Task 1: -

Explain by using examples how prime numbers are important within the field of computing. Give at least two applications/situations where we use prime numbers in the computer security field. Provide 350-450 words on a single page with references, you will not be penalized if you go beyond the word limit.

Prime number: It is a natural integer that is only divisible by 1 or by itself.

Composed number: It is a natural integer that is divisible by two or more prime numbers.

Example for prime number: - {1, 2, 3, 5, 7 ,11, ...}

Encryption: It is a way to hide data during its transmission from one person to another to protect it so that the authorized person can understand the information.

Decryption: It is a way to decode the encrypted data and return it to its original form, and the person who has the authority to obtain the information can do this process so that he has the decryption key.

RSA (Rivest Shamir Adleman): It is an encryption algorithm that is used to transfer data securely over the Internet. It is an asymmetric encryption technique that uses two different public and private keys to perform the encryption and decryption operations. Thus, using this technology, you can encrypt sensitive and important information using the public key and using the private key for decryption, such as between the client and VPN server, SSH, etc.

Prime numbers are very important and are used in encryption in the field of computer security. There is an algorithm called RSA to take advantage of the analysis of prime factors, which are difficult to calculate, so this algorithm is used for large prime numbers that take a long time to complete the analysis into factors. This algorithm is unique and very simple as we calculate the product of multiplying two very large numbers is easy to do. The strength of this algorithm depends on the difficulty of parsing large numbers - specifically the difficulty associated with finding a particular pair of chosen primes to create a large integer.

Diffie-Hellman: It is a key exchange algorithm that enables two parties to use a public key for encryption and decryption over a public communication channel. This algorithm creates a secure communication channel between the two parties and uses this channel to exchange the private key and then uses this private key to perform the symmetric encryption process between the two parties. This algorithm is one of the oldest applications of asymmetric keys.

For example, two people want to send important data while maintaining its confidentiality over an open network, so it becomes difficult to eavesdrop on the message with the presence of many resources within the network and each person has his own key across multiple public connections, so the shared secret is the same every time, so one of the parties needs to create A public and private key each time in order to create a new shared secret.

Example use prime numbers in the computer security field:

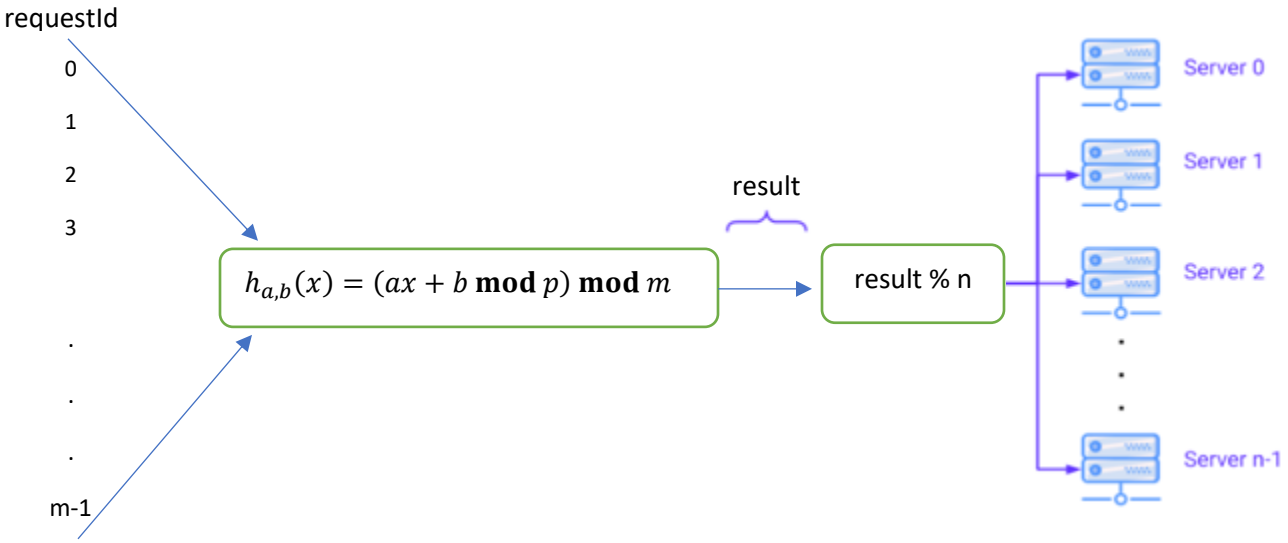
1- Data transfer via emails: When sensitive data is transferred from one person to another, the prime numbers are used in order to encrypt the data and prevent anyone who does not have access permission from obtaining this data.
2- Login to the Internet: When you connect to a network, you need the password in order to get access to the network, which provides a secure and encrypted connection that no one who does not know the password can connect to the network.

Task2: -

Evaluate probability theory to an example involving hashing and load balancing.

For your evaluation:

a) Draw a simple diagram to illustrate the load balancing using hashing process.



b) Assume the number of servers is n.

Where:

n = 25 + your last two digits of your university ID modulo 25

c) Define a Hash function based on Truncation only.

d) Use the client IP address or the request URL as the input of your hash function.

e) To evaluate your Hash function, study the distribution Z of client's requests across the n servers:

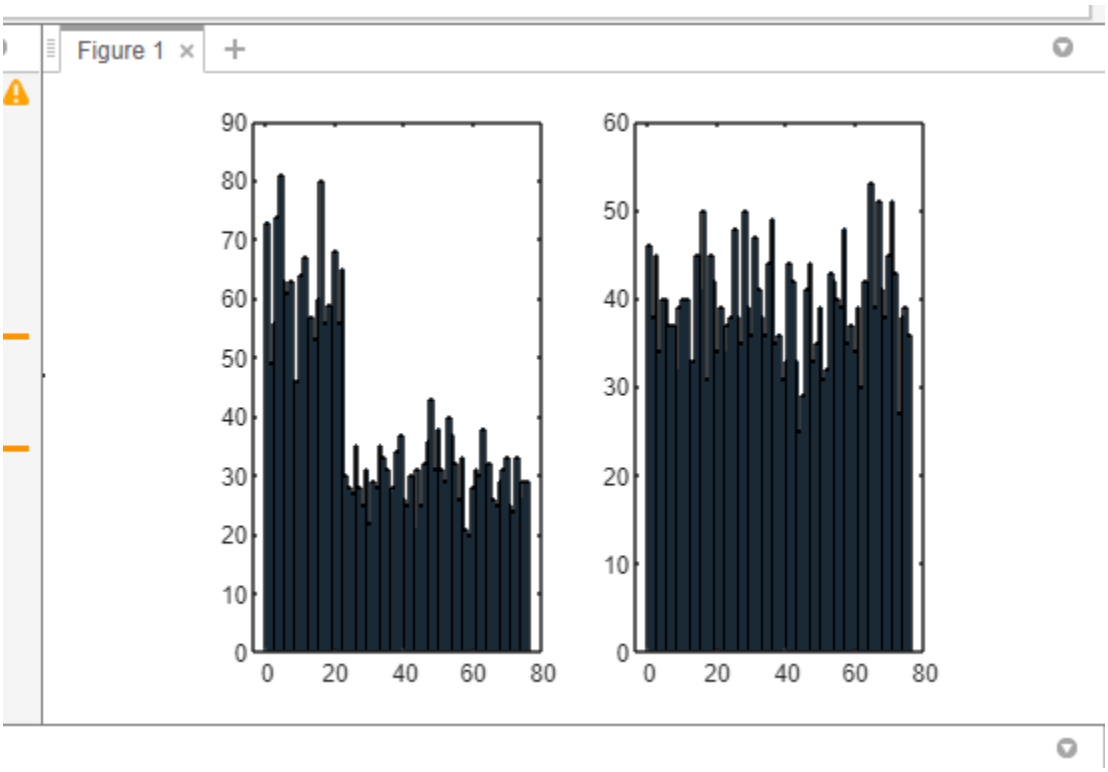
1. Simulate a sample of at least 3000 outcomes for Z.
2. Plot a histogram showing the approximate distribution of Z.
3. Calculate the mean and the standard deviation of the sample.

f) Define a Hash function based on Modular Arithmetic.

g) Repeat parts d and e for the hash function defined in f.

```
clc
clear
n = 77;
reqID = randi([100000000 999999999],1,3000);
for i =1:3000
    x=num2str(reqID(i));
    y =[x(7),x(9)];
    HashValues(i) = str2num(y);
end
HashValues =mod(HashValues,n);
subplot(121); histogram(HashValues,n);
p = 999998727899999 % prime number
a =100;
b =10;

hash_f = mod(a.*reqID+b,p); % Hash function based on modular Arithmetic
hash_f = mod(hash_f,n);
subplot(122); histogram(hash_f,n);
mean(hash_f)
std(hash_f)
```

Result: -



```
p =

   1.0000e+15


ans =

   38.0420


ans =

   22.3892

>>
```

h) Compare the results you found for the hash functions defined in parts c and f.

