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

Student ID: ..... **Signature** .....

COMP 102/112: Test

2021, May 6 ** WITH SOLUTIONS **

Instructions

- Time allowed: **50 minutes**
- Attempt **all** the questions. There are 50 marks in total.
- Write your answers in this test paper and hand in all sheets. (We have different instructions for distance students)
- If you think a question is unclear, ask for clarification.
- Brief Java documentation is provided with the test.
- This test contributes 15% of your final grade.
  (But your mark will be increased to your Test3 mark if that is higher.)
- You may use dictionaries and calculators.
- You may write notes and working on this paper, but make sure your answers are clear.
- You may assume all the programs import the ecs100 library and other standard libraries.

Questions	Marks	
1. Nested Loop	[6]	
2. Defining Classes	[16]	
3. Event-Driven Input	[14]	
4. Files	[14]	
	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. Nested Loop **[6 marks]**

Use a nested loop to complete the printPattern() method. printPattern() asks the user for a string and a count, then prints three lines of text, where each line contains the string repeated count times.

For example, if the user enters `"hello"` and `4`, the output should be:

```
hello hello hello hello
hello hello hello hello
hello hello hello hello
```

Note: UI.print(...) prints a value without a new line.

```
public void printPattern (){
    String s = UI.askString("Enter a string: ");
    int count = UI.askInt("Enter a count: ");
    for (int i = 0; i < 3; i++) {
        for (int j=0; j < count; j++) {
            UI. print (s + " ");
        }
        UI. println ();
    }









}
```

Question 2. Defining Classes **[16 marks]**

This question has two classes: the Box class representing Box objects and the TestBox class simulating a conveyor belt to move boxes.

(a) **[7 marks]** Design a Box class to represent Box objects.

A Box is represented on the screen by a blue square with a green logo on it. The draw() method is done for you.

Complete the Box class by adding the following:

- Fields to store the coordinates of the **bottom** left corner of the box.
  Hint: look at the draw() method.
- A constructor that initializes the fields.
- The erase() method that erases the box from its current position.
- The moveRight(double n) method that moves the box to the right by n units.
- The getLeft() method that returns the coordinate of the left side of the box.

(Question 2 continued on next page)

(Question 2 continued)

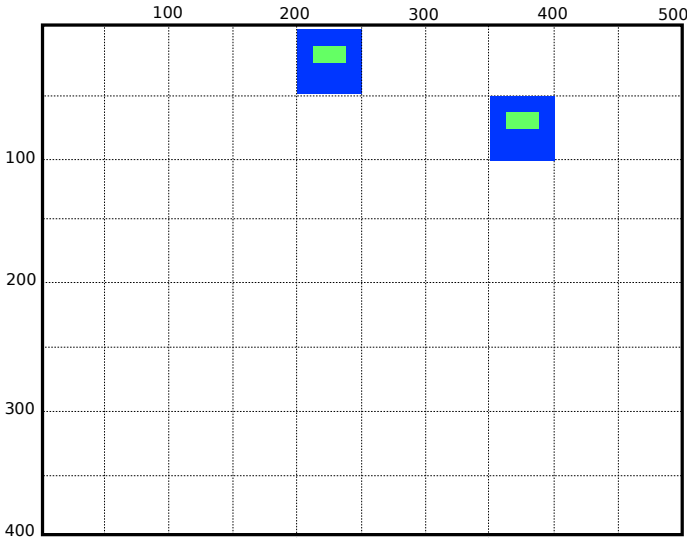
```
public class Box{
   public static final double SIZE = 50;

    private double left ;
    private double bottom;


    public Box(double left , double bottom){

        this. left  = left ;
        this.bottom = bottom;



    }
    public void draw(){                    //draws the box like this :
        UI.setColor(Color.blue);
        UI. fillRect (this. left , this.bottom-SIZE, SIZE, SIZE);
        UI.setColor(Color.green);
        UI. fillRect (this. left +SIZE/4, this.bottom-SIZE*3/4, SIZE/2, SIZE/4);
    }

    public void erase(){
        UI.eraseRect( left , bottom-SIZE, SIZE, SIZE);



    }
    public void moveRight(double n){
        this. left  = this. left  + n;



    }
    public double getLeft(){
        return this. left ;



    }
}
```

(Question 2 continued on next page)

(Question 2 continued)

The TestBox class has two methods for testing the Box class.

(b) **[2 marks]** Complete the twoBoxes() method to create and draw two Box objects with bottom left corner at position (200, 50) and (350, 100).



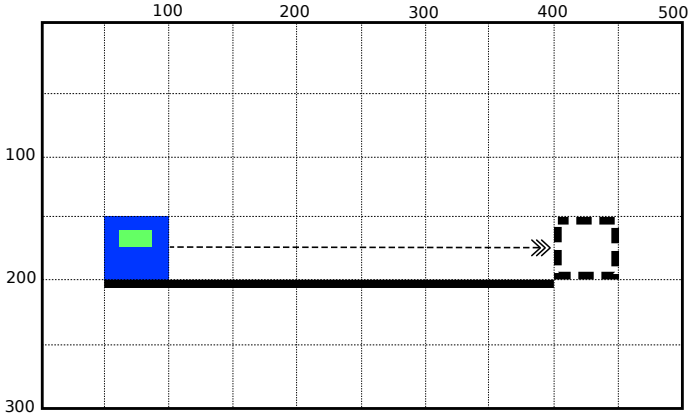

```
public void twoBoxes(){
    UI.clearGraphics();
    Box b1 = new Box(200, 50);
    Box b2 = new Box(350, 100);
    b1.draw();
    b2.draw();




}
```

(Question 2 continued on next page)

(Question 2 continued)

