Week 1:

Watch PyCharm Install Jupyter Notebook, YouTube, 2017

* Watch the first four videos found at this link: [PyCharm: Getting Started](https://www.jetbrains.com/pycharm/documentation/), JetBrains.
* Review the Week 1 Slides.
* Complete the following:
  + 1.1 Assignment: PyCharm (20 points)
  + 1.2 Assignment: GIT (20 points)
  + 1.3 Discussion Board: Interest (10 points)
  + 1.4 Discussion Board: Understanding (10 points)
  + 1.5 Discussion Board: Introduce Yourself (6 points)

Week 2:

* Read:
  + Chapter 2 in *Python for Everybody.*
  + [PEP 8--The Style Guide for Python Code](https://pep8.org), PEP 8
* Complete the following:
  + 2.1 Programming Assignment (40 points)
  + 2.2 Discussion: Interest (10 points)
  + 2.2 Discussion Understanding (10 points)

Assignment:

**Create a program with the following requirements:**

* Using comments, create a header at the top of the program indicating the purpose of the program, assignment number, and your name. Use the [SIUE Style Guide](https://www.cs.siue.edu/programming-style-guide) as a reference.
* Display a welcome message for your user.
* Retrieve the company name from the user.
* Retrieve the number of feet of fiber optic cable to be installed from the user.
* Calculate the installation cost of fiber optic cable by multiplying the total cost as the number of feet times $0.87.
* Print a receipt for the user including the company name, number of feet of fiber to be installed, the calculated cost, and total cost in a legible format.
* Include appropriate comments throughout the program.

Week 3:

* Read Chapter3 in *Python for Everybody.*
* Python Tutorial for Beginners 6: Conditionals and Booleans--If, Else, and Elif Statements, YouTube, 2017:
  + <https://youtu.be/DZwmZ8Usvnk>

Assignment:

This week we will implement “if statements” in a program. Your program will calculate the cost of fiber optic cable installation by multiplying the number of feet needed by $0.87. We will also evaluate a bulk discount. You will prompt the user for the number of fiber optic cable they need installed. Using the default value of $0.87 calculate the total expense. If the user purchases more than 100 feet they are charged $0.80 per foot. If the user purchases more than 250 feet they will be charged $0.70 per foot. If they purchase more than 500 feet, they will be charged $0.50 per foot.

* Your program must have a header. Use the [SUI--Edwardsville Programming Style Guide](https://www.cs.siue.edu/programming-style-guide) for guidance.
* Display a welcome message for your program.
* Get the company name from the user.
* Get the number of feet of fiber optic cable to be installed from the user.
* Evaluate the total cost based upon the number of feet requested.
* Display the calculated information including the number of feet requested and company name.

Week 4:

* Read Chapter 4 in *Python for Everybody.*

Youtube Videos

<https://youtu.be/9Os0o3wzS_I>

https://youtu.be/sugvnHA7ElY

Assignment:

* This week we will modify our If Statement program to add a function to do the heavy lifting.
* Modify your IF Statement program to add a function. This function will perform the cost calculation. The function will have two parameters (feet and price). When you call the function, you will pass two arguments to the function; feet of fiber to be installed and the cost (remember that price is dependent on the number of feet being installed). You probably should have the following:
  + Your program must have a header. Use the [SIU Edwardsville Programming Guide](https://www.cs.siue.edu/programming-style-guide) for guidance.
  + A welcome message
  + A function with two parameters
  + A call to the function
  + The application should calculate the cost based upon the number of feet being ordered
  + A printed message displaying the company name and the total calculated cost

Week 5:

* Read Chapter 5 in *Python for Everybody.*
* Watch Python Tutorial for Beginners 7: Loops and Iterations--For/While Loops, YouTube, 2017
  + https://youtu.be/6iF8Xb7Z3wQ

Assignment:

* Your program must have a header. Use the [programming style guide](https://www.cs.siue.edu/programming-style-guide) for guidance.
* This program will perform various calculations (addition, subtraction, multiplication, division, and average calculation)
* This program will contain a variety of loops and functions.
* The program will add, subtract, multiply, divide two numbers and provide the average of multiple numbers input by the user.
* Define a function named **performCalculation** which takes one parameter. The parameter will be the operation being performed (+, -, \*, /).
  + This function will perform the given prompt the user for two numbers then perform the expected operation depending on the parameter that's passed into the function.
  + This function will print the calculated value for the end user.
* Define a function named **calculateAverage** which takes no parameters.
  + This function will ask the user how many numbers they wish to input.
  + This function will use the number of times to run the program within a for loop in order to calculate the total and average.
  + This function will print the calculated average.
* This program will have a main section which contains a while loop. The while loop will be used to allow the user to run the program until they enter a value which ends the loop.
* The main program should prompt the user for the operation they wish to perform.
* The main program should evaluate the entered data using if statements.
* The main program should call the necessary function to perform the calculation.

