INSTITUTION VISION

To become a premier institution by producing professionals with strong technical knowledge, innovative research skills and high ethical values.

MISSION

IM1: To provide academic excellence in technical education through novel teaching methods.
IM2: To empower students with creative skills and leadership qualities.
IM3: To produce dedicated professionals with social responsibility.

DEPARTMENT

VISION

To be a center of excellence dedicated to providing education in computer applications, fostering a learning environment that cultivates professionals capable of contributing to innovation and social development.

MISSION

DM1: To excel in computer applications education by implementing innovative teaching methods, striving for academic excellence.
DM2: To empower students with creative skills and leadership qualities, fostering an environment that encourages innovation and readies them for successful professional careers.
DM3: To emphasize ethical practices in technology, ensuring that our graduates make meaningful contributions to society by utilizing their expertise for the greater good.

HINDUSTHAN COLLEGE OF ENGINEERING &TECHNOLOGY

(An Autonomous Institution, Affiliated to Anna University, Chennai)

COIMBATORE - 641 032

HINDUSTHAN EDUCATIONAL AND CHARITABLE TRUST

HICET

Certified that this is a bonafide record of work done by

Name :

Register No :

In the **24CA2251 CRYPTOGRAPY AND NETWORK SECURITY LAB** of this Institution ,for the course **MCA DEGREE IInd Semester** during the Academic year **2024 – 2025 (EVEN)**.

Place : Coimbatore

Date :

Faculty In-Charge **Head of the Department**

Submitted for MCA Degree Practical Examination Conducted on

_____

Internal Examiner **External Examiner**

TABLE OF CONTENTS

24CA2251, CRYPTOGRAPY AND NETWORK SECURITY LABORATORY

Continuous Assessment Mark 25 Converted to 75	Model Lab 100 Converted to 25	Total(100)

Faculty In Charge

(Dr. VIGNESHKUMAR K)

Exercise : 1 Date :	**SUBSTITUTION CIPHER ENCRYPTION**

Aim:

To implement a substitution cipher encryption in Python using a randomly generated key that substitutes each alphabet with another unique character.

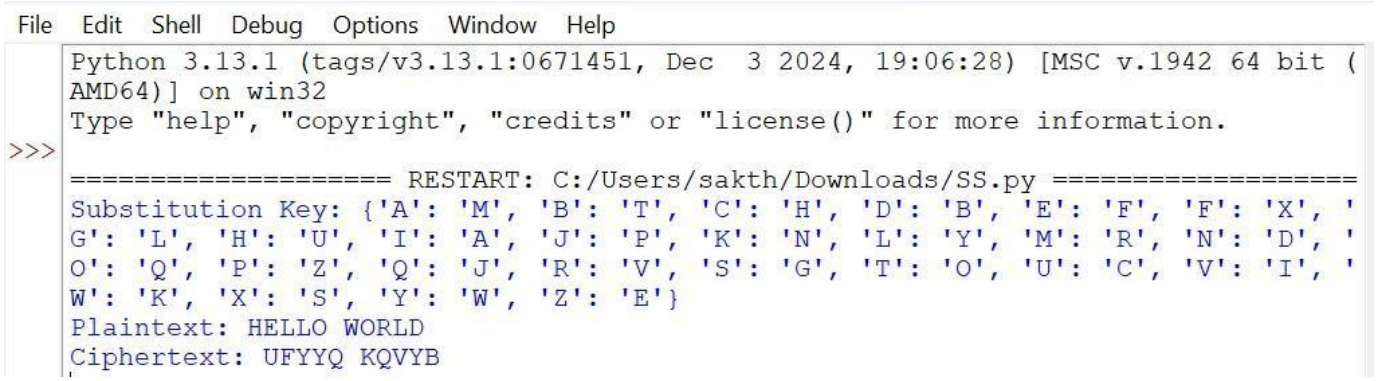
Algorithm:

1. Import required modules: string, random.
2. Define a function generate_key():
   - Create a list of uppercase letters.
   - Shuffle the list to create a random substitution key.
3. Define encrypt(plaintext, key):
   - Convert plaintext to uppercase.
   - Substitute each character using the key if it is an alphabet; else keep it unchanged.
4. Generate a key using generate_key().
5. Define the plaintext message.
6. Call encrypt() function to get the ciphertext.
7. Display the substitution key, original plaintext, and encrypted ciphertext.

Program:

```
import string, random

key = dict(zip(string.ascii_uppercase, random.sample(string.ascii_uppercase, 26)))

def encrypt(text):
    return ''.join(key.get(c, c) for c in text.upper())

plaintext = "HELLO WORLD"
ciphertext = encrypt(plaintext)

print("Substitution Key:", key)
print("Plaintext:", plaintext)
print("Ciphertext:", ciphertext)
```

Output:



Result:

Thus, the above program has been Successfully executed.

Exercise : 2 Date :	**CAESAR CIPHER**

Aim:

To write a Python program that implements the Caesar Cipher encryption technique to encrypt plain text by shifting the characters by a given number of positions.

