

Family Name:

Other Names:

ID Number:

COMP 102: Test 2

Model Solutions

6 May, 2010

Instructions

- Time allowed: **45 minutes**
- There are 45 marks in total.
- Answer **all** the questions.
- Write your answers in the boxes in this test paper and hand in all sheets. You may ask for additional paper if you need it.
- If you think some question is unclear, ask for clarification.
- Brief Java documentation will be supplied with the test.
- This test will contribute 15% 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

[15]

2. Event Driven Input

[10]

3. Two Class programs

[10]

4. Debugging

[10]

TOTAL:

Question 1. Basic Java

[15 marks]

(a) [5 marks] Consider the following printValue method

```
public void printValue(int x, int y){
    if (x < y) {
        Ul.println(x + " before " + y);
        x = x + 50;
    }
    else if (x > y) {
        Ul.println(y + " after " + x);
        y = y + 20;
    }

    Ul.println("Now have " + x + " and " + y);

    if (x > 25){
        Ul.println("Large");
    }
    else if (y > 25) {
        Ul.println("Small");
    }
    else {
        Ul.println("Neither");
    }
}
```

(i) [1 mark] What will be printed if printValue(10, 10) is called?

```
Now have 10 and 10
Neither
```

(ii) [2 marks] What will be printed if printValue(3, 17) is called?

```
3 before 17
Now have 53 and 17
Large
```

(iii) [2 marks] What will be printed if printValue(14, 10) is called?

```
10 after 14
Now have 14 and 30
Small
```

(Question 1 continued)

(b) [2 marks] What will the following fragment of Java print?

```
int i = 0;
UI.print("answer: ");
while ( i <= 8) {
    i = i + 2;
    UI.print(i + " ");
}
```

```
answer:      2 4 6 8 10
```

(c) [4 marks] Write a fragment of Java that will print out every third number from 333 down to 3, as in:

```
333
330
:
6
3
```

```
int k = 333;
while ( k>=3 ){
    UI.println (k);
    k = k-3;
}
```

(Question 1 continued on next page)

(Question 1 continued)

(d) [4 marks] The following `mixup` method calls the `shuffle` method several times. What would be printed out if `mixup()` were called?

```
public void mixup(){
    int diff = 3;
    int sum = 5;
    Ul.println ("diff = " + diff + " sum = " + sum);

    Ul.print ("A: ");
    this.shuffle( diff , sum);

    Ul.print ("B: ");
    this.shuffle(sum, diff );
}

public void shuffle(int diff , int sum){
    Ul.print ( sum + ".." + diff + ".." + sum);
    diff = sum - diff;
    Ul.println (" diff = " + diff );
}
```

```
diff = 3 sum = 5
A: 5..3..5 diff = 2
B: 3..5..3 diff = -2
```

