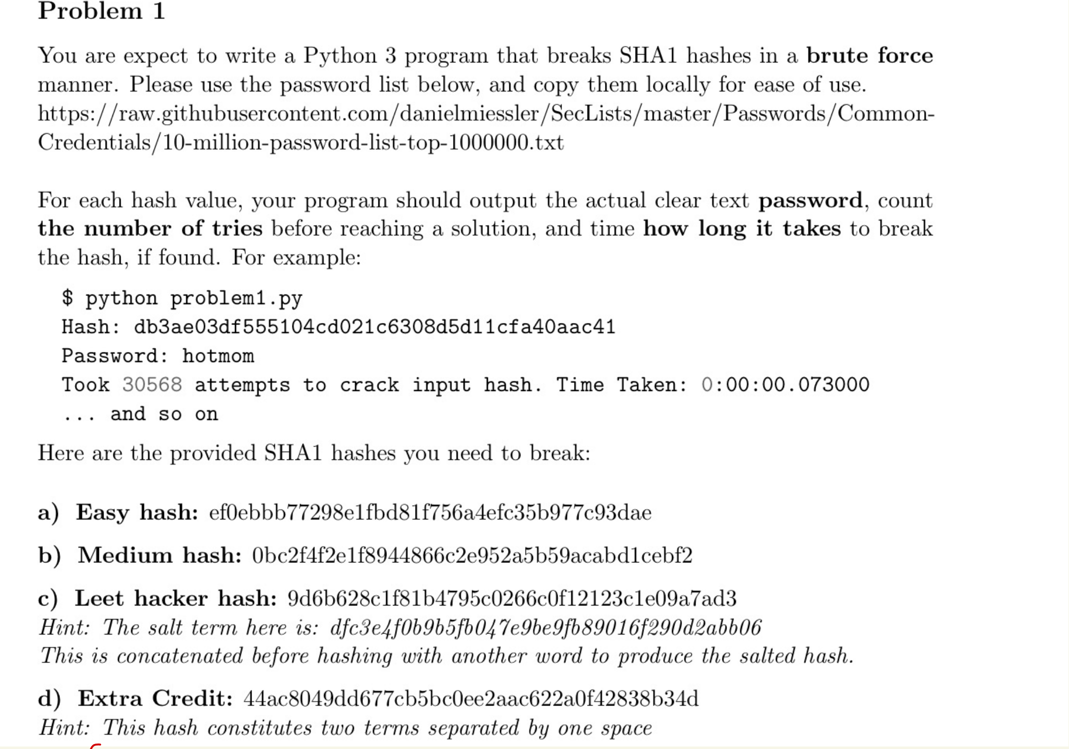


Problem 1

You are expect to write a Python 3 program that breaks SHA1 hashes in a **brute force** manner. Please use the password list below, and copy them locally for ease of use. https://raw.githubusercontent.com/danielmiessler/SecLists/master/Passwords/Common-Credentials/10-million-password-list-top-1000000.txt

For each hash value, your program should output the actual clear text **password**, count **the number of tries** before reaching a solution, and time **how long it takes** to break the hash, if found. For example:

```
$ python problem1.py
Hash: db3ae03df555104cd021c6308d5d11cfa40aac41
Password: hotmom
Took 30568 attempts to crack input hash. Time Taken: 0:00:00.073000
... and so on
```

Here are the provided SHA1 hashes you need to break:

a) Easy hash: ef0ebbb77298e1fbd81f756a4efc35b977c93dae

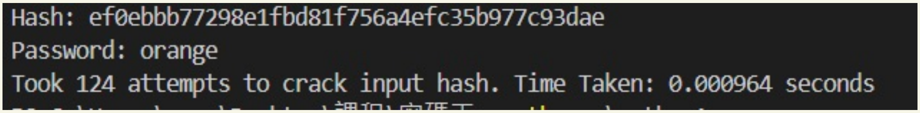
b) Medium hash: 0bc2f4f2e1f8944866c2e952a5b59acabd1cebf2

c) Leet hacker hash: 9d6b628c1f81b4795c0266c0f12123c1e09a7ad3
Hint: The salt term here is: dfc3e4f0b9b5fb047e9be9fb89016f290d2abb06
This is concatenated before hashing with another word to produce the salted hash.

