Computer Science

COMPSCI 230
Assignment ONE

Introduction

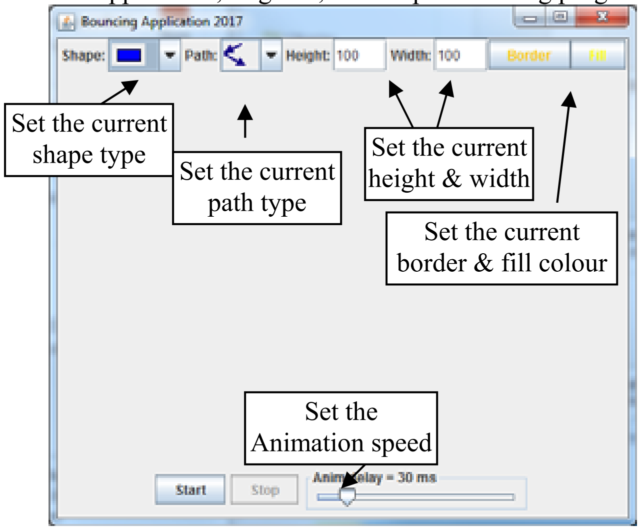
In this programming assignment, you are asked to add extra functions to the bouncing program. The aim of the assignment is to give you experience with oriented programming principles of inheritance and polymorphism.

Due Date

Due:	**11:59 pm Friday 20th April 2018**
Worth:	**5% of the final mark**

Introduction - The Bouncing Program

The application, as given, is a simple bouncing program. Different shapes move around in various paths.


Bouncing Application 2017
Shape:
Path:
Height: 100
Width: 100
Border
Fill
Set the current shape type
Set the current path type
Set the current height & width
Set the current border & fill colour
Set the Animation speed
Start
Stop
Animation delay = 30 ms

Actions

Shape Creation:
The user can create a new shape by clicking anywhere within the panel area of the program. The properties of the newly created shape are based on the current values saved in the appropriate UI fields (e.g. height, width border colour, fill colour and the moving path).

Selecting/deselecting shapes:
A user can select a shape by clicking anywhere on the shape. If a shape is selected, all its handles are shown. The user can change the path types/widths/heights/border colours/fill colours for all selected shapes by changing the current values with the help of the tools provided at the top of the application interface. (But the shape type can't be modified once a shape has been created.) Clicking on a selected shape will deselect it.

Tools

Shape Combo Box:	The 'Shape' combo box sets the current shape type for new shapes. Clicking in the panel area for Shape Creation will create the selected type of the shape. A rectangle can be selected in the program.
Path Combo Box:	Users may select one of several moving paths for shapes from the 'Path' combo box. Selecting a new path changes the path of all currently selected shapes. Additionally, the new path becomes the current path for any new shapes that are created.
Width TextField:	Users may change the current width of new shapes and currently selected shapes by entering a valid number in the width text field and pressing "ENTER".
Height TextField:	Users may change the current height of new shapes and currently selected shapes by entering a valid number in the height text field and pressing "ENTER".
Border Button	Users may change the current border colour of new shapes and currently selected shapes by pressing the border button.
Fill Button	Users may change the current fill colour of new shapes and currently selected shapes by pressing the fill button.
Start Button:	Starts the animation.
Stop Button:	Stops the animation.
Animation Slider:	Users may use the animation delay slider to adjust the speed of the animation.
Popup Menu:	The application has a popup menu, which is activated by clicking the right mouse button anywhere in the panel area (on a windows machine). The popup menu contains a menu item called "Clear All" which allows the user to clear all shapes from the program.

What you are to do

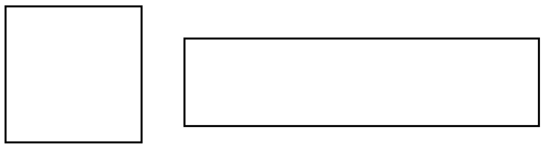
Firstly, become familiar with the program supplied. The files included in the program are as follows:

- `A1.java`
- `AnimationPanel.java`
- `MovingShape.java`

Download all source files from the assignment course page.The program is incomplete. You are required to modify the program and add a few more shapes into the program. Your assignment is divided into several stages for ease of completion. Please complete the assignment in order of the stages.

