Northeastern Illinois University
CS207, Object-Oriented Programming and Data Structures, Summer 2022
Homework 2
Due date: Thursday 6/16/2022 at 11:59 p.m.

Problem 1:

Complete a properly encapsulated class named Shape, which has the following:

- A `boolean` instance variable named `isFilled`.
- A `String` instance variable named `color`.
- A default, no-arg constructor which sets `isFilled` to `true`, and `color` to "`Green`".
- An overloaded constructor which takes two parameters, a `boolean` and a `String` and sets the instance variables accordingly.
- An overridden `toString()` method, which returns a String. The String should contain: The values of the instance variables in the following format:

```
Filled: true
Color: Green
```

Complete a properly encapsulated class named `Circle`, which **inherits** from `Shape` and has the following:

- A `double` instance variable named `radius`.

- A default, no-arg constructor which sets `radius` to `1`.

- An overloaded constructor which takes one double parameter and sets the instance variable radius to the value passed in.

- Another overloaded constructor which takes three parameters, a `double` for `radius`, a `boolean` for `isFilled` and a `String` for `color`, and sets the instance variables accordingly, Hint:(Invoke the matching constructor from the superclass)!!

- A method named `getArea()` which calculates and returns the area of the circle.

- An overridden `toString()` method. The returned String should contain: the value of radius, the area of the circle, then the result of calling the `toString()` method from the superclass, the return String should be formatted as follows:

```
Radius: 2.67
Area: 22.396099868176275
Filled: true
Color: Green
```

Complete a properly encapsulated class named `Rectangle`, which **inherits** from `Shape` and has the following:

- Two `double` instance variable named width and `length`.

- A default, no-arg constructor which sets `length` to 2 and `width` to 1.

- An overloaded constructor which takes two `double` parameter and sets the instance variables `width` and `length` to the values passed in.

- Another overloaded constructor which takes four parameters, a `double` for `width`, a `double` for `length`, a `boolean` for `isFilled` and a `String` for `color`, and sets the instance variables accordingly, Hint:(Invoke the matching constructor from the superclass)!!

- Note that a rectangle has a length that is always greater than width.

- A void `setLW()` method which takes two parameters `x` and `y`, and set the length instance variable to the largest value passed in and the width instance variable to the smallest value passed in (Assume that `x` and `y` are always positive and have different values), also call this method in the constructors, so your instance variables will always have legal values.

- A method named `getArea()` which calculates and returns the area of the rectangle.

- An overridden `toString()` method. The returned String should contain: The value of the length, the value of the width, the area of the rectangle, then the result of calling the `toString()` method from the superclass, the return String should be formatted as follows:

```
Width: 3.2
Length: 4.0
Area: 12.8
Filled: false
Color: Red
```

If you implemented your classes correctly, the output should match the follows:

```
c1:
Radius: 2.67
Area: 22.396099868176275
Filled: true
Color: Green

c2:
Radius: 3.0
Area: 28.274333882308138
Filled: false
Color: Red

r1:
Width: 2.0
Length: 3.0
Area: 6.0
Filled: true
Color: Blue

r2:
Width: 3.2
Length: 4.0
Area: 12.8
Filled: false
Color: Red
```

Instructions:

- Download the needed files and look for TestShape.java.

- Other than uncommenting the code, do not modify the main method in TestShape.java.

- Place Shape.java Circle.java and Rectangle.java in a folder named YourName_HW7

Problem 2:
Create a properly encapsulated class named `Sentence` that has the following:

- A properly encapsulated `String` instance variable named `sentence`.
- A constructor that takes 1 parameter, a `String` and set the instance variable.
- A getter method for `sentence` instance variable.
- Override the Object equals method, The method determines if two Sentence objects are equal by checking their instance variables are equal.

Create a properly encapsulated class named `Word` that inherits from `Sentence` and has the following:

- A properly encapsulated `String` instance variable named `noVowelsWord`.
- A constructor that takes 2 `String` parameters `s` and `w`, `s` is used to set the super class's instance variable and `w` is used to set the `noVowelsWord` instance variable after removing all the vowels. You may **not** use any loops or if statements to do this.
- A getter method for `noVowelsWord` instance variable.
- A method named `isSubstring` that takes no parameters and returns a `boolean`, the method returns `true` if the instance variable `noVowelsWord` is a substring of the super class's instance variable `sentence`, otherwise it returns `false`. You may **not** use any loops or if statements to do this.

TestWord.java

```
public static void main(String[] args)
{
   Word w1 = new Word("Go Cubs Go", "Cub");
   System.out.println("New Word object");
   System.out.println("Superclass word: " + w1.getSentence());
   System.out.println("Subclass word: " + w1.getNoVowelsWord());
   System.out.println("Is word a substring of sentence? " + w1.isSubstring());
   System.out.println();

   Word w2 = new Word("Applepie", "Apple");
   System.out.println("New Word object");
   System.out.println("Superclass word: " + w2.getSentence());
   System.out.println("Subclass word: " + w2.getNoVowelsWord());
   System.out.println("Is word a substring of sentence? " + w2.isSubstring());
   System.out.println();

   System.out.println("w1 and w2 are the same? " + w1.equals(w2));
}
```

Sample run:

```
New Word object
Superclass word: Go Cubs Go
Subclass word: Cb
Is word a substring of sentence? false

New Word object
Superclass word: Applepie
Subclass word: ppl
Is word a substring of sentence? true

w1 and w2 are the same? false
```

Instructions:

- Download the needed files and look for TestWord.java.
- Other than uncommenting the code, do not modify the main method in TestWord.java.
- Place Sentence.java and Word.java in a folder named YourName_HW7

General Instructions:

- No hard copies will be collected.

- Do not send your files through the email!

- You should submit your work by the due date, **No** extensions will be given. (See syllabus for late homework policy).

- DO **NOT** turn in multiple files, only one .zip file.

What to turn in:

There should be five **.java** files(Shape.java Circle.java Rectangle.java Sentence.java & Word.java), put all those files into a zip file and name it <YourFirstName_YourLastName>.zip, submit the zip file into the Dropbox on D2L.

How to zip multiple files?

On Windows: Select all the files > right click > Send to > Comprised File

On Mac: Select all the files > Click/Tap with two fingers > Compress Items