Lab 4

Terrence Jackson

SDEV 300: Building Secure Python Applications

Professor Howard

Apr 9, 2024

Matrix Application

Test Cases

	Input	Expected	Actual	Pass?
Test Case 1: Invalid phone number	Play? Y Phone: apple	Invalid phone number, prompt again	Invalid, please input a phone number in the format XXX-XXX-XXXX	Yes
Test Case 2: Valid Phone Number	Play? yes Phone: 123-456-8794	Print phone number Prompt for zip	Your phone number: 123-456-8794 Enter your zip code+4 (XXXXX-XXXX):	Yes
Test Case 3: Invalid zip	Play? y Phone: 123-456-8794 Zip: pear	Invalid zip, prompt again	Invalid, please input a phone number in the format XXX-XXX-XXXX Enter your zip code+4 (XXXXX-XXXX):	Yes
Test Case 4: Valid zip	Play? y Phone: 123-456-8794 Zip: 12345-1234	Print zip Prompt for first matrix	Your zip code: 12345-1234 Enter your first 3x3 matrix:	Yes

Test Case 5: Invalid matrix	apple pear tangerine	Invalid, prompt again	Invalid input.Please input a matrix using only integers and floats in the following format: X X X X X X X X X Enter your first 3x3 matrix:	Yes
Test Case 6: Matrix Addition	Matrix 1: 4 7 6 2 3 7 9 5 2 Matrix 2: 1 5 2 3 2 3 9 6 9 Selection: a	Print matrices Addition: 5 12 8 5 5 10 18 11 11 Transpose: 5 5 18 12 5 11 8 10 11 Row mean: 8.333 6.666 13.333 Col mean: 9.333 9.333 9.666 Prompt to play again	Your first 3x3 matrix is: 4.00 7.00 6.00 2.00 3.00 7.00 9.00 5.00 2.00 Your second 3x3 matrix is: 1.00 5.00 2.00 3.00 2.00 3.00 9.00 6.00 9.00 You selected Addition. The results are: 5.00 12.00 8.00 5.00 5.00 10.00 18.00 11.00 11.00 The Transpose is: 5.00 5.00 18.00 12.00 5.00 11.00 8.00 10.00 11.00 The row and column mean values of the results are:	Yes

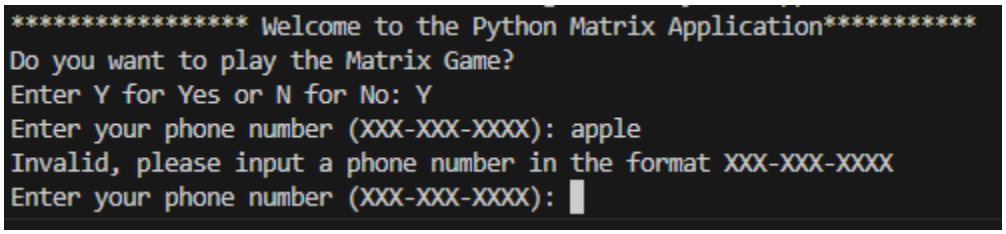
			Row: 8.33 6.67 13.33 Column: 9.33 9.33 9.67	
Test Case 7: Matrix Subtraction	Matrix 1: 3.4 2.9 4.4 0.6 8.6 9.9 5.1 6.2 6.1 Matrix 2: 1.5 6.5 6.3 3.5 4.1 5.1 1.4 7.8 2.5	Print matrices Subtraction: 1.9 -3.6 -1.9 -2.9 4.5 4.8 3.7 -1.6 3.6 Transpose: 1.9 -2.9 3.7 -3.6 4.5 -1.6 -1.9 4.8 3.6 Row: -1.2 2.13 1.9 Column:0.9 -0.23 2.17 Prompt to play again	Your first 3x3 matrix is: 3.40 2.90 4.40 0.60 8.60 9.90 5.10 6.20 6.10 Your second 3x3 matrix is: 1.50 6.50 6.30 3.50 4.10 5.10 1.40 7.80 2.50 You selected Subtraction. The results are: 1.90 -3.60 -1.90 -2.90 4.50 4.80 3.70 -1.60 3.60 The Transpose is: 1.90 -2.90 3.70 -3.60 4.50 -1.60 -1.90 4.80 3.60 The row and column mean values of the results are: Row: -1.20 2.13 1.90 Column: 0.90 -0.23 2.17	Yes

