Computer Science

COMPSCI 230
Assignment TWO

Introduction

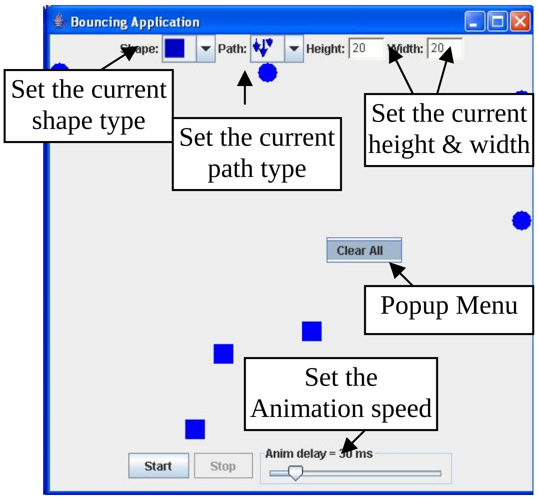
In this programming assignment, you are asked to add extra functions to the bouncing program. The aim of the assignment is to give you experience with oriented programming principles of inheritance and polymorphism.

Due Date

Due:	**17:00 pm Monday 26th August 2013**
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.



<u>Actions</u>

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 etc).

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 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.

<u>Tools</u>

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 or a circle 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".
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:

- `A2.java`
- `AnimationPanel.java`
- `MovingShape.java`
- `MovingRectangle.java`
- `MovingSquare.java`
- `MovingRightAngleTriangle.java`

Download all source files from the assignment course page. The design and implementation of the program will be covered in lectures, please refer to the relevant material. It is strongly recommended to start as early as you can and implement the parts you know as soon as they are taught in lectures.

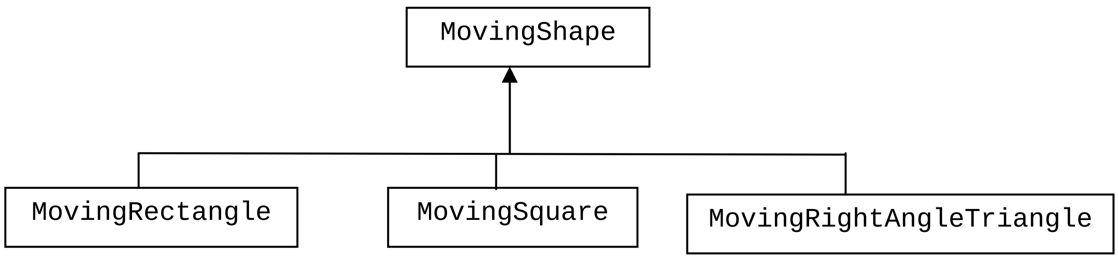
Your assignment is divided into several stages for ease of completion. Please complete the assignment in order of the stages.

Stage 1: Your UPI (2 marks)

Your UPI should be displayed inside the title bar of the `JFrame`.

Stage 2: Restructuring (8 marks)

The original inheritance hierarchy is given below:



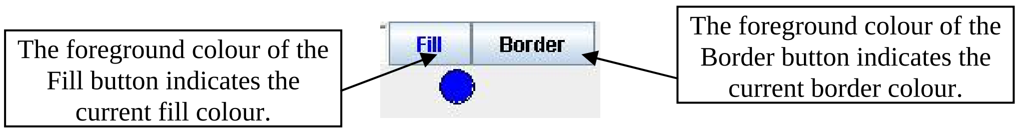
Re-write the inheritance hierarchy for class `MovingShape`, `MovingRectangle`, `MovingSquare` and `MovingRightAngleTriangle`. Use `MovingShape` as the superclass of the hierarchy. Make the hierarchy as deep (i.e., as many levels) as possible.

Stage 3: Fill and Border Colours (20 marks)

In this part you are required to modify classes in A2 which enable users to change the border and/or fill colours of all the currently selected shapes and the current fill and border colour that will be used when creating new shapes.

You are required to add two buttons – 'Fill' and 'Border' two colour choosers and two event handling methods to the `A2` class. You may add the buttons at either the top or the bottom of the panel.

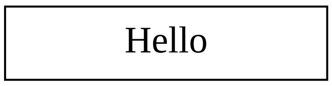
When the user clicks on the 'Fill' or 'Border' colour button, a colour chooser is shown to the user. When a colour is selected, the corresponding foreground colour of the button should be changed. It shows the fill colour that will be used for new shapes. Changing the border colour should behave similarly.


The foreground colour of the Fill button indicates the current fill colour.
Fill
Border
The foreground colour of the Border button indicates the current border colour.