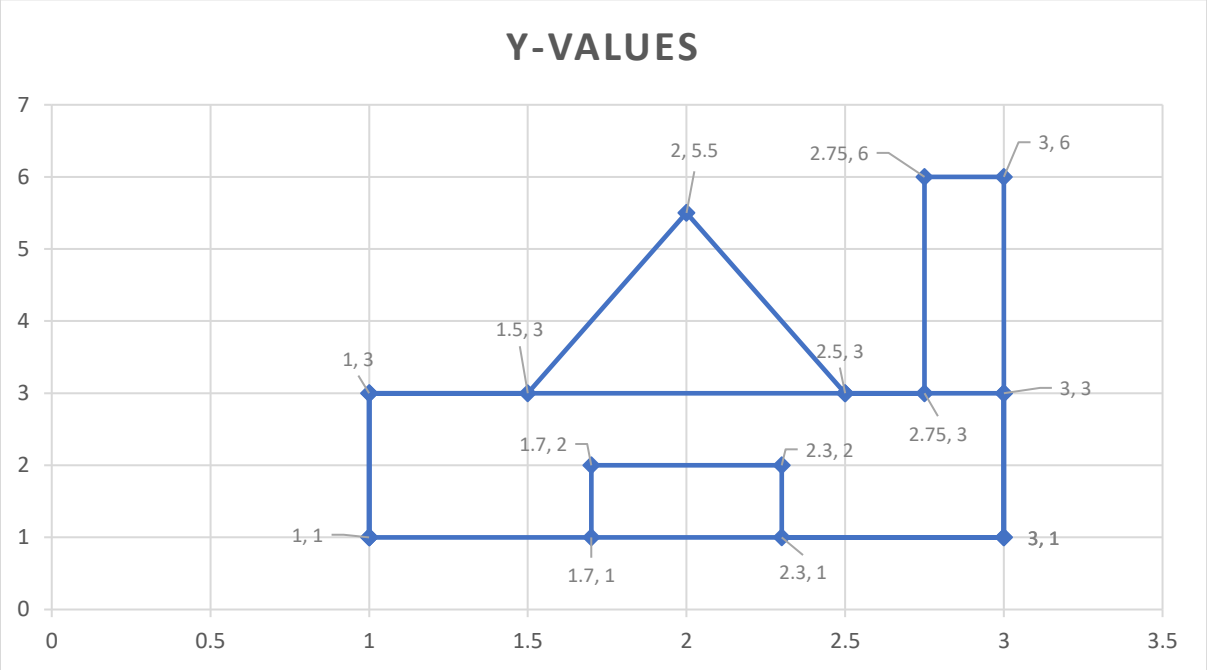
hash function based on modular Arithmetic better than hash function based on truncation.

This is because, in the modular arithmetic process, the IP is distributed among all the servers equally and in an orderly fashion. As for the truncation, as we have noticed in the figure, the first servers are the ones that have the largest number of IPs, which causes pressure on the first servers without the rest of the servers.

Task3: -

For the following shape, define the Cartesian Co-ordinates or the Pixel Co-ordinates; then evaluate the Co-ordinates system used in programming a simple output device by programing the shape. (Use your own dimensions)

I used Cartesian Co-ordinates.