Test Case 8: Matrix Mult	Matrix 1: 6.31 59.2 2.32 6.61 13.6 45.2 39.5 9.46 3.33 Matrix 2: 73.9 9.69 5.17 8.61 38.8 4.09 9.51 54.8 8.82	Print matrices Mult: 998.08 2485.24 295.21 1035.43 3068.69 488.46 3032.17 932.29 272.28 Transpose: 998.08 1035.43 3032.17 2485.24 3068.69 932.29 295.21 488.46 272.28 Row: 1259.51 1530.86 1412.24 Column: 1688.56 2162.07 351.98 Prompt to play again	Your first 3x3 matrix is: 6.31 59.20 2.32 6.61 13.60 45.20 39.50 9.46 3.33 Your second 3x3 matrix is: 73.90 9.69 5.17 8.61 38.80 4.09 9.51 54.80 8.82 You selected Matrix Multiplication. The results are: 998.08 2485.24 295.21 1035.43 3068.69 488.46 3032.17 932.29 272.28 The Transpose is: 998.08 1035.43 3032.17 2485.24 3068.69 932.29 295.21 488.46 272.28 The row and column mean values of the results are: Row: 1259.51 1530.86 1412.24 Column: 1688.56 2162.07 351.98	Yes

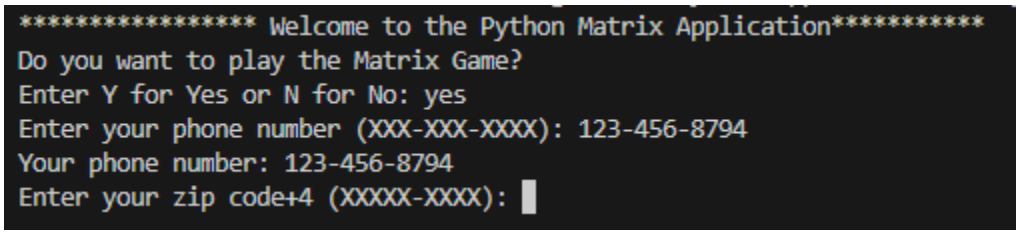
Test Case 9: Matrix Element Mult	Matrix 1: 2.68 8.7 37 5 5.27 5.9 9.7 0.688 1 Matrix 2: 35 40 3.04 6.02 5.9 7 1.99 3.1 18	Print matrices Multiplication: 93.80 348.00 112.48 30.10 31.09 41.30 19.30 2.13 18.00 Transpose: 93.80 30.10 19.30 348.00 31.09 2.13 112.48 41.30 18.00 Row: 184.76 34.16 13.15 Column: 47.73 127.08 57.26 Prompt to play again	Your first 3x3 matrix is: 2.68 8.70 37.00 5.00 5.27 5.90 9.70 0.69 1.00 Your second 3x3 matrix is: 35.00 40.00 3.04 6.02 5.90 7.00 1.99 3.10 18.00 You selected Element by element multiplication. The results are: 93.80 348.00 112.48 30.10 31.09 41.30 19.30 2.13 18.00 The Transpose is: 93.80 30.10 19.30 348.00 31.09 2.13 112.48 41.30 18.00 The row and column mean values of the results are: Row: 184.76 34.16 13.15 Column: 47.73 127.08 57.26	Yes

Test Case 10: Invalid Selection	Select a matrix operation: tangerine	Invalid, prompt again	Invalid choice. Please enter a valid option. Choice:	Y
Test Case 11: Invalid Y/N	Play? Plum	Invalid, prompt again	Invalid, please input 'yes' or 'no. Do you want to play the Matrix Game? Enter Y for Yes or N for No:	Y
Test Case 12: Valid Y/N continue	Play? Y	Prompt for phone number	Enter your phone number (XXX-XXX-XXXX):	Y

Test Case 1:



Test Case 2:



Test Case 3:

```
****************** 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): 123-456-8794
Your phone number: 123-456-8794
Enter your zip code+4 (XXXXX-XXXX): pear
Invalid, please input a phone number in the format XXX-XXX-XXXX
Enter your zip code+4 (XXXXX-XXXX):
```

Test Case 4:

```
****************** 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): 123-456-8794
Your phone number: 123-456-8794
Enter your zip code+4 (XXXXX-XXXX): 12345-1234
Your zip code: 12345-1234
Enter your first 3x3 matrix:
```

Test Case 5:

```
Enter your first 3x3 matrix:
apple pear tangerine
Invalid input.Please input a matrix using only integers and floats in the following format:
  X X X
  X X X
  X X X
Enter your first 3x3 matrix:
```

Test Case 6:

```
Enter your first 3x3 matrix:
4 7 6
2 3 7
9 5 2
Your first 3x3 matrix is:
    4.00    7.00    6.00
    2.00    3.00    7.00
    9.00    5.00    2.00
Enter your second 3x3 matrix:
1 5 2
3 2 3
9 6 9
Your second 3x3 matrix is:
    1.00    5.00    2.00
    3.00    2.00    3.00
    9.00    6.00    9.00
Select a Matrix Operation from the list below:
a. Addition
b. Subtraction
c. Matrix Multiplication
d. Element by element multiplication
Choice: a
You selected Addition. The results are:
    5.00   12.00    8.00
    5.00    5.00   10.00
   18.00   11.00   11.00
The Transpose is:
    5.00    5.00   18.00
   12.00    5.00   11.00
    8.00   10.00   11.00
The row and column mean values of the results are:
Row:        8.33    6.67   13.33
Column:     9.33    9.33    9.67
Do you want to play the Matrix Game?
Enter Y for Yes or N for No:
```

Test Case 7:

```
Enter your first 3x3 matrix:
3.4 2.9 4.4
0.6 8.6 9.9
5.1 6.2 6.1
Your first 3x3 matrix is:
    3.40    2.90    4.40
    0.60    8.60    9.90
    5.10    6.20    6.10
Enter your second 3x3 matrix:
1.5
Invalid input. Please input a matrix using only inte
  X X X
  X X X
  X X X
Enter your second 3x3 matrix:
1.5 6.5 6.3
3.5 4.1 5.1
1.4 7.8 2.5
Your second 3x3 matrix is:
    1.50    6.50    6.30
    3.50    4.10    5.10
    1.40    7.80    2.50
Select a Matrix Operation from the list below:
a. Addition
b. Subtraction
c. Matrix Multiplication
d. Element by element multiplication
Choice: b
You selected Subtraction. The results are:
    1.90   -3.60   -1.90
   -2.90    4.50    4.80
    3.70   -1.60    3.60
The Transpose is:
    1.90   -2.90    3.70
   -3.60    4.50   -1.60
   -1.90    4.80    3.60
The row and column mean values of the results are:
Row:       -1.20    2.13    1.90
Column:     0.90   -0.23    2.17
Do you want to play the Matrix Game?
Enter Y for Yes or N for No:
```

Test Case 8:

```
Enter your first 3x3 matrix:
6.31 59.2 2.32
6.61 13.6 45.2
39.5 9.46 3.33
Your first 3x3 matrix is:
    6.31   59.20    2.32
    6.61   13.60   45.20
   39.50    9.46    3.33
Enter your second 3x3 matrix:
73.9 9.69 5.17
8.61 38.8 4.09
9.51 54.8 8.82
Your second 3x3 matrix is:
   73.90    9.69    5.17
    8.61   38.80    4.09
    9.51   54.80    8.82
Select a Matrix Operation from the list below:
a. Addition
b. Subtraction
c. Matrix Multiplication
d. Element by element multiplication
Choice: c
You selected Matrix Multiplication. The results are:
  998.08 2485.24  295.21
 1035.43 3068.69  488.46
 3032.17  932.29  272.28
The Transpose is:
  998.08 1035.43 3032.17
 2485.24 3068.69  932.29
  295.21  488.46  272.28
The row and column mean values of the results are:
Row:     1259.51 1530.86 1412.24
Column:  1688.56 2162.07  351.98
Do you want to play the Matrix Game?
```

