Quiz #1 Review

All information is stored in the computer in binary.

ANS: T

The computer is a digital device meaning that it stores information in one of two states using binary. We must determine then how to represent meaningful information (such as a name or a program instruction or an image) in binary.

Java is an object-oriented programming language.

ANS: T

Java is classified as a high-level programming language but it is also classified as an object-oriented programming language because it allows the programmer to implement data structures as classes.

System.out.print is used in a program to denote that a documentation comment follows.

ANS: F
Documentation comments follow // marks or are embedded between */ and */. System.out.print is an instruction used to output a message to the screen (the Java console window).

Java byte codes are directly executable whereas Java source code is not.

ANS: F

Neither Java source code nor Java byte codes are executable. Both must be compiled or interpreted into machine code. Java byte codes are useful however in that they are machine-independent but semi-compiled code that allows your Java code to be transmitted over the Internet and executed on another computer even if that other computer is a completely different type.

The Java compiler is able to find all programmer errors.

ANS: F

The Java compiler can find syntax errors but cannot find either logical errors (errors that are caused because of poor logic in writing the program) or run-time errors (errors that arise during the execution of the program).

Java is a case-sensitive language which means Current, CURRENT, and current will all reference the same identifier.

ANS: F

Java is case sensitive which means that Current, CURRENT, and current will all be recognized as different identifiers. This causes problems with careless programmers who do not spell an identifier consistently in terms of upper and lower case characters.

Code placed inside of comments will not be compiled and, therefore, will not execute.

ANS: T

The compiler discards comments; therefore, any code inside a comment is discarded and is not compiled. Your executable program consists only of the code that is compiled.

The word Public is a reserved word.

ANS: F

public is a reserved word, but since Java is case sensitive, Public differs from public and therefore Public is not a reserved word.

Reserved words in Java can be redefined by the programmer to mean something other than their original intentions.

ANS: F

Java reserved words cannot be redefined.

In a Java program, dividing by zero is a syntax error.

ANS: F

Dividing by 0 is not detected at compile time, and because a computer cannot divide by 0, this is a run-time error.

Objects are defined by a class that describes the characteristics common to all instances of the class.

ANS: T

An object is an instance of a class. And, the purpose of a class is to describe these common characteristics.

In Java, identifiers may be of any length up to a limit determined by the compiler.

ANS: F

Java (and Java compilers) do not limit the length of the identifiers you use. Identifiers may be as long as you wish. Good programming practice, however, will limit the lengths of the identifiers you create.

If x is a String, then x = new String("OH"); and x = "OH"; will accomplish the same thing.

ANS: T

In Java, to instantiate (assign a value to) an object, you must use new and the class's constructor. However, since Strings are so common in Java, they can be instantiated in a way similar to assigning primitive types their values. So, both of the above assignment statements will accomplish the same task.

If x is the String "Hi There", then x.toUpperCase().toLowerCase(); will return the original value of x.

ANS: F

x.toUpperCase() returns x as all capital letters, while x.toLowerCase()will return x as all lower case letters. So, this code will first convert x to all upper case letters and then convert the new version to all lower case characters.

If String name = "George W. Bush"; then the instruction name.length(); will return 14.

ANS: T

There are 14 characters between the quote marks including two blanks and a period.

If String a = "ABCD" and String b = "abcd" then a.equals(b); returns false but a.equalsIgnoreCase(b); returns true.

ANS: T

Since "ABCD" is not the same as "abcd", the equals method returns false, but by ignoring case in equalsIgnoreCase, the two are considered to be the same.

Unlike the String class where a method operates on an object (instance) of the class, as in x.length(), the Scanner and Math classes invoke methods directly using the class name, as in Math.abs()or scan.nextInt().

ANS: F

The Math class uses methods known as static methods (or class methods) which are invoked by the class name itself rather by an object of the class. However, with the Scanner class, you need to create an object to invoke the methods of the class.

In order to generate a random number, you must use Math.random().

ANS: F

There is also a Random class available in java.util. This class can generate random int, float, and double values. This is a better mechanism for generating random numbers because you can instantiate several different random number generators.

A double is wider than a float, and a float is wider than an int.

ANS: T

Wider types are larger in size or can store a greater range of values. The double is 64 bits whereas the float is 32 bits and the float, because of the way it is stored, can store a significantly larger range of values than the int.

A variable of type boolean will store either a 0 or a 1.

ANS: F

A boolean variable can store only one of two values, but these values are the reserved words true and false. In C, C++, and C# booleans are implemented as int variables that store only a 0 or a 1, but in Java, the authors of the language opted to use the boolean literals true and false as this is considered to be semantically more understandable (and is much safer).

In Java, 'a' and 'A' are considered to be different values.

ANS: T

Characters values are stored using ASCII, and 'a' and 'A' have different ASCII values as well as different Unicode values.

You cannot cast a String to be a char and you cannot cast a String which stores a number to be an int, float, or double.

ANS: T

There is no mechanism available to cast a String to one of the primitive types, but there are methods available to perform a similar action and return a character at a given location (CharAt) or to return the int, float, or double value equivalent to the number stored in the String.

The values of (double)5/2 and (double)(5/2) are identical.

ANS: F

In the first expression, the (double) cast applies to the int 5, changing it to the double value, 5.0. Then 5.0/2 is calculated, yielding the double value, 2.5. In the second expression the int division is performed first, yielding the value 2. 2 is then changed to a double, yielding the double value, 2.0.

There are three ways that data conversion may occur: by assignment, by promotion, and by casting.

ANS: T

Assignment conversion occurs when a value on the right side of the assignment operator is converted prior to being stored in the variable on the left. Promotion occurs within an expression when values of differing widths are combined. Casting is a programmer's explicit way to control the data conversion process.

You may apply the prefix and postfix increment and decrement operators to instances of the Integer class.

ANS: F

These operators may only be applied to primitive numeric data. An instance of a class is an object and and not a piece of primitive data.

These two ways to set up a String will yield identical results:

```
String myString = "12345";
String myString = 12345;
```

ANS: F

In fact, the second statement won't even compile! The second statement will receive a syntax error about incompatible types. There is no automatic conversion from a number to a String.

These two ways to set up a String will yield identical results:

```
String myString = new String("a bunch of words");
String myString = "a bunch of words";
```

ANS: T

Java has a very strong notion about what Strings are and how they function. The second declaration is merely shorthand for the first. The string reference will be initialized identically by both declarations.

These two ways to set up a String will yield identical results:

```
String myString = new String("123.45");
String myString = "" + 123.45;
```

ANS: T

Java understands the + operator when used to combine Strings with numbers means that the number should be converted into a numeric String, and then concatenation should occur.

These two ways to set up a String will yield identical results:

```
String myString = new String(12345);
String myString = new String("12345");
```

ANS: F

In fact, the first statement won't even compile! There is no overloaded version of the String constructor that accepts a numeric argument.

The following statement will display the value 127.

```
System.out.println("123" + 4);
```

ANS: F

"123" is a String. So, the + will cause String concatenation to occur. The int 4 will be converted into the String "4", and the String "1234" will be created and displayed.

You may use the String.replace() method to remove characters from a String.

ANS: F

The replace() method only replaces single characters with other single characters. The replace() method will not add or delete characters to a String; the String length will remain the same.

If you need to import not only the top-level of a package, but all its secondary levels as well, you should use the following:

```
import package.*.*;
```

ANS: F

The import statement can only be used with a single * (wild card). If you need to import all the secondary levels of a package as well, you must write them out explicitly:

```
import package.A.*;
import package.B.*;
```

The following Java program that will display these three lines when it is run:

```
    *
  * * *
* * * * *
```

```
public class Stars
{
   public static void main(String[] args)
   {
      System.out.println("    *");
      System.out.println("  * * *");
      System.out.print("* * * * *");
   }
}
```

ANS: T

All the methods in the Math class are declared to be static.

ANS: T

The Math class methods are designed to be generally useful in arithmetic expressions, so no instances of anything are required to make use of them. This is achieved by ensuring that all the Math methods are static.

The statements `import java.util.*;` and `import java.util.Random;` both import the Random package?

ANS: T

In the first, all of the classes defined in the java.util library are imported whereas in the latter, only the Random class is imported from the java.util library. The former approach is easier but more time consuming and wasteful of memory, so the latter approach is more appropriate if you only want to use a single class from a given library.

The following statement will create a `Color` object equivalent to `Color.ORANGE` using the `color` method.

```
Color orange = Color.color(255, 165, 0);
```

ANS: F

The color method uses percentages rather than rbg values.

It is often a good idea to group particular elements together in a `scene`?

ANS: T

This makes it much easier to apply transformations such as rotations and position shifts to an entire group at once.

Correct all the syntax errors in the following program.

```
Public Class Program      \\ A problem programZA
(
   Public static voided main[Strings() args]
   {
    system.out.println('This program');   \* oh, my...
*\
    system.out.println('has several syntax errors');\*
                                    lots of errors *\
   }
)
```

ANS:

```
public class Program   // A problem program
{
  public static void main(String[] args)
  {
       System.out.println("This program");   /* oh, my...
*/
     System.out.println("has several syntax errors"); /*
                                          lots of errors */
```

Find all the errors in the following program?

```
public class Enigma
{
  public static void main(String[] args)
  {
     System.out.println("Input a String");
     String x = scan.nextLine();
     int size = x.length;
     char last = x.charAt(size);
     System.out.println("The last character in your string
                                 ", x, " is ", last);
  }
}
```

ANS:

There are a number of syntax errors and one run-time error. First, the `Scanner` class has not been imported so that the `scan.nextString()` will yield a syntax error. Also the can object is never created before it is used. The statement `x.length` requires parentheses as `length` is a method of the class `String`, so this is a method invoked by `x`. Finally, the `System.out.println` statement is not valid because of the use of a comma instead of + to concatenate the various parts of the `String` and does not end one string and create a new one when continuing to the next line. Note that, if the syntax errors were corrected, there would still be a run-time error. This error would arise because `size` will store the length of the `String`, but the last character will be at location `size-1` instead of `size`. Thus, the statement `char last = x.charAt(size);` will result in a run-time error because `size` is beyond the bounds of the `String` `x`.