Family Name: ..... **Other Names:** .....

ID Number: ..... **Signature** .....

COMP 102: Test 2

12 May, 2014

Instructions

- Time allowed: **50 minutes**
- Answer **all** the questions. There are 50 marks in total.
- 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,
  (But your mark will be boosted up to your exam mark if that is higher.)
- You may use calculators and paper translation dictionaries.
- You may write notes and working on this paper, but make sure your answers are clear.

Questions	**Marks**	
1. Understanding Loops	[5]	
2. Writing Loops	[7]	
3. Files	[6]	
4. Using Objects	[7]	
5. Event Driven Input	[7]	
6. Defining Classes	[9]	
7. More Loops	[9]	
	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.

Student ID: .....

Question 1. Understanding Loops [5 marks]

Consider the following printOut method.

```
public void printOut( int max){
    int i = 0;
    int j = max;
    while (i <= j){
        UI.println ( "now: " + i + " and " + j);
        i = i + 2;
        j = j / 2;
    }
    UI.println ("then: " + i + " and " + j);
}
```

max: ☐

i: ☐

j: ☐

What will be printed if printOut(48) is called?

Question 2. Writing loops [7 marks]

Complete the following sumSeries method so that it uses a loop to sum all the integers from 1 to its second argument, and prints out the result.

For example sumSeries(6) should add 1 + 2 + 3 + 4 + 5 + 6 and print out

```
Sum of integers from 1 to 6 = 21
```

Hints:

- Use two variables: one to count up to max, and one to keep the running total.
- The last line of the method is written for you.

```
public void sumSeries(int max){




    UI.println("Sum of integers from 1 to " + max + " = " + total);
}
```

Question 3. Files [6 marks]

An auction room keeps lists of sheep for sale in a text file. The file contains one line for each different breed. Each line has the number of sheep and the breed of the sheep, for example:

```
510 Romney
380 Merino
150 Corriedale
18 Leicester
50 Coopworth
85 Friesian
```

Complete the following totalSheep method that is passed the name of the file, will open a Scanner to read from the file, read all the data in the file, and then will print out the total number of sheep for sale in the file. For example, given the file above, it would print

```
Total sheep: 1193
```

```
public void totalSheep(String fileName){
    try{

        Scanner scan =









        UI. println ("Total sheep: " + total);
    } catch(IOException e){UI.println("Fail: " + e);}
}
```

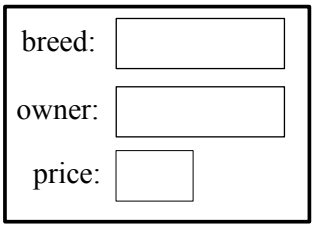
Question 4. Using Objects [7 marks]

Consider the SheepSale class on the facing page

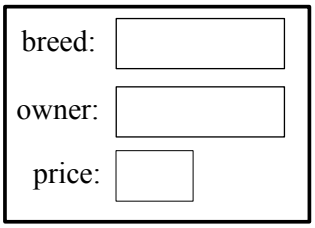
What will the following fragment of code print out? (Note the variables carefully, and keep track of the fields in the objects.)

```
SheepSale ss1 = new SheepSale("Romney", "Smith");

ss1.report ();
ss1.bid(120);
ss1.sold("Murphy");

SheepSale ss2 = new SheepSale("Merino", "Jones");

ss2.bid(250);
ss2.bid(90);
ss1.bid(100);
ss2.sold("Brown");
ss1.report ();
ss2.report ();
```

ss1:



ss2:




```
Report:

Bid:

Sold:

Bid:

Bid:

Bid:

Sold:

Report:

Report:
```

(Question 4 continued on next page)

(Question 4 continued)

The SheepSale class:

```
public class SheepSale{

    private String breed;
    private String owner;
    private double price = 0.0;

    public SheepSale(String b, String o){
        this.breed = b;
        this.owner = o;
    }

    public void report(){
        UI.println("Report: " + this.owner +" : "+ this.breed +" @ $"+ this.price);
    }

    public void bid(double offer){
        this.price = offer;
        UI.println("Bid: " + this.breed +" @ $"+ this.price);
    }

    public void sold(String buyer){
        String oldOwner = this.owner;
        this.owner = buyer;
        UI.println("Sold: " + oldOwner +" --> "+ this.owner +" @ $"+ this.price);
    }

}
```

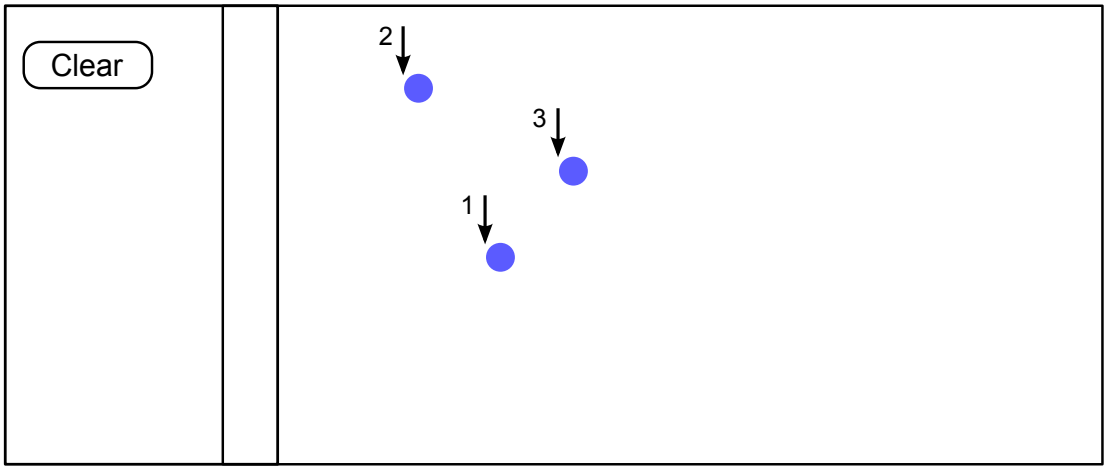
Question 5. Event-Driven Input [7 marks]

