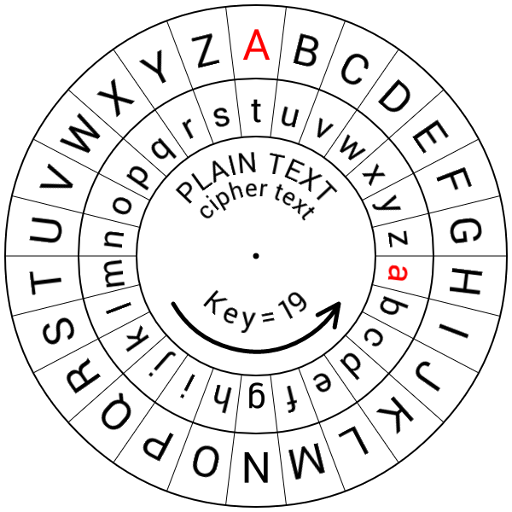
**Mini-Project-1: Caesar Cipher**





**Introduction**

**Objective:** To develop a Python-Based interactive Caesar Cipher for text encryption/decryption.

In today’s world, cryptography plays a crucial role in securing information, and one of the earliest encryption techniques known is the Caesar Cipher. This cipher, also called Caesar’s code, is a basic method of disguising messages by shifting each letter in the text by a fixed number of positions in the alphabet. For instance, shifting each letter by three positions to the left means A becomes X, B becomes Y, C becomes Z, and so on. This simple technique is named after Julius Caesar who used it to keep his communications confidential. Despite its historical significance, the Caesar cipher is easily deciphered and is not considered secure in modern communication practices.

**Problem Statement:**

Develop a Python-based program utilizing the Caesar-Cipher algorithm to enable users to encrypt and decrypt messages.’

1. **User Interaction:**

Create an interactive interface prompting users for message input, encryption or decryption selection, and shift amount.

1. **Cipher Implementation:**

Implement the Caesar Cipher algorithm using Python, utilizing ASCII values and string manipulation for letter shifting.

1. **Conditional Encoding:**

Based on user input, shift the letters in the message to encrypt or decrypt text accordingly.

1. **Handling Special Characters:**

Ensure the program retains non-alphabetic characters (numbers, symbols) unchanged during encryption/decryption.

1. **Interactive Loop:**

Allows users to perform multiple encryption actions until they opt to exit.

1. **Modular design:**

Utilize functions to enhance code readability and maintainability, encapsulating the encryption/decryption logic.

1. **Output Display:**

Present the resulting encoded or decoded text to the user in an understandable format.

1. **Range Consideration:**

Implement a mechanism to ensure shifts remain within the alphabet’s range (0-25) to prevent errors.

**Algorithm:**

Certainly! Here’s a simple algorithm for the Caesar cipher project:

1. **Start:**

- Initialize the program.

- Define the Caesar cipher function (`caesar`).

- Set the variable `should\_continue` to `True` for loop control.

2. **Caesar Cipher Function:**

Define the `caesar` function taking parameters: `start\_text`, `shift\_amount`, and `cipher\_direction`.

- Create an empty string `end\_text` to store the resulting text.

- Define a string of special characters.

- Iterate through each character in the `start\_text`.

- If the character is an alphabet:

- Determine the starting point based on case (lower or upper).

- Calculate the shifted character using the Caesar cipher formula.

- Append the shifted character to `end\_text`.

- If the character is numeric or a special character:

- Directly append it to `end\_text`.

- Print the resulting text indicating the direction (encoded or decoded).

3. **Loop:**

- Run a `while` loop based on the `should\_continue` variable.

- Display the logo and a welcome message.

- Prompt the user for the desired action: encode or decode.

- Take user input for the message and shift amount.

- Ensure the shift amount stays within the range (0-25).

- Adjust the shift amount for decoding if necessary.

- Call the `caesar` function with the provided inputs.

- Ask the user if they want to continue. If ‘no’, set `should\_continue` to `False`.

- Display “Goodbye!” when the loop ends.

4.**End:**

- Terminate the program.

