TOPIC : CLASSES

1. Create class SimpleSpice.
    1. The class should have a String property for the name of the spice, and a String for the level of spiciness.
    2. Set the default name to curry and the spiciness to mild.
    3. Using a string for spiciness is nice for users, but not useful for calculations. Add a heat property to your class with a getter that returns a numeric value for spiciness. Use a value of 5.
    4. Create an instance of SimpleSpice and print out its name, spiciness, and heat.
2. Let's improve the SimpleSpice class so that we can have various spices with different levels of spiciness.
    1. Create a new class, Spice.
    2. Constructor should have a mandatory String argument for the name, and a String argument for the level of spiciness where the default value is "mild" for not spicy.
    3. Add a variable, "heat", to your class, with a getter that returns a numeric value for each type of spiciness (integer values ranging from 1 for "mild" to 10 for "expremelly spicy")
    4. Create a list of Spice objects and give each object a name and a spiciness level.
    5. Add to Spice an init block that prints out the values for the object after it has been created.
    6. Create a list of spices that are spicy or less than spicy. Hint: Use a filter and the heat property.
    7. Because salt is a very common spice, create a helper function called makeSalt().
3. Let's practice inheritance.

1. Create a class, Book, with a title and an author.
2. Add a method, readPage(), that increases the value of a private variable, currentPage, by 1.
3. Create a subclass of Book; name it eBook.
4. eBook also takes in a format, which defaults to "text".
5. In eBooks, counting words makes more sense than pages. Override the readPage() method to increase the word count by 250, the average number of words per page from typewriter days.

4. Interfaces and abstract classes

1. Make a new package, Spices, with a file, Spice, that has a main() function.
2. Copy/paste your Spice class code (from ex 2) into that new file.
3. Make Spice abstract.
4. Create a subclass, Curry. Curry can have varying levels of spiciness, so we don't want to use the default value, but rather pass in the spiciness value.
5. Spices are processed in different ways before they can be used. Add an abstract method prepareSpice to Spice, and implement it in Curry.
6. Curry is ground into a powder, so let's call a method grind(). However, grinding is something that's not unique to curry, or even to spices, and it's always done in a grinder. So we can create an Interface, Grinder, that implements the grind() method. Do that now.
7. Add the Grinder interface to the Curry class.