JP Mayorga 2014057102 4th Summary

Methods are executable code that uses values as arguments. (Compile error: having as member two methods with override equivalent). Methods can return arrays by using empty brackets but shouldn't be used in new code. Formal parameters are separated by commas, each parameter specifier consist of type and the identifier. If a method or constructor doesn't need parameters, only a pair of empty parenthesis appear in the declaration (compile error: two parameters with the same name). Same signature (methods with same name and same arguments), method modifiers refers to the same for variable modifiers. An abstract method introduces the method as a member, with its signature, return type and throws clause, but no implementation and it must be inside an abstract class; one of this can override an abstract method, and a non-abstract method can be overridden by an abstract one.

Method declared static = class method, is always invoked without reference to a particular object. A method is final to prevent subclasses from overriding it or hiding it. A private method inside a final class behaves the same as if it was a final method. A method that is native is usually written in another programming language. A synchronized method requires a monitor before it executes. It is generic if it declares on or more type parameters, defines a set of method for each possible invocation of type parameter. A method can return a type or uses void if it doesn't return anything. Throws class declares a checked exception class. A method body contains the implementation "block of code" or just a semi colon for lack of it. If the method not inherited is declared in a class, or the method not inherited is declared in an interface and the new declaration is abstract, then the new declaration is said to override it. Any subclass can override a method if it isn't private or final (compile error: override with an instance method a static method). If a method is declared static inside a class, it's said that the method will be hidden and can only be accessed by using the keyword super. It is possible for a class to inherit multiple methods with override equivalent signatures. Same methods inside the same class with the same name but signatures, that aren't override equivalent are called overloaded.

A member class and member interface are classes/interfaces declared inside another class declaration. A class inherits all the non-private members directly form superclass/interface. Using the keyword static may change the declaration of a member type that is within the body of a non-inner class. An instance initializer is executed when an instance of the class is going to be created; they can use "this", super or any type of variable. The same for static initializers but this time they are used to create class variables. A constructor is use to create an object for the instance of the class, they are not members, they are not inherited so they cannot be overridden. You can create a generic constructor inside a non-generic class, it is generic if it declares one or more type variables. Its type consists of its signature and the exception types. A return statement is needed if it does not include an expression. If you don't declare a constructor, a default one will be implicit inside the class with no arguments or throws clause. If

you declare the constructor private, it will prevent outside code from creating instances of the class.

An enum type cannot be declared abstract and its always final unless it has one enum constant with a class body. Nested enum types are always static, implicitly. An enum constant defines an instance of an enum type. They must be followed by an argument that will be passed to the constructor. The constructor must be private, if it isn't declared, it will private implicitly and if declared public/protected = compile error.