Introduction to Java Programming

Day 1: Homework

Recap

Before starting your homework, take some time to review the following things we covered in class today:

- Creating a basic program in Java that prints "Hello World" to the Console
- Declaring and manipulating (including casting) different Java variable types: `String`, `int`, `float`, `char`, `boolean`, `long`, `double`
- Using arithmetic operators: `+`, `-`, `*`, `/`, `%`, `++`, `--`
- Using assignment operators: `=`, `+=`, `-=`, `*=`, `/=`
- Using comparison operators: `==`, `!=`, `>`, `<`, `>=`, `<=`, `.equals`, error tolerance for `float`/`double` comparison
- Using logical operators: `&&`, `||`, `!`
- Using conditionals and nested conditionals: `if`, `if/else`, `if/elseif/else`, `switch`
- Using a `while` loop, including `break`
- Commenting our code using `//`
- Order of operations for Java operators; the usage of brackets `()` for code clarity

Installing JDK and Eclipse

As you know, we're starting our Mini Project this Wednesday. To do this Mini Project, you will need to have a JDK (Java Development Kit) and an IDE (Interactive Development Environment) such as Eclipse installed on your personal computer. Follow the instructions in the link below to install both a JDK and Eclipse on your computer: http://www.ntu.edu.sg/home/ehchua/programming/howto/eclipsejava_howto.html

Homework

In class, we got started on building a mini calculator in Java using an online Java IDE/compiler (link: https://repl.it/languages/java). For homework, you'll add the following features to this calculator:

1. Recall that our program crashed when the user tries to divide by 0 in the Console. Fix this so that when the user does this, the program doesn't do anything but doesn't crash (so 14 / 0 returns 14). Be careful, since 0 is parsed as a `double` in your program, you should use an *error tolerance* to compare it to 0.
2. `+`, `-`, `*`, `/`, `%`, `++`, `--` now have an English equivalent, so you should accept the `String` "plus" and treat it the same way as the `String` "+", and so on. This means your calculator should output 11 when it's given 5 `plus` 6, for example. Note: `-` is `minus`, `*` is `times`, `/` is `divide`, `%` is `mod`, `++` is `increment`, `--` is `decrement`.
3. Currently, we store the result of our calculations as a `double`. To make the displayed result look cleaner, change your code so that while it still stores the result as a `double`, it now *displays* it as being rounded to the nearest integer. So while the result might be 3.14159, you should display 3 to the user instead. You must program this rounding function yourself (ie. do **not** use `Math.round`, even if you know what that is).
4. Use a `while` loop to implement the *exponentiation* operation `exp` (eg. 3 `exp` 2 returns 9). You will need to create a new variable to keep track of the current iteration count inside the `while` loop. You can assume the exponent is an integer, and you should explicitly deal with the case where the exponent is 0, because anything raised to the power of 0 is 1, *except* for 0, in which case 0 `exp` 0 should return 0.
5. Using comments to explain how you implemented feature 4 above.