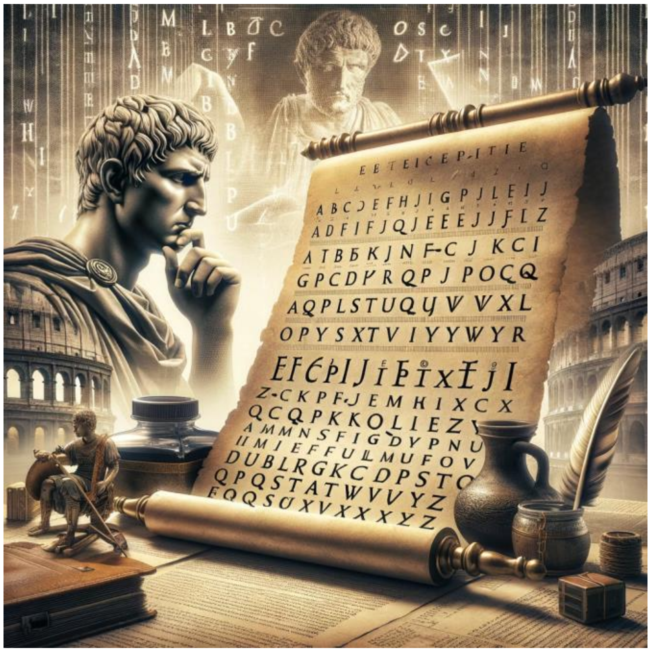
PRODIGY INFOTECH: CS_TASK_01

TASK_01: IMPLEMENT CAESAR CIPHER



PROJECT CAESAR CIPHER

Check out the project on my github :
https://github.com/sajilsaju/PRODIGY_CS_01.git

PROJECT OVERVIEW :

[Created a python program that can encrypt and decrypt text using Caesar cipher algorithm. And also allow users to input their message and a shift value to perform encryption and decryption.]

SAJIL MOHAMED.V

Report Submitted By :

Sajil Mohamed V

Phone : +91 8714194649

E-mail : sajilmohamed1234@gmail.com

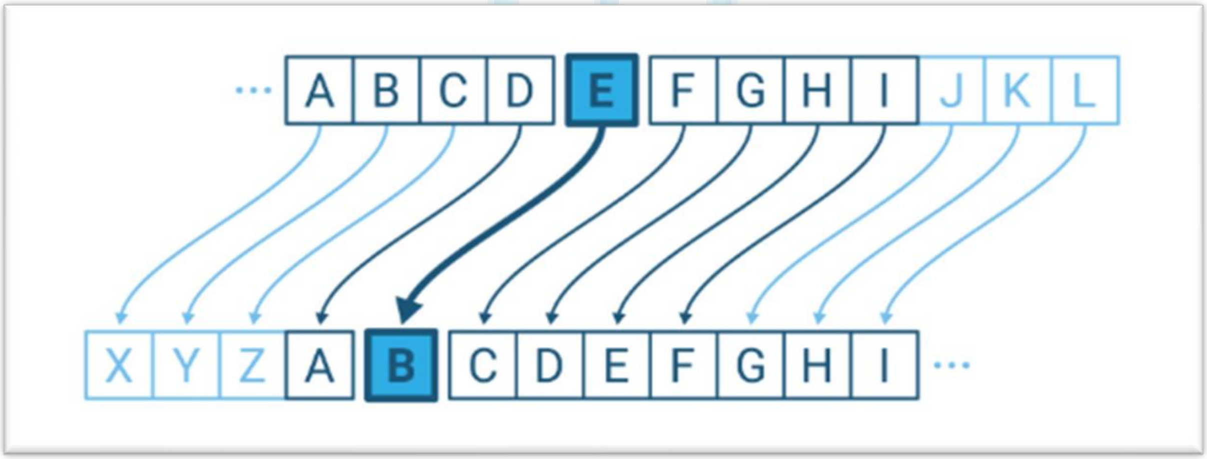
https://www.linkedin.com/in/sajilmohamed1718/

Table Of Contents

1. Introduction

1.1 About Caesar Cipher

The Caesar Cipher is one of the simplest and most widely known encryption techniques. The Caesar cipher is a simple and well-known encryption technique in cryptography. It's a type of substitution cipher that involves shifting each letter in a plaintext message by a fixed number of positions in the alphabet. For example, if the shift is three letters to the left, then the letter D would be replaced by A, and E would be replaced by B.



2. Objectives

The objective of this project to create a python program which allow users to input their message for encrypt and decrypt using the Caesar Cipher algorithm and also allow users to shift their value to perform encryption and decryption.

The following points specify the main objectives:

- To understand the fundamental concepts of classical encryption techniques.
- To implement the Caesar Cipher algorithm in Python.
- To enable encryption and decryption of user-inputted text using a specified shift value.
- To practice text manipulation and handling edge cases, such as letter wrapping.

3. Methodology

3.1 Algorithm Overview

The Caesar Cipher works by shifting each letter in the plaintext by a certain number of places. For example, with a shift of 3, 'A' becomes 'D', 'B' becomes 'E', and so on. If the shift passes 'Z', it wraps around to the beginning of the alphabet.

