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

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

COMP 102: Lab Assessment 3

2020, Dec 18

Instructions

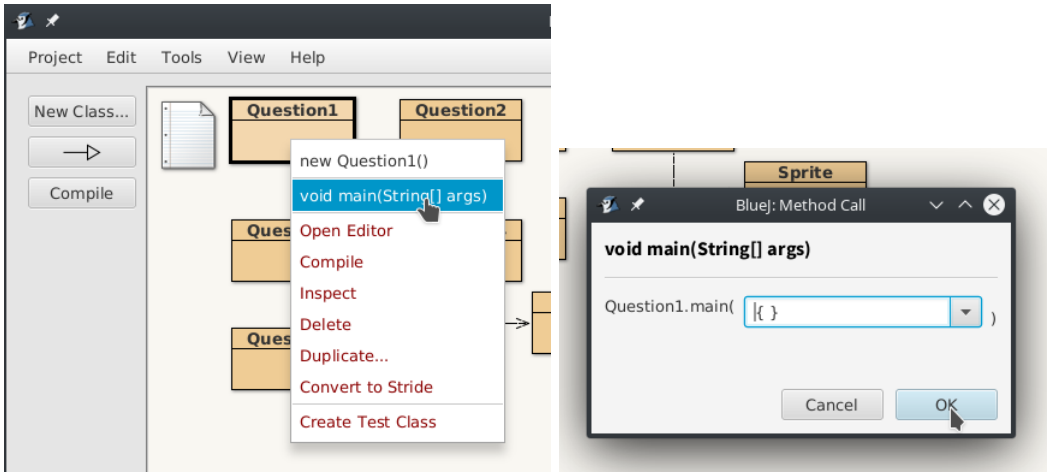
- Time allowed: **TWO AND A HALF HOURS**
- Attempt **all** the questions. There are 120 marks in total.
- To write your answers in BlueJ, follow the instructions on page 2.
- If you think a question is unclear, ask for clarification.
- This test contributes 40% of your final grade.
- You may access the online Java Documentation.
- You may access the lecture notes for COMP102.
- You may use dictionaries and calculators.
- You may not access any other web sites or online help of any kind.
- You may write notes and working on this paper, but make sure your answers are clear.

Questions:

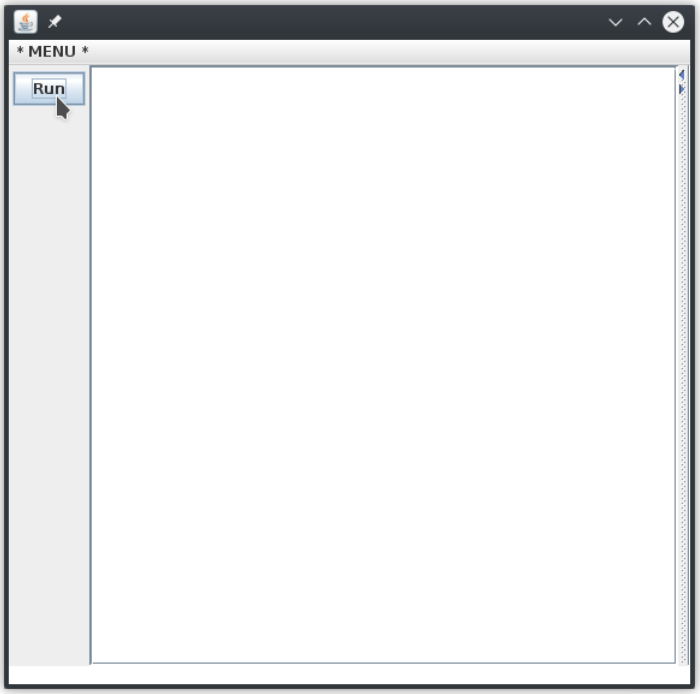
1. Basic Java [27]
2. Design a class [15]
3. ArrayLists of Numbers [23]
4. ArrayLists of Objects [32]
5. 2D Arrays [23]

Running the code:

1. Download the COMP102-LabAssessment-3.zip file from
   https://ecs.wgtn.ac.nz/Courses/COMP102_2020T3/LabAssessment3
2. Unzip the file, and open the project in BlueJ.
3. Each question has been given its own class. To run them, right-click and select 'void main(String[] args)'. Hit 'OK' when it prompts you for arguments.



4. The UI that pops up will have a 'Run' button that will execute the code for the question, once you've written it.



Question 1. Basic Java **[27 marks]**

(a) **[4 marks] Writing methods**.

Define a method called myMethod in Program1. The method must:

- have two parameters:
  1. n, an integer
  2. text, a String
- Print the string stored in text n times to the UI on separate lines.