The DotMaker program on the facing page should allow the user to draw a picture made of blue dots on the graphics pane. Every time the user releases the mouse at a point on the graphics pane, the program should draw a filled blue circle of diameter 10, whose top-left corner is at the point.

The program should have one button called "Clear"; when the user clicks the button, the program should clear the graphics pane.

Complete the DotMaker program.

The diagram shows the result of the user clicking at positions (1), then (2), and then (3).



(Question 5 continued on next page)

(Question 5 continued)

```
public class DotMaker implements                                    {

    private static final DIAM = 10;      // size of the circles


    public DotMaker(){









    }
    public void buttonPerformed(String button){









    }
    public void mousePerformed(String action, double x, double y){









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

Question 6. Defining Classes [9 marks]

For this question, you are to complete part of a program for a computer game involving a flock of sheep:

- The sheep move across the screen from left to right, eating grass as they go.
- As they get fatter, they move more slowly. Initially, a sheep moves 10.0 units in each move. Every bite of grass slows it down by 0.1 unit, until it is moving just 2.0 units.
- If the dog barks at a sheep, the sheep is frightened, and for the next two moves, instead of moving across the screen, it will move up the screen.

Complete the Sheep class on the facing page, which defines Sheep objects.

You need to define:

- Fields: to store the state information about the sheep.
- The constructor: initialises the Sheep. The parameter is the y position of the Sheep. The x position is initially 0.
- The eat method: Parameter is the number of bites of grass. Each bite slows the sheep down, to a limit of 2.0.
- The barkAt method: No parameters. Makes the sheep move upwards instead of to the right for the next two moves.
- The move method: No parameters. Moves the position of the sheep according to its current speed and whether it is frightened.
- The isFrightened method: No parameters. Returns a boolean value that is true if the sheep is currently frightened and false otherwise.

Note, none of the methods need to call the draw method.

(Question 6 continued)

```
public class Sheep{




    public Sheep(double y){



    }
    public void eat( int  bites){



    }
    public void barkAt(){

    }
    public void move(){









    }
    public boolean isFrightened(){



    }
    public void draw(){
        if  (this.isFrightened()){  UI.drawImage("frightened-sheep.png", x, y); }
        else { UI.drawImage("plain-sheep.png", x, y); }
    }
}
```

Question 7. More Loops [9 marks]

The following drawStuff method draws lines and circles on the graphics pane. What will it draw if it is called with an argument of 250?

```
this.drawStuff(250);
```

Sketch the answer on the grid on the facing page.

```
public void drawStuff(int size){
    int diam = 50;
    boolean present = true;

    int left = 0;
    while ( left < size) {
        int top = 0;
        while (top < size){
            if ( left < top ){
                UI.drawLine(left, top, left +diam, top+diam);
            }
            else if (present){
                UI.drawOval(left, top, diam, diam);
                present = false;
            }
            else {
                present = true;
            }
            top = top + diam;
        }

        left = left + diam;
    }
}
```

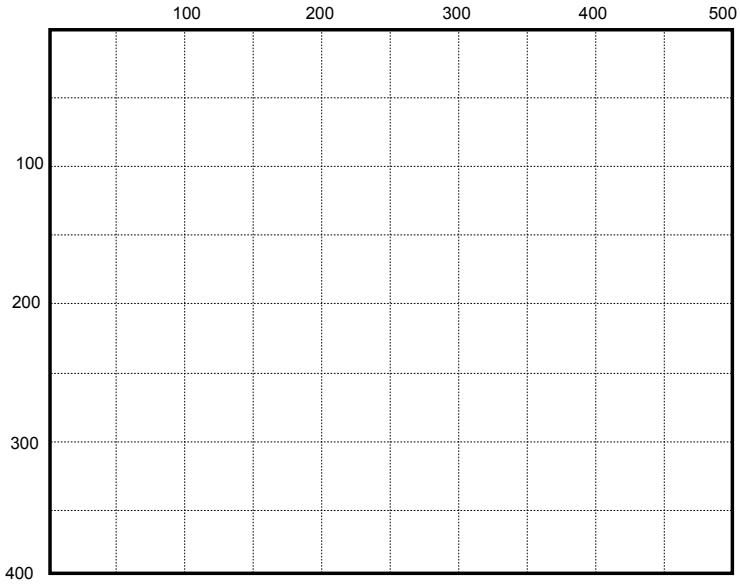
left: ☐

top: ☐

present: ☐

(Question 7 continued on next page)

(Question 7 continued)



* * * * * * * * * * * * * * *