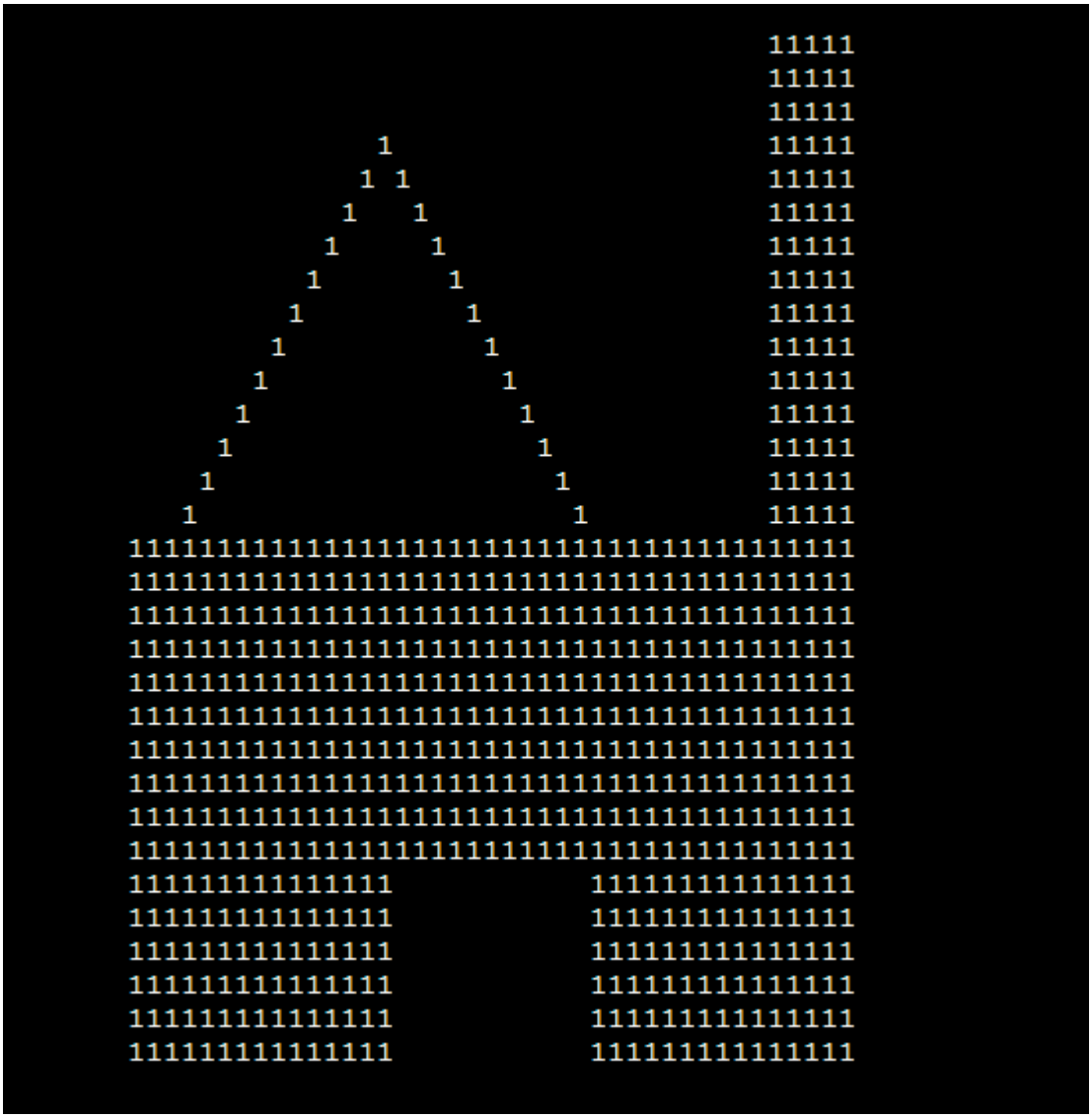



```
#include <stdio.h>

#include <math.h>


int main()

{

   char a[100][100];


   for(int y=0; y<100; y++){

      for(int x=0; x<100; x++){

         a[y][x]=' ';

      }

   }
```

```
// Draw the first Rectangle
int x1=20, y1=20, L=40, W=15;
for(int y=0; y<100; y++){
   for(int x=0; x<100; x++){
      if(x>=x1 && x<= x1+L && y>=y1 && y<=y1+W){
         a[y][x]='1';
      }
   }
}

// Draw the seconde Rectangle
x1=56, y1=5, L=4, W=15;
for(int y=0; y<100; y++){
   for(int x=0; x<100; x++){
      if(x>=x1 && x<= x1+L && y>=y1 && y<=y1+W){
         a[y][x]='1';
      }
   }
}

// Draw the Three Triangle
x1= 34, y1=8; int x2= 22, y2= 20;
double EPSILON= 0.5, m, c;
m= (double) (y2-y1)/(x2-x1);       // the slope of the line
c= m*x1-y1;              //  the y-intercept of the line
for(int y=0; y<100; y++){
   for(int x=0; x<100; x++){
      if(x<=x1 && x>=x2 && fabs(y-m*x+c) < EPSILON){
         a[y][x]='1';
```

```
      }
   }
}
x1= 34, y1= 8, x2= 46, y2= 20;
m= (double) (y2-y1)/(x2-x1);       // the slope of the line
c=m*x1-y1;                         //  the y-intercept of the line

for(int y=0; y<100; y++){
   for(int x=0; x<100; x++){
      if(x>=x1 && x<=x2 && fabs(y-m*x+c) < EPSILON){
         a[y][x]='1';
      }
   }
}

// Draw the Four Rectangle
x1= 35, y1= 30, L=10, W=5;
for(int y=0; y<100; y++){
   for(int x=0; x<100; x++){
      if(x>=x1 && x<=x1+L && y>=y1 && y<=y1+W){
         a[y][x]=' ';
      }
   }
}
// print diagram
for(int y=0; y<100; y++){
   printf("\n");
   for(int x=0; x<100; x++){
         printf("%c",a[y][x]);
```

```
    }
  }
  return 0;
}
```