3.2 Implementation Steps

1. ***Input Handling***: The program prompts the user to enter a message, a shift value, and whether they want to encrypt or decrypt the message.

2. ***Character Processing****:*

- Each character in the input string is examined.
- If the character is an alphabet letter, it is shifted according to the provided shift value.
- Non-alphabet characters (e.g., punctuation, spaces) remain unchanged.

3. ***Wrap-Around Handling****:*

- If the shift results in a character going beyond 'z' or 'Z', it wraps around to the start of the alphabet.

4. ***Output****:*

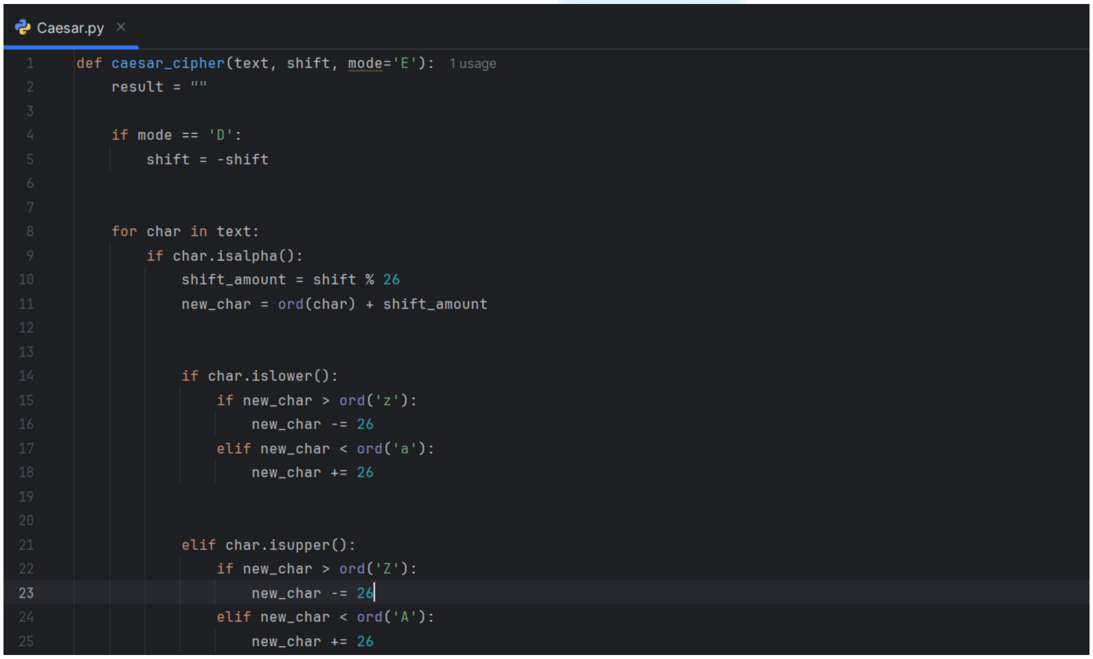
The processed message is then displayed to the user.

3.3 Program Code

The Caesar cipher project were created using “Python program”

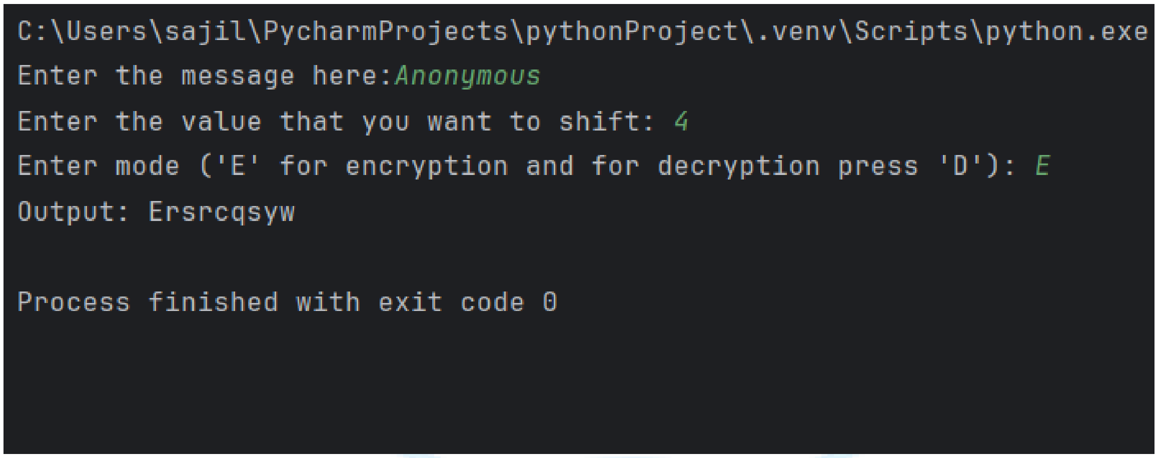
Source code :

```
def caesar_cipher(text, shift, mode='E'):
    result = ""
    if mode == 'D':
        shift = -shift
    for char in text:
        if char.isalpha():
            shift_amount = shift % 26
            new_char = ord(char) + shift_amount


            if char.islower():
                if new_char > ord('z'):
                    new_char -= 26
                elif new_char < ord('a'):
                    new_char += 26


            elif char.isupper():
                if new_char > ord('Z'):
                    new_char -= 26
                elif new_char < ord('A'):
                    new_char += 26


            result += chr(new_char)
        else:
            result += char
    return result
#User input section
message = input("Enter the message here:")
shift_value = int(input("Enter the value that you want to shift: "))
mode = input("Enter mode ('E' for encryption and for decryption press 'D'): ")
#output function
output = caesar_cipher(message, shift_value, mode)
print(f "Output: {output}")
```



```
def caesar_cipher(text, shift, mode='E'):
    result = ""

    if mode == 'D':
        shift = -shift


    for char in text:
        if char.isalpha():
            shift_amount = shift % 26
            new_char = ord(char) + shift_amount


            if char.islower():
                if new_char > ord('z'):
                    new_char -= 26
                elif new_char < ord('a'):
                    new_char += 26


            elif char.isupper():
                if new_char > ord('Z'):
                    new_char -= 26
                elif new_char < ord('A'):
                    new_char += 26
```

