CS 1110, LAB 1: JAVA EXPRESSIONS

Name: _____ **Net-ID**: _____

There is an online version of these instructions at

http://www.cs.cornell.edu/courses/cs1110/2012sp/labs/lab01.php

You may wish to use that version of the instructions.

This lab serves two purposes. First, it is designed to get you started with DrJava immediately. Second, gives you hands on experience with Java expressions, which we talked about on the first day of class. With that said, you are not expected to be be writing any programs for this lab; for the most part you will just be using DrJava as a glorified calculator.

1. Getting Started with DrJava

To open DrJava in a ACCEL-lab computer, find the following folder:

Start ⇒ All programs ⇒ Class Files ⇒ DrJava

Inside the DrJava folder will be a .jar file; double click on this file to start DrJava.

If your primary computer is a laptop, bring it to the lab as it is an *excellent* opportunity to get started with DrJava on your machine. Follow the instructions on the DrJava page. Feel free to ask a consultant for help if you run into problems; that is why they are there.

2. Lab Instructions

Below is a list of expressions. For each expression, you should first **compute the expression in your head, without DrJava**. Write down what you think the value is in the second column of the table. If you have no idea, write "?". Once you have done that, use DrJava to compute the same expression. Simply cut and past the expression from the online version of these instructions into the Interactions pane, and hit the `ENTER` key. You should write the result in the third column.

If the two values are different, you should try to figure out why DrJava gave the answer that it did. Come up with a reasonable explanation and put it in the final column. You are not really being graded on these labs (see Turning In the Lab (cf. Section **??**)), so you should make your best guess at what is happening; your answer will help the instructor and consultants understand how to better aid you.

As part of this lab, you may find it convenient to enlarge the Interactions pane. To do this, 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.

As this is the first lab, it is a little shorter than some of the future labs. With that said, do not waste time! If you don't understand something, ask your lab instructor or a consultant immediately. It is very important that you 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.

Turning in the Lab. Labs are graded on effort, not correctness. When you are finished, you should show your written answers to this lab to your lab instructor, who will record that you did it. You do not actually need to turn in this piece of paper; it is yours to keep. If you do not finish during the lab, finish it within the next few days and show it to your lab instructor next week.

Because the lab is graded on effort, you should focus all of your efforts on ***understanding*** and not just getting the answers correct. The primary purpose of thes labs is for you to have a hands-on session during which the consultants and instructors can help you out.

---

3. Lab Problems

3.1. **int Expressions.**

Expression	**Expected Value**	**Calculated Value**	**Reason for Calculated Value**
5 + 2 * 5			
(5 + 2) * 5			
4 - 3 - 3			
4 - (3 - 3)			
-4 - -4 - -4			
6 / 2			
6 / 4			
7 % 2			
6 % 3			
Integer.MIN_VALUE			
Integer.MIN_VALUE + 1			
Integer.MIN_VALUE - 1			
Integer.MAX_VALUE + 1			

3.2. **double Expressions.**

Expression	Expected Value	Calculated Value	Reason for Calculated Value
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			
Double.MIN_VALUE + 1			
Double.MAX_VALUE			
Double.MAX_VALUE + 1			
Double.MAX_VALUE + Double.MAX_VALUE			

It is important that you understand 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. As a simple example, suppose that the mantissa in Java can have only 3 digits (it can actually have more, but the argument does not really change). Both 3.24E5 and 32.4E5 can be stored in a double in this hypothetical version of Java, but not 32.46E5; the latter number contains 4 digits in the mantissa.

Using this restriction in this hypothetical version of Java, add up the two numbers 1.0E0 and 3.20E-2. 1 + 0.032 = 1.032 = 1.032E0, right? But in our imaginary version of Java, where the mantissa can only hold 3 digits, the 2 would be dropped and 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. What do you (as a non-Java processing human) get for the answer? Now, what would the hypothetical Java — which is limited to a mantissa with only 3 digits — return for the answer? This example should show you why Double.MIN_VALUE + 1 equals 1.

3.3. **Casting Expressions.**

Expression	Expected Value	Calculated Value	Reason for Calculated Value
(**double**) 4			
(**int**) 4			
(**double**) 7 / 4			
(**double**) (7 / 4)			

Given the results in the table above, answer the following question: which operator has higher precedence, casting or division? Explain your answer:

Expression	Expected Value	Calculated Value	Reason for Calculated Value
(**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)			
7 / 4.0 + 5			

3.4. **boolean Expressions.**

Expression	Expected Value	Calculated Value	Reason for Calculated Value
$3 < 5$			
$3 < 5$ && $5 < 3$			
true			
true && false			
true && true			
false			
true || **false**			
true || **true**			
!true			
!false			
!!false			
!false && true			
true && false && true			
true || **false** || **true**			
true || (**false && true**)			
true || **false && true**			
true && (true || **false**)			
true && true || **false**			

Expression	Expected Value	Calculated Value	Reason for Calculated Value
false `&&` (5 / 0 == 1)			
(5 / 0 == 1) `&&` **false**			

For the two expressions in the table above, answer the following question: why does the last expression in the table not work but the one above it does?

3.5. **String Expressions.**

Expression	Expected Value	Calculated Value	Reason for Calculated Value
"Truth " + "is " + "best"			
"Truth" + "is" + "best"			
"Truth " + ("is " + "best")			
56 + "" + 56			
"" + 4 / 2			
("" + 4) / 2			
4 + 2 + ""			
4 + (2 + "")			
"" + 4 + 2			

Given the results in the table above, answer the following question: what does + do if at least one operand is a String?

3.6. **Function Calls.**

In the first function call below, 25 and 4 are called the **arguments** of the call. In the third call below, the arguments are 25 and Math.max(27, 4)

Expression	**Expected Value**	**Calculated Value**	**Reason for Calculated Value**
Math.min(25, 4)			
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)			