Stage 1: The MovingRectangle Class (5 marks)

You are required to add a `MovingRectangle` subclass to your program. This class should draw a rectangle/square based on the mouse-point, size, the current border colour, the current fill colour and, the current moving path saved in the `AnimationPanel`. Some examples are shown as below:

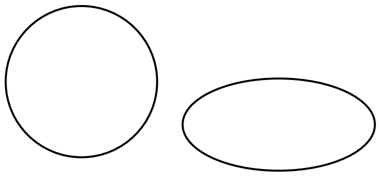


Assessment criteria:

- [1 mark] The class hierarchy should be developed sensibly and in accordance with good object-oriented programming practice.
- [1 mark] The draw method is overridden correctly
- [1 mark] The contains method is overridden correctly
- [1 mark] Users should be able to create a new rectangle using the current fill and border colours, height, width and path in the program.
- [1 mark] Users should be able to change the fill and border colours, width, height and bouncing path of selected rectangles.

Stage 2: The `MovingOval` Class (5 marks)

You are required to add a `MovingOval` subclass to your program. This class should draw a circle/ellipse based on the mouse-point, the current width, the current height, the current border colour, the current fill colour, and the current moving path saved in the `AnimationPanel`. Some examples are shown as below:



Note: You may use the following formula to check if the mouse point is clicked within a circle/ellipse.

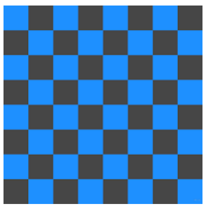
```
Point EndPt = new Point(topLeft.x + width, topLeft.y + height);
dx = (2 * mousePt.x - topLeft.x - EndPt.x) / (double) width;
dy = (2 * mousePt.y - topLeft.y - EndPt.y) / (double) height;
return dx * dx + dy * dy < 1.0;
```

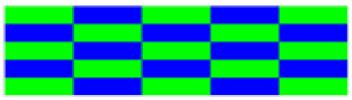
Assessment criteria:

- [1 mark] The class hierarchy should be developed sensibly and in accordance with good object-oriented programming practice.
- [1 mark] The draw method is overridden correctly
- [1 mark] The contains method is overridden correctly
- [1 mark] Users should be able to create a new oval using the current fill and border colours, height, width and path in the program.
- [1 mark] Users should be able to change the fill and border colours, width, height and bouncing path of selected ovals.

Stage 3: The `MovingChecker` Class (10 marks)

You are required to add a `MovingChecker` subclass to your program. This class should draw a checker pattern based on the mouse-point, the current width, the current height, the current border colour, the current fill colour, and the current moving path saved in the `AnimationPanel`. Some examples are shown as below:


(Fill: blue, Border: black)

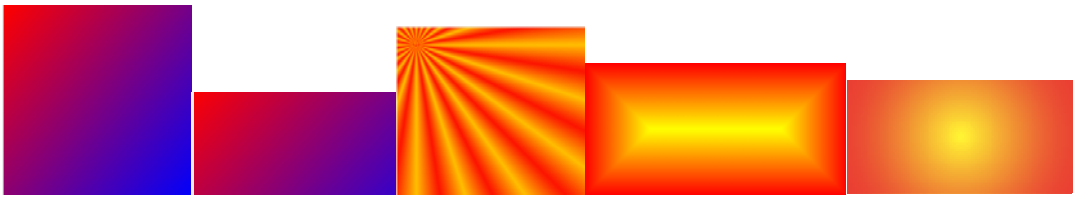

(Fill: green, Border: blue)

Assessment criteria:

- [2 marks] The class hierarchy should be developed sensibly and in accordance with good object-oriented programming practice. Do you need to override the `draw` and/or `contains`?
- [2 marks] You should add two variables to store the number of blocks horizontally and vertically. For example, there are 8 x 8 blocks in the first image and 5 x 5 blocks in the second image.
- [3 marks] Users should be able to create a new checker pattern using the current fill (first) and border (second) colours, height, width and path in the program.
- [3 marks] Users should be able to change the fill and border colours, width, height and bouncing path of selected patterns.