For example, the following code would print "I must not play games in class." 10 times.

```
public void printStuff (){
    String text = "I must not play games in class.";
    int n = 10;
    this.myMethod(n, text);
}
```

Output:

```
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
I must not play games in class.
```

(Question 1 continued on next page)

(Question 1 continued)

Consider the following Circle class, specifying objects with four fields, a constructor and four methods.

```
class Circle {
    private Color color;
    private double size;
    private double x,y;

    public Circle(double x, double y, double size, Color color){
        this.x = x;
        this.y = y;
        this.size = size;
        this.color = color;
    }

    public void grow(){
        this.erase();
        this.size = this.size + 5;
        this.draw();
    }

    public void shrink(double amount){
        this.erase();
        this.size = this.size - amount;
        this.draw();
    }

    public void draw(){
        UI.setColor(this.color);
        UI.fillOval(x-size/2, y-size/2, size, size);
        UI.sleep(100);
    }

    public void erase(){
        UI.eraseOval(x-size/2, y-size/2, size, size);
    }
}
```

(Question 1 continued on next page)

(Question 1 continued)

(b) **[6 marks] Calling methods on objects**.

Given the Circle class on the facing page, complete the useCircle() method in Program1 so that it:

- makes a blue circle at (250,100) with a size of 35
- makes a green circle at (75, 300) with a size of 20
- draws both circles on the screen
- makes the first circle grow 2 times
- makes the second circle grow 3 times
- shrinks the first circle 20 pixels
- shrinks the second circle 12 pixels

(c) **[8 marks] Arrays**.

Complete the averageArray(...) method in Program1 so that it computes and prints the average of the numbers in the nums parameter. If you test your code with the provided button, the answer should be 3.375.

(d) **[9 marks] ArrayLists**.

Complete the countPositives() method in Program1 so that it counts and prints the number of positive values that appear in the ArrayList of Doubles stored in nums.

For example, if the user enters "-1 8.8 -2.1 0 9 2 5", it should print "There were 4 positive numbers."

NB. Zero is not positive or negative.

Question 2. Design a class **[15 marks]**

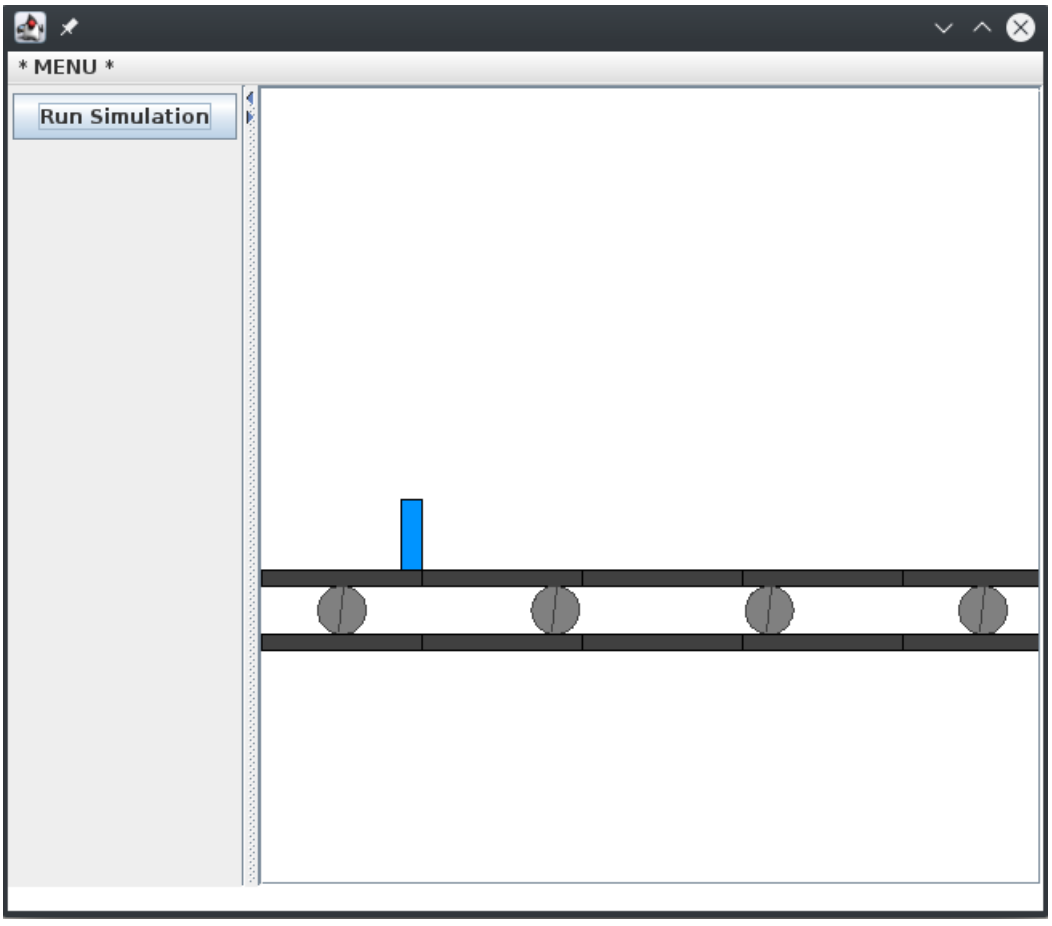
Suppose you are writing a program to simulate a production line where many boxes are moving on a conveyor belt from right to left.

