**Study Guide for COP 2250 Final Exam**

**What is a constructor?**

* It’s like a blueprint for what you will be creating. Default constructor you don’t pass variables to it. Non-default constructor you can pass variables to it and tells the system that you’re using this constructor or the other.

**When defining classes, how do you define the attributes? How about the setters & getters?**

* You define attributes by making variables and use them as parameters for
* Setters: saves information into domain class and are “void” because you don’t return anything
* Getters: gets back that information and assigns it to a variable, if you want an int you set your getter to an “int” so it is *not* void.
* Setters/Getters must be public because you’ll use them for the driver class

**If you need a variable to keep track of the sum of things, what is that variable called?**

* An accumulator.
* We initiate the variable first to 0 and use a loop to accumulate values into variable.

**What is the scope of a variable?**

* The part of the program in which the variable can be accessed
* To make something global we put it on top of the main method and make it “public static”
* Global variables must be static in driver class because ??
* Are there global variables in a domain class? Yes instance variable because they’re accessible to their own class.

**What is encapsulation, and how does it relate to data hiding?**

* Encapsulation is the process of hiding implementation details while publishing an interface. The public methods of a class are the interface.
* It relates to data hiding because it can be seen as a way of achieving data hiding in object-oriented systems.
* When we have a domain class and make those variables private, it’s so we could only access.

**What’s the difference between a class and an object?**

* An object is an entity you can manipulate by calling a method on it (System.out).
* Classes are set of objects with similar behavior (PrintStream class)

**What is a wrapper class? How do you use it, and why?**

* Wrapper class is a class that includes primitive types, such as Integer, Boolean, Float, Long.
* The program automatically coverts primitive type values to a wrapper type object using auto-boxing (that’s the definition of auto-boxing).
* We use wrapper classes because we cannot use primitive types in array lists.
* Unboxing: it is the reverse of auto-boxing, so it is the case where you take a wrapper object and move it to a primitive.
* By the way what is parsing? We use it for JOptionPane because they only use String and to use int you need to parse.
* Parse method is a static helper method that breaks an input line into fields and constructs an object

**Give an example of creating a new instance of a Student class.**

* Student myStudent = new Student();

**What are the different types of loops, and how do they work?**

* While loops, do-while, for loops
* For loops: you use it when you know exactly the number
* While loop – you don’t know how many times
* Do-while: you know the loop will run at least once
* What is something essential to do-while or while loops? Something needs to be in the loop that causes that condition to be false, otherwise you have an infinite loop.
* Example of while loop is when we are reading files because we don’t know how many times the loop will run but we must add a condition

**How should we decide what type of loop to use?**

* We decide what loop we will use based on:
  + If we know how many times the loop will go
  + If it goes at least once
  + For the user’s input

**When you use an if-statement, how should you write the condition?**

* You’re trying to evaluate a condition and it is always Boolean. We compare a set of variables and returns something if it is found true.
* Else if – it doesn’t look at anything else
* Else – if the “if” or “else if” isn’t true then it is the else statement
* If you compare ints in an if you use “==”
* If you compare Strings in an if statement you use “equalsTo”

**How does a switch-case statement work? What is required so the logic won’t fall through?**

* Switch statement: can only use it for discrete statements and certain primitive types.
* The default is when neither of the cases are being called for
* Break statement is used so that all of the cases will not be used. It’s used after each statement so that they can be tested. Used to exit a switch statement.

**When should you use an if-statement vs a switch-case statement?**

* Switch-statement should be used for different cases, and we know numbers, strings, that are set already, but are discrete. They can also be one value per case, while an if-statement is for more values.

**What is a method, and describe the parts of a method’s header.**

* A set of code which is referred to by name and can be called at any point in a program by utilizing the method’s name. The parts of the header:
  + Name
  + Return type
  + List of input parameters enclosed in parentheses

**How do you pass a variable to a method? How do you pass an array to a method?**

**How do you pass an arrayList to a method?**

**What is method overloading?**

* Having two or more methods with the same name, but having different argument lists. (Parameters).

**How are fields in a class initialized?**

**What is an array and how does it work? How do you define an array?**

**How do you change values in an array? How do you initialize an array? (2 ways – what are they?)**

* An array is a static data structure; the size cannot be changed once it is initialized
* To define an array:
  + Datatype[] dataName = new datatype[size];
  + Ex: String[] ericCartman = new String[4];
* To change the value in an array:

**What is an arrayList and how does it work? How do you define an arrayList?**

**How do you add elements to an arrayList? Change? Delete?**

* ArrayList is a dynamic data structure, so items can be added or removed from the list
* To work it, we need to create out ArrayList first, and then add different elements
* To **add** elements, we use “nameList.add(index, “String”)
  + Ex: myChucky.add(2, “Tiffany”);
* To **change** elements, we use, “nameList.set(index, “String”)
  + Ex: myChucky.set(2, “Tiffany”);
* To **delete** elements, we use, “nameList.remove(index)
  + Ex: myChucky.remove(2);

**What is the difference between an array and an arrayList?**

* Array:
  + Has a fixed value when it is initialized
  + Can use primitive types such as “int”
  + The format is also different; it uses blocked brackets instead of the pointed one.
  + We also need forLoops in order to add to an array.
* ArrayList
  + Add to it by using the .add method.
  + Value is flexible, and can continue growing.
  + We can also switch out the place of elements by using the “.set” method.
  + arrayLists cannot use primitive types.

**What is a String? What is concatenation?**

* A string is a sequence of characters. In the Java Programming language, strings are objects.
* Java platform provides the String class to create and manipulate Strings.
* Concatenation is the joining of two Strings
  + We can concatenate using the ‘+’ symbol, or the “.concat()” method

**Coding Practice**

**How many times will the following do-while loop be executed?**

**int x = 11;**

**do**

**{**

**x += 20;**

**} while (x > 100);**

This loop will execute **5** times.

**For the following code, what would be the value of str[2]?**

**String [ ] str = {“abc”, “def”, “ghi”, “jkl”};**

A reference to the element “ghi”.

Why? We start off at position 0, so “ghi” is in position 2. This goes back to how it returns

the reference to, but not the actual object.

**What will be displayed after the following code has been executed?**

**String str1 = “The quick brown fox jumped over the lazy dog.”;**

**String str2 = str1.substring(20,26);**

“jumped\_”  
  
  
**What will be the result of the following code:**

**int[ ] numbers = {40, 3, 5, 7, 8, 12, 10};**

**int value = numbers[0];**

**for (int i=1; i< numbers.length; i++)**

**{ if (numbers[i] < value)**

**value = numbers[i];**

**}**

“3, 5, 7, 8, 12, 10”

**What will be the result of the following code:**

**Int[ ] numbers = {40, 3, 5, 7, 8, 12, 10};**

**int value = 0;**

**for (int i=1; i< numbers.length; i++)**

**{ if (numbers[i] < value)**

**value += numbers[i];**

**}**

“3, 5, 7, 8, 12, 10”

**What does the following code do:**

**String str = “abc456”;**

**char chr = str.charAt(2);**

**If (Character.isLetter(chr))**

**System.out.print(Character.toUpperCase(chr));**

Makes any ‘c’ uppercase