d) Extra Credit: 44ac8049dd677cb5bc0ee2aac622a0f42838b34d
Hint: This hash constitutes two terms separated by one space

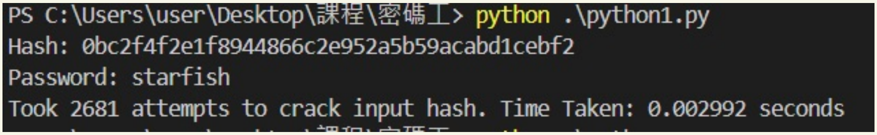
在執行code前要打
pip install requests

(a) orange



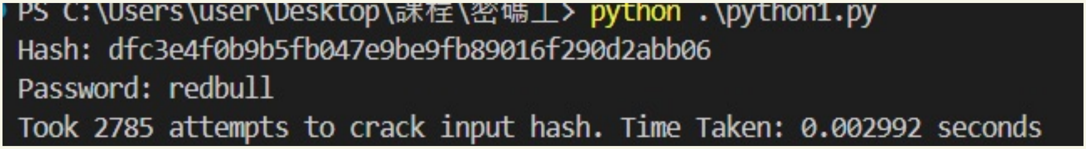
```
Hash: ef0ebbb77298e1fbd81f756a4efc35b977c93dae
Password: orange
Took 124 attempts to crack input hash. Time Taken: 0.000964 seconds
```

(b) starfish



```
PS C:\Users\user\Desktop\課程\密碼工> python .\python1.py
Hash: 0bc2f4f2e1f8944866c2e952a5b59acabd1cebf2
Password: starfish
Took 2681 attempts to crack input hash. Time Taken: 0.002992 seconds
```

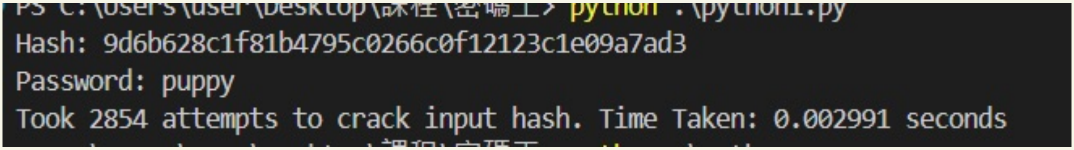
(c)



```
PS C:\Users\user\Desktop\課程\密碼工> python .\python1.py
Hash: dfc3e4f0b9b5fb047e9be9fb89016f290d2abb06
Password: redbull
Took 2785 attempts to crack input hash. Time Taken: 0.002992 seconds
```