Stage 4: The `MovingGradient` Class (10 marks)

You are required to add a `MovingGradient` subclass to your program. You are now required to get creative and add your own gradient pattern that will make the bouncing program more interesting! This class should draw a gradient-rectangle based on the mouse-point, the current width, the current height, the current border colour, the current fill colour, and the current moving path saved in the `AnimationPanel`. Some examples are shown as below:

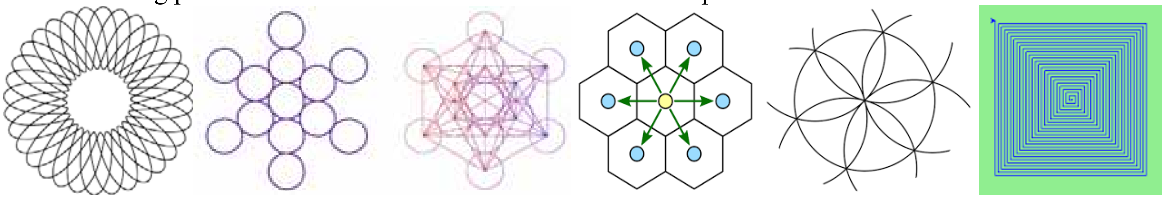

(Fill: Red, Border: blue)

Assessment criteria:

- [3 marks] The class hierarchy should be developed sensibly and in accordance with good object-oriented programming practice. Do you need to override the `draw` and/or `contains`?
- [2 marks] Users should be able to create a new gradient pattern using the current fill (first) and border (second) colours, height, width and path in the program.
- [2 marks] Users should be able to change the fill and border colours, width, height and bouncing path of selected patterns.
- [3 marks] Student's work shows great preparation, creativity or effort.

Stage 5: The `MovingShapePattern` Class (10 marks)

You are required to add a `MovingShapePattern` subclass to your program. You are now required to get creative and add your own pattern that will make the bouncing program more interesting! This class should draw a pattern based on the mouse-point, the current width, the current height, the current border colour, and the current moving path saved in the `AnimationPanel`. Some examples are shown as below:



Assessment criteria:

- [3 marks] The class hierarchy should be developed sensibly and in accordance with good object-oriented programming practice. Do you need to override the `draw` and/or `contains`?
- [2 marks] Users should be able to create a new shape pattern using the current **border** colour (no fill color), height, width and path in the program.
- [2 marks] Users should be able to change the **border** colour, width, height and bouncing path of selected patterns.
- [3 marks] Student's work shows great preparation, creativity or effort.

Stage 6: Adding a New Path (5marks)

In this part, you are required to add your own designed path to the bouncing program. The `MovingPath` is an abstract inner class which contains an abstract method. You are required to add a new subclass which extends the `MovingPath` and implement the `move`() method.

Assessment criteria

- [1 mark] The class hierarchy should be developed sensibly and in accordance with good object-oriented programming practice.
- [2 marks] Users should be able to add a new shape using your own designed path.
- [2 marks] Users should be able to change the bouncing path of selected shapes to your own designed path.

Submission

You may electronically submit your assignment through the Web Dropbox (https://adb.auckland.ac.nz/) at any time from the first submission date up until the final date. An electronic receipt is issued. Remember that you must be logged into Net Account under your own login to use the Assignment Dropbox. You can make more than one submission. However, **every submission that you make replaces your previous submission**. Submit **ALL** your files in every submission. **Only your very latest submission will be marked.** Please double check that you have included all the files required to run your program and the A1.txt in the zip file before you submit it. **No marks will be awarded if your program does not compile and run.**

You are to electronically submit ONE **A1.zip** file containing all the following files:

1. All source files (i.e. new, changed, and unchanged) - Your name, UPI and a comment at the beginning of each file must be included in all your files.
2. All gif files (used as icons in the program)
3. **A1.txt**

What to include inside the A1.txt file

You must include a text file named A1.txt in your submission. This text file must contain the following information:

- Your name, login name and ID number
- How much time did the assignment take overall?
- What areas of the assignment did you find easy and difficult?
- Which topics of the course did the assignment most help you understand?
- Any other comments you would like to make.

DO NOT SUBMIT SOMEONE ELSE'S WORK:

- The work done on this assignment must be your own work. Think carefully about any problems you come across, and try to solve them yourself before you ask anyone for help.
- Under no circumstances should you take or pay for an electronic copy of someone else's work. This will be penalized heavily.
- Under no circumstances should you give a copy of your work to someone else
- The Computer Science department uses copy detection tools on the files you submit. If you copy from someone else, or allow someone else to copy from you, this copying will be detected and disciplinary action will be taken.
- To ensure you are not identified as cheating you should follow these points:
    - Always do individual assignments by yourself.
    - Never give another person your code.
    - Never put your code in a public place (e.g., forum, your web site).
    - Never leave your computer unattended. You are responsible for the security of your account.
    - Ensure you always remove your USB flash drive from the computer before you log off.