Test Case 9:

```
Enter your first 3x3 matrix:
2.68 8.7 37
5 5.27 5.9
9.7 0.688 1
Your first 3x3 matrix is:
    2.68    8.70   37.00
    5.00    5.27    5.90
    9.70    0.69    1.00
Enter your second 3x3 matrix:
35 40 3.04
6.02 5.9 7
1.99 3.1 18
Your second 3x3 matrix is:
   35.00   40.00    3.04
    6.02    5.90    7.00
    1.99    3.10   18.00
Select a Matrix Operation from the list below:
a. Addition
b. Subtraction
c. Matrix Multiplication
d. Element by element multiplication
Choice: d
You selected Element by element multiplication. The results are:
   93.80  348.00  112.48
   30.10   31.09   41.30
   19.30    2.13   18.00
The Transpose is:
   93.80   30.10   19.30
  348.00   31.09    2.13
  112.48   41.30   18.00
The row and column mean values of the results are:
Row:      184.76   34.16   13.15
Column:    47.73  127.08   57.26
Do you want to play the Matrix Game?
Enter Y for Yes or N for No:
```

Test Case 10:

```
Select a Matrix Operation from the list below:
a. Addition
b. Subtraction
c. Matrix Multiplication
d. Element by element multiplication
Choice: tangerine
Invalid choice. Please enter a valid option.
Choice:
```

Test Case 11:

```
Do you want to play the Matrix Game?
Enter Y for Yes or N for No: plum
Invalid, please input 'yes' or 'no.
Do you want to play the Matrix Game?
Enter Y for Yes or N for No:
```

Test Case 12:

```
Do you want to play the Matrix Game?
Enter Y for Yes or N for No: Y
Enter your phone number (XXX-XXX-XXXX):
```

Pylint

```
sdev_300 > lab_4 > pylint_matrix_app.txt
1
2   ------------------------------------------------------------------
3   Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)
4
```

I didn't run into any major Pylint issues with this exercise. The main trouble I had was with figuring out the order/logic for formatting the matrices to print them out with the amount of decimal places and padding I wanted. As I worked through that, Pylint pointed out that I could use an f string instead of formatting a string after the fact. This helped guide me into the final solution.

Password Exercise

Password	**Hash Output**	**Cracked?**
password0	MD5: 305e4f55ce823e111a46a9d500bcb86c SHA256: 1f489582f7ea4c208b70219a2bb6a322227a7516630530a10ed7f2710cfbe447 SHA512: d68e7bc1c722bca2b82c3e525dfbd35465f05cae712308d450a647595fe99778f256001d4751e265dd1a1a4db3e9fbacac1897e7834cffe87c3e1461c6824e94	Yes

password1	MD5: 7c6a180b36896a0a8c02787eeafb0e4c SHA256: 0b14d501a594442a01c6859541bcb3e8164d183d32937b851835442f69d5c94e SHA512: bc547750b92797f955b36112cc9bdd5cddf7d0862151d03a167ada8995aa24a9ad24610b36a68bc02da24141ee51670aea13ed6469099a4453f335cb239db5da	Yes
password2	MD5: 6cb75f652a9b52798eb6cf2201057c73 SHA256: 6cf615d5bcaac778352a8f1f3360d23f02f34ec182e259897fd6ce485d7870d4 SHA512: 92a891f888e79d1c2e8b82663c0f37cc6d61466c508ec62b8132588afe354712b20bb75429aa20aa3ab7cfcc58836c734306b43efd368080a2250831bf7f363f	Yes
password3	MD5: 819b0643d6b89dc9b579fdfc9094f28e SHA256: 5906ac361a137e2d286465cd6588ebb5ac3f5ae955001100bc41577c3d751764 SHA512: 2a64d6563d9729493f91bf5b143365c0a7bec4025e1fb0ae67e307a0c3bed1c28cfb259fc6be768ab0a962b1e2c9527c5f21a1090a9b9b2d956487eb97ad4209	Yes

