



**VIT<sup>®</sup>**  
Vellore Institute of Technology  
(Deemed to be University under section 3 of UGC Act, 1956)

# **CRYPTOGRAPHY AND NETWORK SECURITY**

## **LAB I : WRITE A CODE OF THE FOLLOWING IN YOUR SUITABLE PROGRAMMING LANGUAGE(PYTHON/C/C++/JAVA)**

- i. **CAESER CIPHER**
- ii. **PLAYFAIR CIPHER**

**NAME : OM SUBRATO DEY**

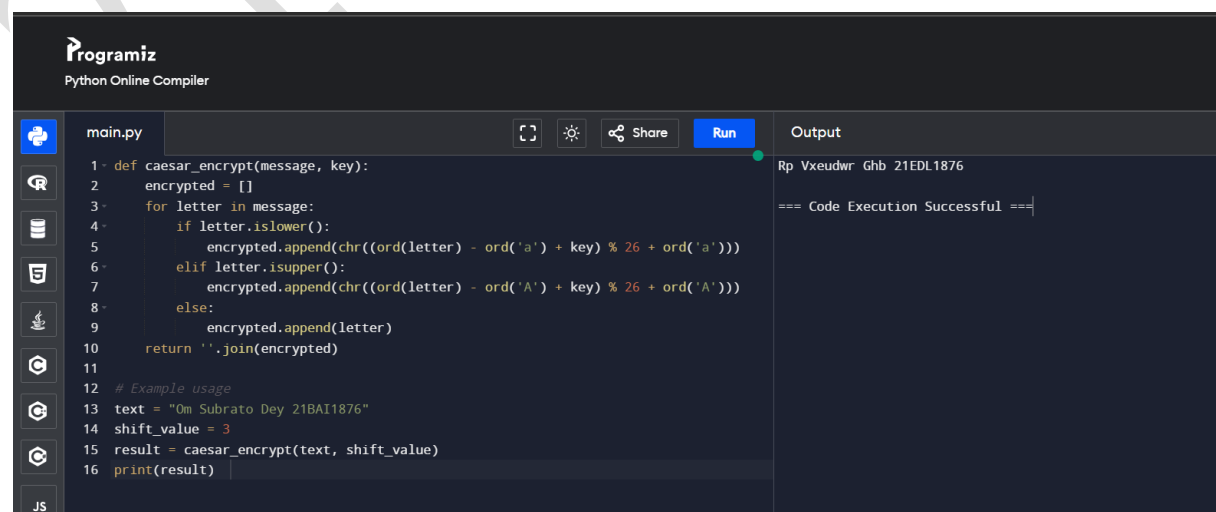
**REGISTER NUMBER : 21BA11876**

## i. CAESER CIPHER:

### CODE:

```
def caesar_encrypt(message, key):  
    encrypted = []  
    for letter in message:  
        if letter.islower():  
            encrypted.append(chr((ord(letter) - ord('a') +  
key) % 26 + ord('a')))  
        elif letter.isupper():  
            encrypted.append(chr((ord(letter) - ord('A') +  
key) % 26 + ord('A')))  
        else:  
            encrypted.append(letter)  
    return ''.join(encrypted)  
  
text = "Om Subrato Dey 21BAI1876"  
shift_value = 3  
result = caesar_encrypt(text, shift_value)  
print(result)
```

### OUTPUT SCREENSHOT:



The screenshot shows the Programiz Python Online Compiler interface. The code editor on the left contains the Python code for the Caesar cipher encryption. The output panel on the right shows the result of the code execution.

```
main.py  
1 def caesar_encrypt(message, key):  
2     encrypted = []  
3     for letter in message:  
4         if letter.islower():  
5             encrypted.append(chr((ord(letter) - ord('a') + key) % 26 + ord('a')))  
6         elif letter.isupper():  
7             encrypted.append(chr((ord(letter) - ord('A') + key) % 26 + ord('A')))  
8         else:  
9             encrypted.append(letter)  
10    return ''.join(encrypted)  
11  
12 # Example usage  
13 text = "Om Subrato Dey 21BAI1876"  
14 shift_value = 3  
15 result = caesar_encrypt(text, shift_value)  
16 print(result)
```

