

Week 4 Deliverables

Overview: In this week, you have studied additional Python language syntax including Arrays and Strings. In particular, you used the numpy, regular expressions, and Panda libraries to help manipulate and store data. The Lab for this week demonstrates your knowledge of this additional Python functionality. Be sure to use the examples in the textbook reading along with the associate libraries, functions and processes when completing the assignments for this week.

Submission requirements for this project include 2 files. (Zipping them into one file is acceptable and encouraged):

- Python Numpy and Pandas Application Code
- Word or PDF file containing your test and pylint results along with the Password cracking activity results

Python Applications for this lab: (total 100 points):

This lab consists of three parts.

1. **(60 points)** allows a user to **enter and validate** their phone number and zipcode+4. Then the user will enter values of two, 3x3 matrices and then select from options including, addition, subtraction, matrix multiplication, and element by element multiplication. You should use numpy.matmul() for matrix multiplication (e.g. np.matmul(a, b)). The program should compute the appropriate results and return the results, the transpose of the results, the mean of the rows for the results, and the mean of the columns for the results.

When entering data, the application should use regular expressions and/or Pandas functionality to check the format of the phone number and zipcode. You should check that each value is numeric for the matrices. The user interface should continue to run until the user indicates they are ready to exit.

A user interface might look similar to this:

```
***** Welcome to the Python Matrix Application*****  
  
Do you want to play the Matrix Game?  
Enter Y for Yes or N for No:  
Y  
Enter your phone number (XXX-XXX-XXXX):  
555-555-55  
Your phone number is not in correct format. Please reenter:  
555-555-5555  
Enter your zip code+4 (XXXXX-XXXX):  
21022-3213
```

Enter your first 3x3 matrix:

1 2 4
4 2 1
3 8 9

Your first 3x3 matrix is:

1 2 4
4 2 1
3 8 9

Enter your second 3x3 matrix:

3 2 1
7 2 5
5 2 1

Your first 3x3 matrix is:

3 2 1
7 2 5
5 2 1

Select a Matrix Operation from the list below:

- a. Addition
- b. Subtraction
- c. Matrix Multiplication
- d. Element by element multiplication

a

You selected Addition. The results are:

4 4 5
11 4 6
8 10 10

The Transpose is:

4 11 8
4 4 10
5 6 10

The row and column mean values of the results are:

Row: 4.33, 7, 9.33

Column: 7.66, 6, 7

```

Do you want to play the Matrix Game?

Enter Y for Yes or N for No:

N

***** Thanks for playing Python Numpy *****

```

If an inappropriate entry is detected, the program should prompt for a correct value and continue to do so until a correct value is entered.

Hints:

1. Use numpy, pandas and regular expressions as appropriate.
2. Create and use functions as often as possible
3. Both integers and float values are acceptable
4. Use comments to document your code
5. Test with many combinations.
6. Use pylint to verify the code style – the goal is a 10!

2. **(15 points)** Document your testing results using your programming environment. You should also include and discuss your pylint results for the application. The test document should include a test table that includes the input values, the expected results and the actual results. A screen capture should be included that shows the actual test results of running each test case found in the test table. Be sure to include multiple test cases to provide full coverage for all code and for each function you develop and test.

3. **(25 points)** Password crackers can easily be written using Python code. You can also generate a hashed password using Python with a variety of hash algorithms. For this exercise, you will create use Python code to generate ten (10) passwords with different hashing algorithms and then use a popular online password cracking website to see if the passwords can be cracked.