Complete the Box class in BlueJ to specify Box objects that are used by the simulation in Program2.

You do not need to modify the Program2 class, only the Box class.

- Each box should have fields that store its state. You will need to add five fields to the Box class so that it stores:
  - the position of the **bottom** of the box, as X,Y coordinates.
    - The X co-ordinate will specify the center point of the box
    - The Y coordinate will specify the base of the box.
  - the width of the box as a double,
  - the height of the box as a double, and
  - the colour of the box.

  Each box should have the default start position at (400,300), where 400 is the x location of the center of the box, and 300 is the position of the conveyor belt (i.e. the bottom of the box).
- The constructor should create a Box object with the specified width, height, and color. It should initialize the position to the default position (400,300). It should then call the draw() method to show the box.
- The moveLeft(double pixels) method should move the box the specified distance to the left. It should not redraw it.
- The getX() method should return the x position of the box.
- The getWidth() method should return the width of the box.

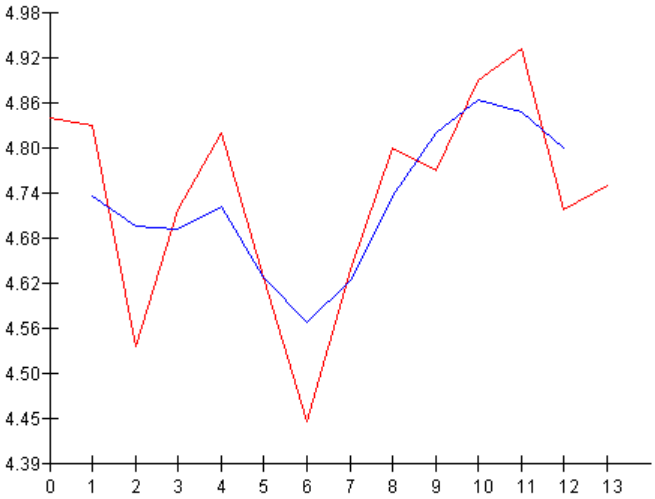


Question 3. ArrayLists of Numbers **[23 marks]**

Complete the findMovingAverage(...) method in the Program3 class.

Suppose you have a sequence of numbers representing the end-of-day stock price of a NZ company over a two week period. For example. the table shown below, which has also been plotted as the red line on the graph.

Day	Price
1	$4.84
2	$4.83
3	$4.54
4	$4.72
5	$4.82
6	$4.63
7	$4.44
8	$4.64
9	$4.80
10	$4.77
11	$4.89
12	$4.93
13	$4.72
14	$4.75



Notice that the red line jumps around a lot, giving the graph a sharp, random-seeming appearance. In order to smooth this data out, analysts often calulate a moving average, as demonstrated by the blue line. Each average is taken from three data points.

Complete the findMovingAverage(...) method so that it will return a list of smoothed values by finding the three-point moving average of the numbers. The code to display the graph has been written for you, so you can test that your method works by comparing it to the image above.

For example, if your data set is the numbers 4, 8, 2, 1, 6, findMovingAverages(...) would:

1. Find the average of the first three numbers: (4+8+2)/3 = 4.67
2. Move along one space: (8+2+1)/3 = 3.67
3. Move along one space: (2+1+6)/3 = 3
4. Return the list:

$$[4.67, 3.67, 3]$$

There are two ways to run the program:

- You can press the 'Run' button to load the data in the table above from the q3data.txt file.
- You can enter your own data (separated by spaces) into the text-field on the left of the UI, and press enter.

Question 4. ArrayLists of Objects **[32 marks]**

This question concerns a program for tracking the books in a collection, as well as their rating (from 0-5).

There are two classes: Program4 and a Book class. You must complete the Program4 class.

Program4 has a collection field which contains an ArrayList of Book objects.