4. Output



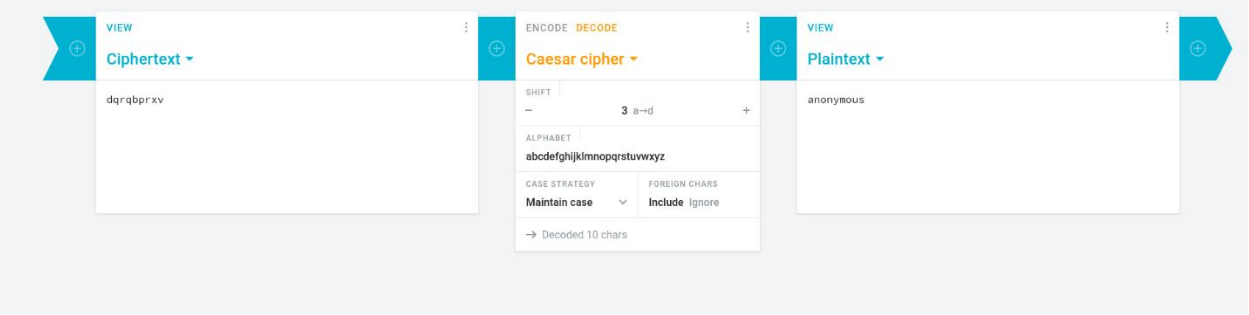
```
C:\Users\sajil\PycharmProjects\pythonProject\.venv\Scripts\python.exe
Enter the message here:Anonymous
Enter the value that you want to shift: 4
Enter mode ('E' for encryption and for decryption press 'D'): E
Output: Ersrcqsyw

Process finished with exit code 0
```

5. Testing and Validation

The program was tested with various inputs, including:

- Different shift values (both positive and negative).
- Messages with mixed case (upper and lower).
- Messages with non-alphabetical characters (e.g., punctuation, spaces).

All tests were successful, with the program accurately encrypting and decrypting messages according to the Caesar Cipher rules.



Ciphertext: dqrqbprxv

SHIFT: 3 a→d

ALPHABET: abcdefghijklmnopqrstuvwxyz

CASE STRATEGY: Maintain case

FOREIGN CHARS: Include Ignore

→ Decoded 10 chars

Plaintext: anonymous

6. Knowledge Gained

1. **Understanding of Classical Encryption Techniques**:
   - Gained insights into one of the earliest forms of encryption, the Caesar Cipher, which laid the groundwork for more complex cryptographic algorithms used today.
2. **Hands-On Experience with Python**:
   - Enhanced my Python programming skills by implementing the Caesar Cipher from scratch, including handling edge cases like letter wrapping and preserving the case of characters.
3. **Conceptualizing Encryption and Decryption**:
   - Learned the core concepts of encryption and decryption, including how shifts and key values are applied to transform plaintext into ciphertext and vice versa.
4. **Algorithmic Thinking**:
   - Improved my ability to think algorithmically by breaking down the encryption and decryption processes into clear, logical steps.
5. **Text Manipulation Techniques**:
   - Gained practical experience in text manipulation, working with strings, and understanding ASCII values for characters, which are fundamental in encryption processes.

7. Conclusion

The Caesar Cipher project provided valuable insights into both classical cryptography and practical Python programming. By completing this task, I have not only gained a deeper understanding of encryption techniques but also sharpened my coding and problem-solving skills. This project is a solid foundation for further exploration into more advanced cryptographic algorithms and cybersecurity practices.

8. Future Works

- Explore the implementation of more complex ciphers.
- Implement additional features, such as brute-force attack simulation to decrypt messages without knowing the shift value.
- Extend the project to include a graphical user interface (GUI) for easier interaction.

9. References

- Caesar Cipher - Wikipedia
- Python documentation: https://docs.python.org/
- www.google.com