Enums and Strings

Objective

See how to define and use the `enum` keyword to define a new type. Consolidate on knowledge of the functionality of class `String` and introduce the very useful `StringBuilder` class.

Part 1 – Using an enum

1. Open the Game project you created previously.
2. Circle/oval is not the only shape! You can give the Ball class a property that dictates its shape. However, the shape must be limited to a list which you define. Let's create this type as an **enum**.
3. Define a new **enum** type called **SHAPE_TYPE** with the following values:

   Rectangle
   ThreeDRectangle
   RoundRectangle
   Oval
   Arc

   Please create this **enum** outside of the Ball class or in its own file.
4. Now you can see the name **Ball** does not look like a good choice!
   Please change the name Ball to **Shape** in your project. The best way to do this is to open the Ball class, right-click on the word **Ball**, and then choose the **Refactor->Rename** menu options. The editor will change all references to Ball.
5. Define a new private field called **shapeType** of type **SHAPE_TYPE**
   as: **private SHAPE_TYPE shapeType;**
6. Create a getter for this field.
7. Set this value inside the constructor.
   Tip: Add a parameter of type SHAPE_TYPE to the constructor.
8. Back in the paint method, you can now examine the getShapeType() to see what to draw. for example:

   if (**getShapeType()** == SHAPE_TYPE.**RoundRectangle**)
   g.drawRoundRectangle(……);
9. Run your application to see different shapes bouncing about!
   You can also change the colour of your shape by creating a new field of type Color (like: private Color colour;)
   Set its value in the shape's constructor and also create a getter method.
   You can then use this in paint when drawing a shape using code like:
   g.setColor(shape.getColour());
   g.drawRect(…);

Part 2 – Using String

1. Expand `main()`, and declare a `String` called Name whose value is any first name of any length greater than three characters.
2. Display its third character using `charAt()` (can also be done with substring).
3. Display it converted to lowercase and to uppercase.
4. Use an enhanced `for` loop to iterate over its characters (use `toCharArray()`) and display each of them tab separated. Throw a line feed after this display.
5. Display whether it `startsWith` a `String` of your choosing.
6. Display whether it `endsWith` a `String` of your choosing.
7. Use `indexOf` to display the position in the `String` of the first occurrence of a character that you know is in the `String`, and also for a character that you know is not in the `String`.
8. Concatenate the 'name' with a surname of your choice to make a variable called '`fullname`' preferably with a space in the middle, then display this `fullname`. Concatenation is ok if it is all done in one statement.

Part 3 – Using StringBuilder

1. Back in main(), create a **`StringBuilder`** object called '**sb**'. Use the constructor that allows you to initialise the object to contain the `String`

   'Bruce Springsteen<space>'.

   (You can use the name of your favorite artist instead!)
2. Now use the `append()` instance method of *`StringBuilder`* to append exactly the text 'is the artist ever' (no error in that!).
3. Use the `toString()` method of the `StringBuilder` to produce a string that you can display to see the current value of the `StringBuilder`.

   You are looking at a strange sentence that needs some amending.
4. Now we would like you to `insert()` an adjective in front of the word 'artist'. Words like 'greatest' obviously spring to mind, but make your own choice.
5. Now use the `replace()` method of `StringBuilder` to replace the word

   'artist' with a noun of your own choice, e.g., 'rock singer'. Display the final result.