Rene Esquivel

CS2302

80219337

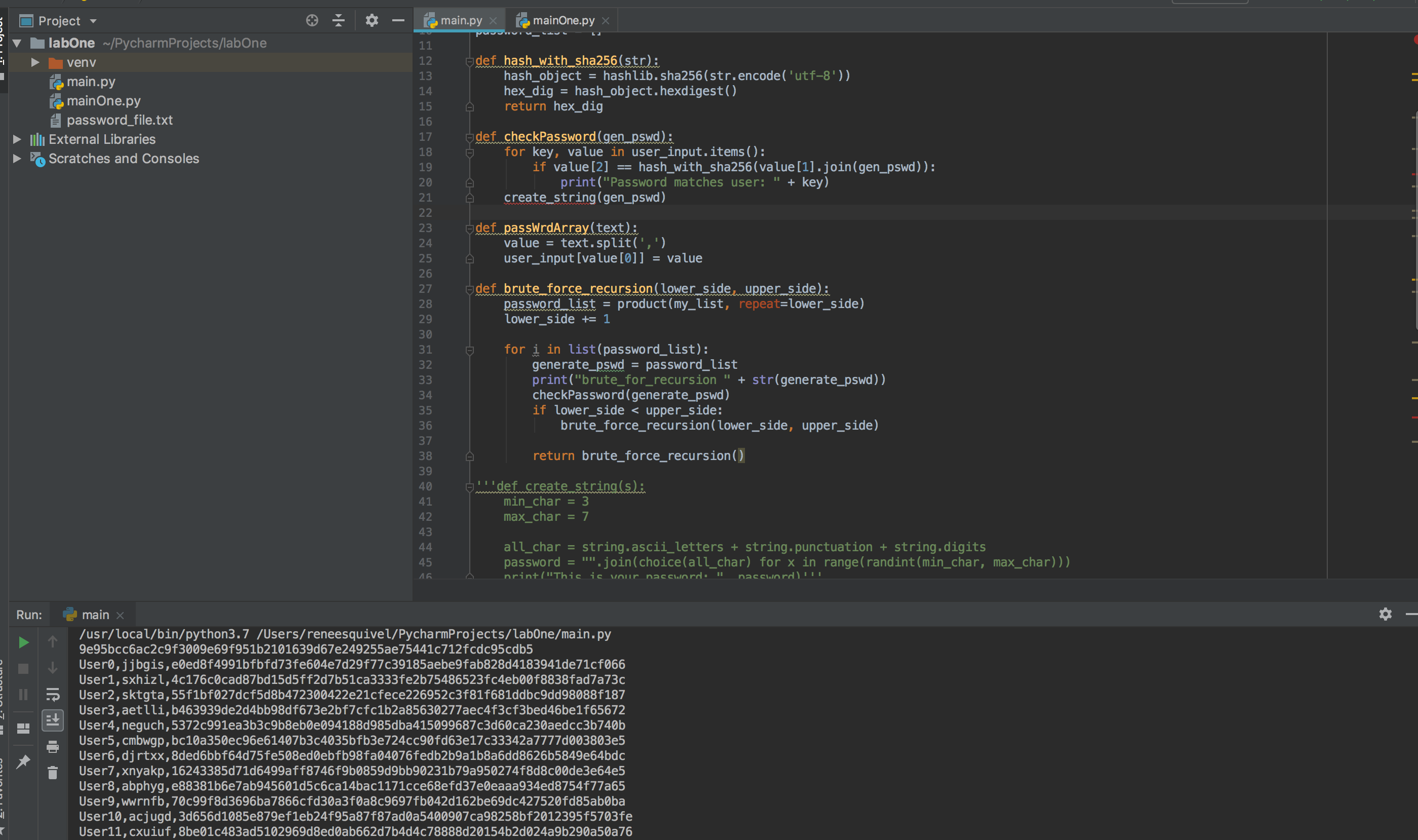
This lab was given a password\_file.txt file with consisted of 100 usernames, salt values, and hashed passwords. My objective in this lab is to display my hacking abilities and be able to find the real passwords associated with each account. There are certain parameters that must be met as well mentioned in the lab.

The proposed solution was to have the .txt file read into my program. From there I am wanting to create a string of arrays then split() the program up into a 2d array. Once I had my 2d array split up and sectioned off I wanted to use the preset hashlib sha256 function to be able to decipher my.

Then use the function for recursion using brute force to crack the code and return the actual password capability using a check password function.

The experiment I ran were to put my functions in different sections to make it ran better. I even added a password creator that allowed me to put in a 3 to 7 digit long password, thus experimenting with different values.

