COMSATS UNIVERSITY ISLAMABAD

ATTOCK CAMPUS

INFORMATION SECURITY LAB

ASSIGNMENT NO: 1

SUBMITTED BY:

AROOJ NOREEN

FA24-BSE-046

SUBMITTED TO:

MS AMBREEN GUL

DEADLINE:

February 28, 2026

Table of Contents:

INTRODUCTION

The **Caesar Cipher** is one of the earliest and simplest encryption techniques in classical cryptography. It is a type of substitution cipher in which each letter of the plaintext is shifted forward or backward by a fixed number of positions in the alphabet.

For example, if the shift value is 3:
A → D
B → E
C → F

The shift value acts as the key for both encryption and decryption. Only alphabetic characters are shifted, while spaces, numbers, and special symbols remain unchanged.

Although the Caesar Cipher is useful for understanding basic encryption concepts, it is not secure for modern communication because it has only 26 possible keys and can easily be broken using brute force or frequency analysis.

PROGRAM CODE:

```
# Caesar Cipher Implementation
# Lab Assignment 1 - Information Security
```

```python
def caesar_encrypt(text, shift):
    encrypted_text = ""

    for char in text:
        # Check if character is uppercase letter
        if char.isupper():
            # Shift within range A-Z
            encrypted_char = chr((ord(char) - ord('A') + shift) % 26 + ord('A'))
            encrypted_text += encrypted_char

        # Check if character is lowercase letter
        elif char.islower():
            # Shift within range a-z
            encrypted_char = chr((ord(char) - ord('a') + shift) % 26 + ord('a'))
            encrypted_text += encrypted_char

        else:
            # Keep spaces and special characters unchanged
            encrypted_text += char

    return encrypted_text


def caesar_decrypt(ciphertext, shift):
    decrypted_text = ""

    for char in ciphertext:
        # Check if character is uppercase letter
        if char.isupper():
            decrypted_char = chr((ord(char) - ord('A') - shift) % 26 + ord('A'))
            decrypted_text += decrypted_char

        # Check if character is lowercase letter
        elif char.islower():
            decrypted_char = chr((ord(char) - ord('a') - shift) % 26 + ord('a'))
            decrypted_text += decrypted_char

        else:
            # Keep spaces and special characters unchanged
            decrypted_text += char

    return decrypted_text


# Main Program
```

```
message = input("Enter your message: ")
shift_value = int(input("Enter shift value: "))

encrypted = caesar_encrypt(message, shift_value)
print("Encrypted Message:", encrypted)

decrypted = caesar_decrypt(encrypted, shift_value)
print("Decrypted Message:", decrypted)
```

CODE EXPLANATION:

1. Function Definition

def caesar_encrypt(text, shift):

This line defines the encryption function that takes the original message (text) and the shift value (shift) as input parameters.

2. Initialize Result Variable

encrypted_text = ""

An empty string is created to store the encrypted output.

3. Loop Through Characters

for char in text:

This loop processes each character of the input message individually.

4. Check Uppercase Letters

if char.isupper():

This condition checks whether the character is an uppercase letter.

5. Apply Shift to Uppercase

encrypted_char = chr((ord(char) - ord('A') + shift) % 26 + ord('A'))

This line shifts the uppercase character using ASCII values and modular arithmetic to ensure it remains within the alphabet range.

6. Check Lowercase Letters

elif char.islower():

This condition verifies if the character is lowercase.

7. Apply Shift to Lowercase

encrypted_char = chr((ord(char) - ord('a') + shift) % 26 + ord('a'))

This performs shifting for lowercase letters while preserving alphabet boundaries.

8. Preserve Special Characters

```
else:
  encrypted_text += char
```

Non-alphabetic characters such as spaces and symbols remain unchanged.

9. Return Encrypted Text

return encrypted_text

Returns the final encrypted message.

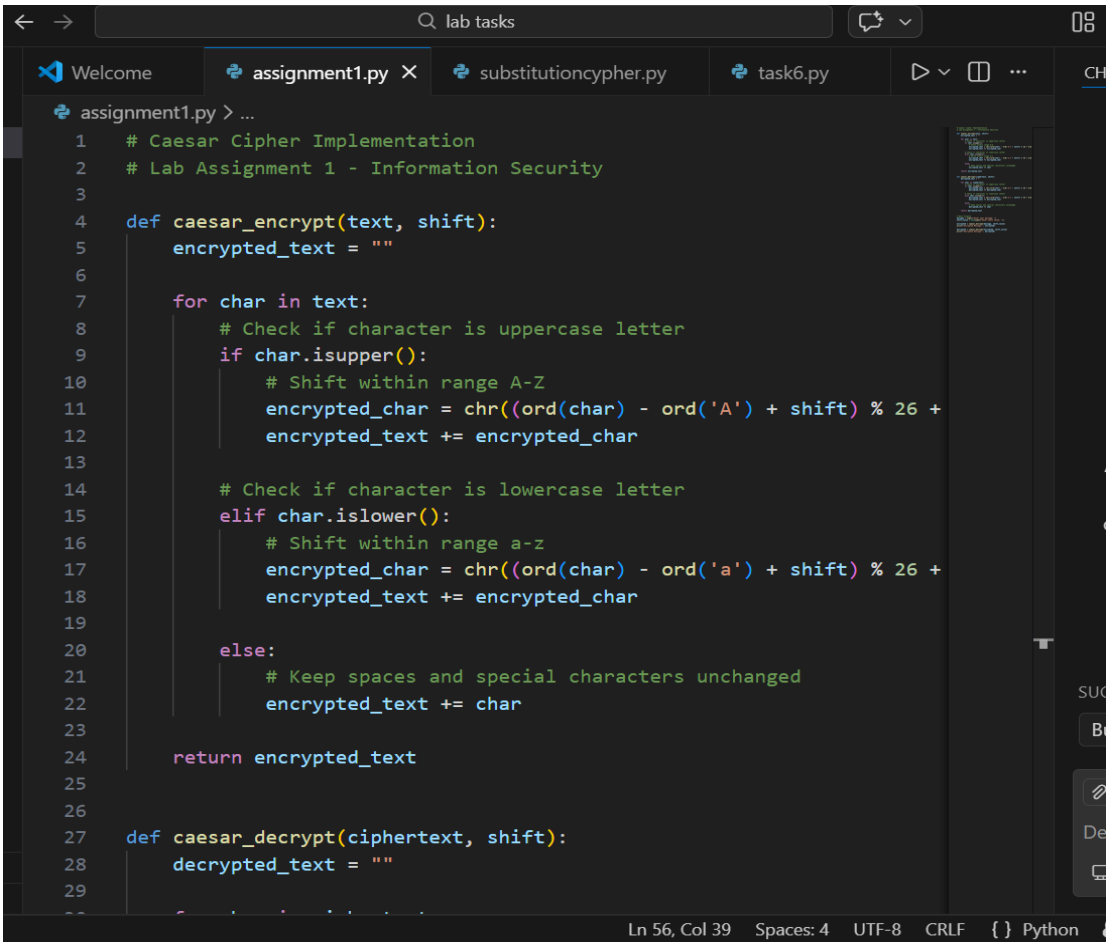
10.Decryption Function

The decryption function follows the same logic as encryption, except that the shift value is subtracted instead of added. This restores the original message.

SECURITY ANALYSIS:

The Caesar Cipher provides basic encryption by shifting characters using a fixed key. However, the algorithm is vulnerable to brute force and frequency analysis attacks because it has only 26 possible keys. Therefore, it is not suitable for modern secure communication systems.

CODE SCREENSHOTS:



```python
def caesar_decrypt(ciphertext, shift):

    for char in ciphertext:
        # Check if character is uppercase letter
        if char.isupper():
            decrypted_char = chr((ord(char) - ord('A') - shift) % 26 +
            decrypted_text += decrypted_char

        # Check if character is lowercase letter
        elif char.islower():
            decrypted_char = chr((ord(char) - ord('a') - shift) % 26 +
            decrypted_text += decrypted_char

        else:
            # Keep spaces and special characters unchanged
            decrypted_text += char

    return decrypted_text


# Main Program
message = input("Enter your message: ")
shift_value = int(input("Enter shift value: "))

encrypted = caesar_encrypt(message, shift_value)
print("Encrypted Message:", encrypted)

decrypted = caesar_decrypt(encrypted, shift_value)
print("Decrypted Message:", decrypted)
```

OUTPUT:

```
PS C:\Users\HEC\Desktop\courses\information security\lab tasks> & C:\Users\HEC\AppData\Local\Python\pythoncore-3.14-64\python.exe "c:/Users/HEC/Desktop/courses/information security/lab tasks/assignment1.py"
Enter your message: arooj
Enter shift value: 4
Encrypted Message: evssn
Enter shift value: 4
Encrypted Message: evssn
Decrypted Message: arooj
PS C:\Users\HEC\Desktop\courses\information security\lab tasks>
```