Objectives of this lab:

• Practice if, else-if, and switch statements
• Learn about command line arguments
• Practice datatype conversions

Exercise Preparation:

• Start a terminal application and prepare your lab4 directory:
  ◦ mkdir ~/cs170/lab4
  ◦ cp ~cs170005/share/lab4/* ~/cs170/lab4
  ◦ cd ~/cs170/lab4
  ◦ ls
  ◦ You should see 2 files: Arithmetic1.java and Arithmetic2.java. If you do not see these files, ask the TA for help.

Command Line Arguments:

• When you run a Java program using additional parameters, the additional parameters are called “command line arguments.”
• Example:
  ◦ java MyProgram x y z
  ◦ x y and z are command line arguments
• We can use these command line arguments to control or change the purpose of our programs.
• The values of the command line arguments are passed (ie given) to a Java program through the parameter to the main method.
• Remember that the main method is required of all Java programs:
  ```java
  public static void main( String[] args ) {
    ....
  }
  ```
  ◦ args is the parameter to the main method.
  ◦ This variable, args, contains the values of the command line arguments in this manner:
    ▪ args[0] contains the value of the 1st command line argument
    ▪ args[1] contains the value of the 2nd command line argument
    ▪ args[2] contains the value of the 3rd command line argument
    ▪ and so forth
  ◦ Regardless of how it is typed on the command line, the arguments are of type String.
    ▪ Example: java MyProgram 3 4
    ▪ args[0] will contain the String value “3”.
    ▪ args[1] will contain the String value “4”.
  ◦ If you want to use the command line arguments to do calculations, you will need to convert them to a numerical datatype.
- Recall that you can use the `Integer.parseInt(String value)` and `Double.parseDouble(String value)` to convert Strings to integers or doubles.

The programs `Arithmetic1.java` and `Arithmetic2.java`
- Both programs take 3 command line arguments:
  - two integers followed by:
  - one of three arithmetic operators: `+`, `-`, or `/`
    - NOTE: we are only supporting these 3 operators. There's a reason we aren't supporting `*`, but don't worry about it.
- Both programs print the result of an arithmetic expression
- Sample correct output when the program is executed: (The program `Arithmetic2` would produce the same results.)

```
>> java Arithmetic1 1 2 +
1+2=3

>> java Arithmetic1 3 2 -
3-2=1

>> java Arithmetic1 3 2 @
3@2=Error: invalid operator!
```

**Task 1: Complete `Arithmetic1.java`**
- Enter the following commands in a Terminal:
  - `cd ~/cs170/lab4`
  - `gedit Arithmetic1.java &`
- You should see the “skeleton” of a (incomplete) program in your editor. If you see a blank window, ask your TA for help.
- The comments in the file will guide you in what to do.
- Remember to make small changes, compile, and check your work. Don't try to write the entire program first, before you test it out.
  - You may need to comment out portions of code to achieve a compile.
  - After you have compiled your program, you can execute it (from your `~/cs170/lab4` directory) by typing a command in the form of
    - `java Arithmetic1 int1 int2 operator`
    - Specific example: `java Arithmetic1 2 4 /`
- Tasks:
  - Add a statement to obtain the second operand. The first operand has been done for you.
  - Complete the `if, else if, else` statement that performs the selected operation based on the operator entered via the command line. The `-` operator has been done for you.
    - Add the clauses to handle `+` and `/`
    - The last `else` clause should contain the code to print out “Error! Invalid operator” if the user has entered something other than `+,-, or /`.
Task 2: Complete `Arithmetic2.java`

- Open `Arithmetic2.java` for editing in `gedit`.
- Again, you should see the “skeleton” of an (incomplete) program. If you do not, ask your TA for help.
- The comments in the file will guide you in what to do. Several things will be very similar to `Arithmetic1.java`.
- Tasks:
  - Add a statement to obtain the second operand. The first operand has been done for you.
  - Complete the switch statement that performs the selected operation and prints the result. The `-` operator has been done for you.
    - Add the cases to handle `+` and `/`
    - The `default` case should handle the case when the operator is not one of the 3 supported operators (ie `+`, `-`, or `/`) and print out “Error. Invalid operator.”

Turning in your work:

- Save all your work and close `gedit`.
- Turn in your lab 4 work using these commands:
  ```
  cd ~/cs170/lab4
  /home/cs170xxx/turnin Arithmetic1.java lab4a
  /home/cs170xxx/turnin Arithmetic2.java lab4b
  ```
  - Note: you will need to replace `xxx` with your section number.