**Advanced Encryption Standard**

**CSE 459: Cryptography & Network Security**

Submitted by

Name: D. Tejaswi

Roll No: AP22110011503

Section: CSE Y

Lab Date: 03/02/2025

Submission Date: 09/02/2025

**A picture containing text

Description automatically generated**

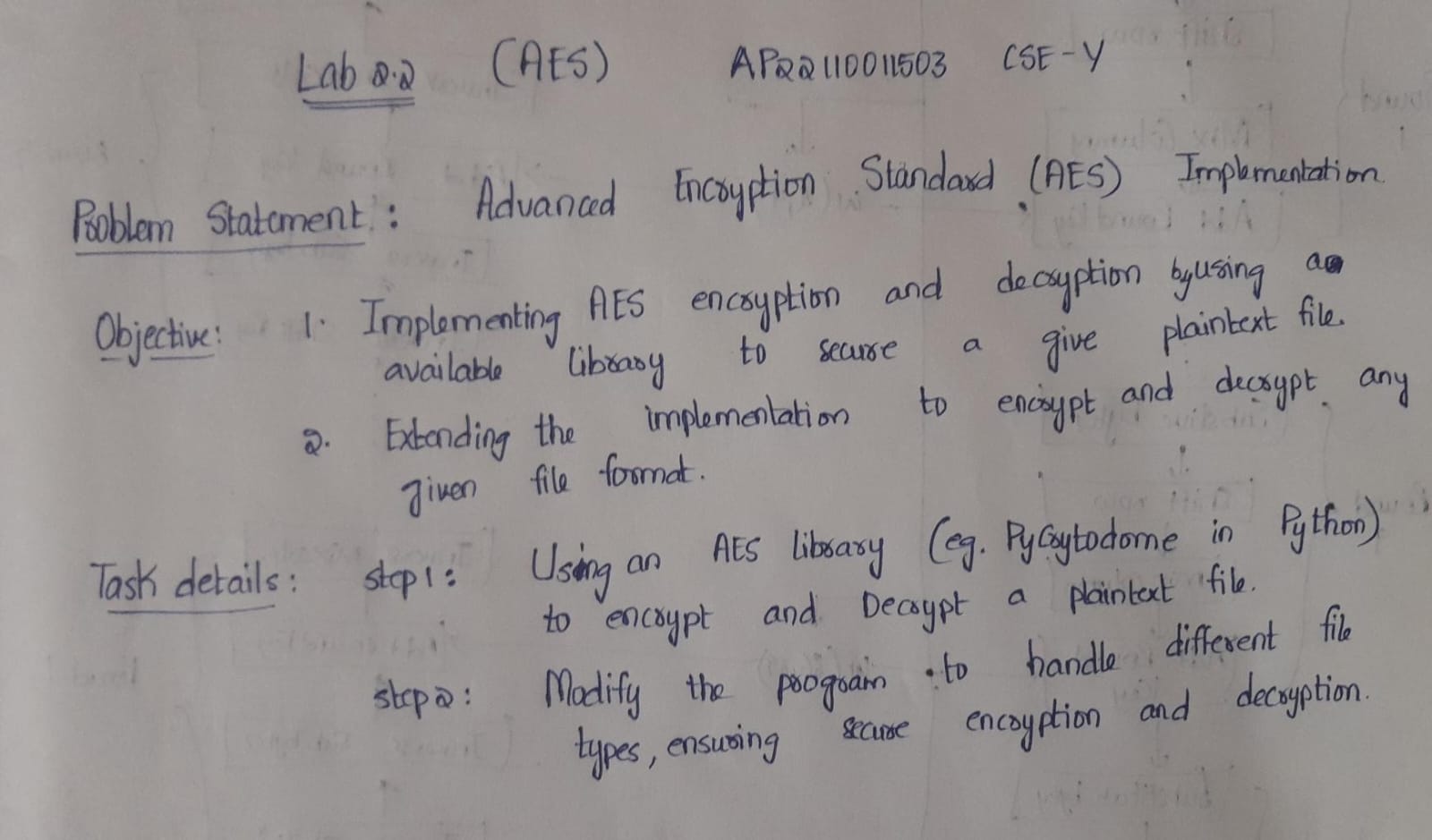
**Department Computer Science and Engineering**

