Computer Science I – Prepa Tec Campus Eugenio Garza Lagüera
Activity 7: Java Operators Exercise!

Section 1: Java arithmetic operators

What is the output to the following programs? Consider that each block of code is independent to each other.

`int quotient = 7 / 3;`
`int remainder = 7 % 3;`
`System.out.println("quotient = " + quotient);`
`System.out.println("remainder = " + remainder);`	

`double result = (1 / 2) * 2;`
`System.out.println("(1 / 2) * 2 equals " +`
`result);`	

`int result = 3 * 7 % 3 - 4 - 6;` `System.out.println("result is " + result);`	

Section 2: Special assignment operator

You can precede the simple assignment operator (=) with an arithmetic operator, such as +, to produce a kind of special-purpose assignment operator. For example, the following code will increase the value of the variable amount by 5.

```
amount += 5;
```

This statement is just a shorthand for:

```
amount = amount + 5;
```

What is the output to the following programs? Consider that each block of code is independent to each other.

`int result = 11;`
`result /= 2;`
`System.out.println("result #1 " + result);`	

`double result = 10.5;`
`result += 13.2;`
`result *= 2+3/2;`
`result = -result;`
`System.out.println("result #2 " + result);`	

`long x = 10;`
`long y = 10;`
`x *= y+3;`
`y /= 2;`
`System.out.println("result #3 " + ( x + y ));`
`System.out.println("result #3 " + x + y);`	

Section 3: Increment and Decrement Operators

Java has two special operators that are used to increase or decrease the value of a variable by 1.

The **increment operator** is written as two plus signs (++). For example, the following will increase the value of the variable `count` by 1:

```
count++;
```

This is a Java statement. If the variable `count` has the value 5 before this statement is executed, it will have the value 6 after the statement is executed. The statement is equivalent to:

```
count = count + 1;
```

The **decrement operator** is similar, except that it subtracts 1 rather than adds 1 to the value of the variable. The decrement operator is written as two minus signs (`--`). For example, the following will decrease the value of the variable `count` by 1:

```
count--;
```

If the variable `count` has the value 5 before this statement is executed, it will have the value 4 after the statement is executed. The statement is equivalent to

```
count = count -1;
```

You can use the increment operator and the decrement operator with variables of any numeric type, but they are used most often with variables of integer types, such as the type `int`.

In expressions, you can place the ++ or `--` either before or after the variable, but the meaning is different, depending on where the operator is placed. For example, consider the code

Code	Explanation

`int n = 3;`
`int m = 4;`
`int result = n * (++m);`
	After this code is executed, the value of n is 3, the value of `m` is 5, and the value of `result` is 15.
	
	Thus, `++m` both changes the value of m and returns that changed value to be used in the arithmetic expression.

`int n = 3;`
`int m = 4;`
`int result = n * (m++);`
	After the code is executed, the value of n is 3 and the value of `m` is 5, but the value of `result` is 12.

The two expressions `n*(++m)` and `n*(m++)` both increase the value of `m` by 1, but the first expression increases the value of m *before* it does the multiplication, whereas the second expression increases the value of `m` *after* it does the multiplication. Both `++m` and `m++` have the same effect on the final value of m, but when you use them as part of an arithmetic expression, they give a different value to the expression.

What is the ouptut of the following code?

`int n = 2;`
`n++;`
`System.out.println("n is " + n);`
`n--;`
`System.out.println("n is " + n);`	

`int u = 2;`
`int v = 3;`
`int x = 5;`
`int w = 7;`
`int result = u + v * w + x;`
`System.out.println(result);`	

`int u = 2;`
`int v = 3;`
`int x = 5;`
`int w = 7;`
`int y = 11;`
`int result = u + y % (--v) * w + x;`
`System.out.println(result);`	

`double u = 4;`
`int v = 2;`
`int w = 7;`
`double result = u++ / v + u++ * w;`
`System.out.println(result);`	