This algorithm outlines the steps involved in implementing the Caesar cipher project using Python, including user interaction, cipher logic, loop control, and termination.

**Project Abstract**

This Python project implements a simplified Caesar cipher for encoding and decoding messages. Using a defined set of characters, the program shifts letters in a message by a user-specified amount, wrapping around the alphabet. Users can choose to encode or decode their messages with a specified shift. The interface prompts for input, allowing users to input their message, select the encryption or decryption process, and specify the shift amount. This basic implementation demonstrates the fundamental concept of the Caesar cipher and provides a simple command-line tool for encoding and decoding messages.

**Source Code**

The source code of this Python project is shown as follows:

logo = """

░█████╗░░█████╗░███████╗░██████╗░█████╗░██████╗░  ░█████╗░██╗██████╗░██╗░░██╗███████╗██████╗░

██╔══██╗██╔══██╗██╔════╝██╔════╝██╔══██╗██╔══██╗  ██╔══██╗██║██╔══██╗██║░░██║██╔════╝██╔══██╗

██║░░╚═╝███████║█████╗░░╚█████╗░███████║██████╔╝  ██║░░╚═╝██║██████╔╝███████║█████╗░░██████╔╝

██║░░██╗██╔══██║██╔══╝░░░╚═══██╗██╔══██║██╔══██╗  ██║░░██╗██║██╔═══╝░██╔══██║██╔══╝░░██╔══██╗

╚█████╔╝██║░░██║███████╗██████╔╝██║░░██║██║░░██║  ╚█████╔╝██║██║░░░░░██║░░██║███████╗██║░░██║

░╚════╝░╚═╝░░╚═╝╚══════╝╚═════╝░╚═╝░░╚═╝╚═╝░░╚═╝  ░╚════╝░╚═╝╚═╝░░░░░╚═╝░░╚═╝╚══════╝╚═╝░░╚═╝

"""

def caesar(start\_text, shift\_amount, cipher\_direction):

end\_text = ""

special\_characters = '!"#$%&\'()\*+,-./:;<=>?@[\\]^\_`{|}~'

for char in start\_text:

if char.isalpha():

start = ord('a') if char.islower() else ord('A')

shifted = (ord(char) - start - shift\_amount) % 26 + start

end\_text += chr(shifted)

elif char.isnumeric() or char in special\_characters:

end\_text += char

else:

end\_text += char

print(f"The {cipher\_direction}d text is {end\_text}")

should\_continue = True

while should\_continue:

print(logo)

print("Welcome to Caesar-Cipher!!!")

direction = input("Type 'encode' to encrypt, type 'decode' to decrypt:\n").lower()

text = input("Type your message:\n")

shift = int(input("Type the shift number:\n"))

shift %= 26 # Considering the alphabet range

if direction == 'decode':

shift = -shift # Reversing the shift for decryption

caesar(start\_text=text, shift\_amount=shift, cipher\_direction=direction)

result = input("Type 'yes' if you want to go again. Otherwise type 'no'.\n").lower()

if result == "no":

print("Goodbye!")

should\_continue = False

**Output and Screenshots**

Sample outputs and screenshots of this Python project are shown as follows:

**Sample-Output-1:** 

**Sample-Output-2:**



**Sample-Output-3:**



**Sample-Output-4:**



Sample-Output-5:



**Conclusion**

This Caesar cipher project served as an insightful journey into encryption techniques and basic Python programming. Through this hands-on experience, I gained a deeper understanding of string manipulation, ASCII values, and conditional statements. I learned how to implement a fundamental encryption algorithm, allowing message encoding and decoding based on user input. Moreover, this project reinforced the significance of user interaction within programs, employing loops and conditionals to create an interactive and user-friendly interface. Overall, it was an enriching experience that not only honed my coding skills but also introduced me to the intriguing world of cryptography, sparking my curiosity to explore further in this field.

**Reference**

Following resources were the reference for our project:

1. <https://en.wikipedia.org/wiki/Caesar_cipher>
2. <https://www.udemy.com/>
3. https://www.geeksforgeeks.org/