Computing Logic

There are two kinds of programing languages. Interpretive Language and Compiled language.

Interpretive- executes the code in order 1st to last. It takes the code, runs it through an interpreter, and then is able to read and execute. Because it must run through an interpreter, it is slower than compiled and should not be used to program large projects.

Compiled- All parts of this runs all at once or not in any specific order. Much faster and can handle more.

Because we are nobs we will be learning Python, which is an interpretive language.

Different Development Styles

Waterfall- begins by starting to develop ideas, goes to the actual developmental stage, testing happens, and then you have the product. This style is very rigid because as soon as a problem occurs you must restart the cycle. It is expensive and time consuming.

Rapid- The idea is to create a minimally viable program, just the bare minimum. Once you have that base program. Then you can go back and add meat to the bones. Once you've done this a few times you can add the sexy shit to finish your completed program. (You do however have to be skilled to do it well)

Agile - doing one thing at a time. Get done with one thing? Time to add more shit to it. You must keep a rigid structure to be able to do this properly. A sub category is Scrum. You plan out sprints where you only have one larger focus. From there it is broken into daily plans to accomplish said goal. From there you reassess and restart the process.

Python we're learning on PyCharm

```
name="Madi"
print("Hello "+name+".")
name="Olivia"
print("Hello "+name+".")
name=input("What is your name?\n")
print("Hello "+name+".")
```

= name is a variable. Variables can be set equal to values. The value in this line is "Madi"

↳ "Madi" is a string. Strings are show in parenthesis. Strings are characters within the paranthesis.

= print is the function. Print is displayed as print(). Whatever is placed within the parenthesis is displayed onto the terminal . The string "Hello " is concatenated by the + to the other strings and variables.

= the variable name now has the value of Olivia. Because Python is an interpretive language, the code is ran in order. Up until now name's value was Madi but from now on it's Olivia.

= we are changing the value of name again. Name now equals what the user inputs into the running box after the string "What is your name?" is displayed. The final part, \n formats this so what the user inputs is displayed on a new line.

displays

```
print(7)
print(7.0)
print(int(7.0))
print(int(7.6))
```

```
7
7.0
7
7
```

Numbers in computer language are placed into two categories: Integers and Floating Point Numbers (floats).

Integers – whole numbers

Floating Point Numbers (floats) – fractions and decimals (same thing)

Other programs make using said numbers but Python keeps track of which kind of number is being used.

= the function print( ) is printing 7. Python immediately establishes this as an integer and displays it.

= Python immediately establishes 7.0 as a float and displays according. 7.0 is a float because of the .0 .

= int labels what follows in the ( ) as an integer whether it was to begin with or not. Therefore 7.0 is now displayed as the integer 7

= the same as the line before, 7.6 is converted to an integer and is displayed as 7. Integer is a floor operator. It always rounds down to the base integer shown.

```
num1=5
num2=12
num3=num1+num2
print("my number is "+str(num3))
```

= num1 is the variable which is set equal to the value 5. It's important to note: you can have numbers in variables, but you can not start the variable with a number because it confusing the system.

= same shit bro.

= the variable num3 is equal to the values of num1 and num2

= prints to the terminal the string "my number is" and then converts the variable which is equal to an integer to a string that can be printed along with the other string.

Extra Notes (some important shit that I wasn't sure where to fit in)

- When executing a program, it essentially follows order of operations, the innermost paranthesis goes first and the outermost paranthesis goes last.

- Conversely to floor functions, ceil functions round up. So if you have 6.4 it would immediately round up to 7.

- You can't have integers and strings within the same line so you must convert the integers to strings.

- Just as you can't start a variable with a number, you can't start a variable with an underscore (it becomes something else)

- Snake case is using an underscore in between the words in the variable.
  example: snake_case = good

- It doesn't matter whether you use single or double quotes you just have to stick with it to not confuse the system.

- Use ctrl s to save, just makes sure its saved

- add +, subtract −, multiply *, divide /

- Under the project area, click on file, then Git, then add

- Click green check mark in top left, add comment in black, above click files you want to commit, then hit push, and push.

- Escape Sequence – backslash then n, \n. Makes it look pretty and adds a new line for response.

- Scripting – the process of programming, mainly used in Python. Creating on a file to automate mundane tasks.

- Product Life Cycles – process of taking, developing, and maintaining a product (though it is applicable to the creation of most things) Keep in mind this process does get broken down further.

- Remember to Ask about Modules aka libraries. Was exhausted and didn't understand :;