ITCS 209 Object Oriented Programming	Name:	Lab Score	Challenge Bonus
	ID:		
	Section:		

Lab04: Classes, Objects, Methods

Change Log:

- [30 Jan] For Task 1.1, change class name from "`FourBoxPrinter`" to "`FourRectanglePrinter`"
- [30 Jan] For Task 2.4, change the expected output – as shown in red color, and upload a new `ItemTester.java` file in the starter code folder on MyCourses

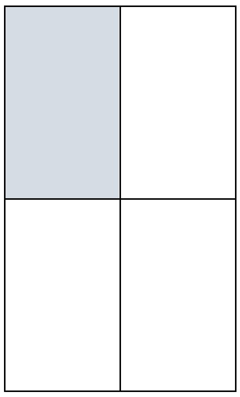
Objectives:

- Students can construct objects of the premade class, and call methods to modify their properties.
- Students can create classes, constructor, setter/getter, instance methods.

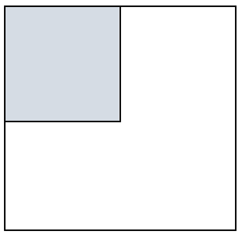
Task 1: Working with a premade class in Java named "`Rectangle`" class. [Ref. Java Concepts Book]

(For more detail, please read Java API https://docs.oracle.com/javase/8/docs/api/java/awt/Rectangle.html)
Note that in this task, your program will not produce a drawing. It will simply print the location of the rectangles.

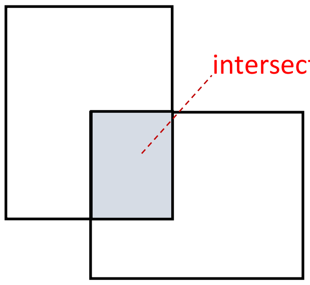
1.1 Modify the given class called "**`FourRectanglePrinter`**" to write a program that constructs a `Rectangle` object named "`box`" representing a box with a top-left corner (100, 50) with width 150 and height 200, prints its location by calling `System.out.println(box)`, and then **`translates`** (there is a method in the `Rectangle` class to do this task) and prints it three more times, so that if the rectangle were drawn, they would form one large rectangle.



1.2 Write a program called "**`GrowSquarePrinter`**" that constructs a Rectangle object named "`square`" representing a square with a top-left corner (100, 100) and side length 50, prints its location by calling `System.out.println(square)`, applies the **`translate`** and **`grow`** methods (please read the API document) and calls `println` statement again. The calls to **`translate`** and **`grow`** should modify the `square` object so that it has twice the size and the same top-left corners as the original.



1.3 Write a program called "`IntersectionPrinter`" that constructs two rectangle objects (define your own location and size), prints their locations by calling `System.out.println(r1)` and `System.out.println(r2)`, finds the intersections of two rectangles using **`intersection`** method (please read the API documentation), prints that intersection rectangle. If the rectangles do not overlap, the program should print the "`DO NOT OVERLAP`" message. **Explain how you check whether the rectangles overlap or not.**



Task 2: Implementing a new user-defined class

2.1 Implement a class "`Item`". Each item has a name (a text), a price (a decimal point number), and a calorie (a whole number). Please choose an appropriate data type for each attribute. All attributes must be available to access inside the class only, so you must use an appropriate access modifier.

2.2 Implement two constructor methods as follow:

`public Item(String _name)` This method takes an input of *name* and assigns to the attribute *name*, the set default values of price and calorie as 0.

`public Item(String _name, double _price, int _cal)` This method takes input parameters and assigns them to each corresponding attribute.

2.3 Create the following methods to retrieve the attribute values, set new values, and manipulate the values.

`public String getName()` // return item's name

`public double getPrice()` // return item's price

`public int getCalorie()` // return item's calorie

`public void setPrice(double _price)` // set new value to item's price

`public void setCalorie(int _cal)` // set new value to item's calorie

`public void printItemInfo()`

// print all information of an item in the following format
// Name: xxx, Price: yyy, Calorie: zzz

`public void raisePrice(double percent)`

// raise the item's price by a certain percentage

`public double priceWithDiscount()`

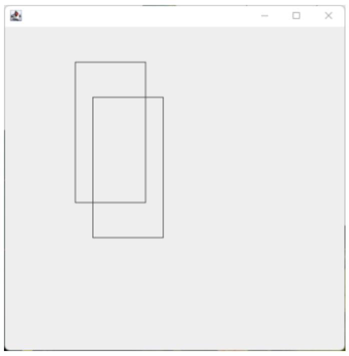
// return the price after the discount. The amount of discount depends on the item based on its price.
// If the price is less than 100, no discount is provided
// If the price is between 100 – 500 (inclusive), give a 10 percent discount
// If the price is more than 500, give a 20 percent discount
// Note that, unlike `raisePrice` method, this method should NOT change the original item's price.

2.4 Execute the **`ItemTester`** class (already provided in the code starter package) to test all methods above. Here is an expected output. [The re

```
** First Item **
Name: Chocolate, Price: 0.0, Calorie: 0
After setting price and calorie
Price: 550.0, Calorie: 200

** Second Item **
Name: Ice cream, Price: 200.0, Calorie: 375
After raising price by 20%
New price: 240.0

** Calculate Price with Discount **
Chocolate => 440.0
Ice cream => 216.0

** Final Information **
Name: Chocolate, Price: 550.0, Calorie: 200
Name: Ice cream, Price: 240.0, Calorie: 375
```

Challenge Bonus (Optional):

To visualize a `Rectangle` object, you can draw shapes inside a Java `Frame` window by constructing a **`component`** object and adding it to the frame. In the Swing toolkit, the `JComponent` class represents a blank component. The `TwoRectangleComponent` class (in the code stater package) is an example component that extends the `JComponent` class *(You will learn more about extending class in the later lecture).* This class has a "**`paintComponent`**" method which contains instructions to draw two rectangles. This method receives an object of type `Graphics`. It is cast to be `Graphics2D` to draw two-dimensional graphics. Then, the "**`draw`**" method of the `Graphics2D` class can draw shapes such as rectangles as shown below.



```
public void paintComponent(Graphics g) {
    Graphics2D g2 = (Graphics2D) g;

    Rectangle box = new Rectangle(x, y, w, h);
    g2.draw(box);
    box.translate(25, 50);
    g2.draw(box);
}
```

Your task is to create a `FourRectangleComponent` class to draw four rectangles as mentioned in Task 1.1. The given `DrawingDemo` class can be used to test your component. A user must input the value of x, y, width, and height of the rectangle. Note you must click × button to close the frame window and exit the program.

```
Input number 2 to draw two rectangles, or 4 to draw
four rectangles in the frame window: 4
Input x: 100
Input y: 50
Input width: 150
Input height: 200
```