password4	MD5: 34cc93ece0ba9e3f6f235d4af979b16c SHA256: b97873a40f73abedd8d685a7cd5e5f85e4a9cfb83eac26886640a0813850122b SHA512: 11961811bd4e11f23648afbd2d5c251d2784827147f1731be010adaf0ab38ae18e5567c6fd1bee50a4cd35fb544b3c594e7d677efa7ca01c2a2cb47f8fb12b17	Yes
SDEV 300 Building Secure Python Applicationspassword0	MD5: a5f296d3ffb39afb88d66be32b711b7c SHA256: a4d6e05bb575cfc96d13be337c20962e9d75dde2a8271f5439fd58339db9a8cc SHA512: 9b89817baece493621e39569122f4aa8ee393b4fa2e4c1071b3d089df063900368511d945898e8d7eafc6751cf551b2bef6d86575c71752ce49e056b50aeb16d	No
SDEV 300 Building Secure Python Applicationspassword1	MD5: bd58db5bd4062a70e9349e16af7f5a8f SHA256: 4a49193dcc87e27aa6f16288ff27a172a70b6f44cac09fa9735d01dc07725464 SHA512: 4acafbab9f42c0e306d79a18b8c97ca0f4d62304e3b75169297e2d33f4efefca513adceb7fe68d9e9e4daa71f1ca8df404c8470e3e4e8074d7b032cfd859ce4b	No

SDEV 300 Building Secure Python Applicationspassword2	MD5: 0262b28aa50e457d70bacd6b94964d67 SHA256: 9c6dc4fd63c6efd2a18cd94c0a2d60fc5760092a12b4060c714027a397c0de1b SHA512: 80a6b834856b20c452e990473e0e762e71c4d90d7acac94525c0b7dc741c15827deccb694e85238a925751b0aeb52a9cd0543fb822135863b87caba654273872	No
SDEV 300 Building Secure Python Applicationspassword3	MD5: f5685f3bbbc8a086eaaf6c1c83d8ab51 SHA256: bff1bf711c9b4f277771f6d15d290de04b7f870c7f91dfe1cc95d1e613531aa5 SHA512: 5519462e880c41a8b8e6542b96023302de3d1e3a8007692f6761b70aaaea45499821ef4d1cb0705f244a7a0ac05ada08a0e74d7c1591ee53c159c57574cc2887	No
SDEV 300 Building Secure Python Applicationspassword4	MD5: 845a35fde2542ca0771390bbfd48da4f SHA256: e26c3dbc23f04ef086da5340277ee3af326cc042550ff2413b29428fc4daf9df SHA512: 125a8167819608c8ee094581ba1feb183865da17165f250ff6a178a066d48cc19e990d0e7289aaa127a70cbbbb6554564a10c62d891404c06b203d499440b0cd	No

c<v6Vs'NfF	MD5: d884d2f35e69fc4bf08e2eacb7c64054 SHA256: bc15070a95fe377431967671dbeea2f2d306c479728a56ffa06a3241fe17c1a7 SHA512: e0bd297cf4e1a804667bc2cd661139e2e41796ace04e1dc8cfc0de47e7e3a2805c9008b0b8458dbc126b74a8dddb6e7c0902f51ffe966cf166684d013a0f47b7	No
|*lCYS#yfo!2Ggc	MD5: 1c53d9f014bedd4ccdf1382b75aada10 SHA256: 867129f755aae4fee7944f6bdb346d53b95099f9e6b2842d59cb4516cf920213 SHA512: 52304291d399381988e268acd62d07ec4b1bdd7a8bc1bbd2fbb3e0c83258750839b20aee7381977010e9ded20e51d2a07bbf3b39a553809092b4bfdf8870a64e	No
y.Oiz4w/5LWnY?0uP?ZK	MD5: 17b7aff43ae73e51dc96a23d87f9a9aa SHA256: d07f6d04a0ffb9d9d0b91088b7a69dc756b674dd477e269a18db33c038f0ace9 SHA512: 244e3b6879f9bcdda5839755d87ad53cc9b4bb4069f558f2ff846bb70cccbf88aa286d228fd1216e9fac8c345e3ad120fa54cd2cecc55fc183433b24f977b0ad	No

