Project 07

This assignment is worth 50 points (5.0% of the course grade) and must be **completed and turned in before 11:59 on Monday, October 30, 2017.**

Assignment Overview

- Functions
- Lists and Tuples
- Strings
- File manipulation

Assignment Background

The Internet has become an essential part of our lives. We use it to read emails, take courses (CSE231, there is a website for this course! Surprise!), and watch cat videos. However, connecting everything through a network needs to break information into pieces or small packets with identification. Just like letters need to have source and destination addresses, these packets needs to have IP addresses as source and destination.

Although the existence of the Internet is for good, some individuals use it to extract personal and confidential information like social security numbers, university records, etc. The task for this project is to determine the country of origin of the IP addresses that attacked the network. Specifically, one input file for this project was generated from the attacks on the CSE network in one day—the source is http://facweb.cse.msu.edu/climer/listdates.php For this project, we provide a file from that site, and for the blind test case we simply pick a different day. For privacy reasons, the IP addresses of the attacks will only have the 24 most significant digits (first three numbers, each ranging 0-255), addresses of that type are known as Class C.

Project Specifications

1. You must implement the following functions:

   a) **`open_file(message)`** has one parameter: `message`, a string. This function prompts the user to enter a filename displaying the message provided. The program will *try* to open a file. An error message should be shown if the file cannot be opened. This function will loop until it receives proper input and successfully opens the file. It returns a file object.

   b) **`read_ip_location(file)`** receives a file object as parameter (such as returned from `open_file()`) and reads a range of IP addresses corresponding to a specific two-digit country code (e.g. US for United States). *This function returns a list of 3-tuple items,* where each tuple contains `(start_ip, end_ip, country_code)`. Both `start_ip` and `end_ip` should be 12-digit integer numbers. That is to make the IP lookup easier. First split the ip address at periods, then add leading zeros to each number to make each of them three digits, i.e. the string 102.1.2.0 becomes the integer 102001002000. Hint: use the string `zfill()` method (zero fill) and do operations as strings to build a string of digits and only convert to an int after the whole number has been built. Here are the steps:

```
102.1.2.0
102  1  2  0  # after split(".")
102  001  002  000  # after zfill()
102001002000
```

c) **`read_ip_attack(file)`** receives a file object as a parameter (such as returned from `open_file()`) and reads detected IP addresses corresponding to an attack to a network. *This function returns a list of tuples*, where each tuple `(ip_int,ip_str)` includes a 12-digit integer number representing the IP address, and a string of the IP address with .xxx appended to the end of the IP address. The IP addresses in the file are Class-C IP addresses which means that each one has three of the four numbers that make up an IP address (the fourth is left out of the data to ensure privacy by identifying sub-networks rather than individual machines—yes, we maintain privacy of attackers!). To make a 12-digit integer out of three numbers, follow the steps explained in *b)* and add 000 to the right of the IP address, i.e. 102.1.2 should result in the tuple `(102001002000, "102.1.2.xxx")`.

d) **`read_country_name(file)`** receives a file object as parameter and reads the country code and its corresponding full name. This function returns a list of 2-tuples, where each tuple contains `(country_code, full name)`. **Note: The file has the country name first, make sure to append them to the list in the required order.**

e) **`locate_address(ip_list, ip_attack)`** This function searches for the IP address of *one* attack (`ip_attack`, an `int`) in the list of IP addresses (`ip_list`). If the IP attack address is found in the list, return the country code (string). An IP attack address is found, if all the numbers from the address are between the start address and end address. Note: the `ip_list` is the list of tuples returned by the `read_ip_location()` function.

f) **`get_country_name(country_list, country_code)`** This function searches for a `country_code` (string) in the list of countries (`country_list`) and return the name corresponding to the code (string). Note: `country_list` is the list of tuples returned by the `read_country_name()` function.

g) **`main()`** The main function calls the three functions to read the files, each returning a list. Then the program loops to determine the countries that attacks came from. Next the program determines the top ten countries with respect to the number of attacks. If two countries have the same number of attacks, the countries will be listed in reverse alphabetical order (that ordering is easiest to do because the number of attacks are also in the reverse order according to Python sort). The best way to handle this is to create a list of tuples with two items in each tuple: the country code and the attack count for that country. Then sort that list. See note below about sorting with `itemgetter`.