11111
11111
11111
1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1 1 11111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
1111111111111111111111111111111111111111
111111111111111 111111111111111
111111111111111 111111111111111
111111111111111 111111111111111
111111111111111 111111111111111
111111111111111 111111111111111
111111111111111 111111111111111

Task4: -

A) Describe the tringle Δ(ABC) by vector coordinates.

$$\vec{A} = (2, 5.5),\ \vec{B} = (1.5, 3),\ \vec{C} = (2.5, 3)$$

B) Find the area of the tringle Δ(ABC).

$$\text{Area of a Triangle} = \frac{1}{2}|x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)|$$

Area of a Triangle = $\frac{1}{2}\ |2(3-3) + 1.5(3-5.5) + 2.5(2-3)|$

Area of a Triangle = 3.125

C) Find the angle $\angle ABC$ (use dot-product)

$$\vec{A} \cdot \vec{C} = (\mathrm{ai} + \mathrm{bj}).(\mathrm{di} + \mathrm{ej}) = \mathrm{ad} + \mathrm{be}$$

$$\vec{A} \cdot \vec{C} = 0.5 \times 1 + 2.5 \times 0 = 0.5$$

$$|\mathrm{A}| = \sqrt{(\mathrm{a2} - \mathrm{a1})^2 + (\mathrm{b2} - \mathrm{b1})^2},\ \text{and}\ |\mathrm{C}| = \sqrt{(\mathrm{d2} - \mathrm{d1})^2 + (\mathrm{e2} - \mathrm{e1})^2}$$

$$|\mathrm{A}| = \sqrt{(2 - 1.5)^2 + (5.5 - 3)^2} = \sqrt{6.5}\ ,\ |\mathrm{C}| = \sqrt{(2.5 - 1.5)^2 + (3 - 3)^2} = \sqrt{1}$$

$$\theta\ = \cos^{-1}\frac{\vec{A}\cdot\vec{C}}{|A|\times|C|} = \cos^{-1}\frac{0.5}{\sqrt{6.5}\times\sqrt{1}} = 78.69°$$

D) Construct a scaling process to scale the tringle Δ(ABC) by a factor of 2 in the x direction and by a factor of 3 in the y direction with fixpoint B.

Before scaling: $\vec{A}\begin{pmatrix}2\\5.5\end{pmatrix}$ $\vec{C}\begin{pmatrix}2.5\\3\end{pmatrix}$ $\vec{B}\begin{pmatrix}1.5\\3\end{pmatrix}$

$$P\begin{pmatrix}1.5 + 2\,(\,P_x - 1.5\,)\\3\ \ + 3\,(\,P_y - \ \ 3\ \ \,)\end{pmatrix}$$

After scaling: $\vec{A}\begin{pmatrix}2.5\\10.5\end{pmatrix}$ $\vec{C}\begin{pmatrix}3.5\\3\end{pmatrix}$ $\vec{B}\begin{pmatrix}1.5\\3\end{pmatrix}$