**Student:** Samander Abayneh

**Professor:** Raymond Chan

**Course:** OPD435NBB

**Date:** March 20, 2020

**input**

* get data (command line arguments/options) from the user using the functions provided by the argparse module
* according to the arguments/options given at the command line, take appropriate processing action.

**processing**

* based on the file(s) specified, read the contents of each file and use appropriate objects to store it
* based on the command line arguments/options, process the data accordingly, which includes
  + data preprocessing (split a multi-day record into single day record)
  + record processing (preform required computation)

**output**

* output the required report based on the processed data

**g\_login function**:- will return unformatted list. once it read the input with command that has the word “last” then it will get the data from the system

**r\_login function**: - accepts one argument which is the log file name that will be given, it reads the file and return an unformatted list

g**et\_list function** takes 2 argument that are formatted list record and the pt of the column in the record. If pt is 0, it will take the username which is the first column in the list. Otherwise, it will take IP address which is the third column.

To make every values are unique not duplicated, all the items will be put in a set. After collecting all the values, it will return a list of usernames or a list of IP Addresses

**f\_rec function**:- accepts one argument which can be parsed from g\_loginord function or r\_login function. It reads a list of unformatted records and convert all the records which have a different day of login and logout time, to the same day then it returns a new list of records.

d1 = time.strftime("%j",time1) To return day of year

r\_list.append(item.split()) To save the record to a list

e\_time = time.ctime(next\_day).split() To convert to a readable time

if d1 != d2: To check if next day is on the same day with end day or not

**daily\_usage**:- It accepts 2 arguments. Username or IP address is the first argument. Second argument is the list of formatted records, which will get from function f\_rec. It gathers all the records in the same day then sum all the total time. It will return a dictionary with the key is the date and the value is total time for each date, and the Total time of all the dates

**Weakily\_usage**:- It accepts 2 arguments. Username or IP address is the first argument. Second argument is the list of formatted records, which will get from function f\_rec. It collects all the records in the same weekday of the year then sum all the total time. It will return a dictionary with the key is the date and the value is total time for each date, and the Total time of all the dates

**CT function**: - It takes one argument. It takes the result of the daily and weekly usage function and it gets dictionary and the total time. It sorts the key in the dictionary in reverse with text alignment and save it to the list.

**gen\_text** **function**: - when user run a command on different situation it will generate texts as an output and saves the test into a list.