For example, the following Python code can be used to hash a password input using MD-5, SHA-256 and SHA-512 algorithms.

```

import hashlib

# input a message to encode
print('Enter a message to encode:')
message = input()

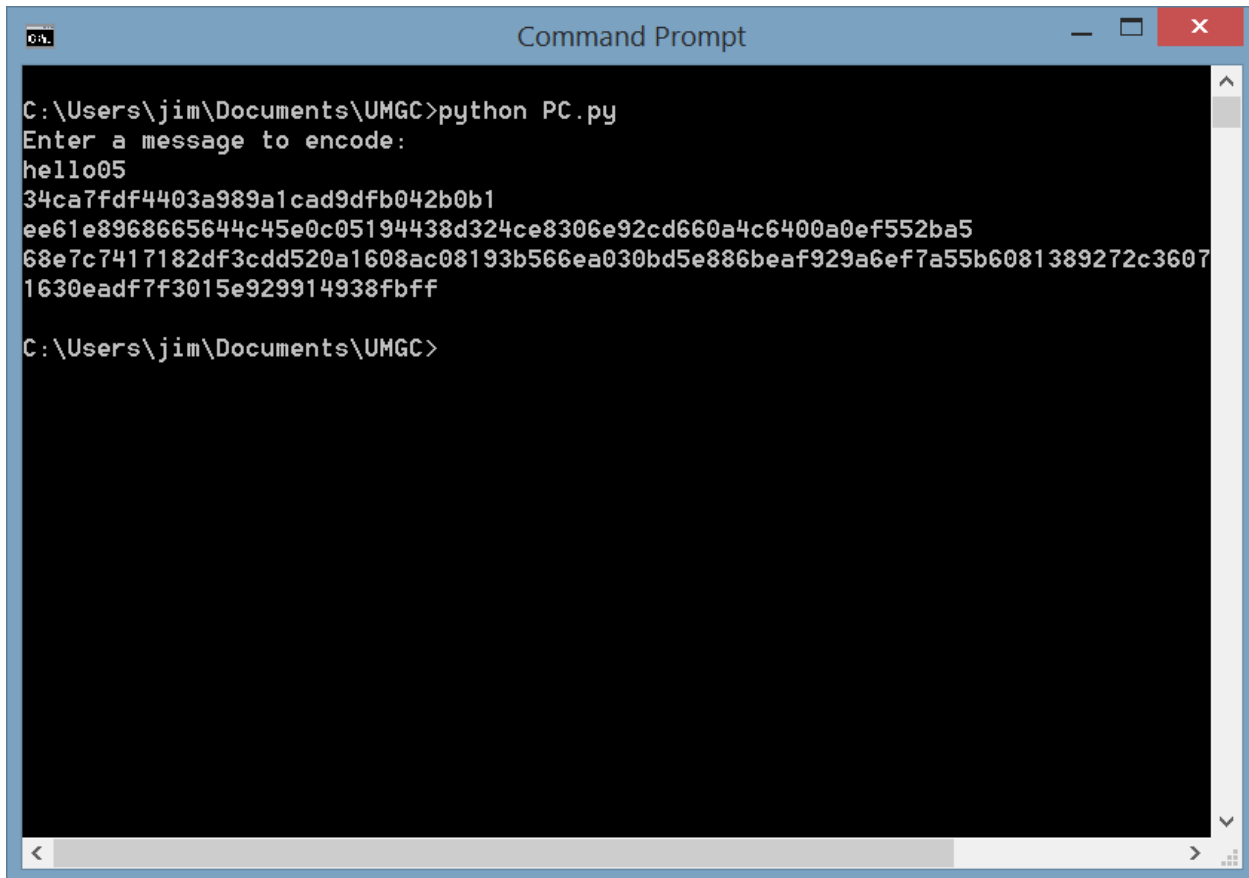
# encode it to bytes using UTF-8 encoding
message = message.encode()

# hash with MD5 (very weak)
print(hashlib.md5(message).hexdigest())

```

```
# Lets try a stronger SHA-2 family  
print(hashlib.sha256(message).hexdigest())  
print(hashlib.sha512(message).hexdigest())
```

When you run this at the command prompt, you enter a password and then you will receive the hashed values for the MD-5, SHA-256 and SHA-512 algorithms, respectively.



```
C:\Users\jim\Documents\UMGC>python PC.py  
Enter a message to encode:  
hello05  
34ca7fdf4403a989a1cad9dfb042b0b1  
ee61e8968665644c45e0c05194438d324ce8306e92cd660a4c6400a0ef552ba5  
68e7c7417182df3cdd520a1608ac08193b566ea030bd5e886beaf929a6ef7a55b6081389272c3607  
1630eadf7f3015e929914938fbff  
C:\Users\jim\Documents\UMGC>
```

In this case, hello05 was entered and resulted in the 3 outputs. Notice, the SHA-512 is quite long and extends over two lines.

If we take those 3 output hashes and input them into the Crackstation.net URL, the passwords are hacked in each case:

CrackStation Password Hashing Security Defuse Security

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
34ca7fdf4403a989a1cad9dfb042b0b1
ee61e8968665644c45e0c05194438d324ce8306e92cd660a4c6400a0ef552ba5
68e7c7417182df3cdd520a1608ac08193b566ea030bd5e886beaf929a6ef7a55
b6081389272c36075e12a2b415aa278067511630eadf7f3015e929914938fbff
```

I'm not a robot reCAPTCHA Privacy - Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1 sha1_bin), QubesV3.1BackupDefaults

Hash	Type	Result
34ca7fdf4403a989a1cad9dfb042b0b1	md5	hello05
ee61e8968665644c45e0c05194438d324ce8306e92cd660a4c6400a0ef552ba5	sha256	hello05
68e7c7417182df3cdd520a1608ac08193b566ea030bd5e886beaf929a6ef7a55 b6081389272c36075e12a2b415aa278067511630eadf7f3015e929914938fbff	sha512	hello05

Color Codes: Green Exact match, Yellow Partial match, Red Not found.

[Download CrackStation's Wordlist](#)

Notice in each case, the original password of hello05 was cracked.

You can salt a password by adding some random text of a phrase to the beginning of your actual password. This will add some strength to preventing a quick decryption. For this simple Python code, you can simulate a salt by adding a string to the front of password and then run the output into the Crackstation. For example, consider this salted password:

`wellwhatdoyouknowaboutthathello05`

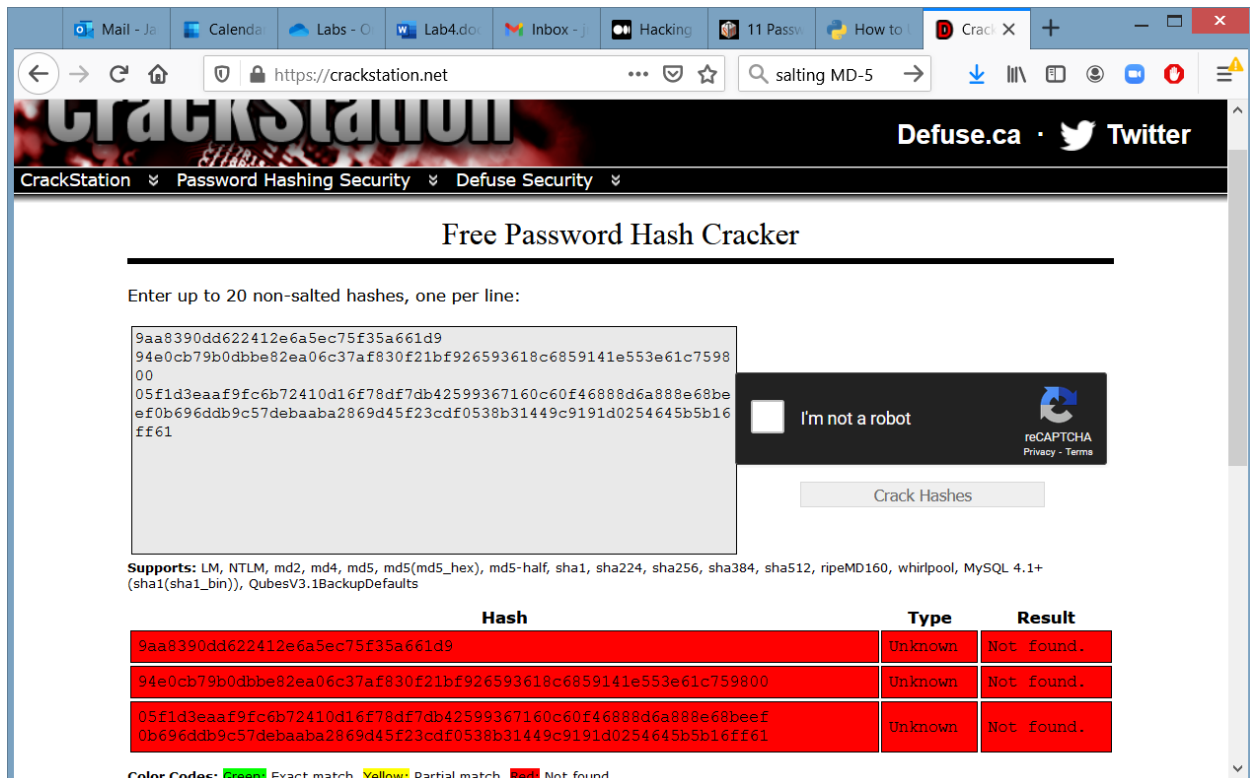
In this case the phrase “wellwhatdoyouknowaboutthat” is placed in front of hello05. The resulting output hashes are:

9aa8390dd622412e6a5ec75f35a661d9

94e0cb79b0dbbe82ea06c37af830f21bf926593618c6859141e553e61c759800

05f1d3eaaf9fc6b72410d16f78df7db42599367160c60f46888d6a888e68beef0b696ddb9c57debaaba2869
d45f23cdf0538b31449c9191d0254645b5b16ff61

When use those hash values as input into the Crackstation, the decoder fails:



For your activity, experiment with the Python script using MD-5, SHA-256 and SHA-512 hash algorithms for at least 20 different passwords. Be sure to experiment with “easy” passwords, salted passwords as well as randomly generated passwords (e.g. from sites such as Norton password generator (<https://my.norton.com/extspa/passwordmanager?path=pwd-gen>)).

For your report, prepare a table that shows the input password, the resulting hashes and if the Crackstation.net site was able to crack the password. An example table is shown below:

Table 1. Password Cracking Activity Results

Password	Hash output	Did Crackstation work?
password01	af88a0ae641589b908fa8b31f0fc6e1 4b8f353889d9a05d17946e26d014efe99407cba8bd9d 0102d4aab10ce6229043 746a5a2664633cb15829e80cc8d5dd7368b1d939756e 7b069df9df482e2afc3c44029ec71ffbf7cc9916719d861 b60fc34b5 bd6a4f2cb0fe7747d99d5b219162	Yes
...		

Be sure to describe what you learned from this password cracking activity and what would you recommend as possible strong passwords after completing this activity in your report?

Any submissions that do not represent work originating from the student will be submitted to the Dean’s office and evaluated for possible academic integrity violations and sanctions.