Day 13: Abstract Classes

Objective

Today, we will extend what we learned yesterday about *Inheritance* to *Abstract Classes*. Because this is a very specific object oriented concept, submissions are limited to the few languages that use this construct. Check out the Tutorial tab for learning materials and an instructional video.

Task

Given a *Book* class and a *Solution* class, write a *MyBook* class that does the following:

- Inherits from *Book*
- Has a parameterized constructor taking these 3 parameters:
  1. string $title$
  2. string $author$
  3. int $price$
- Implements the *Book* class' abstract *display()* method so it prints these 3 lines:
  1. `Title:`, a space, and then the current instance's $title$.
  2. `Author:`, a space, and then the current instance's $author$.
  3. `Price:`, a space, and then the current instance's $price$.

Note: Because these classes are being written in the same file, you must not use an access modifier (e.g.: `public`) when declaring *MyBook* or your code will not execute.

Input Format

You are not responsible for reading any input from stdin. The *Solution* class creates a *Book* object and calls the *MyBook* class constructor (passing it the necessary arguments). It then calls the *display* method on the *Book* object.

Output Format

The $void\ display()$ method should print and label the respective $title$, $author$, and $price$ of the *MyBook* object's instance (with each value on its own line) like so:

```
Title: $title
Author: $author
Price: $price
```

Note: The $\$$ is prepended to variable names to indicate they are placeholders for variables.

Sample Input

The following input from stdin is handled by the locked stub code in your editor:

```
The Alchemist
Paulo Coelho
248
```

Sample Output

The following output is printed by your *display()* method:

```
Title: The Alchemist
Author: Paulo Coelho
Price: 248
```