SAMPLE RUN:



RUN TIME:

|  |  |  |  |
| --- | --- | --- | --- |
| \_new\_module | 1 | 0 | 0 |
| \_relax\_case | 5 | 0 | 0 |
| \_\_init\_\_ | 2 | 0 | 0 |
| acquire | 2 | 0 | 0 |
| release | 2 | 0 | 0 |
| \_\_init\_\_ | 2 | 0 | 0 |
| \_\_enter\_\_ | 2 | 0 | 0 |
| \_\_exit\_\_ | 2 | 0 | 0 |
| cb | 2 | 0 | 0 |
| \_get\_module\_lock | 2 | 0 | 0 |
| \_call\_with\_frames\_removed | 3 | 1 | 0 |
| \_verbose\_message | 28 | 0 | 0 |
| \_requires\_builtin\_wrapper | 1 | 0 | 0 |
| \_\_init\_\_ | 2 | 0 | 0 |
| \_\_enter\_\_ | 2 | 0 | 0 |
|  | 8 | 0 | 0 |
| \_\_exit\_\_ | 2 | 0 | 0 |
| \_r\_long | 3 | 0 | 0 |
| \_\_init\_\_ | 2 | 0 | 0 |
| cached | 2 | 0 | 0 |
| parent | 2 | 0 | 0 |
| has\_location | 2 | 0 | 0 |
|  | 26 | 0 | 0 |
| spec\_from\_loader | 1 | 0 | 0 |
| \_init\_module\_attrs | 2 | 0 | 0 |
| module\_from\_spec | 2 | 0 | 0 |
| \_load\_unlocked | 2 | 1 | 0 |
| find\_spec | 2 | 0 | 0 |
| create\_module | 1 | 0 | 0 |
| exec\_module | 1 | 0 | 0 |
| is\_package | 1 | 0 | 0 |
| \_path\_join | 26 | 0 | 0 |
| find\_spec | 1 | 0 | 0 |
| \_path\_split | 2 | 0 | 0 |
| \_\_enter\_\_ | 4 | 0 | 0 |
| \_\_exit\_\_ | 4 | 0 | 0 |
| \_path\_stat | 7 | 0 | 0 |
| \_find\_spec | 2 | 0 | 0 |
| \_find\_and\_load\_unlocked | 2 | 2 | 0 |
| \_find\_and\_load | 2 | 2 | 0 |
| \_handle\_fromlist | 3 | 0 | 0 |
| \_path\_is\_mode\_type | 1 | 0 | 0 |
| \_path\_isfile | 1 | 0 | 0 |
| cache\_from\_source | 2 | 0 | 0 |
| \_get\_cached | 1 | 0 | 0 |
| \_check\_name\_wrapper | 1 | 0 | 0 |
| \_classify\_pyc | 1 | 0 | 0 |
| \_validate\_timestamp\_pyc | 1 | 0 | 0 |
| \_compile\_bytecode | 1 | 0 | 0 |
| spec\_from\_file\_location | 1 | 0 | 0 |
| create\_module | 1 | 0 | 0 |
| exec\_module | 1 | 1 | 0 |
| get\_code | 1 | 0 | 0 |
| \_\_init\_\_ | 1 | 0 | 0 |
| get\_filename | 1 | 0 | 0 |
| get\_data | 1 | 0 | 0 |
| path\_stats | 1 | 0 | 0 |
| \_path\_importer\_cache | 6 | 0 | 0 |
| \_get\_spec | 1 | 0 | 0 |
| find\_spec | 1 | 0 | 0 |
| \_get\_spec | 1 | 0 | 0 |
| find\_spec | 5 | 0 | 0 |
| \_\_init\_\_ | 3 | 0 | 0 |
| \_\_init\_\_ | 3 | 0 | 0 |
| decode | 2 | 0 | 0 |
| join | 1 | 0 | 0 |
| \_get\_sep | 2 | 0 | 0 |
| dirname | 1 | 0 | 0 |
| \_optimize\_charset | 4 | 0 | 0 |
| \_compile | 1 | 1 | 0 |
| compile | 1 | 1 | 0 |
| escape | 1 | 0 | 0 |
| \_\_call\_\_ | 4 | 0 | 0 |
| \_combine\_flags | 6 | 0 | 0 |
| \_\_new\_\_ | 4 | 0 | 0 |
| name | 12 | 0 | 0 |
| value | 2 | 0 | 0 |
| \_compile | 13 | 0 | 0 |
| \_compile\_charset | 4 | 0 | 0 |
| \_mk\_bitmap | 2 | 0 | 0 |
| \_missing\_ | 1 | 0 | 0 |
| \_create\_pseudo\_member\_ | 1 | 0 | 0 |
| \_\_or\_\_ | 1 | 0 | 0 |
| \_\_and\_\_ | 1 | 0 | 0 |
| \_high\_bit | 2 | 0 | 0 |
|  | 1 | 0 | 0 |
|  | 2 | 0 | 0 |
| \_decompose | 1 | 0 | 0 |
| \_power\_of\_two | 3 | 0 | 0 |
|  | 2 | 0 | 0 |
| \_simple | 2 | 0 | 0 |
| \_generate\_overlap\_table | 1 | 0 | 0 |
| \_get\_iscased | 1 | 0 | 0 |
| \_get\_literal\_prefix | 1 | 0 | 0 |
| \_\_get\_\_ | 14 | 0 | 0 |
| \_compile\_info | 1 | 0 | 0 |
| isstring | 2 | 0 | 0 |
| \_code | 1 | 0 | 0 |
| compile | 1 | 1 | 0 |
| \_\_init\_\_ | 1 | 0 | 0 |
| groups | 10 | 0 | 0 |
| opengroup | 4 | 0 | 0 |
| closegroup | 4 | 0 | 0 |
| \_\_init\_\_ | 14 | 0 | 0 |
| \_\_len\_\_ | 21 | 0 | 0 |
| \_\_getitem\_\_ | 57 | 0 | 0 |
| \_\_setitem\_\_ | 3 | 0 | 0 |
| append | 16 | 0 | 0 |
| getwidth | 17 | 0 | 0 |
| \_\_init\_\_ | 1 | 0 | 0 |
| \_\_next | 318 | 0 | 0 |
| match | 47 | 0 | 0 |
| get | 260 | 0 | 0 |
| getuntil | 4 | 0 | 0 |
| tell | 22 | 0 | 0 |
| seek | 1 | 0 | 0 |
| \_escape | 2 | 0 | 0 |
| \_uniq | 4 | 0 | 0 |
| \_parse\_sub | 8 | 0 | 0 |
| \_parse | 11 | 0 | 0 |
| \_parse\_flags | 2 | 0 | 0 |
| fix\_flags | 1 | 0 | 0 |
| parse | 1 | 0 | 0 |
| getpreferredencoding | 2 | 0 | 0 |
|  | 1 | 0 | 0 |
|  | 1 | 0 | 0 |
| hash\_with\_sha256 | 1 | 0 | 0 |
| main | 1 | 0 | 0 |
|  | 1 | 2 | 0 |
| \_\_init\_\_ | 1 | 1 | 0 |
| \_TemplateMetaclass | 1 | 0 | 0 |
| Template | 1 | 0 | 0 |
| Formatter | 1 | 0 | 0 |
|  | 1 | 1 | 0 |

CONCLUSION:

I was unable to get the code to work in the manner I wanted it to work. I was able to import the .txt file, but after that my code didn’t seem to leave the main. The main take away from this lab is that I learned how to bring a file in from a .txt wordpad. One lesson learned was to get better work with functions and calling them in different function methods. I also learned how to look at strings and arrays in a different way and adjust to try and get the program to work, I’m seeking more practice in this area. This was my first try with python and feel that I am having trouble with the way it works.

APPENDIX, SOURCE CODE:

# CS2302  
import os  
import hashlib  
from itertools import product  
import cProfile  
import string  
from random import \*  
  
user\_input = {}  
my\_list = "0123456789"  
password\_list = []  
  
def myFunc():  
  
 cProfile.run('MyFunc()')  
  
def hash\_with\_sha256(str):  
 hash\_object = hashlib.sha256(str.encode('utf-8'))  
 hex\_dig = hash\_object.hexdigest()  
 return hex\_dig  
  
def checkPassword(gen\_pswd):  
 for key, value in user\_input.items():  
 if value[2] == hash\_with\_sha256(value[1].join(gen\_pswd)):  
 print("Password matches user: " + key)  
 create\_string(gen\_pswd)  
  
def passWrdArray(text):  
 value = text.split(',')  
 user\_input[value[0]] = value  
  
def brute\_force\_recursion(lower\_side, upper\_side):  
 password\_list = product(my\_list, repeat=lower\_side)  
 lower\_side += 1  
  
 for i in list(password\_list):  
 generate\_pswd = password\_list  
 print("brute\_for\_recursion " + str(generate\_pswd))  
 checkPassword(generate\_pswd)  
 if lower\_side < upper\_side:  
 brute\_force\_recursion(lower\_side, upper\_side)  
  
 return brute\_force\_recursion()  
  
def create\_string(s):  
 min\_char = 3  
 max\_char = 7  
  
 all\_char = string.ascii\_letters + string.punctuation + string.digits  
 password = "".join(choice(all\_char) for x in range(randint(min\_char, max\_char)))  
 print("This is your password: ", password)  
  
def file\_line(array\_file):  
 value = array\_file.split(',')  
 user\_input[value[0]] = value  
  
  
def pswd\_creator(lower, upper):  
 password\_list = (my\_list, lower)  
 lower += 1  
  
 for i in my\_list(password\_list):  
 generate\_pswd = "".join(i)  
 print("pswd\_creator" + generate\_pswd)  
 rev\_password(generate\_pswd)  
 if lower < upper:  
 pswd\_creator(lower, upper)  
  
def main():  
  
 hex\_dig = hash\_with\_sha256('This is how you hash a string with sha256')  
 print(hex\_dig)  
  
 dirpath = os.path.dirname('password\_file.txt')  
 filepath = os.path.join(dirpath, 'password\_file.txt')  
 open(filepath, 'r')  
  
 fr = open('password\_file.txt', 'r')  
 text = fr.read()  
 print(text)  
 fr.close()  
  
  
main()

“I certify that this project is entirely my own work.

I wrote, debugged, and tested the code being presented, performed the

experiments, and wrote the report. I also certify that I did not share my code or

report or provided inappropriate assistance to any student in the class.

Rene Esquivel

80219337