CRYPTO LAB ASSIGNMENT

Name: Paras Joshi Roll No: 251100690034

Q1. **Develop a program that shows the working on S-Box and inverse used in AES- take bit number and loop through the look up mechanism after your base S and S inverse operation is working.**

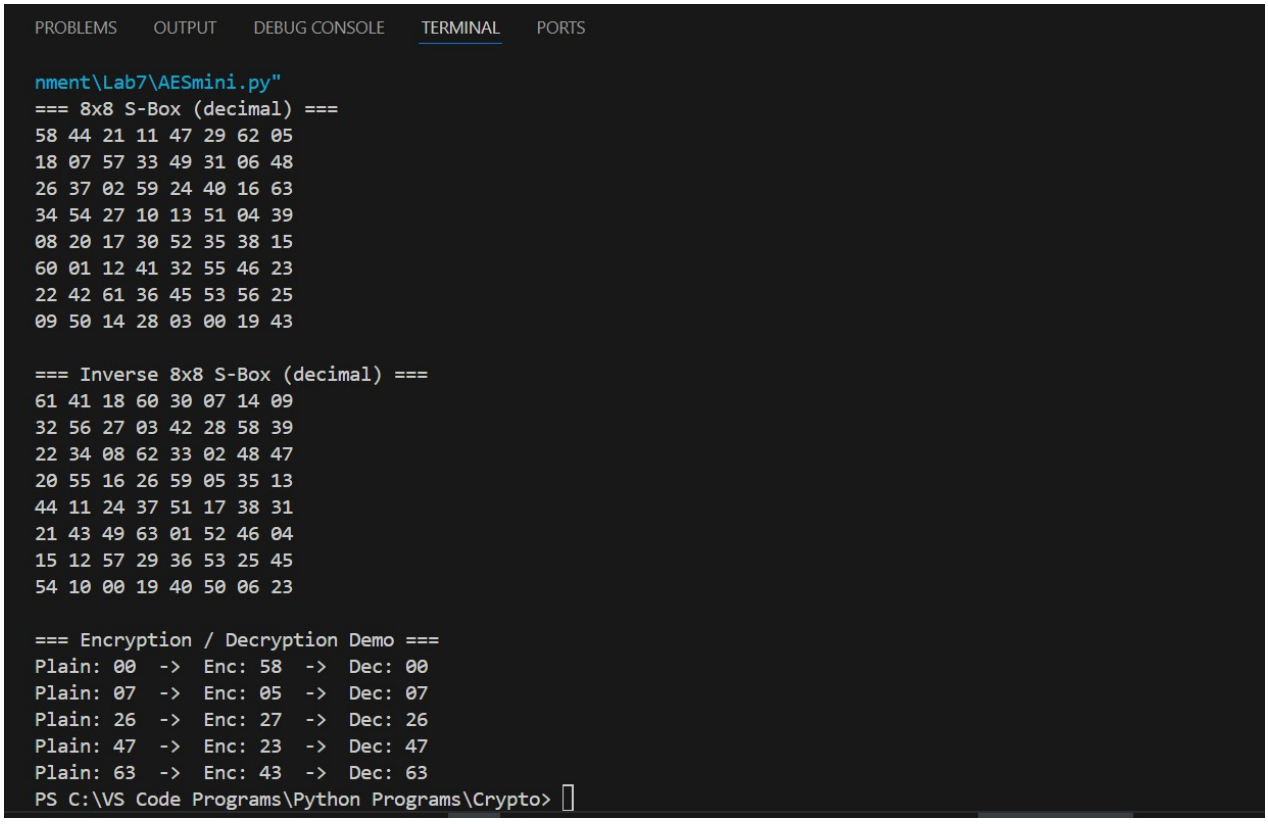
A. In this, I have taken 8x8 matrix for the substitution box (s-box) and found the s-box inverse. I have taken values from 0-63 and it has been shuffled for the s-box values. The lookup mechanisms have been added for both s-box and s-inverse box.

This is the code part.

```
# Mini AES Demo with 8x8 S-Box (decimal only)

# Example 8x8 S-Box with values 0–63
shuffled S_BOX = [
    [58, 44, 21, 11, 47, 29, 62, 5],
    [18, 7, 57, 33, 49, 31, 6, 48],
    [26, 37, 2, 59, 24, 40, 16, 63],
    [34, 54, 27, 10, 13, 51, 4, 39],
    [8, 20, 17, 30, 52, 35, 38, 15],
    [60, 1, 12, 41, 32, 55, 46, 23],
    [22, 42, 61, 36, 45, 53, 56, 25],
    [9, 50, 14, 28, 3, 0, 19, 43],
]
```

```
INV_S_BOX = [[0] * 8 for _ in
range(8)] for i in range(8):
    for j in range(8):
        val = S_BOX[i]
        [j]
        row, col = divmod(val, 8)
        INV_S_BOX[row][col] = (i << 3) | j # reverse mapping

def sbox_lookup(value):
    """Encrypt: Lookup in S-
    Box""" row, col =
    divmod(value, 8) return
    S_BOX[row][col]

def inv_sbox_lookup(value):
    """Decrypt: Lookup in Inverse S-Box"""
    row, col = divmod(value, 8)
    return INV_S_BOX[row][col]

# === Demo ===
if __name__== "__main__":
    print("=== 8x8 S-Box (decimal)
    ===") for row in S_BOX:
        print(" ".join(f"{x:02d}" for x in row))

    print("\n=== Inverse 8x8 S-Box (decimal)
    ===") for row in INV_S_BOX:
        print(" ".join(f"{x:02d}" for x in row))

    # Try encryption +
    decryption samples = [0, 7,
    26, 47, 63]
    print("\n=== Encryption / Decryption Demo
```

This contains Both encryption and decryption and the resulting output we get is this.