2. We provided the function `bar_plot(count_list, countries)` that plots `count_list` vs `countries`. This function receives two parameters: `count_list` which is a list of `integers` and contains the number of attacks made by the top 10 countries sorted in the reverse order (highest to lowest). `countries` is a list of country codes (`strings)` and contains the top 10 country abbreviations that made the most attacks. That is, the country codes (`countries`) are in the same order as the counts (`count_list`). Your program should prompt the user for plotting and plot only if user responds `yes`. Check sample output to see an example interaction.

3. Hints and suggestions

   a) The usage of the `.split()` is essential towards the completion of this project.
   b) Sorting of the top ten attacks requires sorting on both the number of attacks and on the country codes. Use the sorting technique from this weeks lab with
   `from operator import itemgetter`
   and within the sort call include `key = itemgetter(x,y)` where you decide the appropriate values for `x` and `y`.
   c) Use the `csv.reader()` to read the file of IP address locations. Note: do not call the `csv.reader` in the `open_file()` function.

```
import csv

fp = open(filename)
fp_csv = csv.reader(fp)
```

Deliverables

The deliverable for this assignment is the following file:

`proj07.py` – the source code for your Python program

Be sure to use the specified file name and to submit it for grading via the **Mimir system** before the project deadline.

Sample Output

Function Test `get_country_name`

```
Reads country_names_small.txt
Instructor data for 'AU','AT','AR': 'AUSTRALIA','AUSTRIA','ARGENTINA'
```

Function Test `locate_address`

```
Uses data ip_data = [(0, 255255255, 'ZZ'), (1000000000,
1000000255, 'AU'), (1000001000, 1000003255, 'CN'), (1000004000,
1000007255, 'AU'), (1000008000, 1000015255, 'CN'), (1000016000,
1000031255, 'JP'), (1000032000, 1000063255, 'CN'), (1000064000,
1000127255, 'JP'), (1000128000, 1000255255, 'TH'), (1001000000,
1001000255, 'CN')]
Instructor Data for ip_addresses 128015000, 1000002000, 1000120000,
1000005000, 1000055000:'ZZ','CN','JP','AU','CN'
```

Function Test `read_country_name`

```
Reads country_names_small.txt
Instructor country data: [('AF', 'AFGHANISTAN'), ('AL', 'ALBANIA'),
('DZ', 'ALGERIA'), ('AS', 'AMERICAN SAMOA'), ('AD', 'ANDORRA'), ('AO',
'ANGOLA'), ('AI', 'ANGUILLA'), ('AQ', 'ANTARCTICA'), ('AG', 'ANTIGUA
AND BARBUDA'), ('AR', 'ARGENTINA'), ('AM', 'ARMENIA'), ('AW',
'ARUBA'), ('AU', 'AUSTRALIA'), ('AT', 'AUSTRIA'), ('AZ',
'AZERBAIJAN'), ('BS', 'BAHAMAS'), ('BH', 'BAHRAIN'), ('BD',
'BANGLADESH'), ('BB', 'BARBADOS')]
```