Algorithm:

1. Start the program
2. Take input from the user: a string (text) and an integer (shift value).
3. Define a function 7aesar_cipher(text, shift):
4. Initialize an empty string result.
5. For each character in the input text:
    a. If the character is an alphabet:
        i. Determine the base ASCII (A for uppercase or a for lowercase).
        ii. Shift the character using the Caesar Cipher formula:
6. Return the encrypted string.
7. Call the function with the input values.
8. Display the encrypted text.
9. End the program

Program:

```
def caesar_cipher(t, s):
   return ''.join(chr((ord(c) - (o := ord('A') if c.isupper() else ord('a')) + s) % 26 + o) if c.isalpha() else c for c in t)

text = input("Text: ")
shift = int(input("Shift: "))
print("Encrypted:", caesar_cipher(text, shift))
```

Output:

```
Enter text: Hello World
Enter shift value: 3
Encrypted text: Khoor Zruog
```

Result:

Thus, the above program has been Successfully executed.

Exercise : 3 Date :	**HILL CIPHER ENCRYPTION**

Aim:

To implement a 2x2 Hill Cipher encryption technique using linear algebra and matrix multiplication in Python.

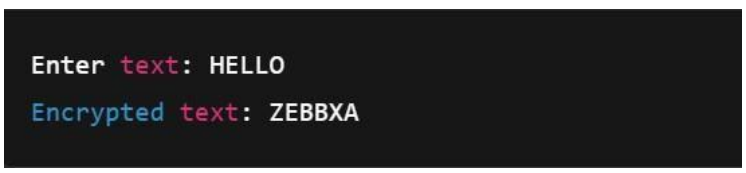
Algorithm:

1. Import numpy for matrix operations.
2. Define hill_cipher_encrypt(text, key_matrix):
   - Convert the input text to uppercase and remove spaces.
   - Pad the text with 'X' if its length is odd. ○ Convert text to numerical vector using A=0 to Z=25.
   - For every pair of numbers, apply matrix multiplication with the key matrix.
   - Convert resulting numbers back to letters.
3. Define a 2x2 key matrix.
4. Get user input for plaintext.
5. Call the hill_cipher_encrypt() function.
6. Print the encrypted result.

Program:

```
import numpy as np

def hill_encrypt(t, k):
    t = t.upper().replace(" ", "")
    n = len(k)
    t += 'X' * (-len(t) % n)
    v = [ord(c) - 65 for c in t]
    return ''.join(chr(int(c) + 65) for i in range(0, len(v), n) for c in np.dot(k, v[i:i+n]) % 26)

k = np.array([[3, 3], [2, 5]])
print("Encrypted:", hill_encrypt(input("Text: "), k))
```

Output:



Result:

Thus, the above program has been Successfully executed.

Exercise : 4 Date :	**IMPLEMENT THE DES ALGORITHM LOGIC**

Aim:

To implement the DES (Data Encryption Standard) encryption and decryption using Python.

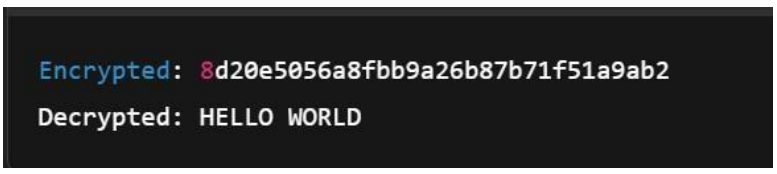
Algorithm:

1. Import DES from Crypto.Cipher and utility modules for padding.
2. Define a key of 8 bytes.
3. Create the cipher object using DES and the key.
4. Encrypt and decrypt the message using padding.
5. Display the encrypted and decrypted result.

Program:

```
from Crypto.Cipher import DES
from Crypto.Util.Padding import pad, unpad

k = b'abcdefgh'
c = DES.new(k, DES.MODE_ECB)
t = b"HELLO WORLD"
e = c.encrypt(pad(t, 8))
d = unpad(c.decrypt(e), 8)

print("Encrypted:", e.hex())
print("Decrypted:", d.decode())
```

Output:



Result:

Thus, the above program has been Successfully executed.

Exercise : 5 Date :	**RC4 LOGIC USING CRYPTOGRAPHY; BLOWFISH ENCRYPTION OF "HELLO WORLD"**

Aim:

To demonstrate RC4 logic and encrypt "Hello world" using Blowfish algorithm.

Algorithm:

1. Import Blowfish from Crypto.Cipher.
2. Use a key (user-defined).
3. Encrypt the message using Blowfish with padding.
4. Output encrypted result.

Program:

```
from Crypto.Cipher import Blowfish
from Crypto.Util.Padding import pad, unpad

c = Blowfish.new(b'secretkey123', Blowfish.MODE_ECB)
t = b"Hello world"
e = c.encrypt(pad(t, Blowfish.block_size))
d = unpad(c.decrypt(e), Blowfish.block_size)

print("Encrypted:", e.hex())
print("Decrypted:", d.decode())
```

Output:

Result:

Thus, the above program has been Successfully executed.

