Assignment 3

Avdhesh Kumar

U19ME191

1 Verify the integrity of a software

[Download Visual Studio Code - Mac, Linux, Windows](#) hash code from VS code site.

Linux .deb (64 bit)	5d14d85ee907045b0fdc00d02d5f0f2a4336d37747f52862ed64fd52f92dec5b

Hash generated using terminal in Ubuntu

```
akgbot@akgbot-VirtualBox:~/Downloads$ ls
code_1.60.2-1632313585_amd64.deb
akgbot@akgbot-VirtualBox:~/Downloads$ sha256sum code*
5d14d85ee907045b0fdc00d02d5f0f2a4336d37747f52862ed64fd52f92dec5b  code_1.60.2-16
32313585_amd64.deb
```

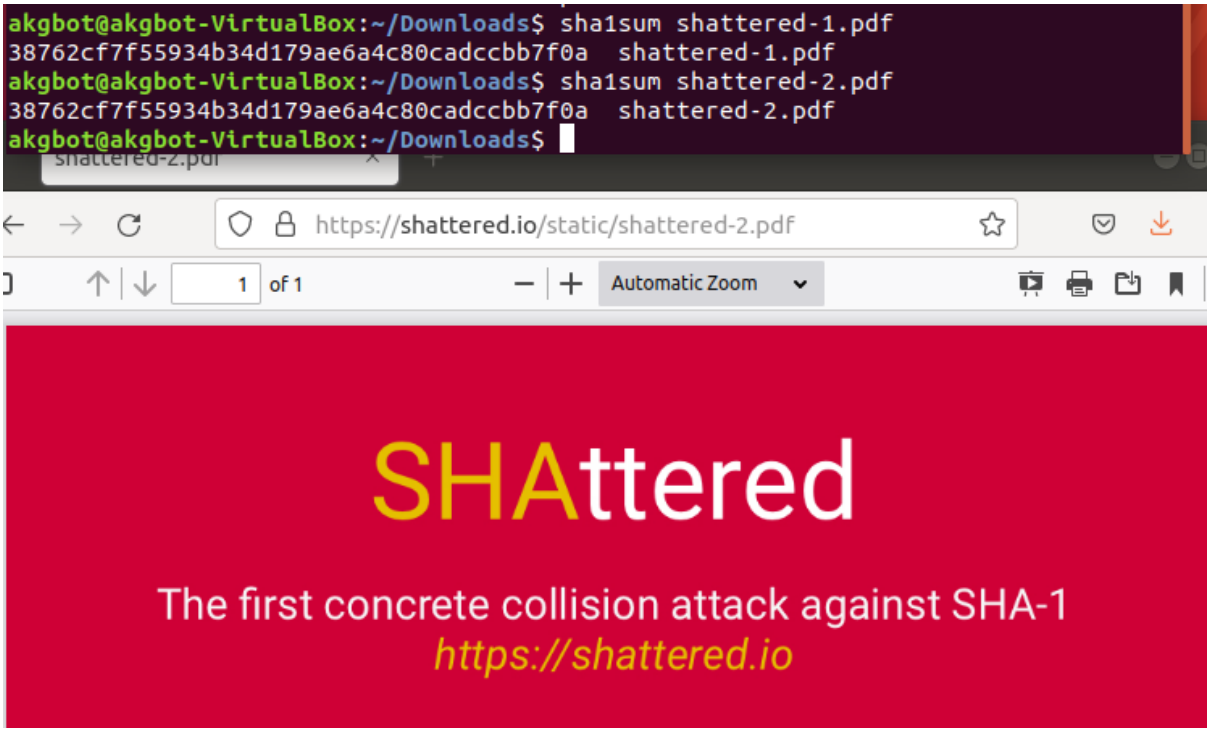
2Generate hashcodes of name using SHA-256 and SHA-512 algorithm

```
akgbot@akgbot-VirtualBox:~/Downloads$ echo "Avdhesh" > data.txt
akgbot@akgbot-VirtualBox:~/Downloads$ ls
 data.txt  'Linux .deb (64-bit ARM).deb'
akgbot@akgbot-VirtualBox:~/Downloads$ sha256sum data.txt
6635562d357785dfae3d9995d973d3015966d615dde0b3152b2662ab65cf48d4  data.txt
akgbot@akgbot-VirtualBox:~/Downloads$ sha215sum da*
sha215sum: command not found
akgbot@akgbot-VirtualBox:~/Downloads$ sha512sum da*
b90b52ecb03f2711505c6441ff9bdfb4267b695b1e2ed7e9f190b664528c22bc27c937bbc97a558c
f776b1d298a6930b0c2c47338874be2cc5ab9697af921691  data.txt
akgbot@akgbot-VirtualBox:~/Downloads$
```

3Using SHA-512, give an example of the avalanche effect using two different inputs

