CS 458 – Fall 2024

Introduction To Information Security

Assignment #4

1 SHA-256 programming

a)

```
PS C:\Users\lenin\iit_archive\cs\cs458\Assignment4> & C:/Users/lenin/AppData/Local/Programs/Python/Python312/python.exe c:/Users/lenin/iit_archive/cs/cs45
8/Assignment4/a.py
SHA-256 hash value of the name 'Shree Ramachandran Muthukumaran' in hexadecimal: 73d9a284482abdc1b7fe9c9eb19a730d21af809cf1ee85c6cf6f489ad73e6e76
PS C:\Users\lenin\iit_archive\cs\cs458\Assignment4>
```

b) Histogram Data:

1-bit count: 107, Frequency: 1

1-bit count: 109, Frequency: 1

1-bit count: 110, Frequency: 1

1-bit count: 111, Frequency: 3

1-bit count: 112, Frequency: 10

1-bit count: 113, Frequency: 10

1-bit count: 114, Frequency: 12

1-bit count: 115, Frequency: 15

1-bit count: 116, Frequency: 12

1-bit count: 117, Frequency: 23

1-bit count: 118, Frequency: 21

1-bit count: 119, Frequency: 21

1-bit count: 120, Frequency: 45

1-bit count: 121, Frequency: 31

1-bit count: 122, Frequency: 39

1-bit count: 123, Frequency: 37

1-bit count: 124, Frequency: 43

1-bit count: 125, Frequency: 34

1-bit count: 126, Frequency: 53

1-bit count: 127, Frequency: 55

1-bit count: 128, Frequency: 60

1-bit count: 129, Frequency: 45

1-bit count: 130, Frequency: 47

1-bit count: 131, Frequency: 50

1-bit count: 132, Frequency: 46

1-bit count: 133, Frequency: 44

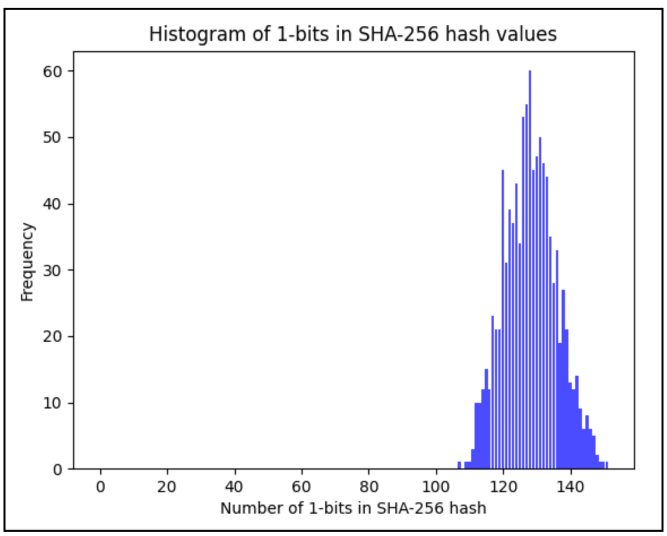
1-bit count: 134, Frequency: 35

1-bit count: 135, Frequency: 28

1-bit count: 136, Frequency: 33

1-bit count: 137, Frequency: 19
1-bit count: 138, Frequency: 27
1-bit count: 139, Frequency: 21
1-bit count: 140, Frequency: 13
1-bit count: 141, Frequency: 12
1-bit count: 142, Frequency: 14
1-bit count: 143, Frequency: 9
1-bit count: 144, Frequency: 6
1-bit count: 145, Frequency: 8
1-bit count: 146, Frequency: 6
1-bit count: 147, Frequency: 5
1-bit count: 148, Frequency: 2
1-bit count: 149, Frequency: 1
1-bit count: 150, Frequency: 1
1-bit count: 151, Frequency: 1

