Course Name:	**Information and Cyber Security Laboratory**	**Semester:**	**VII**
Date of Performance:	**16 / 07 / 2025**	**Batch No.:**	**B – 1**
Faculty Name:	**Prof. Makarand Kulkarni**	**Roll No.:**	**16014022050**
Faculty Sign & Date:		**Grade/Marks:**	**___ / 25**

Experiment No.: 1

Title: To implement Caesar Cipher encryption-decryption

Aim and Objective of the Experiment:
To implement CAESER cipher encryption- decryption.

COs to be achieved:
CO1: Explain various security goals, threats, vulnerabilities and controls with various cryptographic algorithms for software security.

Books/Journals/Websites referred:
1.

Tools required:
C/C++/Python IDE/compiler

Theory:

Julius Caesar used an additive cipher to communicate with his officers. For this reason, additive ciphers are sometimes referred to as the Caesar cipher. Caesar used a key of 3 for his communications. Additive ciphers are sometimes referred to as shift ciphers or Caesar cipher.

The Caesar cipher involves replacing each letter of the alphabet with the letter standing three places further down the alphabet.

For example,

plain: meet me after the party
cipher: PHHW PH DIWHU WKH SDUWB

Then the algorithm can be expressed as follows.
For each plaintext letter p, substitute the cipher Text letter C:
$C = E(3, p) = (p + 3) \bmod 26$
A shift may be of any amount, so that the general Caesar algorithm is

C = E(k, p) = (p + k) mod 26
Where takes on a value in the range 1 to 25.The decryption algorithm is simply
p = D(k, C) = (C - k) mod 26

Implementation details:

1. **Enlist all the Steps followed and various options explored**
    - Took user input for text and key.
    - Checked each character—only processed letters.
    - Shifted letters using Caesar cipher logic.
    - Preserved case and non-letter characters.
    - Displayed encrypted output.

2. **Explain your program logic and methods used.**
    - Used ord() and chr() to convert letters.
    - Applied modulo % 26 to wrap alphabet.
    - Restored original casing with isupper().

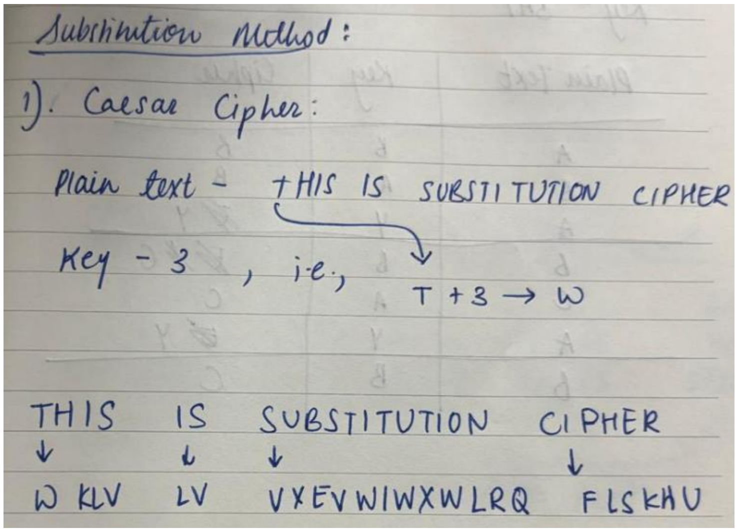
C/C++/Python Code implemented:

```python
def caesar_cipher(text, key):
    encrypted_text = ""
    for char in text:
        if char.isalpha():
            is_upper = char.isupper()
            char = char.lower()
            original_position = ord(char) - ord('a')
            new_position = (original_position + key) % 26
            new_char = chr(new_position + ord('a'))
            if is_upper:
                new_char = new_char.upper()
            encrypted_text += new_char
        else:
            encrypted_text += char
    return encrypted_text

# Input and usage
text = input("Enter the text to encrypt: ")
key = int(input("Enter the key (integer): "))
encrypted_text = caesar_cipher(text, key)
```

```
print("Encrypted text:", encrypted_text)
```

Output/ program results after execution:




```
PS C:\Users\admin\OneDrive\Desktop\sem 7\informawtion and cybersec\lab>
 python -u "c:\Users\admin\OneDrive\Desktop\sem 7\informawtion and cybe
rsec\lab\caesar.py"
Enter the text to encrypt: this is substitution cipher
Enter the key (integer): 3
Encrypted text: wklv lv vxevwlwxwlrq flskhu
PS C:\Users\admin\OneDrive\Desktop\sem 7\informawtion and cybersec\lab>
```

Post Lab Subjective/Objective type Questions:

1. Use the additive cipher with key = 15 to encrypt the message "hello".

```
 python -u "c:\Users\admin\OneDrive\Desktop\sem 7\informawtion and cybe
rsec\lab\caesar.py"
Enter the text to encrypt: hello
Enter the key (integer): 15
Encrypted text: wtaad
PS C:\Users\admin\OneDrive\Desktop\sem 7\informawtion and cybersec\lab>
```

2. **A plaintext was encrypted with a Caesar cipher with a shift of 7 (A maps to H). The resulting ciphertext is:**
   Kvu'a qbknl h ivvr if paz jvcly
   What was the original plaintext?
   - Caesar cipher shifts each letter forward by a fixed number—in this case, +7. To decrypt, we shift each letter backward by 7.
   - Process each letter:
     - Take the letter K. It's the 11th letter of the alphabet.
     - Subtract the shift: 11 - 7 = 4. The 4th letter is D.
     - So, K → D.
   - Repeat this for each letter in the ciphertext.

   - Kvu'a → Don't
     - K → D, V → O, U → N
     - ' → ' (punctuation preserved)
     - A → T

   - qbknl → judge
     - Q → J, B → U, K → D, N → G, L → E

     …and continue for the rest of the sentence.

   - **Plain Text: Don't judge a book by its cover**

Conclusion:
Successfully learnt and implemented different encryption and decryption techniques to make data being transferred more secure.

Signature of faculty in-charge with Date: