Cryptography and Network Security
Assignment 1: Caesar Cipher
Name: Mrunal Khade.
PRN: 2020BTECS00057

Q.1 Implementation of Caesar Cipher.

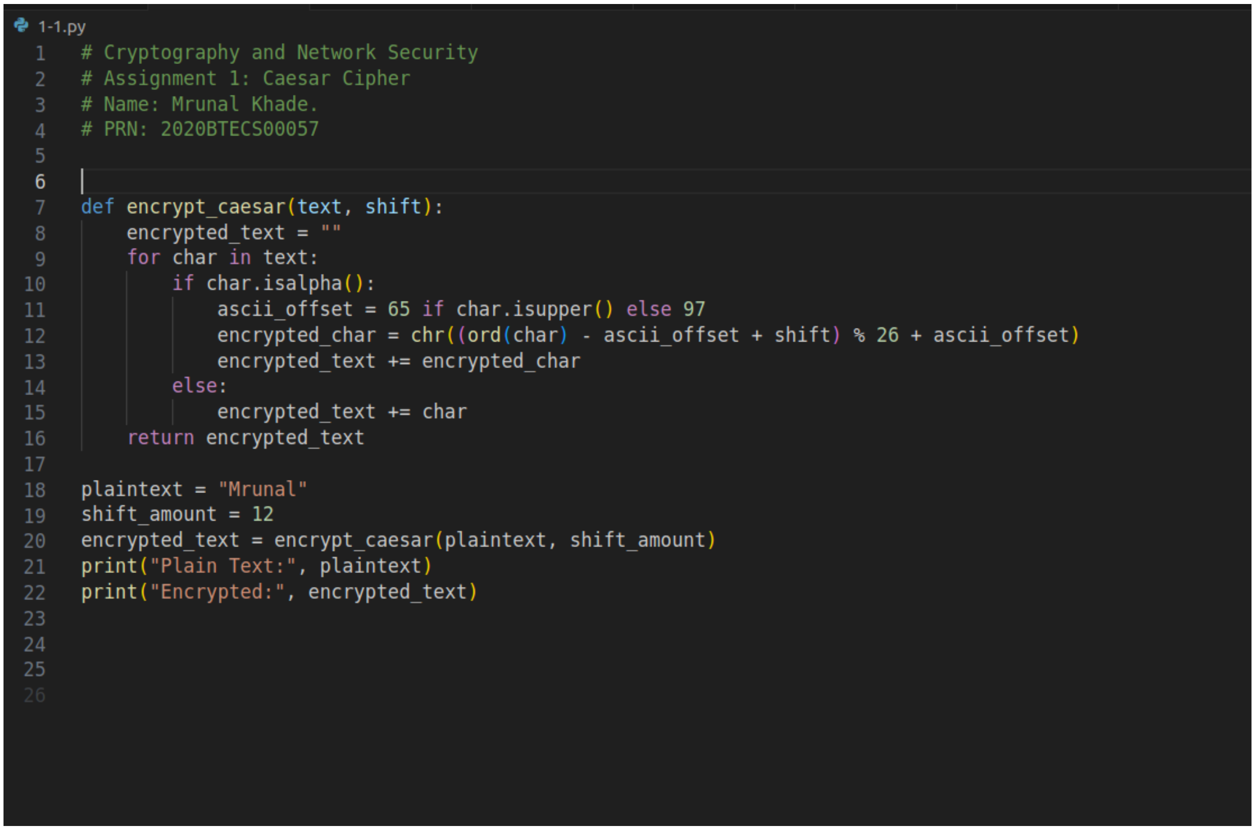
**Theory:**
A Caesar Cipher is a simple substitution cipher where each letter in the plaintext is shifted a certain number of places down or up the alphabet.

