**Assignment 3 [Python]**

**Saranya P**

<<Data Analytics >>

**Py Me Up, Charlie**

**Table of Contents**

[**Background** 2](#_Toc18260478)

[**Pre-Work** 2](#_Toc18260479)

[**Assignment Instruction** 4](#_Toc18260480)

[**PyBank** 4](#_Toc18260481)

[**Expected Output:** 5](#_Toc18260482)

[**PyPoll** 5](#_Toc18260483)

[**Expected Output:** 6](#_Toc18260484)

[**Script Explanation** 7](#_Toc18260485)

[**PyBank** 7](#_Toc18260486)

[***Terminal Snapshot*** 9](#_Toc18260487)

[***Text File*** 9](#_Toc18260488)

[**PyPoll** 10](#_Toc18260489)

[***Terminal Snapshot*** 13](#_Toc18260490)

[***Text File*** 13](#_Toc18260491)

[**Hints and Considerations** 14](#_Toc18260492)

[**GitHub** 14](#_Toc18260493)

**# Unit 3 | Assignment – Py Me Up, Charlie**

# **Background**

It's time to put away the Excel sheet and join the big leagues. Welcome to the world of programming with Python. In this homework assignment, you'll be using the concepts you've learned to complete \*\*2\*\* Python Challenges, PyBank and PyPoll.

Both of these challenges encompasses a real-world situation where your newfound Python scripting skills can come in handy. These challenges are far from easy so expect some hard work ahead!

# **Pre-Work**

1. Create a new repository for this project called `python-challenge`. \*\*Do not add this homework to an existing repository\*\*.

**<Sara Comment> :**

***A repository Python-Challenge have been created.***

<https://github.com/SaranyaPandiaraj/Python-Challenge>

2. Clone the new repository to your computer.

**<Sara Comment> :**

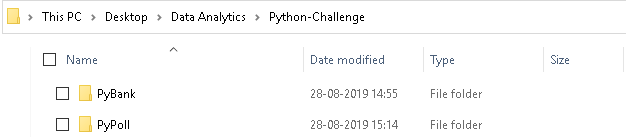
***The Python-Challenge have been cloned to my computer using the below command***

***<<git clone*** [***https://github.com/SaranyaPandiaraj/Python-Challenge.git***](https://github.com/SaranyaPandiaraj/Python-Challenge.git)***>>***

3. Inside your local git repository, create a directory for both of the Python Challenges. Use folder names corresponding to the challenges: \*\*PyBank\*\* and \*\*PyPoll\*\*.

**<Sara Comment> :**

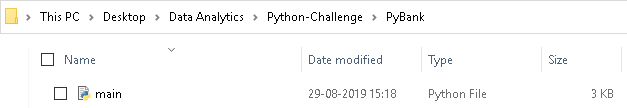
***Two Folders PyBank & PyPoll have been created inside my local git repository.***



4. Inside of each folder that you just created, add a new file called `main.py`. This will be the main script to run for each analysis.