Function Test `read_ip_attack`

```
Reads 2017-09-27-attacks-small.txt
Instructor attack data: [(31167062000, '31.167.62.xxx'),
(189004072000, '189.4.72.xxx'), (181014170000, '181.14.170.xxx'),
(71129073000, '71.129.73.xxx'), (140000068000, '140.0.68.xxx'),
(78100228000, '78.100.228.xxx'), (85137197000, '85.137.197.xxx'),
(144139172000, '144.139.172.xxx'), (154066252000, '154.66.252.xxx'),
(47008002000, '47.8.2.xxx'), (203221093000, '203.221.93.xxx'),
(182071170000, '182.71.170.xxx'), (124107220000, '124.107.220.xxx'),
(103242238000, '103.242.238.xxx'), (75159231000, '75.159.231.xxx'),
(105107116000, '105.107.116.xxx'), (71129073000, '71.129.73.xxx'),
(41096153000, '41.96.153.xxx'), (49148105000, '49.148.105.xxx'),
(24122027000, '24.122.27.xxx'), (65175193000, '65.175.193.xxx'),
(89142199000, '89.142.199.xxx'), (90104247000, '90.104.247.xxx'),
(85186223000, '85.186.223.xxx'), (121223018000, '121.223.18.xxx'),
(81064115000, '81.64.115.xxx'), (204044110000, '204.44.110.xxx'),
(185120125000, '185.120.125.xxx'), (71129073000, '71.129.73.xxx'),
(179211028000, '179.211.28.xxx')]
```

Function Test `read_ip_location`

```
Reads dbip-country-small.csv
Instructor data: [(0, 255255255, 'ZZ'), (1000000000, 1000000255,
'AU'), (1000001000, 1000003255, 'CN'), (1000004000, 1000007255,
'AU'), (1000008000, 1000015255, 'CN'), (1000016000, 1000031255,
'JP'), (1000032000, 1000063255, 'CN'), (1000064000, 1000127255,
'JP'), (1000128000, 1000255255, 'TH'), (1001000000, 1001000255,
'CN')]
```

Test 1

```
Enter the filename for the IP Address location list: wrong_filename.csv
File is not found! Try Again!
Enter the filename for the IP Address location list: wrong_filename.csv
File is not found! Try Again!
Enter the filename for the IP Address location list: dbip-country.csv
Enter the filename for the IP Address attacks: invalid_filename.txt
File is not found! Try Again!
Enter the filename for the IP Address attacks: 2017-09-27-attacks-small.txt
Enter the filename for the country codes: country_names.txt
Do you want to display all data? yes
The IP Address: 31.167.62.xxx      originated from SAUDI ARABIA
The IP Address: 189.4.72.xxx       originated from BRAZIL
The IP Address: 181.14.170.xxx     originated from ARGENTINA
The IP Address: 71.129.73.xxx      originated from UNITED STATES
The IP Address: 140.0.68.xxx       originated from INDONESIA
The IP Address: 78.100.228.xxx     originated from QATAR
The IP Address: 85.137.197.xxx     originated from SPAIN
The IP Address: 144.139.172.xxx    originated from AUSTRALIA
The IP Address: 154.66.252.xxx     originated from SOUTH AFRICA
The IP Address: 47.8.2.xxx         originated from INDIA
The IP Address: 203.221.93.xxx     originated from AUSTRALIA
The IP Address: 182.71.170.xxx     originated from INDIA
The IP Address: 124.107.220.xxx    originated from PHILIPPINES
The IP Address: 103.242.238.xxx    originated from INDIA
The IP Address: 75.159.231.xxx     originated from CANADA
The IP Address: 105.107.116.xxx    originated from ALGERIA
The IP Address: 71.129.73.xxx      originated from UNITED STATES
The IP Address: 41.96.153.xxx      originated from ALGERIA
The IP Address: 49.148.105.xxx     originated from PHILIPPINES
The IP Address: 24.122.27.xxx      originated from CANADA
The IP Address: 65.175.193.xxx     originated from UNITED STATES
The IP Address: 89.142.199.xxx     originated from SLOVENIA
The IP Address: 90.104.247.xxx     originated from FRANCE
The IP Address: 85.186.223.xxx     originated from ROMANIA
The IP Address: 121.223.18.xxx     originated from AUSTRALIA
The IP Address: 81.64.115.xxx      originated from FRANCE
The IP Address: 204.44.110.xxx     originated from UNITED STATES
The IP Address: 185.120.125.xxx    originated from ISRAEL
The IP Address: 71.129.73.xxx      originated from UNITED STATES
The IP Address: 179.211.28.xxx     originated from BRAZIL

Top 10 Attack Countries
Country  Count
```

```
US           5
IN           3
AU           3
PH           2
FR           2
DZ           2
CA           2
BR           2
ZA           1
SI           1

Do you want to plot? no
```