**Algorithm:**

**Encryption Algorithm (Caesar Cipher)**

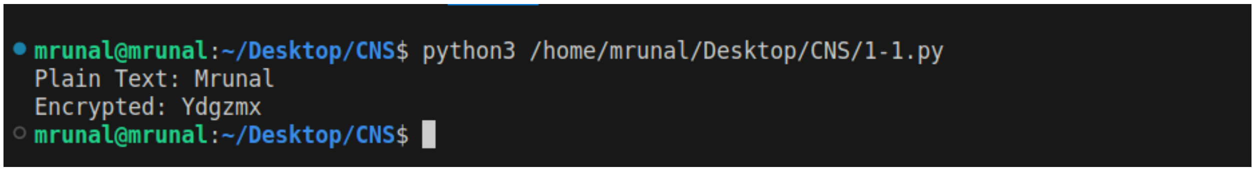
1. Start with a plaintext message and a fixed integer key (the key represents the number of positions to shift each letter).
2. Initialize an empty string to store the ciphertext.
3. For each character in the plaintext message:
   - If the character is an uppercase letter (A-Z), perform the following steps:
     - Determine the position of the letter in the alphabet (A=0, B=1, ..., Z=25).
     - Add the key to this position (modulo 26 to handle wrapping around the alphabet).
     - Convert the new position back to a letter (A=0, B=1, ..., Z=25).
     - Append the resulting letter to the ciphertext.
   - If the character is a lowercase letter (a-z), perform the same steps as above but for the lowercase alphabet.
4. If the character is not a letter (e.g., space, punctuation, or a digit), leave it unchanged and append it to the ciphertext.
5. Continue this process for each character in the plaintext until the entire message is encrypted.
6. The resulting ciphertext is the encrypted message.

**Input:**



```python
# Cryptography and Network Security
# Assignment 1: Caesar Cipher
# Name: Mrunal Khade.
# PRN: 2020BTECS00057


def encrypt_caesar(text, shift):
    encrypted_text = ""
    for char in text:
        if char.isalpha():
            ascii_offset = 65 if char.isupper() else 97
            encrypted_char = chr((ord(char) - ascii_offset + shift) % 26 + ascii_offset)
            encrypted_text += encrypted_char
        else:
            encrypted_text += char
    return encrypted_text

plaintext = "Mrunal"
shift_amount = 12
encrypted_text = encrypt_caesar(plaintext, shift_amount)
print("Plain Text:", plaintext)
print("Encrypted:", encrypted_text)
```

**Output:**



```
mrunal@mrunal:~/Desktop/CNS$ python3 /home/mrunal/Desktop/CNS/1-1.py
Plain Text: Mrunal
Encrypted: Ydgzmx
mrunal@mrunal:~/Desktop/CNS$ 
```