You are also required to add two methods (`setCurrentBorderColor` and `setCurrentFillColor`) to the `AnimationPanel` class in order to set the colours of all the currently selected shapes and the colours that will be used when creating new shapes and add two methods to the `AnimationPanel` to return the fill/border colours (`getCurrentFillColor` and `getCurrentBorderColor`).

You should add the set and get methods to the `MovingShape` class in order to set or get the corresponding fill/border colour of shapes. You should modify the `draw` method of `MovingRectangle` class (and all subclasses) in order to use the fill/border color attribute stored in the superclass to fill/draw the shape.

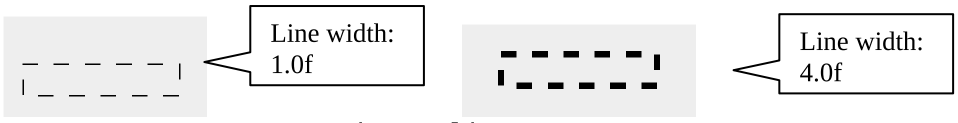
Stage 4: MovingMessage Class (20 marks)



You are required to add the `MovingMessage` shape to the bouncing program. The `MovingMessage` is just a text rectangle. The text message should be placed to the centre of a rectangle vertically and horizontally.

You are also required to add a text field control which allows users to enter the message. When a text message is entered, your program should change the message of all the currently selected `MovingMessage` shapes and the current message that will be used when creating new `MovingMessage` shapes. Again, you are required to add get and set methods to the `AnimationPanel` class.

Stage 5: MovingOutline Class (20 marks)



You are required to add the `MovingOutline` shape to the bouncing program. The `MovingOutline` is just an outline of a rectangle.

You are also required to add a control to adjust the line width of all `MovingOutline` shapes in the bouncing program. When a value is selected from the slider, your program should change the line width of all the currently selected `MovingOutline` shapes and the current line width that will be used when creating new `MovingOutline` shapes. Again, you are required to add get and set methods to the `AnimationPanel` class.

Inheritance hierarchy (Stage 4 & Stage 5)
The `MovingShape` is an abstract class which contains two abstract methods: draw and contains. You are required to add TWO new subclasses. You may need to implement some or all abstract methods for the new shapes. Why? You may also need to add a private instance field to store a specific property of the new shape. You will need to think carefully on the structure of the inheritance hierarchy.

A2 and AnimationPanel (Stage 4 & Stage 5)
Next, you are required to add a new `ImageIcon` to the '`Shape`' combo box control for each new type of shape and modify the `createNewShape` method in the `AnimationPanel` class which allows users to create each new subclass instance.

Stage 6: Adding a New Path (15 marks)

In this part, you are required to add your own designed path to the bouncing program. You are required to implement the following methods for the new path:

- constructor(s) to create a new object.
- move(…) to move the shape according to the path.

Bonus: Adding your own feature (10 marks)

You are now required to get creative and add your own special feature that will make the Bounce program more interesting! You can do anything you like. You can add another new shape, a new moving path, a scaling function, etc.

To get the marks for this part, the feature should be non-trivial to implement and reasonably interesting. You should be sure to explain what your extra feature is in the A2.txt file (see below), including a very short overview of how you implemented it (what methods you added, in which classes, etc). You shouldn't need to write more than a few sentences.

Any bonus awarded for this advanced part can be used to make up for lost marks in any stage of this assignment. (i.e. The marks add up to 100 with the Bonus is included.)

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. 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 A2.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 **A2.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. **A2.txt**

What to include inside the A2.txt file

You must include a text file named A2.txt in your submission. There will be a 5 mark allocated for this section. 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?
- What areas of the assignment did you find difficult?
- Which topics of the course did the assignment most help you understand?
- Any other comments you would like to make.
- What is the extra feature that you have implemented in the bonus section? If you don't specify it in this file, the markers might not notice it and you won't get the marks!
    - Please put a very brief description of how you implemented this feature – what methods/classes/interfaces you added or modified.

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.

Marking

- **Style – 10 marks**
    - Comments at the top of each class (containing your name, upi, date and a brief description of the class), good variable names, good method names, correct indentation and uses helper methods to simplify and structure the code.
- **Correctness – 90 marks**
    - Stage 1 – Your UPI – 2 marks
    - Stage 2 – Restructuring – 8 marks
    - Stage 3 – Fill and Border Colours – 20 marks
    - Stage 4 – Adding a new shape: `MovingMessage` – 20 marks
    - Stage 5 – Adding a new shape: `MovingOutline` – 20 marks
    - Stage 6 – Adding a new path – 15 marks
    - A2.txt – 5 marks
    - Bonus – Add your own feature to the bouncing program