CS1110 Lab 01. Expressions, variables, declarations, assignments Spring 2011

Name _____ Cornell net id _____

This introductory lab deals with Java expressions and assignment statements, for the most part treating Java just as a calculator.

Below is a list of expressions, some followed by questions. Cut and paste each expression from the online pdf or html version of this handout (available from the course webpage, http://www.cs.cornell.edu/courses/cs1110/2011sp) into the Interactions pane in DrJava, hit the enter key to have the expression evaluated, record its value on this sheet (next to the expression), and answer any questions. Do not simply write down what you think is the value of an expression; write down only what DrJava says it is. When a row has two expressions/questions, do the left-column one first, and then the right-column one.

When finished, show this sheet to your lab instructor, who will record that you did it. If you do not finish during the lab, finish it within the next few days and show it to your lab instructor next week. This paper is yours to keep.

You may find it convenient to enlarge the Interactions pane: put the mouse on the horizontal bar running across the entire window, hold down the mouse button and drag the bar upwards. Also, you can use the up-arrow key to obtain a previous expression; then you can edit the expression and hit the return key to have the modified expression evaluated.

Don't waste time! If you don't understand something, ask your lab instructor or a consultant immediately! You should understand HOW each expression is evaluated, so if an answer doesn't make sense, ask someone. The lab instructors and consultants are in the lab to help. They will look over your shoulder from time to time and give you advice.

INT EXPRESSIONS	5 + 2
5 + 2 * 5	(5 + 2) * 5
4 - 3 - 3	4 - (3 - 3)
-4 - -4 - -4	
6 / 2	
6 / 4	Why isn't 6/4 = 1.5?
7 % 2	8 % 3
6 % 3	What is the name of operator %?
Integer.MIN_VALUE	Integer.MAX_VALUE
Integer.MIN_VALUE + 1	Integer.MAX_VALUE - 1
Integer.MIN_VALUE - 1	Why does Integer.MIN_VALUE - 1 have such a funny value?
Integer.MAX_VALUE + 1	

DOUBLE EXPRESSIONS	
5.0 + 2.0	(5 + 2.1) * 5
4.0 - 3 - 3	4.0 - (3 - 3)
6.0 / 2	6.0 / 4
6.0 % 3	-6.0 % 4
Double.MIN_VALUE	Why isn't Double.MIN_VALUE negative?
Double.MIN_VALUE + 1	Double.MAX_VALUE
Double.MAX_VALUE + 1	Double.MAX_VALUE + Double.MAX_VALUE

We want you to figure out why an addition like Double.MIN_VALUE + 1 equals 1. Here is the explanation:

In Java, the mantissa in "mantissa E exponent" can only hold a certain maximum number of digits. Let's see how this causes Double.MIN_VALUE + 1 to give 1:

Imagine a Java world where the mantissa can have only 3 digits. Both 3.24E5 and 32.4E5 can be stored in a double in this imaginary java world, but not 32.46E5 because it contains 4 digits in the mantissa.

Using this restriction in this imaginary Java world, add up the two numbers 1.0E0 and 3.20E-2. 1 + 0.032 = 1.032 = 1.032E0, right? But in our imaginary Java world where the mantissa can only hold 3 digits, the 2 would be dropped and imaginary Java will return 1.03E0. In essence, the least significant digits are simply removed.

Now, using the same restriction, YOU add up the two numbers 1.00E0 and 1.00E-4 —remembering that at all times, each mantissa is only 3 digits. What would imaginary Java return? This example should show you why Double.MIN_VALUE + 1 equals 1.

CASTING	
(**double**) 4	(**int**) 4
(**double**) 7 / 4	(**double**) (7 / 4)
Which operator has higher precedence, casting or division? (deducible from the row above)	
(**int**) 5.3	(**double**) (**int**) 5.3
(**int**) -5.3	
7 / 4 + 5	(**double**) 7 / 4 + 5
7 / (**double**) 4 + 5	(**double**) (7 / 4 + 5)

BOOLEAN EXPRESSIONS	
3 < 5	3 < 5 && 5 < 3
true	**true** && **false**
true && **true**	What is the name of operator && ?
false	**true** || **false**
true || **true**	What is the name of operator || ?
!**true**	What is the name of operator ! ?
!**false**	!!**false**
true && **false** && **true**	**true** || **false** || **true**
true || (**false** && **true**)	**true** && (**true** || **false**)
false&& (5 / 0 == 1)	(5 / 0 == 1) && **false**
Why does the lefthand expression above work whereas the righthand one doesn't?	

STRING EXPRESSIONS	"Truth " + "is " + "best"
"Truth " + ("is " + "best")	56 + "" + 56
"" + 4 / 2	("" + 4) / 2 gives an error message. Why?
4 + 2 + ""	4 + (2 + "")
What does + do if at least one operand is a String?	

VARIABLES, DECLARATIONS, ASSIGNMENTS

It is important that you learn the difference between a declaration and an assignment statement.

In Java, variables must be declared before they are used. In the newest version of DrJava, in the interactions pane, variables must be declared before they are used. You can change this to allow uses of variables without declarations. Use menu item Edit->Preferences; in the window that opens, click on "Interactions pane" in the left column and then uncheck the appropriate box.

To the right, write the purpose of a declaration like "**int** j;".

To the right, explain how the assignment statement "j= j + 5;" is executed.

int j;	(There will be no answer from the declaration to the left)
j	Does a newly declared variable have a value?
j= 2;	(To the left is your first assignment statement)
j	j+4
j= j + 9;	
j	(You can see what assigning to j did)
int k= 5;	j + k
double w= j + k;	w
w;	(if you follow an expression with a semicolon, you don't see its value)

FUNCTION CALLS	
Math.min(25, 4)	(Note: In the function call on the left, the two constants 25 and 4 are called the **arguments** of the call.)
Math.max(25, 4)	Math.min(25, Math.max(27, 4))
Math.abs(25)	Math.abs(- 25)
Math.ceil(25.6)	Math.floor(25.6)
Math.ceil(- 25.6)	Math.floor(- 25.6)