**Algorithm:**

**Decryption Algorithm (Caesar Cipher)**

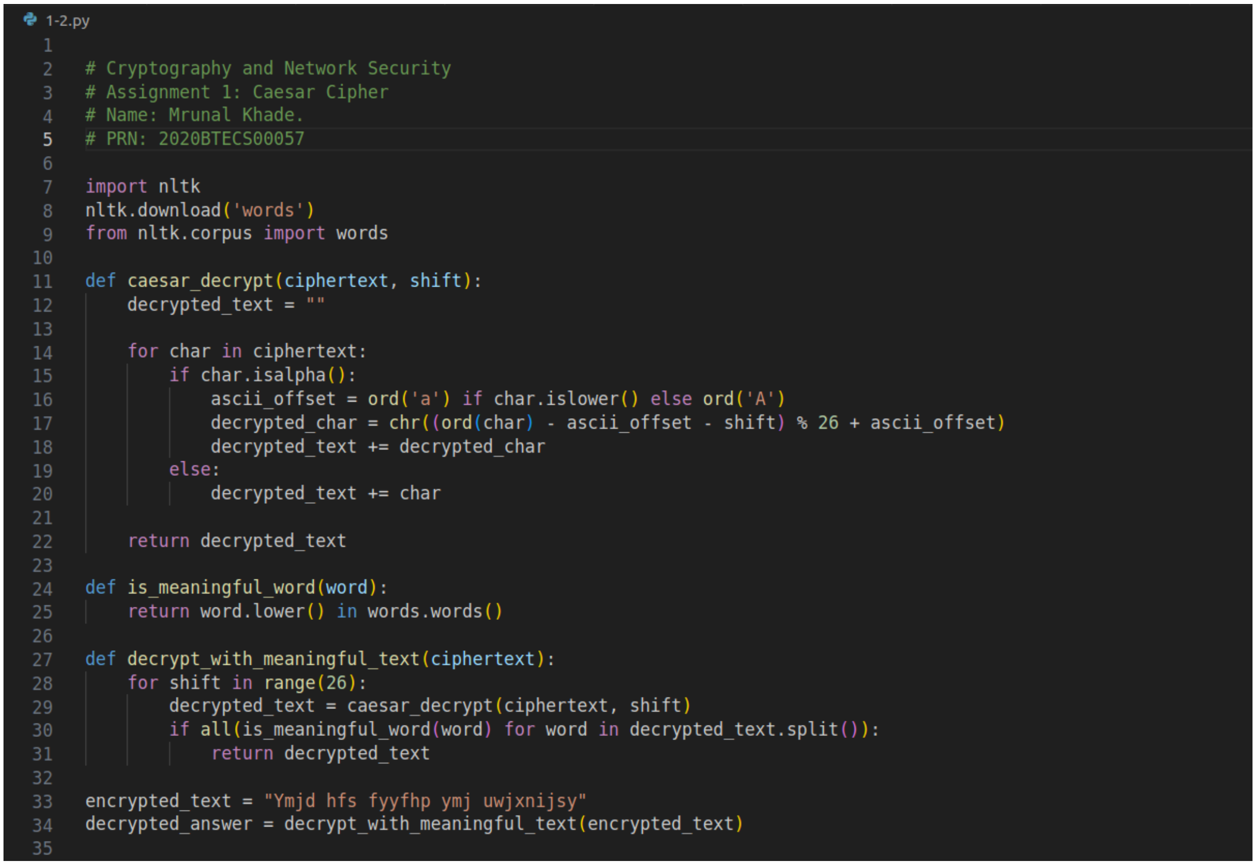
1. Start with a ciphertext message and the same fixed integer key used for encryption.
2. Initialize an empty string to store the plaintext.
3. For each character in the ciphertext message:
   - If the character is an uppercase letter (A-Z), perform the following steps:
     - Determine the position of the letter in the alphabet (A=0, B=1, ..., Z=25).
     - Subtract the key from this position (modulo 26 to handle wrapping around the alphabet).
     - Convert the new position back to a letter (A=0, B=1, ..., Z=25).
     - Append the resulting letter to the plaintext.

- If the character is a lowercase letter (a-z), perform the same steps as above but for the lowercase alphabet.

