCircle(file.readNumber(), file.readNumber(), file.readNumber());  
        }  
    }  
    // Close the file  
    file.close();  
}
```
