Fill in the Blanks (20 points – 2 points each). Provide the MOST appropriate answers.

a) The Java compiler will convert the source code in a _______.java_____________ (give extension) file into byte code in a ______.class______________ (give extension) file.

b) The _______ associativity __________ of Java operators determines the order of evaluation when the operators have the same precedence.

c) Give the values for P and Q after the execution of the Java statements below:
   ```java
   int P = 5;
   int Q = P++;
   P ___6____ Q ___5____
   ```

d) Give the value for X after the execution of the Java statements below:
   ```java
   int A = 5;
   double X;
   X = 10.0 - A / 2;
   X ___8.0____
   ```

e) The next() method in the Java Scanner class will read a String up until an occurrence of the _______ delimiter ____________, which by default is "white space".

f) The Java code segment:
   ```java
   int X;
   X = 10.0/2;
   ```
   will generate the following error: _____Type mismatch – possibly lossy conversion_____

g) List the Java relational operators _____ ==, !=, <, >, <=, >= ______.

h) (Choose the correct answer) Using a Java keyword as a variable name is an example of a [compilation error, run-time error, logic error] _____compilation error_____.

i) (Choose the correct answer) The "dangling else" in Java is an example of a [compilation error, run-time error, logic error] _____logic error_____.

j) Consider the Java code segment below:
   ```java
   StringBuilder S1, S2, S3;
   S1 = new StringBuilder("Hello");
   S2 = S1;
   How many StringBuilder objects exist after execution of this segment? _____1_______
True / False (14 points – 2 points each). Explain false statements for full credit.

a) A Java .class file that was generated on a Windows machine cannot be run on a Mac.  
   False – any system that has a JRE can run the program.

b) A programmer could redefine the String class in Java to be something different if he/she wanted to do so.  
   True

c) The Java code segment below will output "Yes":
   ```java
double X = 3.33333;
double Y = 10.0/3.0;
if (X == Y)
    System.out.println("Yes");
```
   False – the precision of the numbers is different and they are thus not equal.

d) The Java code segment below will output "1 2 3 4 5"
   ```java
for (int X = 1; X < 5; X++)
    System.out.print(X + " ");
System.out.println();
```
   False – The output only goes to 4

e) A Java do while loop will execute the loop body 1 or more times.  
   True

f) The Java code segment below will output "OneTwoThree":
   ```java
int test = 1;
switch (test)
{
    case 1: System.out.print("One");
    case 2: System.out.print("Two");
    case 3: System.out.print("Three");
}
System.out.println();
```
   True

g) The Java code segment below will output "Yes".
   ```java
String S1 = new String("Hello");
String S2 = new String("Hello");
if (S1 == S2)
    System.out.println("Yes");
```
   False – the objects have the same data but the references are not the same.
Tracing (16 points)  Give all output produced by the execution of the Java program below. Be careful to note the print() vs. println() statements. Use the bottom of the page for your output and clearly mark it by drawing a box around it.  If you need extra space use the back of this page.

```java
public class Trace1A {
    public static void wacky(int one, int two) {
        for (int i = one; i < two; i++) {
            for (int j = i; j >= 0; j--)
                System.out.println("i = " + i + " and j = " + j);
        }
        int temp = one;
        one = two;
        two = temp;
        System.out.println("one is " + one + " and two is " + two);
    }
    public static void logical(int p, int q, int r) {
        boolean b1 = (p <= q) && (q <= r);
        boolean b2 = !((p <= q) && (q <= r));
        boolean b3 = (p > q) || (q > r);
        System.out.print("b1 is " + b1 + " and b2 is " + b2);
        System.out.println(" and b3 is " + b3);
    }
    public static void main(String [] args) {
        System.out.println("Starting trace");
        int p = 3, q = 5, r = 7;
        logical(p, q, r);
        p = 7; q = 5; r = 3;
        logical(p, q, r);
        int low = 3, high = 5;
        wacky(low, high);
        System.out.println("low is " + low + " and high is " + high);
    }
}
```

Output:
Starting trace
b1 is true and b2 is false and b3 is false
b1 is false and b2 is true and b3 is true
i = 3 and j = 3
i = 3 and j = 2
i = 3 and j = 1
i = 3 and j = 0
i = 4 and j = 4
i = 4 and j = 3
i = 4 and j = 2
i = 4 and j = 1
i = 4 and j = 0
one is 5 and two is 3
low is 3 and high is 5