Test 2

```
Enter the filename for the IP Address location list: dbip-country.csv

Enter the filename for the IP Address attacks: 2017-09-27-attacks.txt

Enter the filename for the country codes: country_names.txt


Do you want to display all data? Yes
```

**** Too much data is output here to print in this document – see feedback on Mimir ****

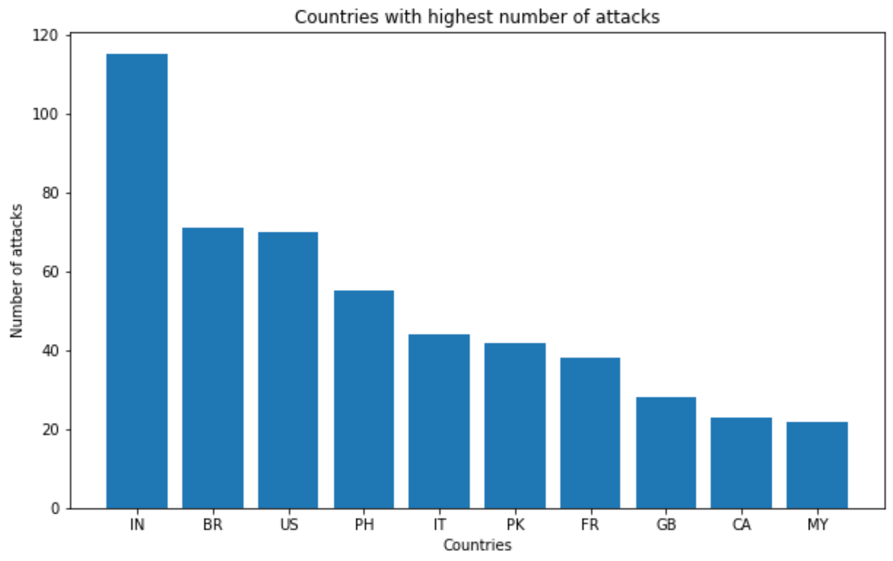
```
Top 10 Attack Countries
Country  Count
IN         115
BR          71
US          70
PH          55
IT          44
PK          42
FR          38
GB          28
CA          23
MY          22


Do you want to plot? No
```

Test 3 Blind Test (some other day of data – no plot)

Test 4 Bar Plot (same as test2, but with plot)

```
Enter the filename for the IP Address location list: dbip-country.csv

Enter the filename for the IP Address attacks: 2017-09-27-attacks.txt

Enter the filename for the country codes: country_names.txt
```

```
Do you want to display all data? no

Top 10 Attack Countries
Country  Count
IN         115
BR          71
US          70
PH          55
IT          44
PK          42
FR          38
GB          28
CA          23
MY          22


Do you want to plot? Yes
```



Grading Rubric

```
General Requirements:
  (5 pts) Coding Standard 1-9
          (descriptive comments, function headers, etc...)


Implementation:
  (4 pts) open_file function (no Mimir test)
     -2 not using try/except and -2 for not using a while loop
  (3 pts) get_country_name
  (3 pts) locate_address
  (5 pts) read_country_name
  (5 pts) read_ip_attack
  (5 pts) read_ip_location
  (5 pts) Test1
  (5 pts) Test2
  (5 pts) Test3: blind test
  (5 pts) Test4: Bar plot (no Mimir test)
```