The Book class below defines Book objects, which stores the title, author, and rating of books.

The title and author are both strings, such as "Harrow the Ninth", or "Tamsyn Muir"

The rating is stored as a double, but is always between 0 and 5.

```
class Book {
    private String title;
    private String author;
    private double rating;

    /** Constructs a new Book object */
    public Book(String t, String a, double r) {
        this.title = t;
        this.author = a;
        this.rating = r;
    }

    /** Returns the title of this book */
    public String getTitle() { return this.title; }

    /** Returns the author of this book */
    public String getAuthor() { return this.author; }

    /** Returns the rating of this book */
    public double getRating() { return this.rating; }

    /** Returns a string representation of this book suitable for printing */
    public String toString() { ... }
}
```

(a) **[15 marks]** Complete the searchByTitleOrAuthor(...) method that will print every book that has a title or author that contains the text specified in the parameter. For example, searchByTitleOrAuthor("Time") might print:

'This Is How You Lose the Time War', by Amal El-Mohtar and Max Gladstone (5.0/5.0)
'A History of the Brave Companions', by Timeon of Dorne (2.0/5.0)
'A Brief History of Time', by Stephen Hawking (4.2/5)

If no matching books are found, this method should display a "no matching books found" message.

(b) **[17 marks]** Complete the addBook(...) method that will create a book with the specified title, author, and rating, and then add the book to the collection **if and only if**:

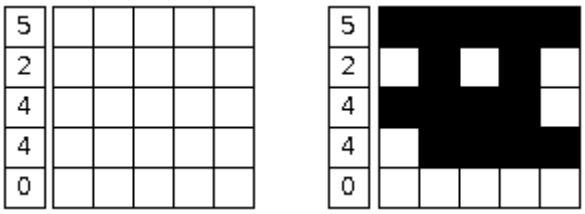
- The rating is valid (between 0 and 5 inclusive)
- A book with that title is not already in the collection

The method should return true if the book was successfully added, or false otherwise.

Question 5. 2D Arrays **[23 marks]**

The Program5 class contains a program for two highly simplified versions of a grid-based logic puzzle. The program displays a grid of filled and unfilled squares, and an array of numbers on the left that indicates the number of filled squares that should be in each row. The user can click on the grid to toggle a square between filled and unfilled.

For example, here is one puzzle and a potential solution.



(These are based on nonogram puzzles, which have many more constraints)

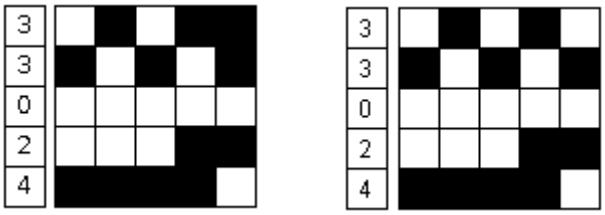
- The grid field is declared to hold a 2D array of booleans. A true value indicates that the square is filled in, while a false value indicates that the square is blank.
- The rowCounts field is a 1D array that represents the number of filled squares that should appear in each row, from the top down.

The program has the fields:

```
private boolean[ ][ ] grid ;
private int[ ] rowCounts;
```

(a) **[8 marks]** Complete the checkRowCountsSimple() method in Program5 so that it counts the number of filled squares in each row of the grid, and checks that the counts match the values stored in rowCounts. Your method should still work whatever the size of the grid array.

For example, the grid on the left should be valid, but the grid on the right is not, as it does not have the correct number of filled squares on the first row.



The test code provided lets you create new grids to test your code in two ways:

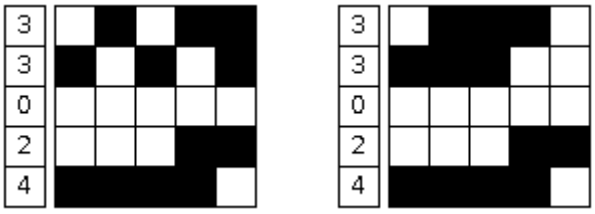
- Press the 'Make New Grid' button. Generated grids will be square, with a size of 4, 5, or 6, and random row counts.
- You can enter row counts in the text field to specify a grid. Entering "5 2 4 4 0" and pressing enter will generate the example grid shown above.

(Question 5 continued on next page)

(Question 5 continued)

(b) **[15 marks]** Complete the checkRowCountsContinuous() method in Program5 so that it counts the number of **connected** filled squares in each row, and checks that the counts match the values stored in rowCounts. Your method should still work whatever the size of the grid array.

For example, the grid shown on the left is no longer valid, as the first two rows have gaps between filled cells. The grid on the right does not, and would be a valid solution.



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