Q2. Implement Ceaser's cipher that takes n and m- where n is the shift position (K) and m is number of characters you are considering (26, 27 (space), 28 (comma) or 29 (full stop))

A. I have taken all the characters containing 26 letters along with that I have taken 27 as ‘space’, 28 as ‘,’ and 29 as ‘.’. With this I am going to take my input as “CRYPTO IS AMAZING” and

encrypted that text into “ET RVQ.KU.COC,KPI” using the shift key as 2.

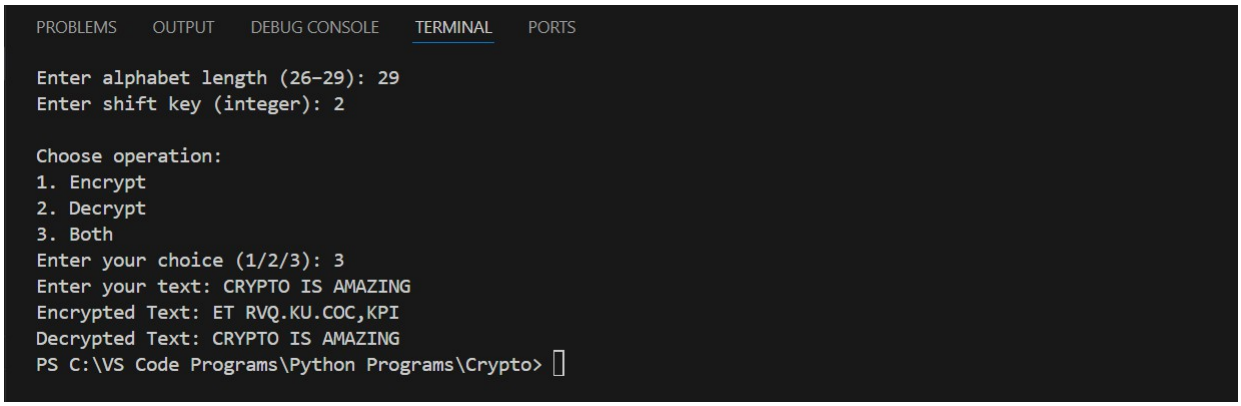
To decipher/decrypt it, I used the same key and converted back to original text.

This is the code part.

```
class
    ConfigurableCaesarCipher:
    def __init__(self, M,
    key):
        if M < 26 or M > 29:
            raise ValueError("M must be between 26 and 29")
        self.M = M
        self.key = key
        self.alphabet = [chr(i + ord('A')) for i in
        range(26)] if M >= 27:
            self.alphabet.append('
        ') if M >= 28:
            self.alphabet.append(','
        ) if M == 29:
            self.alphabet.append('.')
        self.char_to_index = {char: idx for idx, char in
enumerate(self.alphabet)}
        self.index_to_char = {idx: char for idx, char in
enumerate(self.alphabet)}

    def encrypt(self, text):
        text = text.upper()
        encrypted = []
        for char in text:
            if char in self.char_to_index:
                idx = self.char_to_index[char]
                shifted_idx = (idx + self.key) %
                self.M
                encrypted.append(self.index_to_char[shifted_idx])
```

```
        text = text.upper()
        decrypted = []
        for char in text:
            if char in self.char_to_index:
                idx = self.char_to_index[char]
                shifted_idx = (idx - self.key) %
                self.M
                decrypted.append(self.index_to_char[shifted_idx])
            else:
                decrypted.append(char)
        return ''.join(decrypted)

def main():
    print("Welcome to the Configurable Caesar Cipher
    🔐") try:
        M = int(input("Enter alphabet length (26–29):
        ")) key = int(input("Enter shift key (integer):
        ")) cipher = ConfigurableCaesarCipher(M, key)

        print("\nChoose
        operation:") print("1.
        Encrypt") print("2.
        Decrypt") print("3.
        Both")
        choice = input("Enter your choice (1/2/3): ")

        text = input("Enter your text: ")

        if choice == '1':
            print("Encrypted Text:",
        cipher.encrypt(text)) elif choice == '2':
            print("Decrypted Text:",
        cipher.decrypt(text)) elif choice == '3':
            encrypted = cipher.encrypt(text)
            decrypted = cipher.decrypt(encrypted)
            print("Encrypted Text:", encrypted)
            print("Decrypted Text:", decrypted)
        else:
            print("Invalid choice. Please select 1, 2, or 3.")

    except ValueError as ve:
        print("Error:", ve)

if __name__ == "__main__":
    main()
```

Output:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Enter alphabet length (26-29): 29
Enter shift key (integer): 2
Choose operation:
1. Encrypt
2. Decrypt
3. Both
Enter your choice (1/2/3): 3
Enter your text: CRYPTO IS AMAZING
Encrypted Text: ET RVQ.KU.COC,KPI
Decrypted Text: CRYPTO IS AMAZING
PS C:\VS Code Programs\Python Programs\Crypto>