Week 6:

* Read Chapters 6 and 8 in *Python for Everybody.*
* Watch Python Tutorial for Beginners 2: Strings - Working with Textual Data, YouTube, 2017
  + https://youtu.be/k9TUPpGqYTo
* Watch Python Lists || Python Tutorial || Learn Python Programming, YouTube, 2016
  + https://youtu.be/ohCDWZgNIU0
* Watch Python Tutorial: Slicing Lists and Strings, YouTube, 2015
  + <https://youtu.be/ajrtAuDg3yw>

Assignment:

This week we will create a program which works with lists. Your goal is to create a program which contains a list of temperatures. Your program will populate the list based upon user input. Your program will determine the number of temperatures in the program, determine the largest temperature, and the smallest temperature.

* Your program must have a header. Use the [programming style guide](https://www.cs.siue.edu/programming-style-guide) for guidance.
* Create an empty list called temperatures.
* Allow the user to input a series of temperatures along with a sentinel value which will stop the user input.
* Evaluate the temperature list to determine the largest and smallest temperature.
* Print the largest temperature.
* Print the smallest temperature.
* Print a message tells the user how many temperatures are in the list.

Week 8

* Read Chapters 9 and 10 in *Python for Everybody.*
* Watch Python Tutorial for Beginners 4: Lists, Tuples, and Sets, YouTube, 2017

Videos:

Watch Python Tutorial for Beginners 5: Dictionaries - Working with Key-Value Pairs, YouTube, 2017

<https://youtu.be/W8KRzm-HUcc>

Watch Python Tutorial: Working with JSON Data Using the JSON Module, YouTube, 2017

<https://youtu.be/daefaLgNkw0>

<https://youtu.be/9N6a-VLBa2I>

Assignment:

We will create a program which performs three essential operations. It will process this .txt file: [Gettysburg.txt](https://content.bellevue.edu/cst/dsc/510/resources/new/gettysburg.txt). (Click the link to download the text file).  Calculate the total words, and output the number of occurrences of each word in the file.

* Open the file and process each line.
* Either add each word to the dictionary with a frequency of 1 or update the word’s count by 1.
* Nicely print the output, in this case from high to low frequency. You should use string formatting for this. (See discussion 8.3).

We want to achieve each major goal with a function (one function, one action). We can find four functions that need to be created.

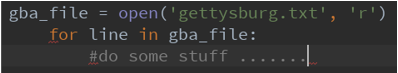
**add\_word:** Add each word to the dictionary. Parameters are the word and a dictionary. No return value.

**Process\_line:** There is some work to be done to process the line: strip off various characters, split out the words, and so on. Parameters are a line and the dictionary. It calls the function **add word** with each processed word. No return value.

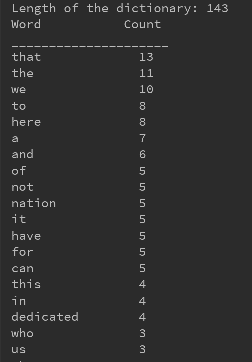
**Pretty\_print:** Because formatted printing can be messy and often particular to each situation (meaning that we might need to modify it later), we separated out the printing function. The parameter is a dictionary. No return value.

**main:** We will use a main function as the main program. As usual, it will open the file and call process\_line on each line. When finished, it will call pretty\_print to print the dictionary.

In the **main**function, you will need to open the file. We will cover more regarding opening of files next week but I wanted to provide you with the block of code you will utilize to open the file, see below.



**Example Output from the Program:**



Week 9:

* Read Chapter 7 in *Python for Everybody.*
* Watch Python Tutorial: File Objects - Reading and Writing to Files, YouTube, 2016

<https://youtu.be/Uh2ebFW8OYM>

<https://youtu.be/q5uM4VKywbA>

<https://youtu.be/NIWwJbo-9_8>

Assignment:

Last week we got a taste of working with files. This week we’ll really dive into files by opening and closing files properly.

For this week we will modify our Gettysburg processing program from week 8 in order to generate a text file from the output rather than printing to the screen. Your program should have a new function called **process\_file** which prints to the file (this method should almost be the same as the **pretty\_print** function from last week. Keep in mind that we have print statements in main as well. Your program must modify the print statements from **main** as well.