**<Sara Comment> :**

***Main.py python script file have been created in each of the PyBank and PyPoll Folder.***





5. Push the above changes to GitHub or GitLab.

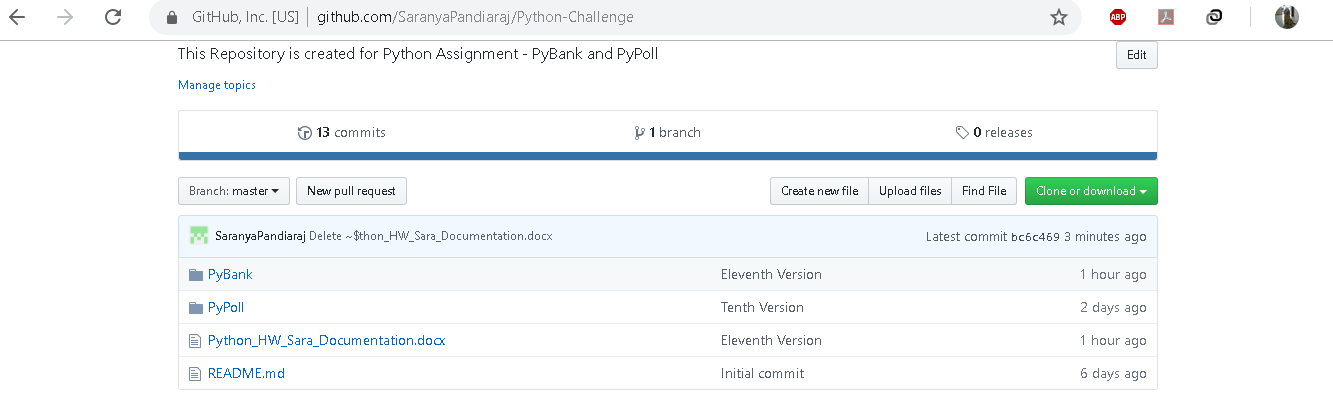
**<Sara Comment> :**

***The above changes have been pushed to GitLab using the below commands.***

***git add.***

***git commit -m “Initial Version”***

***git push***



# **Assignment Instruction**

## **PyBank**



\* In this challenge, you are tasked with creating a Python script for analyzing the financial records of your company. You will give a set of financial data called [budget\_data.csv](PyBank/Resources/budget\_data.csv). The dataset is composed of two columns: `Date` and `Profit/Losses`. (Thankfully, your company has rather lax standards for accounting so the records are simple.)

Your task is to create a Python script that analyzes the records to calculate each of the following:

* The total number of months included in the dataset
* The net total amount of "Profit/Losses" over the entire period
* The average of the changes in "Profit/Losses" over the entire period
* The greatest increase in profits (date and amount) over the entire period
* The greatest decrease in losses (date and amount) over the entire period

### **Expected Output:**

\* As an example, your analysis should look similar to the one below:

Financial Analysis

----------------------------

Total Months: 86

Total: $38382578

Average Change: $-2315.12

Greatest Increase in Profits: Feb-2012 ($1926159)

Greatest Decrease in Profits: Sep-2013 ($-2196167)

\* In addition, your final script should both print the analysis to the terminal and export a text file with the results.

## **PyPoll**



\* In this challenge, you are tasked with helping a small, rural town modernize its vote-counting process. (Up until now, Uncle Cleetus had been trustfully tallying them one-by-one, but unfortunately, his concentration isn't what it used to be.)

\* You will be give a set of poll data called [election\_data.csv](PyPoll/Resources/election\_data.csv). The dataset is composed of three columns: `Voter ID`, `County`, and `Candidate`. Your task is to create a Python script that analyzes the votes and calculates each of the following:

* The total number of votes cast
* A complete list of candidates who received votes
* The percentage of votes each candidate won
* The total number of votes each candidate won
* The winner of the election based on popular vote.

### **Expected Output:**

\* As an example, your analysis should look similar to the one below:

Election Results

-------------------------

Total Votes: 3521001

-------------------------

Khan: 63.000% (2218231)

Correy: 20.000% (704200)

Li: 14.000% (492940)

O'Tooley: 3.000% (105630)

-------------------------

Winner: Khan

-------------------------

\* In addition, your final script should both print the analysis to the terminal and export a text file with the results.

# **Script Explanation**

## **PyBank**

* The total number of months included in the dataset

**<Sara Comment> :**

***The Total Number of months is calculated by appending the Date column ( row[0] ) from the dataset to the Months List ( Months = [] ) and then the length of the Months List is derived to get the Total Months included in the dataset.***

**Months = Months + [row[0]]**

**Total\_Months = len(Months)**

* The net total amount of "Profit/Losses" over the entire period

**<Sara Comment> :**

***The Net Total Amount is calculated by appending the Profilt/Losses column ( row[1] ) from the dataset to the Amount List ( Amount = [] ) and then the Sum of the Amount List is derived to get the Net Total Amount of “Profit/Losses” over the entire period.***

**Amount.append(int(row[1]))**

**Total\_Amount = sum(Amount)**

* The average of the changes in "Profit/Losses" over the entire period

**<Sara Comment> :**

***The Average of the Change is calculated by appending the change in the Amount Column from the dataset to the Change\_Profit\_Loss List ( Change\_Profit\_Loss = [] ) and then the average of the Change\_Profit\_Loss is derived to get the average of the changes in “Profit/Losses” over the entire period.***

**for change in range(len(Amount)-1):**

**Change\_Profit\_Loss.append(Amount[change+1] - Amount[change])**

**Average\_Change = average(Change\_Profit\_Loss)**

* The greatest increase in profits (date and amount) over the entire period

**<Sara Comment> :**

***The Greatest Increase in Profits is calculated by taking the maximum value in the Change\_Profit\_Loss List.***

**Greatest\_Increase = max(Change\_Profit\_Loss)**

* The greatest decrease in losses (date and amount) over the entire period

**<Sara Comment> :**

***The Greatest Decrease in Profits is calculated by taking the minimum value in the Change\_Profit\_Loss List.***

**Greatest\_Decrease = min(Change\_Profit\_Loss)**

\* As an example, your analysis should look similar to the one below:

Financial Analysis

----------------------------

Total Months: 86

Total: $38382578

Average Change: $-2315.12

Greatest Increase in Profits: Feb-2012 ($1926159)

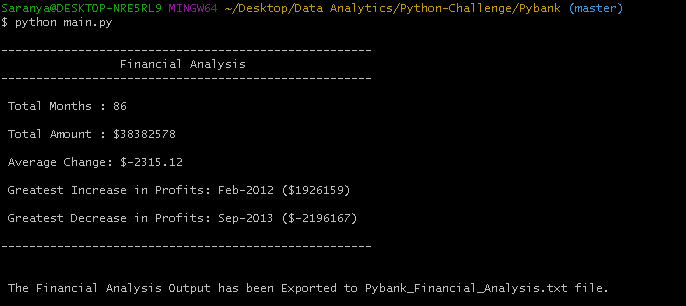
Greatest Decrease in Profits: Sep-2013 ($-2196167)

\* In addition, your final script should both print the analysis to the terminal and export a text file with the results.

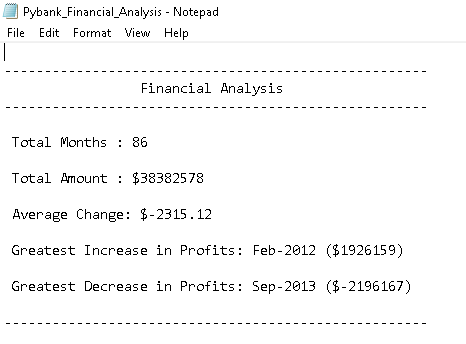
**<Sara Comment> :**

***The Financial Analysis are displayed as per the expected output in the terminal and also the same output has been exported to a text file (Pybank\_Financial\_Analysis.txt)***

### ***Terminal Snapshot***



### ***Text File***



## **PyPoll**

* The total number of votes cast

**<Sara Comment> :**

***The Total Number of Votes is calculated by appending the VoterID column ( row[0] ) from the dataset to the Votes List ( Votes = [] ) and then the length of the Votes List is derived to get the Total Number of Votes Cast.***

**Votes.append(int(row[0]))**

**Total\_Votes = len(Votes)**

* A complete list of candidates who received votes

**<Sara Comment> :**

***A Dictionary Candidates = {} has been created to store the Candidate Name as well the Total Number of Votes each candidate won. Here the Candidate Name is referred as the Key and the Vote Count as Value.***

***Logic:***

***Candidate Name 🡪 Key***

***Vote Count 🡪 Value***

***If a Candidate name is not present in the Candidates dictionary, The Candidate Name will be added in the dictionary having the Vote Count to be initialized to 1.***

***If a Candidate name is already present in the Candidates dictionary, The Candidate Name will be preserved and Vote Count will be incremented by 1 with the previous value.***

**if row[2] in Candidates.keys():**

**Candidates[row[2]] += 1**

**else:**

**Candidates[row[2]] = 1**

***The Candidate list is derived from the Candidates dictionary by accessing the Candidates.Items function in a loop. The Key of the Candidates Dictionary is the complete list of Candidates who received votes.***

**for Name, VoteCount in Candidates.items():**

**txt\_file.write(Name + " : " + str(Percent[Name]) + "% (" + str(VoteCount) + ")\n\n")**

* The percentage of votes each candidate won

**<Sara Comment> :**

***The Percentage of Votes is derived as Vote Count of each candidate by the Total Number of Candidates. A Percent = {} dictionary has been created to store the Candidate Name and the % of Votes the Candidates won.***

***The Vote Count for each Candidate is derived from the Candidates dictionary by accessing the Candidates.Items function in a loop.***

***The Total Votes is already derived by calculating the length of the Votes List.***

***The Percentage of votes each candidate won has been formatted and the value is stored in the Percent Dictionary.***

**for Name, VoteCount in Candidates.items():**

**Percent[Name] = format((VoteCount/Total\_Votes) \* 100 , '.3f')**

* The total number of votes each candidate won

**<Sara Comment> :**

***The Vote Count for each Candidate is derived from the Candidates dictionary by accessing the Candidates.Items function in a loop. The Value of the Candidates Dictionary is the Vote Count of each candidate.***

**for Name, VoteCount in Candidates.items():**

**txt\_file.write(Name + " : " + str(Percent[Name]) + "% (" + str(VoteCount) + ")\n\n")**

***Candidates Dictionary Function Outputs :***

**#print(Candidates.keys()) --> dict\_keys(['Khan', 'Correy', 'Li', "O'Tooley"])**

**#print(Candidates.values()) --> dict\_values([2218231, 704200, 492940, 105630])**

**#print(Candidates.items()) --> dict\_items([('Khan', 2218231), ('Correy', 704200), ('Li', 492940), ("O'Tooley", 105630)])**

* The winner of the election based on popular vote.

