Progress:

1/11 review 1 - 77

1/12 78 – 139

1/13 None

**Chapter 1 A quick tip**

1. **Source file – Class – Method – Statement**. A source file holds 1 class. A class has 1+ methods. A method is a set of statements
2. Void: no return value.
3. System.out.println VS. System.out.print: print后回车 VS. 不回车
4. String uses “+” to concatenate e.g. s = s + “” + “is a dog”;
5. String.length 用法
6. Math.random() returns a number from 0 to 1. (int) Math.random() \* 5

**Chapter 2 Classes and Objects**

1. OO **inheritance** Abstract to a superclass-subclasses.
2. Subclasses inherit the methods of the superclass.
3. Subclass can **overrides** the methods of the superclass.
4. **Overidding** means that a subclass redefines one of its inherited methods when it needs to change or extend the behavior of that method.
5. Design a class: 1 object knows – **instance variable - state** 2 object does – **methods - behavior**
6. **A class is not an object**, it is a blueprint for an object.
7. Create an object. Two classes in need. One for type of object. The other one to test new class.
8. Two uses of main: 1 to test real class 2 to launch Java application.
9. Objects live in Heap. Memory in Java – **Heap** – Garbage-Collectible Heap.
10. Put all of application files into a Java Archive jar.file.

**Chapter 3 Primitives and References**

1. Variable: primitive hold the value. Primitive: Boolean, char, byte, short, int, long, float, double. **Object reference variable holds** bits that represent **a way** to access an object. All reference variables are the same size.
2. Int 32 bits, float 32 bits, double 64 bits. Float f = 32.5f. No ‘f’, then java thinks it as a double.
3. Cant start a name with a number.
4. Dog myDog = new Dog(); Left: Tells JVM to allocate soace for a reference variable myDog. Right: Tells the JVM to allocate space for a new Dog object on the heap. Link the object and the reference variable.
5. References marked ‘final’ cannot be redirected.
6. An Array variable is a remote control to an array object. Array can hold primitives and references. **An array is an object.**
7. A reference variable has a value of null when it is not referencing any object.

**Chapter 4 Methods use Instance Variables**

1. **A method uses parameters. A caller passes arguments. Inside of Method, its parameter-variable. Outside, its argument.** Just copy pass from caller to method.
2. ArrayList for returning multiple values
3. Accesser-Getter; Mutator-Setter
4. **Encapsulation** puts a force-field around instance variables, so no one can set them inappropriate**.** Mark **instance variables private** and provide **public getters and setters** for access control.
5. Instance variables always get a default value. Int=0, float=0, Boolean=false,reference=null.
6. Instance variable: inside a class but not within a method. **Local variable (no default value)**: within a method. Method parameters are local variables.
7. Use the .equals() method if you want to know if two objects are equal. E.g. String.

**Chapter 5 Writing a program**

1. Flow chart first. For each class, prep code - test code - real code. Prep code= instance variables declarations + methods declaration + method logic.
2. Extreme Programming (XP): Write the test code first.
3. Integer.paresInt(“5”) Covereting a String to an int.
4. Enhanced for: for (int cell : locationCells){} cell holds one element from the array.
5. Math.random(), import java.io.\*; Cast operator: (int); Pre and post: i++, ++i

**Chapter 6 Get to know the Java API**

1. ArrayList – a class in Java library add(object elem); remove(int index); remove(Object elem); contains(Object elem); isEmpty(); indexOf(Object elem); size(); get(int index); ArrayList<Egg> myList = new ArrayList<Egg>();