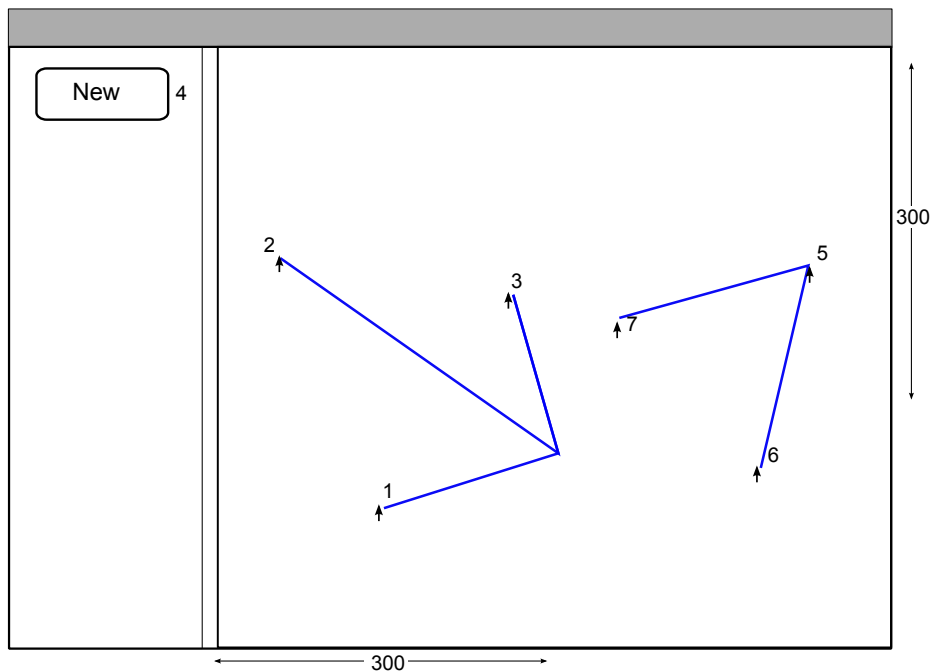
Question 2. Event-Driven Input

[10 marks]

(a) [6 marks] Consider the `PatternWriter` class on the facing page, which has one button and the graphics pane responds to the mouse. It has two fields that store a position (`xPos` and `yPos`).

Sketch below what the program would draw on the graphics pane if the user took the following actions in sequence:

1. press and release mouse at point 1
2. press and release mouse at point 2
3. press and release mouse at point 3
4. press the "New" button
5. press and release mouse at point 5
6. press and release mouse at point 6
7. press and release mouse at point 7



(b) [4 marks] Modify the `PatternWriter` class so that it has an additional button called "Clear", that will clear the canvas and set both `xPos` and `yPos` to 300.

(Answer by modifying the code on the facing page)

(Question 2 continued on next page)

(Question 2 continued)

```
public class PatternWriter implements UIButtonListener, UIMouseListener{
    private int xPos = 300;
    private int yPos = 300;

    public PatternWriter(){ //Set up window with one button
        UI.setMouseListener(this);
        UI.addButton("New", this);
        -----> UI.addButton("Clear", this);
    }

    public void buttonPerformed(String cmd){
        if (cmd.equals("New") ){
            xPos = -1;
        }
        -----> else if (cmd.equals("Clear")){
            UI.clearGraphics();
            this.xPos = 300;
            this.yPos = 300;
        }
    }

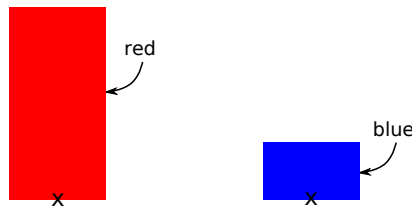
    public void mousePerformed(String action, double x, double y) {
        if (action.equals("released")){
            if (this.xPos < 0){
                this.xPos = x;
                this.yPos = y;
            }
            else {
                UI.drawLine(this.xPos, this.yPos, x, y);
            }
        }
    }

    public static void main(String[] arguments){
        new PatternWriter();
    }
}
```

Question 3. Two Class programs

[10 marks]

For this question, you are to complete a small program with two classes. The Platform class represents a platform (in a video game) which can change its height between one of two states - high or low. It is displayed as a solid rectangle on a DrawingCanvas. When a platform is high, its height is 100 units and it is displayed in red; when it is low, its height is 30 units and it is displayed in blue. The figures shows two platforms; the left one is in its high state, and the right one is in its low state. The "x" shows the position of the platform.



The flipPlatforms method in the TestPlatforms class below creates two Platform objects, one at position (100, 200) (the center of the bottom of the platform) and the other at position (300, 250). It then makes the platforms go up and down 5 times.