先找到 redbull，在每個 string 前
+‘redbull’再去找 得到“puppy

☆做法在註解

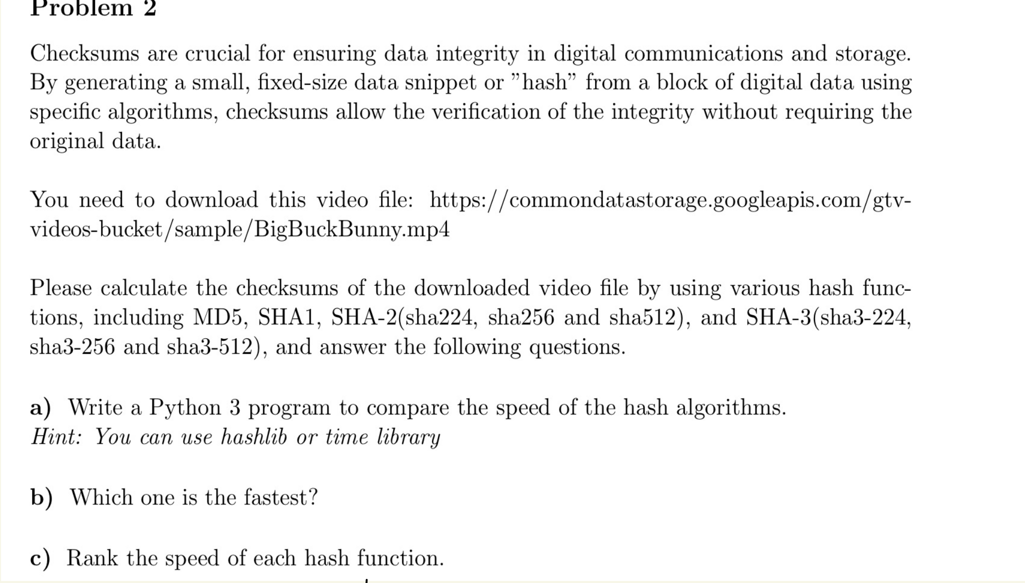


```
PS C:\Users\user\Desktop\課程\密碼工> python .\python1.py
Hash: 9d6b628c1f81b4795c0266c0f12123c1e09a7ad3
Password: puppy
Took 2854 attempts to crack input hash. Time Taken: 0.002991 seconds
```

故 password 為 redbullpuppy

(d)

X



Problem 2

Checksums are crucial for ensuring data integrity in digital communications and storage. By generating a small, fixed-size data snippet or ”hash” from a block of digital data using specific algorithms, checksums allow the verification of the integrity without requiring the original data.

You need to download this video file: https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/BigBuckBunny.mp4

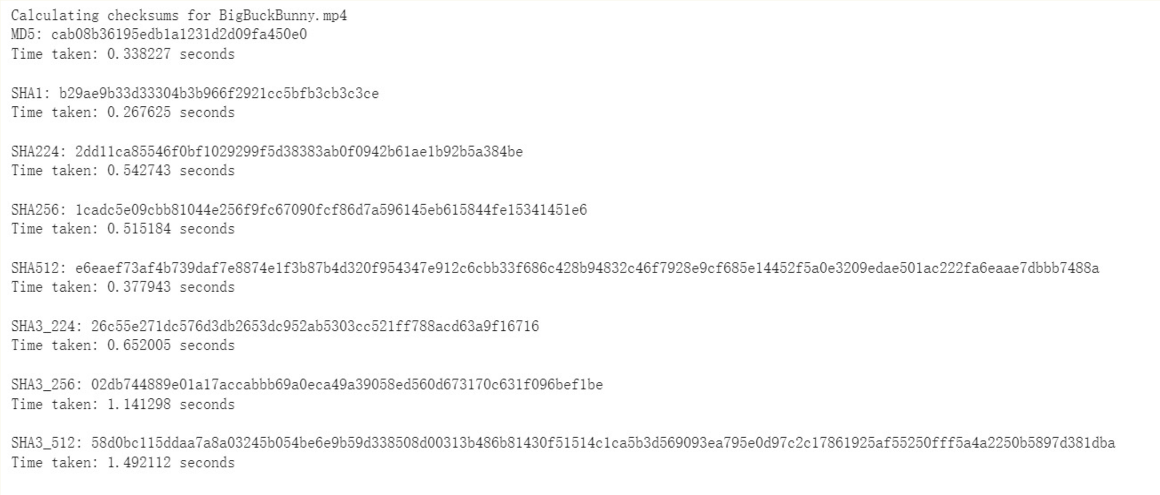
Please calculate the checksums of the downloaded video file by using various hash functions, including MD5, SHA1, SHA-2(sha224, sha256 and sha512), and SHA-3(sha3-224, sha3-256 and sha3-512), and answer the following questions.

a) Write a Python 3 program to compare the speed of the hash algorithms.
Hint: You can use hashlib or time library

b) Which one is the fastest?

c) Rank the speed of each hash function.