**<Sara Comment> :**

***The winner of the election is derived by comparing the Vote Count of each candidate and whoever has the highest Vote Count is declared as the Winner.***

***Candidate[Name] will give you the Vote Count value of eachCandidate and Name is the Candidate Name.***

**for Name in Candidates.keys():**

**if Candidates[Name] > Count:**

**Winner = Name**

**Count = Candidates[Name]**

\* As an example, your analysis should look similar to the one below:

Election Results

-------------------------

Total Votes: 3521001

-------------------------

Khan: 63.000% (2218231)

Correy: 20.000% (704200)

Li: 14.000% (492940)

O'Tooley: 3.000% (105630)

-------------------------

Winner: Khan

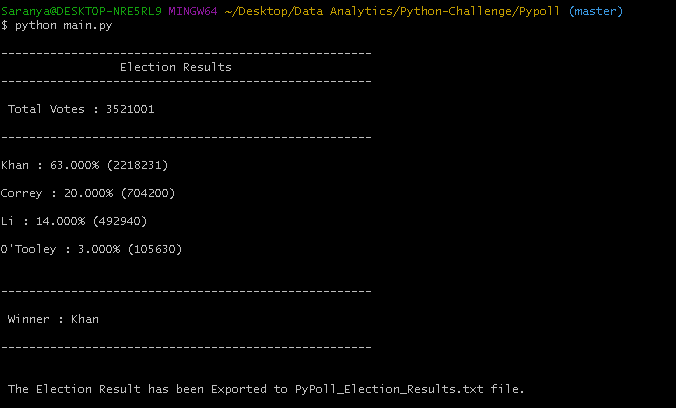
-------------------------

\* In addition, your final script should both print the analysis to the terminal and export a text file with the results.

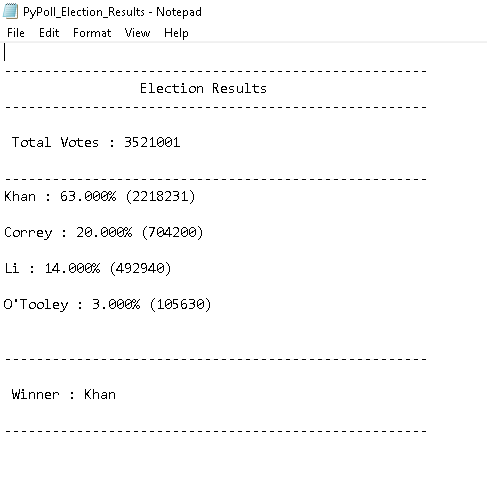
**<Sara Comment> :**

***The Election Results are displayed as per the expected output in the terminal and also the same output has been exported to a text file (PyPoll\_Election\_Results.txt)***

### ***Terminal Snapshot***



### ***Text File***



# **Hints and Considerations**

\* Consider what we've learned so far. To date, we've learned how to import modules like `csv`; to read and write files in various formats; to store contents in variables, lists, and dictionaries; to iterate through basic data structures; and to debug along the way. Using what we've learned, try to break down you tasks into discrete mini-objectives. This will be a \_much\_ better course of action than attempting to Google Search for a miracle.

\* As you will discover, for some of these challenges, the datasets are quite large. This was done purposefully, as it showcases one of the limits of Excel-based analysis. While our first instinct, as data analysts, is often to head straight into Excel, creating scripts in Python can provide us with more robust options for handling "big data".

\* Your scripts should work for each dataset provided. Run your script for each dataset separately to make sure that the code works for different data.

\* Feel encouraged to work in groups, but don't shortchange yourself by copying someone else's work. You get what you put in, and the art of programming is extremely unforgiving to moochers. Dig your heels in, burn the night oil, and learn this while you can! These are skills that will pay dividends in your future career.

\* Start early, and reach out for help often! Challenge yourself to identify \_specific\_ questions for your instructors and TAs. Don't resign yourself to simply saying, "I'm totally lost." Come prepared to show your effort and thought patterns, we'll be happy to help along the way.

\* Always commit your work and back it up with GitHub pushes. You don't want to lose hours of your work because you didn't push it to GitHub every half hour or so.

\* \*\*Commit often\*\*.

# **GitHub**

**<Sara Comment> :**

The link to the Python Assignment is in the below link.

<<https://github.com/SaranyaPandiaraj/Python-Challenge>>

The Exported Output File is present in the Output Folder of each of the Challenge.