MODULE 2: Hello.java

1. install java
2. go to module_one directory
3. create a file Hello.java using vim
4. list all of the files in module_one using ls
5. compile Hello.java by typing **javac Hello.java**
6. list all of the files in module_one using ls.
    6.1. Hello.java is the java code.
    6.2. Hello.class is the ***java byte code*** - which is the code a computer understands.
    6.3. Open Hello.class with the command ***vim Hello.class***. What do you see?
7. run Hello by typing **java Hello**

COMMAND LINE ARGUMENTS:

Let's edit the file to include an input parameter or a ***command line argument***.
String args[] is a list of words (or character sequences separated by spaces) that you can input to the program. args[0] holds the first word you give the file. For example, the command **java Hello world**, has ***world*** stored in args[0].

1. change System.out.println("hello world"); to System.out.println(“hello ” + args[0]);
2. recompile Hello.java with the command: **javac Hello.java**
3. run Hello world with the command: ***java Hello name***

Note that args[0] holds the first word, args[1] holds the second word, args[2], holds the third word.

CHALLENGE ONE:
In the command: **java Hello sunny world**, where is *sunny* stored and where is *world stored*?

CHALLENGE TWO:
what does *args[48]* hold?

MULTIPLE ARGUMENTS:

We can find out how many parameters we have by printing "args.length" with the command: **System.out.println(args.length)**

CHALLENGE THREE:
Let's expand our *Hello* program.

1. List the current files in your directory with ***ls***
2. Use vim to open the file. Should you use vim to open Hello.class or Hello.java? If you don't remember, try opening both with vim and see which one doesn't look like a garbled mess

3. Edit the correct file to take in the input name and age and then print:
4. ***hello, name. you are age years old***
5. Then, compile and run your program using ***javac Hello.java*** and ***java Hello***