Histogram Plot:



c)

```
PS C:\Users\lenin\iit_archive\cs\cs458\Assignment4> & C:/Users/lenin/AppData/Local/Programs/Python/Python312/python.exe c:/Users/lenin/ii
t_archive/cs/cs458/Assignment4/c.py
Elapsed time for 1000 hashes: 0.004813 seconds
Hashes per second: 207762.23
Time to compute 2^128 hashes: 5.19e+25 years
Time to compute 2^256 hashes: 1.77e+64 years
PS C:\Users\lenin\iit_archive\cs\cs458\Assignment4>
```

d) Mathematica output of Binomial[256,128] / 2^{256}:

Input

$$\frac{\binom{256}{128}}{2^{256}}$$

$\binom{n}{m}$ is the binomial coefficient

Exact result

2 884 329 411 724 603 169 044 874 178 931 143 443 870 105 850 987 581 016 304 218
283 632 259 375 395 /
57 896 044 618 658 097 711 785 492 504 343 953 926 634 992 332 820 282 019 728
792 003 956 564 819 968

Decimal form

More digits

0.0498191099361401512382797171174812407196399526166936976843972104…

Mathematica output of Binomial[256,100] / 2^{256}:

Input

$$\frac{\binom{256}{100}}{2^{256}}$$

$\binom{n}{m}$ is the binomial coefficient

Exact result

192 233 475 654 278 015 300 682 501 747 228 291 604 991 135 720 640 682 342 516
624 794 462 315 /
1 809 251 394 333 065 553 493 296 640 760 748 560 207 343 510 400 633 813 116 524
750 123 642 650 624

Decimal form

More digits

0.0001062502846516473129959966572697976526356402651884393760247326…

Output of my python program:

```
PS C:\Users\lenin\iit archive\cs\cs458\Assignment4> & C:/Users/lenin/AppData/Local/Programs/Python/Python312/python.exe c:/Users/lenin/ii
t_archive/cs/cs458/Assignment4/d.py
Probability of exactly 128 bits set to one: 4.982e-02
Probability of exactly 100 bits set to one: 1.063e-04
```

2 RSA public-key cryptosystem

Bob's Public Key:

- $N = 143 = p \cdot q = 11 \cdot 13$
- $e = 7$.

Alice's Public Key:

- $N = 39 = p \cdot q = 3 \cdot 13$
- $e = 5$

Message: $M = 3$

<u>Bob Signs the Message:</u>

To sign the message, Bob uses his private key (d_{Bob}).

The private key d_{Bob} is computed using the relation: $d_{Bob} \cdot e_{Bob} \equiv 1 \pmod{\phi(N_{Bob})}$,

where $\phi(N_{Bob}) = (p - 1)(q - 1)$

For $N_{Bob} = 143$, $\phi(N_{Bob}) = (11 - 1)(13 - 1) = 120$

$d_{Bob} \cdot 7 \equiv 1 \pmod{120}$

$d_{Bob} = 103$.

Signature $S = M^d \pmod{N_{Bob}}$

$S = 3^{103} \pmod{143}$

$S = 16$

<u>Bob Encrypts the Signature:</u>

The encrypted value $E = S^{eAlice} \pmod{N_{Alice}}$

$E = 16^5 \pmod{39}$

$E = 22$

<u>Alice Decrypts the Encrypted Signature:</u>

Alice decrypts E using her private key (d_{Alice}). The private key is computed similarly:

$d_{Alice} \cdot e_{Alice} \equiv 1 \pmod{\phi(N_{Alice})}$

For $N_{Alice} = 39$, $\phi(N_{Alice}) = (3 - 1)(13 - 1) = 24$

$d_{Alice} \cdot 5 \equiv 1 \pmod{24}$

$d_{Alice} = 5$.

$S' = E^{dAlice} \pmod{N_{Alice}}$

$S' = 22^5 \pmod{39}$

$S' = 16$

<u>Alice Verifies the Signature:</u>

To verify the signature, Alice has to check if : $M = S'^{eBob} \pmod{N_{Bob}}$

$M = 16^7 \pmod{143}$

$M = 3$,

Since M is the expected value, we can conclude that the signature is valid, proving both the integrity of the message and Bob's identity as the signer.