01-Introduction to Classes Notes

a **Class** is a description of the **data** an object will contain and **methods** to process that data.
(a class is also known as a user defined data type)

an **Object** is an **instance of a class**.

Object Oriented Programming is a programming paradigm/style of programming where the **focus is on the Object (data)** rather than the process.

Process Oriented Programming (aka Procedural Programming) is a programming paradigm/style where the **focus is on the process rather than the data**.

Problem: increment a variable called counter

Object-Oriented: `counter.add(1);  // object.method(parameter)`

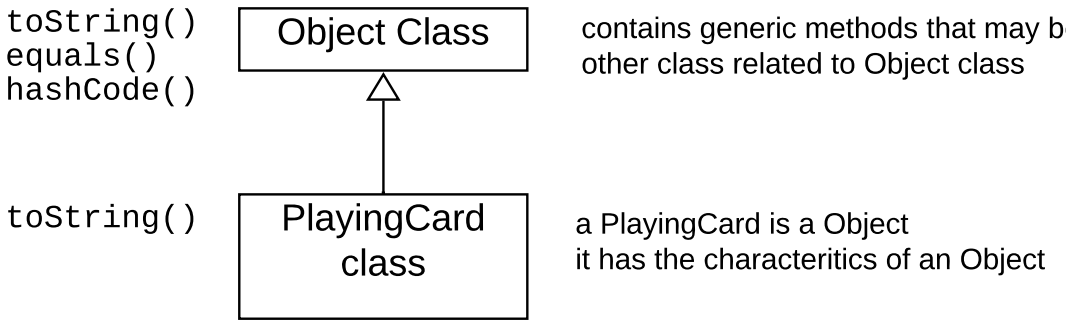
Process Oriented: `add 1 to counter; // process is first, data second`

Principles of Object Oriented Programming (OOP):

- Encapsulation - Protect the data from access outside the class
- Inheritance
- Polymorphmism
- Abstraction*

* Although defined as a principle of OOP when the concept was first presented, some IT professionals do not include Abstraction as a principle of OOP

Every class is a member of the Object class hierarchy:



A class should usually have:

1. One or more constructors (ctor) to allow users to instantiate objects of the class with various initial values.

2. A default ctor is a constructor that takes no parameters; initialze to default values.

3. Getters and Setters to allow users to retrieve or change values in an object.

4. It's OK to limit what the user can Get/Set. Setters should validate data.

5. `toString()` method to return what an object of the class should like as a String. If one is not provided, the Object class toString() is used and displays
`package-name.class-name@location`

6. `equals()` method that compares two objects of the class to decide if they are equal. YOU determine what it means for objects to equal.

7. Any other methods that make class easier for the user to use or provide a common or frequently used process.

Note: Most IDEs, Eclipse, IntelliJ, NetBeans, MS Visual Studio, will generate standard getters, setters, constructors, toString(), equals() et al. Take advantage of this to save you time.