Exercise : 6 Date :	**TRIPLE DES (3DES) IMPLEMENTATION**

Aim:

To implement Triple DES (3DES) encryption and decryption in Python.

Algorthim:

1. Import DES3 and padding utilities.
2. Use a 16 or 24-byte key.
3. Encrypt and decrypt using Triple DES ECB mode.
4. Pad and encrypt plaintext.
5. Decrypt the cipher.
6. Display both.

Program:

```
from Crypto.Cipher import DES3
from Crypto.Util.Padding import pad, unpad

k = DES3.adjust_key_parity(b'Sixteen byte key123456'.ljust(24, b'0'))
c = DES3.new(k, DES3.MODE_ECB)
e = c.encrypt(pad(b"Hello World", 8))
d = unpad(c.decrypt(e), 8)

print("Encrypted:",e.hex())
print("Decrypted:",d.decode())
```

Output:

```
Encrypted: c4bbaf2f947275364ad47f12b9c8ad5f
Decrypted: Hello World
```

```
Encrypted: b'...binary...'
Decrypted: Hello123
```

Result:

Thus, the above program has been Successfully executed.

Exercise : 7 Date :	**DIFFIE-HELLMAN KEY EXCHANGE IN HTML + JAVASCRIPT**

Aim:

To implement the Diffie-Hellman key exchange using JavaScript for secure key sharing.

Algorthim:

1. Define prime number (p) and generator (g).
2. Each user chooses a private key and computes public key.
3. Exchange public keys and compute shared secret.
4. Verify both shared keys match.
5. Display the output.

Program:

```
<!DOCTYPE html>
<html>
<head>
  <title>Diffie-Hellman</title>
</head>
<body>
  <h2>Diffie-Hellman Key Exchange</h2>
  <script>
    const p = 23;  // Prime number
    const g = 5;   // Base (generator)

    const a = 6;   // Private key A
    const b = 15;  // Private key B

    const A = Math.pow(g, a) % p;  // Public key A
    const B = Math.pow(g, b) % p;  // Public key B

    const sharedKeyA = Math.pow(B, a) % p;  // Shared key computed by A
    const sharedKeyB = Math.pow(A, b) % p;  // Shared key computed by B

    document.write("Public Key A: " + A + "<br>");
    document.write("Public Key B: " + B + "<br>");
    document.write("Shared Key (A): " + sharedKeyA + "<br>");
    document.write("Shared Key (B): " + sharedKeyB + "<br>");
  </script>
</body>
</html>
```

Output:

```
Public Key A: 8
Public Key B: 2
Shared Key (A): 2
Shared Key (B): 2
```

Result:

Thus, the above program has been Successfully executed.

Exercise : 8 Date :	**SHA-1 MESSAGE DIGEST CALCULATION**

Aim:

To calculate the message digest of a string using SHA-1.

Algorthim:

1. Import hashlib.
2. Define the message.
3. Encode the message.
4. Encode the message to bytes.
5. Use SHA-1 to generate digest.
6. Print the hex digest.

Program:

```
import hashlib

text = "Hello world"
sha1_hash = hashlib.sha1(text.encode()).hexdigest()
print("SHA-1 Digest:", sha1_hash)
```

Output:

```
SHA-1 Digest: 2ef7bde608ce5404e97d5f042f95f89f1c232871
```

Result:

Thus, the above program has been Successfully executed.

Hindusthan College of Engineering and Technology

(An Autonomous Institution, Approved by AICTE, New Delhi,
Accredited with 'A++' Grade by NAAC, Affiliated to Anna University)
Valley Campus, Pollachi Highway, Coimbatore – 641032

Department of Computer Applications

PO	PROGRAM OUTCOME
PO1	Foundation Knowledge: Apply Knowledge of mathematics programming logic and coding fundamentals for solution architecture and problem solving.
PO2	Problem Analysis: Identify, review, formulate and analyse problems for primarily focussing on customer requirements using critical thinking frameworks.
PO3	Development of Solution: Design, develop and investigate problems with as an innovative approach for solutions incorporating ESG/SDG goals.
PO4	Modern Tool Usage : Select, adapt and apply modern computational tools such as development of algorithms with an understanding of the limitations including human biases
PO5	Individual and Team Work: Function and communicate effectively as an individual or team leader in diverse and multidisciplinary groups. Use methodologies such as agile.
PO6	Project Management and Finance : Use the principles of project management such as scheduling, work breakdown structure and be conversant with the principles of finance for profitable project management.
PO7	Ethics: Commit to professional ethics in managing software projects with financial aspects. Learn to use new technologies for cyber security and insulate customers from malware.
PO8	Life-Long Learning : Change management skills and the ability to learn, keep up with contemporary technologies and ways of working.