The Platform class contains fields, a constructor, and four methods: high(), low(), draw(), and erase(). The fields and parts of the constructor and methods are written for you; you are to complete the constructor and the methods.

```
public class TestPlatform{

    public void flipPlatforms (){
        Platform p1 = new Platform(100, 200);
        p1.high();
        Platform p2 = new Platform(300, 250);
        int count = 0;
        while (count < 5){
            Ul.sleep(500);
            p1.low();
            p2.high();
            Ul.sleep(500);
            p1.high();
            p2.low();
            count++;
        }
    }
}
```

(Question 3 continued on next page)

(Question 3 continued)

```
public class Platform{
    private final int width = 50;           // constant: the width of the platform
    private final int upHeight = 100;      // constant: the height when it is high
    private final int downHeight = 30;     // constant: the height when it is low
    private int xPos;                       // middle of the the Platform
    private int yPos;                       // bottom of the Platform
    private int height = this.downHeight; // current height of the platform.
    /** Construct a new Platform object and draw it on the canvas.*/
    public Platform(                          ){
        int x, int y
        this.xPos = x;
        this.yPos = y;
        this.draw()
    }
    public void high(){    /** Makes the platform be high */

        this.erase();
        this.height = this.upHeight;
        this.draw();
    }
    public void low(){    /** Makes the platform be low */

        this.erase();
        this.height = this.downHeight;
        this.draw();
    }
    public void draw(){    /** Draws platform on canvas: red if high, blue if low */

        if (this.height == this.upHeight){
            UI.setColor(Color.red);
        }
        else{
            UI.setColor(Color.blue);
        }
        int left = this.xPos - this.width/2;
        int top = this.yPos - this.height;
    } UI.fillRect ( left , top, this.width, this.height);
    public void erase(){    /** Erases the platform */
        int left = this.xPos - this.width/2;
        int top = this.yPos - this.height;
        UI.clearRect(left , top, this.width, this.height);
    }
}
```

Question 4. Debugging

[10 marks]

The `doTotal` method is intended to read a file of numbers containing four numbers per line, and print out how many numbers are in the file and the sum of the last number on each line of the file. To assist in debuggin, at the end of each line, it prints out the number of numbers read so far, and the sum of the last numbers of the lines so far.

For example, given a file containing the numbers shown on the left, it should print the data on shown on the right.

4 16 83 10	count: 4, total: 10
72 13 26 20	count: 8, total: 30
12 81 14 2	count: 12, total: 32
7 23 9 8	count: 16, total: 40
76 10 43 15	count: 20, total: 55
61 9 33 10	count: 24, total: 65
	Final count: 24, grand total: 65

The following version of `doTotal` has errors:

```
public void doTotal(String fname){
    try{Scanner sc = new Scanner(new File(fname));
        int count = 0;
        int total = 0;
        while (sc.hasNextInt()){
            int j = 0;
            while (j <= 4){
                int num = sc.nextInt();
                count = count + j;
                j++;
            }
            total = total + sc.nextInt ();
            UI.printf ("count: %2d, total: %2d\n", count, total);
        }
        UI.printf ("Final count: %2d, grand total: %2d\n", count, total);
    }catch(IOException e){UI.println("File error: "+e);}
}
```

(a) [5 marks] What would `doTotal` print out if it were called on a file containing the numbers shown above on the left?

```
count: 10, total: 13
count: 20, total: 15
count: 30, total: 25
count: 40, total: 35
Final count: 40, grand total: 35
```

(Question 4 continued)

(b) [5 marks] Write a correct version of the doTotal method.

```
public void doTotal(String fname){
    try{
        Scanner sc = new Scanner(new File(fname));
        int count = 0;
        int total = 0;

        int num = 0;
        while (sc.hasNext()){
            int j = 0;
            while (j < 4){
                num = sc.nextInt();
                count = count + 1;
                j++;
            }
            total = total + num;
            UI.printf ("count: %2d, total: %2d\n", count, total);
        }
    }
```

OR

```
-----
while (sc.hasNext()){
    int j = 0;
    while (j < 3){
        int num = sc.nextInt();
        j++;
    }
    count = count + 4;
    total = total + sc.nextInt ();
    UI.printf ("count: %2d, total: %2d\n", count, total);
}
```

```
UI.printf ("Final count: %2d, grand total: %2d\n", count, total);
} catch(IOException e){UI.println("File error: "+e);}
}
```