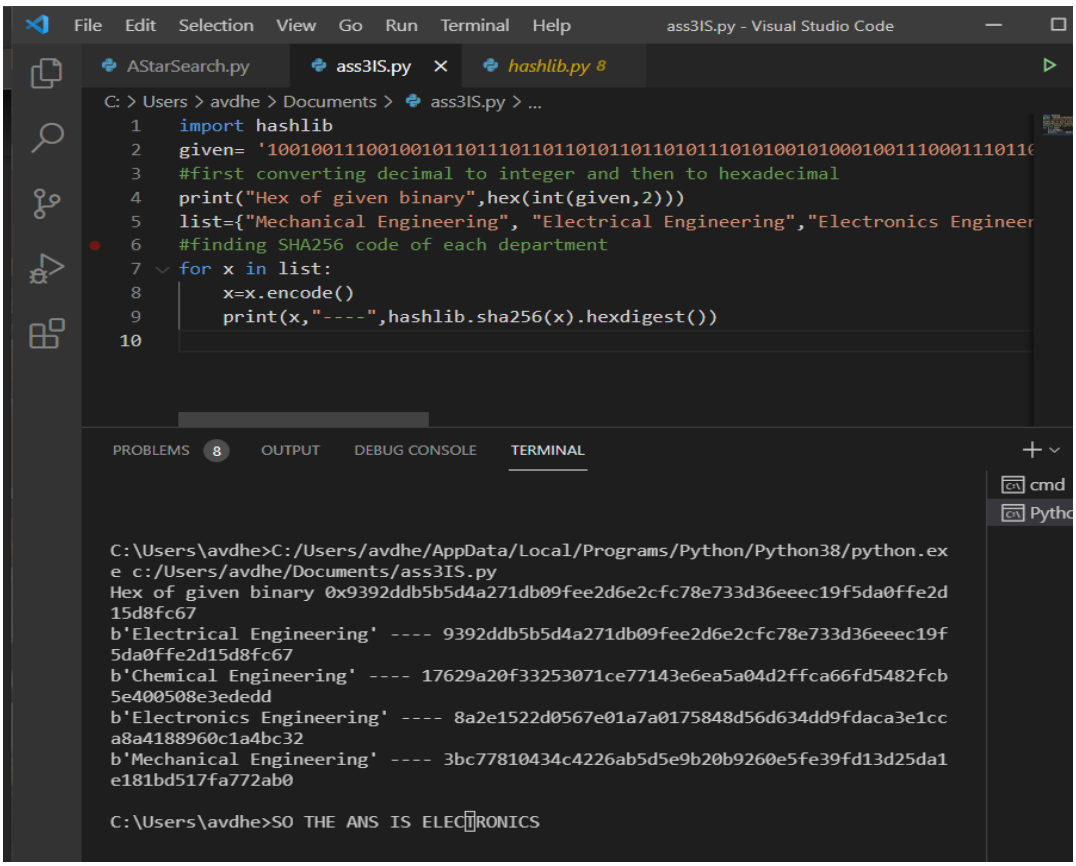
```
akgbot@akgbot-VirtualBox:~/Downloads$ echo "akg" > data1.txt
akgbot@akgbot-VirtualBox:~/Downloads$ echo "AKG" > data21.txt
akgbot@akgbot-VirtualBox:~/Downloads$ echo "AKG" > data2.txt
akgbot@akgbot-VirtualBox:~/Downloads$ ls
 data1.txt   data21.txt   data2.txt   data.txt  'Linux .deb (64-bit ARM).deb'
akgbot@akgbot-VirtualBox:~/Downloads$ sha512sum data1.txt
6dc94e1025312aca5039d61b4e88b8df94c03e060603d0a0ecc8665bc7d1766d2a1cc01b35dfc548
41c17006e8b09d13392795cf0f92e5c376b4ff2ed94d6588  data1.txt
akgbot@akgbot-VirtualBox:~/Downloads$ sha512 data2.txt

Command 'sha512' not found, but can be installed with:

sudo apt install hashalot

akgbot@akgbot-VirtualBox:~/Downloads$ sha512sum data2.txt
947a421864026311405ce6ee0968231c48d4ecd79d48e9bdb8b22f63f9f2327c5640f8c9195f1d2e
b9e7f67e0b2a2634f492482b52db51d5902151c22df4127e  data2.txt
```

4Verify the collision attack on SHA-1 (160-bit). Write the steps/commands/outputs used to verify the same.



Both the hashes are same.

5 Find the (svnit) department name whose hash code generated using SHA256 is as follows.



6 Find the "Input" (or message) whose last 20 bits of hashcode (SHA256) are as follows. [Note: Second Preimage/Weak Collision Resistance Assignment]

01010101010101010101

```python
import hashlib
input='01010101010101010101'
#first converting decimal to integer and then to hexadecimal
HEX= hex(int(input,2))[2:]

for i in range(2**20):

    HASH= hashlib.sha256(str(i).encode()).hexdigest()
    if(HASH[-5:] == HEX):
            print(i)
            break
```

```
888311

C:\Users\avdhe>C:/Users/avdhe/AppData/Local/Programs/Python/Python38/python.ex
e "c:/Users/avdhe/Downloads/A_Search/A_Search/code/ASS3 is q6.py"
888311
```

7Find two "Inputs" (or messages) whose last 20 bits of hashcode (SHA256) are same. [Note: Strong Collision Resistance Assignment]

```python
c=0 # counter variable to count 2 nos
for i in range(2**20):

    HASH1= hashlib.sha256(str(i).encode()).hexdigest()

    for j in range(i+1,2**20):
        HASH2= hashlib.sha256(str(j).encode()).hexdigest()
        if(HASH1[-5:] == HASH2[-5:]):
            print(i,j)
            c=c+1
            break
        if(c==2):
            break
```

```
C:\Users\avdhe>C:/Users/avdhe/AppData/Local/Programs/Python/Python38/python.ex
e "c:/Users/avdhe/Downloads/A_Search/A_Search/code/ASS3 is q6.py"
0 567559
1 658079

C:\Users\avdhe>
```