**School of Engineering and Sciences**

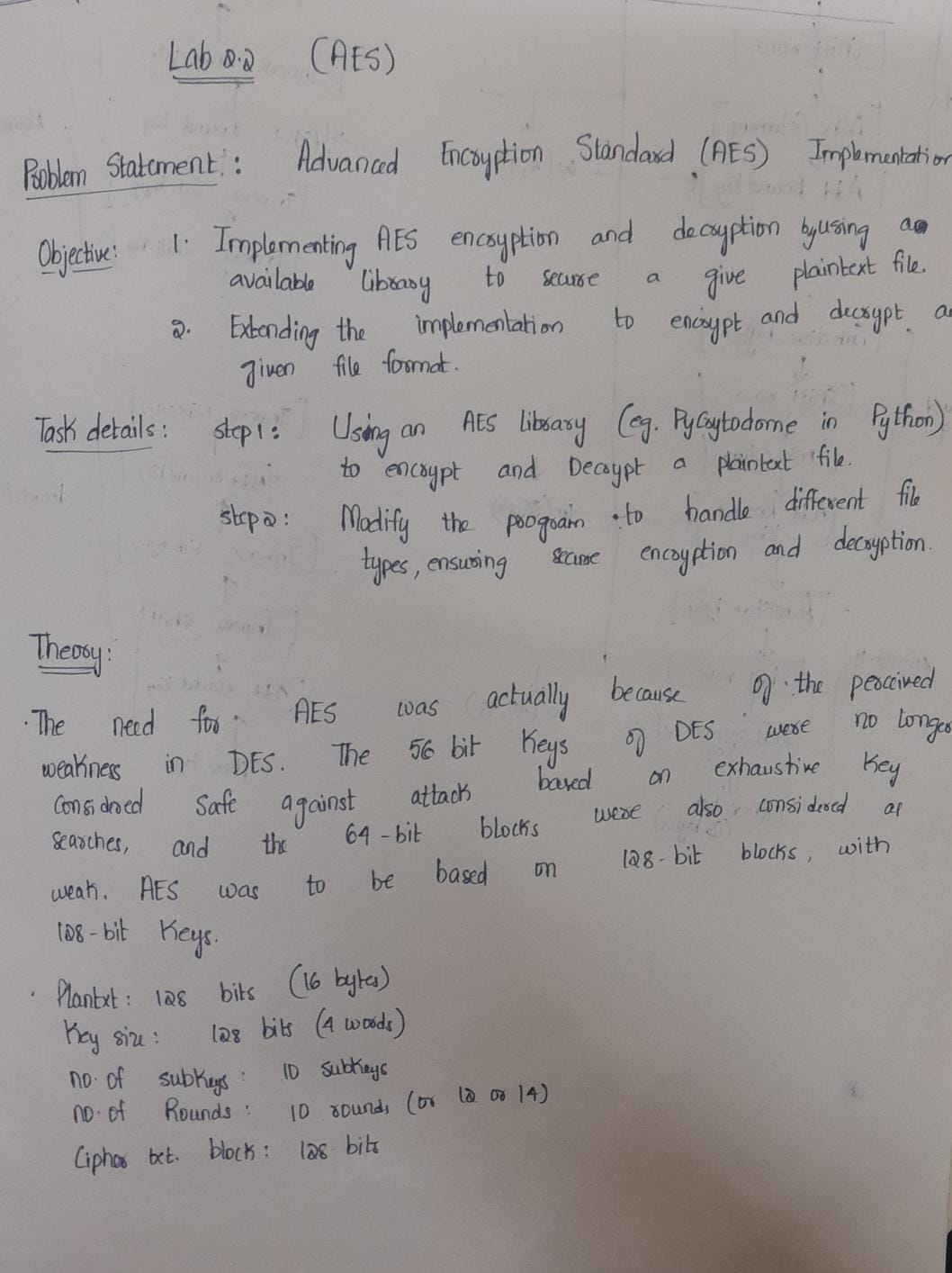
**SRM University–AP**

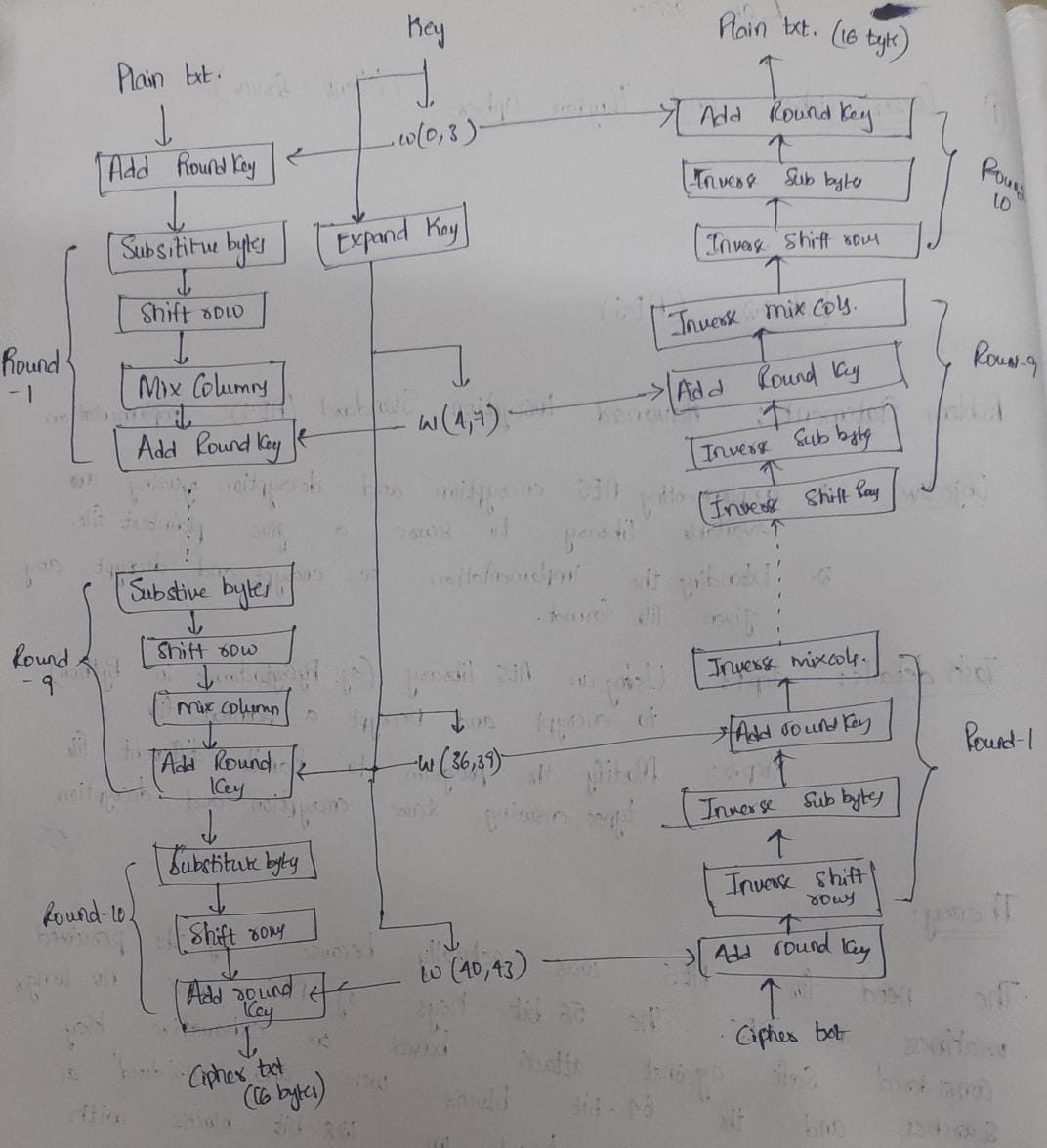
**Amaravati, Andhra Pradesh – 522 240, India**

1. **Question**

****

1. **Algorithm Description**

****

****

1. **Solution**

#### 1) Implement AES using the available library to encrypt and decrypt a given plain text file.

from Crypto.Cipher import AES

from Crypto.Util.Padding import pad, unpad

import base64

import os

KEY = os.urandom(16)  # Replace with a fixed key for actual use

def encrypt\_message(message):

    cipher = AES.new(KEY, AES.MODE\_CBC, iv=KEY)

    encrypted\_bytes = cipher.encrypt(pad(message.encode(), AES.block\_size))

    return base64.b64encode( encrypted\_bytes).decode()

def decrypt\_message(encrypted\_message):

