CP213 Lesson 5

* You declare a method to be static when it is used by multiple objects, for example a game board could have static methods to read and write to it
* The main method is an example of a static method
* The purpose of making methods static is so you don’t have to create any objects for example you can use Math.pow(x,y) without creating a math object because the .pow method is static
* A static method has no “this” operator
* A static method cannot refer to an instance variable or involve a nonstatic method of the class
* Static methods cannot call nonstatic methods but they can call other static methods
* A static variable exists only once, all objects of the class can change the static variable
* A static variable should always be defined as private unless it is also defined as constant (final), since constants don’t change it’s ok to make it public
* Difference between static and non-static methods:
  + A static method belongs to the class and you do not need to create an instance of the class to access the static method
  + A non-static method belongs to an **object** of the class, and you have to create an instance of the class to access the non-static method
* When to define a static method:
  + If your method is not using any instance variables (not dependant on instance variable creation)
  + If the definition of the method will never be changed or overridden
* When you make an object = to another object, you aren’t duplicating objects they refer to the same location in memory and are thus the same object
* A method cannot change the value of a variable of a primitive type that is an argument to the method
* A method **can** change the values of the variables of a class type that is an argument to
* The Null keyword:
  + Used to indicate that a variable has “no real value”
  + You can use == and != instead of .equals to test if a class variable has a null value.
* By writing “new Object” you create an anonymous object(i.e. it is not assigned to a variable)
* Use lambda expressions to make your code shorter
* Privacy leaks:
  + When using primitive instance variables, defining them as private is enough
  + This is not the case for class type instance variables
* A statement that is always true for every object of the class is class a **class invariant**
* To duplicate objects, create a copy constructor, make sure to make **new** instance variables instead of referencing the old ones
* Creating proper getter and setters:

public Date getBirthDate() {

return born; // dangerous

}

public Date getBirthDate() {

return new Date(born); // correct

}

* The exception to this is the String class, because it contains no mutator methods
* A class that contains no methods that change any of the data (other than constructors) is called an **immutable class** and objects of such a class are then immutable objects.
* A deep copy of an object is a copy that has no references in common with the original object (with the exception of references to immutable objects)
* A copy that is not a deep copy is called a shallow copy and can cause privacy leaks.

Keywords:

* Anonymous class
* Class invariant
  + A statement that is always true for every object of the class is class a **class invariant.** This is helpful for defining methods that are used in your constructor. For example in the Person class from the lesson the class invariant was that a person’s birth date should come before their death.
* Class reference
* Class type
* Copy constructor
* Deep copy
  + A deep copy of an object is a copy that has no references in common with the original object (with the exception of references to immutable objects). A copy that is not a deep copy is called a shallow copy and can cause privacy leaks.
* Immutable class
* Lambda expression
* Mutable class
* Null
  + Used to indicate that a variable has “no real value”. You can use == and != instead of .equals to test if a class variable has a null value.
* Pass by reference
* Pass by value
* Privacy leak
* Shallow copy
  + Anything that is not a deep copy of an object is a shallow copy. This means most likely that this copy shares references to mutable objects with the original object, which can lead to privacy leaks.
* Static instance variable
* Static method
  + A method is declared static if it needs to be used by multiple objects. Static methods can only refer to other static methods. Define your method as static if it does not change any instance variables.