1. What is a method/function?

A method/function is a block of reusable code

2. Write the Java syntax to declare a method

<access\_modifier> [static\_keyword] <return\_type> <name>([parameters]){

<body>

}

3. What does the access modifier do?

The access modifier specifies where in your code a function can be called

public: function can be called from anywhere

package\*: function can be called from anywhere in the same package

protected: function can be called within the same class and subclasses

private: function can only be called within the same class

\*the default access modifier, can only be accessed by not indicating an access modifier

4. What does the return type do?

Indicates the data type that is sent back when the method ends

5. What does it mean to return data?

When a method is called, if the return type is not void, it will send a piece of data back to where it was called from. The original method call will then be equal to the data that was returned. For example, let’s say a method named “getRandomNumber()” returns 4. In this specific case, the line

int num = getRandomNumber();

is equivalent to

int num = 4;

6. How do you call a method?

To call a method, type the method’s name followed by parentheses and pass in the necessary arguments inside. For example, let’s say a method named “multiply()” takes two ints and returns their product. To call the method, we would type

multiply(4, 6);

For a method that takes no arguments, leave the parentheses empty

doSomething();

7. What are some examples of method calls that you have used before?

System.out.println();

input.nextLine();

string1.equals(string2);

8. What is the difference between an argument and a parameter?

A parameter is part of the method header and indicates the data required to be passed into the parentheses from the method call. An argument is the actual value that gets passed into the method call

9. What is a method signature?

The method signature is like a method’s fingerprint, such that each signature must be unique

10. What two parts of the method header define the method signature?

The signature is defined by the method’s name and the parameters

11. What is method overloading?

Method overloading is when two methods have the same name but different parameters and, therefore, different method signatures

12. Write a Java function called “add()” that takes two ints and returns their sum

public static int add(int a, int b){

return a + b;

}

13. Write a Java function called “concatenate()” that takes two Strings, combines them, and returns the new String

public static String concatenate(String a, String b){

return a + b;

}

14. Write a Java function called “printGreeting()” that takes a name and prints out a greeting, such as “Hello, <name>”

public static void printGreeting(String name){

System.out.println(“Hello, “ + name);

}