    cipher = AES.new(KEY, AES.MODE\_CBC, iv=KEY)

    decrypted\_bytes = unpad(cipher.decrypt(base64.b64decode(encrypted\_message)), AES.block\_size)

    return decrypted\_bytes.decode()

def send():

    message = open("/content/shift.java",'r').read()

    encrypted\_message = encrypt\_message(message)

    print(f"Client (Encrypted): {encrypted\_message}")

    return encrypted\_message

def receive(msg):

    decrypted\_message = decrypt\_message(msg)

    print(f"Server (Decrypted): {decrypted\_message}")

    return decrypted\_message

encrypted\_msg = send()

receive(encrypted\_msg)

**Output:**

Client (Encrypted): HNq+GnFATHXRrtDSVwVY9T1JPAfeRXqLheZ8SlXq2fQ7RlHXqk1o53vcViJApUqyiIuHITvP86s2h2aO6fniH8RGPKBsfDHyHGjgG+p25DeBXTT7Y6YnW3EqpyI83Tpb9oaiRldHDoXTVTjGlVAiezRVQ0Xew5WeMDdgaVc+vsJMa14cYJVj2SY/vJqnZzgEHVnrO0whakrcHb5owoReX/ZYYyDe4Qf2evrhJ0P6rGW4Qh4wzZR5HVav81PlNTe64qvO6bUOqWZjVtD9JkiaV+iR7BEEj1AaBEWHQDFkrLn325U8DteN8TJMw8e6d2Gp2l4tx6H9nNuRrgcpda5nQJgHU1LTj0YAVXFWKdythcvBCDmDYaHuFar1vVzVMKY8

Server (Decrypted): public class shift {

public static void main(String[] args) {

int num = 8;

System.out.println("Left Shift: " + (num << 1));

System.out.println("Right Shift: " + (num >> 1));

System.out.println("Unsigned Right Shift: " + (num >>> 1));

}

}

'

public class shift {\n public static void main(String[] args) {\n int num = 8;\n System.out.println("Left Shift: " + (num << 1));\n System.out.println("Right Shift: " + (num >> 1));\n System.out.println("Unsigned Right Shift: " + (num >>> 1));\n }\n}\n

2) Extend the program (2) to encrypt and decrypt a given file.

def encrypt\_message(message):

    cipher = AES.new(KEY, AES.MODE\_CBC, iv=KEY)  # Using key as IV (Not recommended for real applications)

    encrypted\_bytes = cipher.encrypt(pad(message, AES.block\_size))  # No need to encode, message is already bytes

    return base64.b64encode(encrypted\_bytes).decode()

def decrypt\_message(encrypted\_message):

    cipher = AES.new(KEY, AES.MODE\_CBC, iv=KEY)

    decrypted\_bytes = unpad(cipher.decrypt(base64.b64decode(encrypted\_message)), AES.block\_size)

    return decrypted\_bytes  # Return as bytes, no need to decode

def send(image\_path):

    with open(image\_path, 'rb') as f:

        message = f.read()  # Read image data as bytes

    encrypted\_message = encrypt\_message(message)

    print(f"Client (Encrypted): {encrypted\_message[:50]}...")  # Print only first 50 chars for readability

    return encrypted\_message

def receive(msg):

    decrypted\_message = decrypt\_message(msg)

    print(f"Server (Decrypted): {len(decrypted\_message)} bytes received.")  # Print size instead of raw data

    return decrypted\_message

image\_path = "/content/Screenshot 2025-02-07 175816.png"

# Encrypt and send image

encrypted\_msg = send(image\_path)

# Receive and decrypt image

decrypted\_data = receive(encrypted\_msg)

# Save decrypted image

decrypted\_image\_path = r"D:\decrypt.png"

with open(decrypted\_image\_path, "wb") as f:

    f.write(decrypted\_data)

print(f"Decrypted image saved at: {decrypted\_image\_path}")

print("Decrypted file size:", os.path.getsize(decrypted\_image\_path))

**Output:**

Client (Encrypted): qL/+TOR4bREdW7NHEE3GYLail1Y+eV56GMazFqqVAFqn9cT/00...

Server (Decrypted): 368258 bytes received.

Decrypted image saved at: D:\decrypt.png

Decrypted file size: 368258

1. **Code Repository:**

Provide GitHub link for the assignment