Experiment No. 2: Design and Implementation of Ceaser Cipher

Aim: To design and Implementation of a Ceaser Cipher

Theory:

The Caesar cipher, named after Julius Caesar, is one of the simplest and most widely known encryption techniques. It is a substitution cipher where each letter in the plaintext is shifted a certain number of places down or up the alphabet. The key for the Caesar cipher is the number of positions each letter in the plaintext is shifted. For example, with a shift of 3, 'A' would be replaced by 'D', 'B' would become 'E', and so on.

The encryption process of the Caesar cipher operates by shifting the alphabet by a fixed number of positions. This fixed number is the key. For instance, with a key of 3, 'A' would be replaced by 'D', 'B' would become 'E', and so forth. The process continues until all letters in the plaintext are substituted according to the key.

To decrypt a message encrypted with the Caesar cipher, one simply needs to shift the alphabet in the opposite direction by the same key. For example, if the encryption key was 3, decryption would involve shifting each letter in the ciphertext back by 3 positions.

In the Caesar cipher, the key space is limited to the number of letters in the alphabet, since the key represents the number of positions shifted. For the English alphabet, the key space is 25, as a key of 0 or 26 would result in no change to the plaintext.

Experimentation with the Caesar cipher can involve various aspects such as cryptanalysis, security analysis, and studying its cryptographic properties. One of the simplest forms of cryptanalysis against the Caesar cipher is brute force, where all possible keys are tried until the correct one is found. Given the limited key space, brute force is easily feasible even without computers.

However, the Caesar cipher is highly vulnerable to brute force attacks and frequency analysis. Frequency analysis involves analyzing the frequency of letters in the ciphertext to deduce the key. Since the Caesar cipher merely shifts letters, the frequency distribution of letters remains the same, making it susceptible to this form of attack.

Despite its vulnerabilities, the Caesar cipher forms the basis for more complex encryption techniques, such as the Vigenère cipher and the ROT13 algorithm. These variations introduce additional complexity by using multiple Caesar ciphers with different shift values or other modifications.

Algorithm/Procedure:

```
function encrypt(text, shift):
    encrypted_text = ""
```

```
    for each character in text:
        if character is an uppercase letter:
            encrypted_text += shift_character(character, shift, 'A')
        else if character is a lowercase letter:
            encrypted_text += shift_character(character, shift, 'a')
        else:
            encrypted_text += character
    return encrypted_text

function shift_character(character, shift, base):
    alphabet_size = 26
    shifted_char = ((character - base) + shift) % alphabet_size + base
    return shifted_char

function decrypt(encrypted_text, shift):
    return encrypt(encrypted_text, -shift)
```

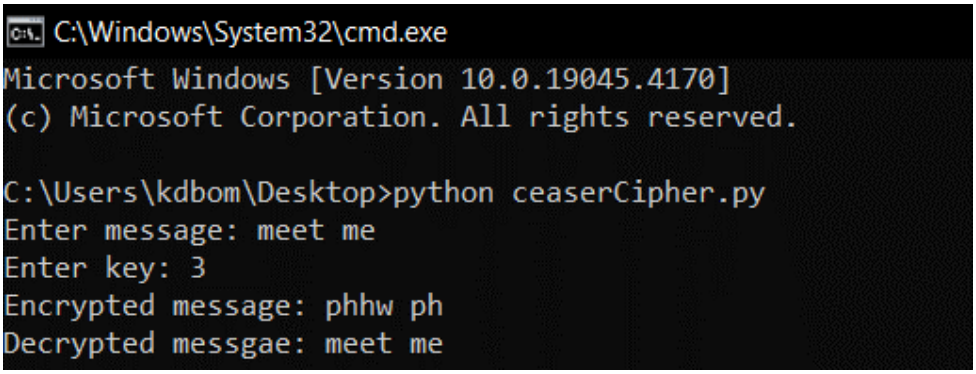
Source Code:

```
def encrypt(plainText, key):

        cipherText = ""
        for letter in plainText:
                index = letters.find(letter)
                if index == -1:
                        cipherText += letter
```

```
            else:
                new_index = index + key
                cipherText += letters[new_index]

            if new_index >= length:
                new_index = new_index%length
                cipherText += letters[new_index]

        return str(cipherText)

def decrypt(cipherText, key):

        plainText = ""
        for letter in cipherText:
            index = letters.find(letter)
            if index == -1:
                plainText += letter
            else:
                new_index = index - key
                plainText += letters[new_index]

            if new_index >= length:
                new_index = new_index%length
                plainText += letters[new_index]
```

```
        return str(plainText)


letters = "abcdefghijklmnopqrstuvwxyz"

length = len(letters)


msg = str(input("Enter message: "))

key = int(input("Enter key: "))


ct = encrypt(msg, key)

print("Encrypted message: " + ct)


pt = decrypt(ct, key)

print("Decrypted messgae: " + pt)
```

Output:



Conclusion:

In conclusion, the Caesar cipher remains a fundamental encryption technique, offering a basic yet illustrative example of substitution ciphers. Through experimentation, we've explored its encryption and decryption processes, key space limitations, susceptibility to cryptanalysis methods like brute force and frequency analysis, as well as its extensions into more complex cryptographic algorithms. While the Caesar cipher holds historical significance and serves as an educational tool, its inherent vulnerabilities underscore the need for more sophisticated encryption methods in modern security contexts.