Output  
Rp Vxewdwr Ghb 21EDL1876  
=== Code Execution Successful ===

## ii. PLAYFAIR CIPHER:

### CODE:

```
def forge_key_square(keyword):  
    # Remove duplicates while preserving order  
    keyword = ''.join(sorted(set(keyword),  
key=keyword.index))  
  
    # Create the 5x5 matrix  
    alphabet = 'ABCDEFGHIKLMNOPQRSTUVWXYZ'  
    square = []  
  
    for char in keyword + alphabet:  
        if char not in square and char != 'J': # 'I' and  
'J' are treated as the same letter  
            square.append(char)  
  
    # Split the list into a 5x5 grid  
    return [square[i:i+5] for i in range(0, 25, 5)]  
  
def prepare_message(message):  
    message = ''.join(filter(str.isalnum,  
message.upper())).replace('J', 'I')  
    prepared = []  
    i = 0  
    while i < len(message):  
        first = message[i]  
        if i + 1 < len(message):  
            second = message[i + 1]  
        else:  
            second = 'X'  
        if first == second:  
            prepared.append(first + 'X')  
            i += 1
```

```

        else:
            prepared.append(first + second)
            i += 2
    if len(prepared[-1]) == 1:
        prepared[-1] += 'X'
    return prepared

def locate_position(square, char):
    for row in range(5):
        for col in range(5):
            if square[row][col] == char:
                return row, col
    return None

def encrypt_digraph(square, digraph):
    row1, col1 = locate_position(square, digraph[0])
    row2, col2 = locate_position(square, digraph[1])

    if row1 == row2:
        return square[row1][(col1 + 1) % 5] +
square[row2][(col2 + 1) % 5]
    elif col1 == col2:
        return square[(row1 + 1) % 5][col1] + square[(row2 +
1) % 5][col2]
    else:
        return square[row1][col2] + square[row2][col1]

def playfair_encryption(message, keyword):
    key_square = forge_key_square(keyword)
    prepared_msg = prepare_message(message)
    encrypted_msg = ''

```

```

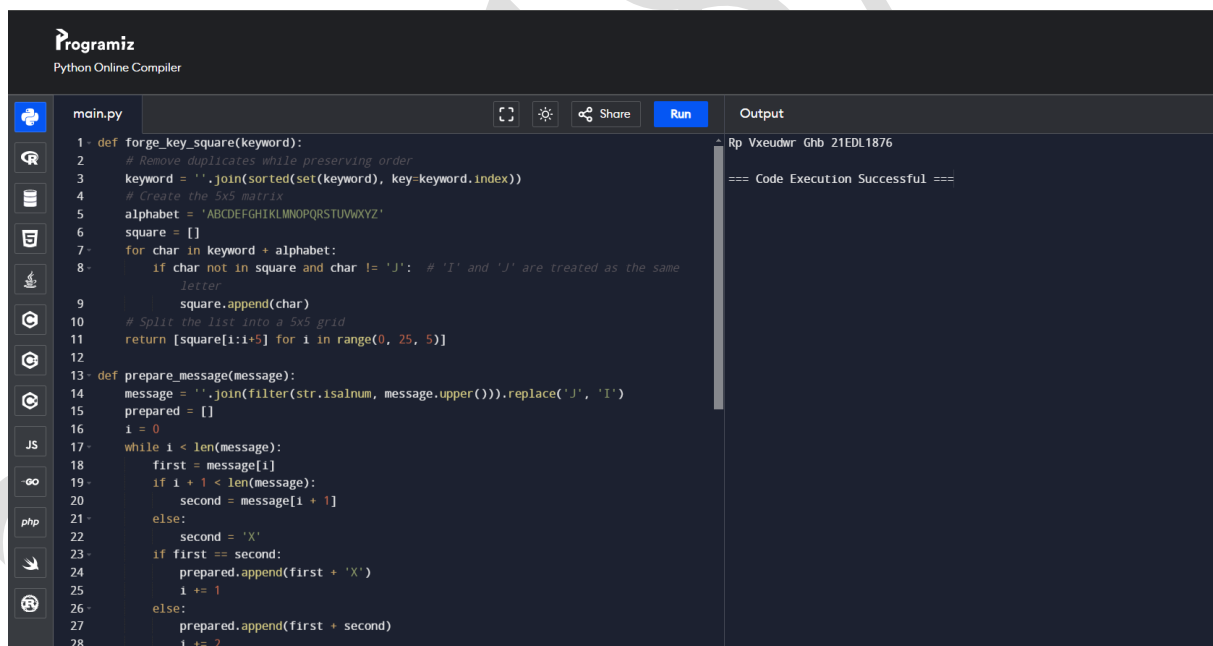
        for digraph in prepared_msg:
            encrypted_msg += encrypt_digraph(key_square,
            digraph)

        return encrypted_msg

keyword = "KEYWORD"
message = "Om Subrato Dey 21BAI1876"
ciphertext = playfair_encryption(message, keyword)
print(ciphertext)

```

## **OUTPUT SCREENSHOT:**



The screenshot shows the Programiz Python Online Compiler interface. The code in the editor is as follows:

```

1- def forge_key_square(keyword):
2-     # Remove duplicates while preserving order
3-     keyword = ''.join(sorted(set(keyword), key=keyword.index))
4-     # Create the 5x5 matrix
5-     alphabet = 'ABCDEFGHIKLMNOPQRSTUVWXYZ'
6-     square = []
7-     for char in keyword + alphabet:
8-         if char not in square and char != 'J': # 'I' and 'J' are treated as the same letter
9-             square.append(char)
10-    # Split the list into a 5x5 grid
11-    return [square[i:i+5] for i in range(0, 25, 5)]
12-
13- def prepare_message(message):
14-     message = ''.join(filter(str.isalnum, message.upper())).replace('J', 'I')
15-     prepared = []
16-     i = 0
17-     while i < len(message):
18-         first = message[i]
19-         if i + 1 < len(message):
20-             second = message[i + 1]
21-         else:
22-             second = 'X'
23-         if first == second:
24-             prepared.append(first + 'X')
25-             i += 1
26-         else:
27-             prepared.append(first + second)
28-             i += 2

```

The output on the right shows the execution result:

```

Rp Vxeudwr Ghb 21EDL1876
=== Code Execution Successful ===

```