1.

Match the following:  (2.5)

1.  Assembler

2.  Compiler

3.  Interpreter

4.  Bytecode

5.  Operating System

a. Java's machine language

b. Translates instructions written in high level language to bytecode

c. Translates instructions in mnemonic form (ex: LOAD, ADD) to machine language

d. Translates bytecode instruction to the computer's machine language

e. Monitors the overall activity of the computer.

**1 = c**

**2 = b**

**3 = d**

**4 = a**

**5 = e**

2.

Why do we say Java  is platform independent           (1)

**Java compiles code into bytecode (the .class file) which will run on any computer. It does not need to be rewritten for Linux, Windows, Mac, etc. It does, however, need a JVM. So Java is platform independent, but each platform must have a JVM and the JVM is platform dependent.**

3.

Explain the term garbage collection.(1)

**Garbage collection is automated memory management in Java. Garbage collection is basically the reclaiming of unused memory. As a java program runs, some objects are no longer needed. The unused objects still take up heap memory. Garbage collection “destroys” these unused objects to free up that memory and allow it to be reused.**

4.

What is wrong with the following statements:                         (1.5)  
               int number one = 100  
               Double x = 10.5;  
               string name = "John David"

**Int number one is not a valid variable name (number\_one would work, though it’s still not a good name). That line and the 3rd line are also missing semi-colons at the end.**

5.

Identify the storage in bytes for each of the data types: (2.5)

a. int

b. long

c. short

d. char

e. 40 bits

1. **4**
2. **8**
3. **2**
4. **2**
5. **5**

6.

.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  , \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  and \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ are the 3 different

   categories of primitive data types.(1.5)

**int, long, and float**

7.

Differentiate primitive and reference variable with examples (1)

**Primitive variables store the actual values whereas reference types store a memory location that contains the actual value. So if int x = 3, x will have the value of 3. With something like String name = “Tom”, name will point to a memory location (e.g. 2100) and that memory location will store Tom. If I create a second String coolestPersonEver = “Tom”, that second variable (coolestPersonEver) can also point to 2100.**  
8.                                                                                                      (2)

a.

String s1 = “Hello”;

String s2 = “Hello”;

The address that the variable s1 refers to is the same as the address s2 refers to(True/False)

Explain why?

**True. String is a reference type thus s1 points to a memory location that stores “Hello”. When you create a second string with the same value, Java points to the same memory location.**

b.

String s3 = new String(“Hello”);

String s4 = new String(“Hello”);

The address that the variable s3 refers to is the same as the address s4 refers to(True/False)

**False. By explicitly creating a new string, you’re telling java to store the value in a new memory location.**

9.

When will the expression  !(x>0) become true?  (1)

**Anytime the value stored in x is less than 0 this expression will be true. Put another way, any time x is negative, this expression will be true.**

10.  
  
   if(6<2\*5)

  System.out.print("Hello");

  System.out.print("World");

  If the above code is executed, what would be the output ?    (1)

**HelloWorld. Java will calculate 2\*5 and then evaluate the answer (10) against 6. Since 6 is always less than 10, the print statements will run.**