* Your program must have a header. Use the [programming style guide](https://www.cs.siue.edu/programming-style-guide) for guidance.
* Create a new function called process\_fie. This function will perform the same operations as pretty\_print from week 8 however it will print to a file instead of to the screen.
* Modify your main method to print the length of the dictionary to the file as opposed to the screen.
* This will require that you open the file twice. Once in main and once in process\_file.
* Prompt the user for the filename they wish to use to generate the report.
* Use the filename specified by the user to write the file.
* This will require you to pass the file as an additional parameter to your new process\_file function.

Week 10:

* Read Chapter 13 in Python for Everybody.
* Read [Building a Basic RestFul API in Python](https://www.codementor.io/sagaragarwal94/building-a-basic-restful-api-in-python-58k02xsiq), Codementor Community, 2017
* Watch REST API & RESTful Web Services Explained, YouTube, 2017

Video:

<https://youtu.be/LooL6_chvN4>

Assignment:

We’ve already looked at several examples of API integration from a Python perspective and this week we’re going to write a program that uses an open API to obtain data for the end user.

* Create a program which uses the Request library to make a GET request of the following API: [Chuck Norris Jokes](https://api.chucknorris.io/jokes/random).
* The program will receive a JSON response which includes various pieces of data. You should parse the JSON data to obtain the “value” key. The data associated with the value key should be displayed for the user (i.e., the joke).
* Your program should allow the user to request a Chuck Norris joke as many times as they would like. You should make sure that your program does error checking at this point. If you ask the user to enter “Y” and they enter y, is that ok? Does it fail? If it fails, display a message for the user. There are other ways to handle this. Think about included string functions you might be able to call.
* Your program must include a header as in previous weeks.
* Your program must include a welcome message for the user.
* Your program must generate “pretty” output. Simply dumping a bunch of data to the screen with no context doesn’t represent “pretty.”

Week 11:

* Read Chapter 14 in Python for Everybody.
* Read [Hello Markdown](http://www.firstpythonnotebook.org/markdown/), First Python Notebook, 2017
* Read [Markdown](https://www.fullstackpython.com/markdown.html), Full Stack Python
* Watch Python OOP Tutorial 1: Classes and Instances, YouTube, 2017

Videos:

<https://youtu.be/ZDa-Z5JzLYM>

<https://youtu.be/BJ-VvGyQxho>

<https://youtu.be/rq8cL2XMM5M>

<https://youtu.be/RSl87lqOXDE>

<https://youtu.be/3ohzBxoFHAY>

<https://youtu.be/jCzT9XFZ5bw>

Assignment:

This week we’re going to demonstrate our knowledge of Python object oriented programming concepts by creating a simple cash register program.

* Your program must have a header.
* Your program must have a welcome message for the user.
* Your program must have one class called CashRegister.
  + Your program will have an instance method called addItem which takes one parameter for price. The method should also keep track of the number of items in your cart.
  + Your program should have two getter methods.
    - getTotal – returns totalPrice
    - getCount – returns the itemCount of the cart
* Your program must create an instance of the CashRegister class.
* Your program should have a loop which allows the user to continue to add items to the cart until they request to quit.
* Your program should print the total number of items in the cart.
* Your program should print the total $ amount of the cart.
  + The output should be formatted as currency. Be sure to investigate the locale class. You will need to call locale.setlocale and locale.currency.

Final Assignment:

The class project is due at the end of this week.

##### Weather Program

For your class project we will be creating an application to interacts with a webservice in order to obtain data. Your program will use all of the information you’ve learned in the class in order to create a useful application.

Your program must prompt the user for their city or zip code and request weather forecast data from [OpenWeatherMap](http://openweathermap.org/" \t "_blank). Your program must display the weather information in a **READABLE** format to the user.

**Requirements:**

* Create a header for your program just as you have in the past.
* Create a Python Application which asks the user for their zip code or city.
* Use the zip code or city name in order to obtain weather forecast data from [OpenWeatherMap](http://openweathermap.org/" \t "_blank).
* Display the weather forecast in a readable format to the user.
* Use comments within the application where appropriate in order to document what the program is doing.
* Use functions including a main function.
* Allow the user to run the program multiple times to allow them to look up weather conditions for multiple locations.
* Validate whether the user entered valid data. If valid data isn’t presented notify the user.
* Use the **Requests** library in order to request data from the webservice.
  + Use Try blocks to ensure that your request was successful. If the connection was not successful display a message to the user.
* Use **Python 3**
* Use try blocks when establishing connections to the webservice. You must print a message to the user indicating whether or not the connection was successful

**Deliverables:**

* Final Program in a .py file (Due week 12)