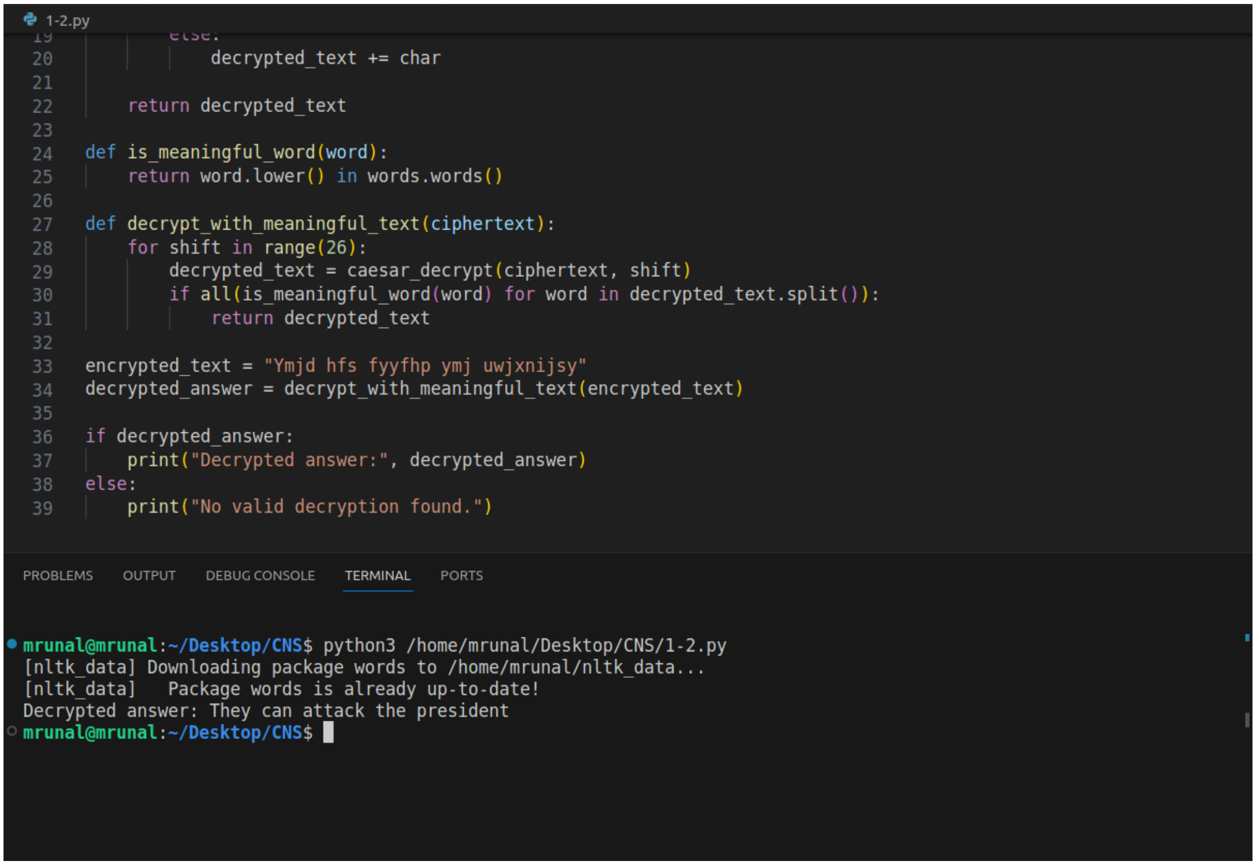
4. If the character is not a letter (e.g., space, punctuation, or a digit), leave it unchanged and append it to the plaintext.
5. Continue this process for each character in the ciphertext until the entire message is decrypted.
6. The resulting plaintext is the decrypted message.

However, it serves as a basic example of a substitution cipher and is not suitable for secure communication.

### Input and Output:



```python

# Cryptography and Network Security
# Assignment 1: Caesar Cipher
# Name: Mrunal Khade.
# PRN: 2020BTECS00057

import nltk
nltk.download('words')
from nltk.corpus import words

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

    for char in ciphertext:
        if char.isalpha():
            ascii_offset = ord('a') if char.islower() else ord('A')
            decrypted_char = chr((ord(char) - ascii_offset - shift) % 26 + ascii_offset)
            decrypted_text += decrypted_char
        else:
            decrypted_text += char

    return decrypted_text

def is_meaningful_word(word):
    return word.lower() in words.words()

def decrypt_with_meaningful_text(ciphertext):
    for shift in range(26):
        decrypted_text = caesar_decrypt(ciphertext, shift)
        if all(is_meaningful_word(word) for word in decrypted_text.split()):
            return decrypted_text

encrypted_text = "Ymjd hfs fyyfhp ymj uwjxnijsy"
decrypted_answer = decrypt_with_meaningful_text(encrypted_text)

```



```
        else:
            decrypted_text += char

    return decrypted_text

def is_meaningful_word(word):
    return word.lower() in words.words()

def decrypt_with_meaningful_text(ciphertext):
    for shift in range(26):
        decrypted_text = caesar_decrypt(ciphertext, shift)
        if all(is_meaningful_word(word) for word in decrypted_text.split()):
            return decrypted_text

encrypted_text = "Ymjd hfs fyyfhp ymj uwjxnijsy"
decrypted_answer = decrypt_with_meaningful_text(encrypted_text)

if decrypted_answer:
    print("Decrypted answer:", decrypted_answer)
else:
    print("No valid decryption found.")
```

```
mrunal@mrunal:~/Desktop/CNS$ python3 /home/mrunal/Desktop/CNS/1-2.py
[nltk_data] Downloading package words to /home/mrunal/nltk_data...
[nltk_data]   Package words is already up-to-date!
Decrypted answer: They can attack the president
mrunal@mrunal:~/Desktop/CNS$ 
```