Fw=qliYQ10u@"hx_->5T-g NSX	MD5: 4b5aab614583766a4189dfcc186c9adf SHA256: ecaafd5de712aa6dad972181a891e6d5f9d3de26d8e6a0b50f55c93555575452 SHA512: 4b600be93013c4efe19f4cf24030bec1df281f0ee44e764deb3e276148e57e8b6822b82948ee56cb5d17b0021ac9f734bc269339eaa740f755c936af957c8de6	No
?s,oq"hce\oC?R<aiXf\v#46D Aa`+G	MD5: 1c4b1a32b6509f347c8c7226f96b5650 SHA256: a3399f83c0b5bb9828bb467d22c3cf7652bac4569414adf27f9b08f506c8bca9 SHA512: 1c9f1508f015f8e9d59aafcb246948a405b6b3bdd1f336b5fc4b4c535f1ee6a95164558e8d17d68697afd8b549ca9ae72101bff5b84aff2bf800c71d6bf34206	No
SDEV 300 Building Secure Python Applications9W\&Ag~k[Q	MD5: 6cd7e583e357f9ee3c9cec3ace34a9e9 SHA256: a30f8f8a1795ab3b776dc6d66186d77510beb432551c2876a62110ab5f0067f4 SHA512: d4f9d86015f964c35ac4cb6e823539b0ffabfc492efcb009330bd772e9cae961aebe3386c817e4bb351a75e1548d4fa23c7a7feb7176083c4afc2ee6beb0c312	No

SDEV 300 Building Secure Python Applicationsa"kS4b2@`?S|Ux8	MD5: 7894f5c33a761c07e067e27d81943625 SHA256: 187a7622360ff1f20878d83530651f9cfb966109b9484eb360c6314dc1d7878c SHA512: cb7dad1d99707bb1e8604313c352d4a7a7c8d02f3a6c56f192d25b6c3f74d59c4ec01fc0022596103243904afff629f14dfee3adf80d80a3e634f5eca442df35	No
SDEV 300 Building Secure Python Applications's&=_NQ4Gz2><58)f52V	MD5: dfc4976a6fde8815ce01d76536ad55de SHA256: 78f9fda59f3be8ad627eaeb75b0d297f3d3f54d73cf9e01b7047bf113607600c SHA512: 1df65378feef910310f0722b85659c3041cd60aee1372dbb69f5649a5f17d958647f2d68865f7b4cdb13b7d78af34753f9123b21785af0d725c8f5ae8671830f	No
SDEV 300 Building Secure Python ApplicationsXmInPb;[@fqP*g-]p)l7]oZ\y	MD5: 596738cd106256d3634fac3d55827409 SHA256: fca320c0bf73d6d495c9cfe617bed1808b087e2cdd311aeb0cab72814ec998aa SHA512: 19a53d0ef797528380113b6631f90799678318850ca01a8ab186ff0002c7ffd2f824a611250481e2e58e1085abad684cd07e38a4fae5eaccb27400209406deb1	No

SDEV 300 Building Secure Python Applications>is$slo?u[}Vr(& "sh1vpJg)~IW7tC	MD5: d2df1afcb4ba622baef24919eb93016d SHA256: 17f2b1acc3c597a7730645865eb958a450d4069e65d0959327709a500209e3f8 SHA512: 77d65c3768ef50c67973d489389e43c97b7987f9715ba3f87be83a6a981f4ddee08b09062d9e587bff3d81b4606f45eb782f4872ab50718334b207319bd49df4	No

Reflection

The trend that stood out to me was that the different hashing algorithms did not have any impact on whether a password was cracked or not. The easy passwords were all cracked across all three hashing algorithms. Moving on to salting passwords and more complex passwords led to a complete failure to crack even the simpler MD5 hash. As a user, this activity has convinced me that it is important to generate sufficiently secure passwords. A user will never know what hashing algorithm is used (if any). In the same vein, a user will never know if a salt is used to help secure their password. The best way to ensure security is through a complex password. As a developer, I think I need to experiment more with the various hashing algorithms. After this activity, I am not yet clear on the pros and cons of each. As I build new applications that require passwords, I now know about salting and how useful it is with making even the simplest of passwords more difficult to crack.