(c) **[7 marks]** Complete the conveyerBelt() method that will:

- draw a conveyor belt represented as a black rectangle with a very small height of 5 units. This is done for you.
- create a Box object that sits on the left end of the conveyor belt
- draw the box.
- make the box move step by step to just past the right end of the belt. Each step should move by 50 units, pausing for 100 milliseconds between steps.




```
public void conveyerBelt(){
    UI.clearGraphics();
    double leftEnd = 50;
    double rightEnd = 400;
    double belt = 200;
    UI.fillRect(leftEnd, belt, (rightEnd-leftEnd), 5);        //draw the conveyor belt
    Box b = new Box(leftEnd, belt);
    b.draw();

    double boxX = b.getLeft();
    while (boxX<rightEnd) {
        UI.sleep(100);
        b.erase();
        b.moveRight(50);
        b.draw();
        boxX=b.getLeft();
    }









}
```

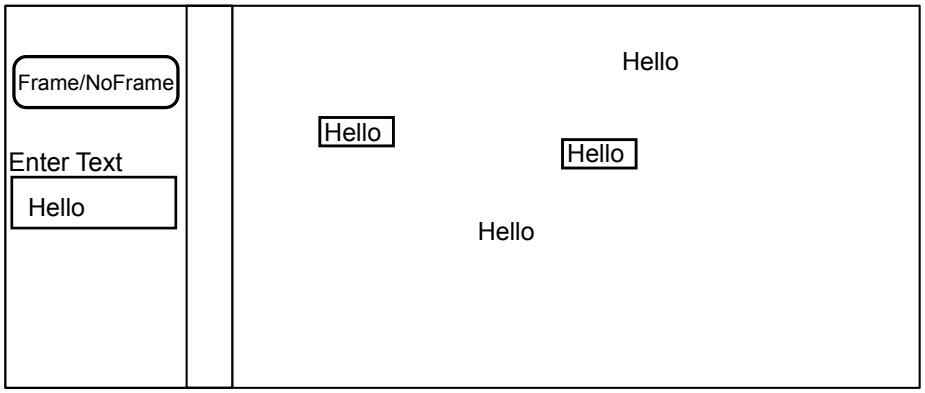
Question 3. Event-Driven Input **[14 marks]**

Complete the Program3 class so that it allows the user to draw framed and unframed text boxes on the graphics pane.

The program should have one button which allows the user to swap between "Frame" and "NoFrame", and one text field which allows the user to enter the text to be drawn.

When the user releases the mouse at a position on the window, the program should draw a text string, where the bottom left of the text string should be at this position.

A text box with a frame should have a rectangle drawn at the boundary of the text string. You may assume each character is 10 pixels high and 5 pixels wide.




```
//Some methods are done for you.
public void setupGUI() {
    UI.initialise();
    UI.addButton("Frame/NoFrame", this::setFrame);
    UI.addTextField("Enter text", this::setText);
    UI.setMouseListener(this::doMouse);
    UI.setWindowSize(500,400);
    UI.setDivider(0.0);
}

public static void main(String[] args) {
    new Program3().setupGUI();
}
```

```
public class Program3{
    public static final double CHAR_HEIGHT= 10;
    public static final double CHAR_WIDTH= 5;

    // Fields
    private boolean frame = true;
    private String text ="";



    public void setupGUI(){...}    // shown on facing page

    /** Swap between "Frame" and "NoFrame" */
    public void setFrame(){


        this.frame = !this.frame;



    }
    /** Change the text to be drawn. */
    public void setText(String s){



        this.text = s;

    }
    /** Respond to the mouse events. */
    public void doMouse(String action, double x, double y){


        if ( action.equals("released")){
            if (this.frame){
                double width = this.text.length() * CHAR_WIDTH;
                UI.drawRect(x, y-CHAR_HEIGHT, width, CHAR_HEIGHT);
            }
            UI.drawString(this.text, x, y);

        }

    }
}
```

Question 4. Files **[14 marks]**

Suppose a file consists of some data with comments added. Comments start with a < and end with a >.

Complete the printWithoutComments() method so that it will ignore all comments and only print the tokens that are not inside comments, with each token on a separate line.

For example, suppose a file contains:

```
Toyota Aygo <good>
Mitsubishi <might be better> Star
< on special > <cool> Skoda Citigo
Fiat 500 <check this one
and <later> this one> Mini Cooper
```

If the user chose this file, printWithoutComments() should print out

```
Toyota
Aygo
Mitsubishi
Star
Skoda
Citigo
Fiat
500
Mini
Cooper
```

Notes:

- Comments can be anywhere in the file and can consist of one word, multiple words or multiple lines.
- Every < is at the beginning of a token and every > is at the end of a token.
- Comments can be nested inside other comments (like the last comment in the example above).
- You can get **10 marks** for an answer that assumes there are no nested comments.
- For **full marks**, your answer must also remove comments that have nested comments.

```
    public void printWithoutComments() {
        try {
            String filename = UIFileChooser.open("Choose a file");
            Scanner sc = new Scanner(new File(filename));
            while (sc.hasNext()) {
                String s = sc.next();
                if (s.startsWith("<")) {
                    String next = s;
                    while (sc.hasNext()) {
                        if (next.endsWith(">")){
                            break;
                        }
                        next = sc.next();
                    }
                }
                else {
                    UI.println(s);
                }
            }
\\OR (for full marks)
            Scanner sc = new Scanner(Path.of(filename));
            while (sc.hasNext()) {
                String token = sc.next();
                if (token.startsWith("<")) { depth++; }
                if (depth==0){ UI.println(token); }
                if (token.endsWith(">")){ depth = Math.max(depth-1, 0); }
            }









        } catch(IOException e) {UI.println("File error");}
    }
```

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