(b) SHA1 最快



```
Calculating checksums for BigBuckBunny.mp4
MD5: cab08b36195edb1a1231d2d09fa450e0
Time taken: 0.338227 seconds

SHA1: b29ae9b33d33304b3b966f2921cc5bfb3cb3c3ce
Time taken: 0.267625 seconds

SHA224: 2dd11ca85546f0bf1029299f5d38383ab0f0942b61ae1b92b5a384be
Time taken: 0.542743 seconds

SHA256: 1cadc5e09cbb81044e256f9fc67090fcf86d7a596145eb615844fe15341451e6
Time taken: 0.515184 seconds

SHA512: e6eaef73af4b739daf7e8874e1f3b87b4d320f954347e912c6cbb33f686c428b94832c46f7928e9cf685e14452f5a0e3209edae501ac222fa6eaae7dbbb7488a
Time taken: 0.377943 seconds

SHA3_224: 26c55e271dc576d3db2653dc952ab5303cc521ff788acd63a9f16716
Time taken: 0.652005 seconds

SHA3_256: 02db744889e01a17accabbb69a0eca49a39058ed560d673170c631f096bef1be
Time taken: 1.141298 seconds

SHA3_512: 58d0bc115ddaa7a8a03245b054be6e9b59d338508d00313b486b81430f51514c1ca5b3d569093ea795e0d97c2c17861925af55250fff5a4a2250b5897d381dba
Time taken: 1.492112 seconds
```

(c)

由快至慢排序

SHA1, MD5, SHA512, SHA256, SHA224, SHA3_224, SHA3_256, SHA3_512

Problem 3

Given the transposition cipher:

UONCS VAIHG EPAAH IGIRL BIECS

TECSW PNITE TIENO IEEFD OWECX

TRSRX STTAR TLODY FSOVN EOECO

HENIO DAARQ NAELA FSGNO PTE

Please decrypt this ciphertext.

Hint: How to determine the dimension of the rectangle?

1) Vowel Frequencies can help us to determine the dimensions of the rectangle. In English, approximately **40%** of plaintext consists of vowels. Therefore, for the correct dimension, each row of the rectangle should be approximately 40% vowels.

2) For example: ”ASAIR ITFNM IMTKL SOIEE M”. There are 21 letters.
Because we know that the message completely fills the rectangle, this suggests either a 3X7 or a 7X3 rectangle.
Consider our choice between 3X7 and 7X3 as an example. For a 3X7 rectangle, each row should contain approximately 2.8 vowels.
Let us note the difference between this estimate and the actual count.

For a 3X7 rectangle:

							Number of vowels	Difference
A	I	T	M	T	S	E	3	0.2
S	R	F	I	K	O	E	3	0.2
A	I	N	M	L	I	M	3	0.2

The average difference of each row is 0.2.

For a 7X3 rectangle:

			Number of vowels	Difference
A	F	L	1	0.2
S	N	S	0	1.2
A	M	O	2	0.8
I	I	I	3	1.8
R	M	E	1	0.2
I	T	E	2	0.8
T	K	M	0	1.2

The average difference of each row is 0.88.

So in this case, 3X7 rectangle is more likely.

用手攵寫算後

1x98 the avg differelce is 2.2

2x49 . . . . 1.5

7x14 . . . . 0.657

14x7 . . . 0.557

49x2 . . . . 0.551

98x1 . . . . 0.496

故用 14x7 來試 try y tryse

经由湊和一些網路上的網站得出了明文

T	H	E	Q	U	E	S
T	I	O	N	O	F	W
A	G	E	A	N	D	P
R	I	C	E	C	O	N
T	R	O	L	S	W	I
L	L	H	A	V	E	T
O	B	E	F	A	C	E
D	I	N	S	I	X	T
Y	E	I	G	H	T	I
F	C	O	N	G	R	E
S	S	D	O	E	S	N
O	T	A	P	P	R	O
V	E	A	T	A	X	I
N	C	R	E	A	S	E