

PRODIGY INFOTECH INTERNSHIP

ASSIGNMENT 1

NAME: Arya Sunil Kumar

EMAIL: asunikumar369@gmail.com

INTERNSHIP: Cyber Security

Objective:

The objective of this assignment was to create a Python program that performs encryption and decryption using the Caesar Cipher algorithm. I wanted the program to allow a user to:

- Choose between encryption or decryption,
- Input a message,
- Enter a shift value (key),
- And receive the correctly processed output.

This assignment helped me understand the basic principles of classical cryptography, especially substitution ciphers, while also giving me hands-on experience in building a text-based tool using Python.

Skills Gained:

Working on this assignment helped me strengthen a variety of technical and problem-solving skills:

- I deepened my understanding of cryptographic concepts, especially how substitution ciphers work.
- I practiced writing modular Python code with custom functions and clear structure.
- I improved at string manipulation, particularly using ASCII conversions through `ord()` and `chr()`.
- I got better at handling user input and input validation, making the program more user-friendly and error-free.

- I learned how to preserve letter cases and how to skip special characters during encryption/decryption.

Tools Used:

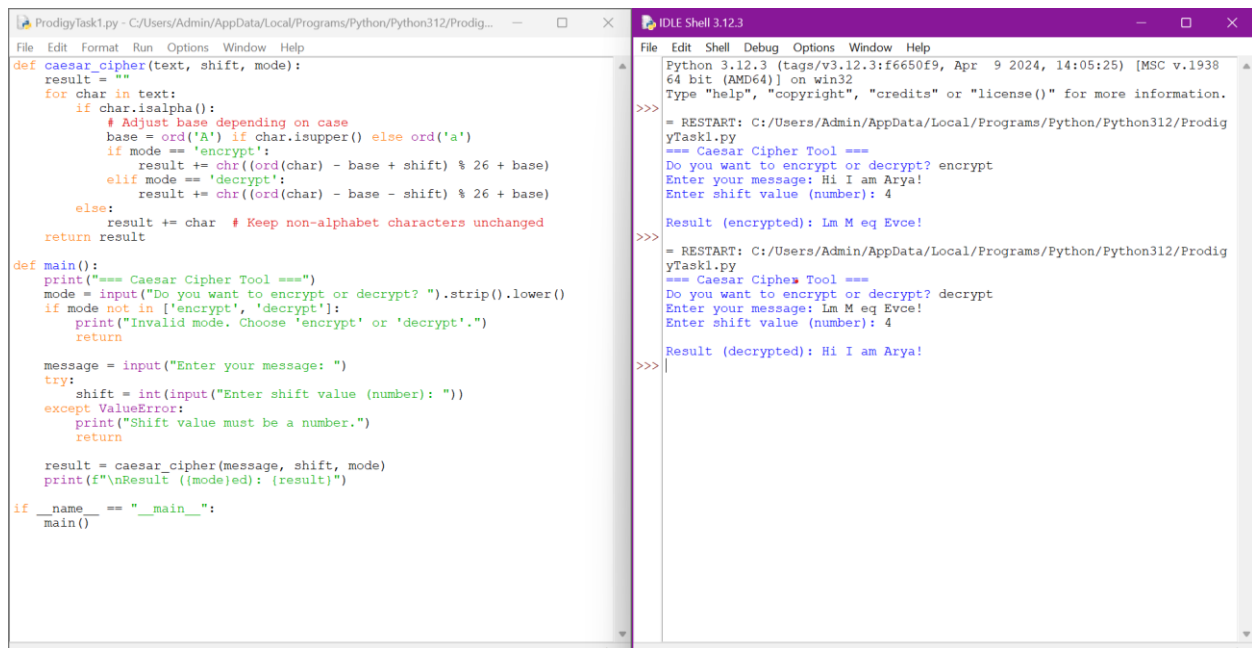
For this assignment, I used the following tools and technologies:

- Python 3.x: My primary programming language for implementation.
- VS Code / IDLE / PyCharm (any of these): To write, run, and test my Python code.

How I Achieved It:

- I started by researching how the Caesar Cipher works. I understood that it's a substitution cipher where each alphabet in the message is shifted by a fixed number of positions. For example, with a shift of 3, A becomes D, B becomes E, and so on. If we reach Z, it wraps around to A.
- Then, I began planning my code. I decided to write a single function named `caesar_cipher()` that could handle both encryption and decryption. Inside this function, I used modular arithmetic to shift the characters properly. I relied on the built-in `ord()` and `chr()` functions in Python to work with ASCII values for accurate letter transformations.
- I made sure to handle both uppercase and lowercase letters separately so that the case of the original text is preserved. I also ensured that non-alphabetic characters like commas, spaces, and numbers were not altered during encryption or decryption.
- For the user interaction part, I used `input()` prompts to ask the user whether they wanted to encrypt or decrypt a message. I also asked them to input the message and the shift value. To make the program more robust, I added error handling in case the user entered an invalid shift value (like a letter instead of a number).
- After coding, I thoroughly tested the program with different types of inputs:
 - A simple sentence with letters only
 - Sentences with punctuation and numbers
 - Very large shift values
 - Negative shift values

Glimpse of the Assignments:



The image shows a screenshot of a Python IDE with two windows. The left window displays the source code for a Caesar Cipher tool, and the right window shows the interactive shell output.

```
def caesar_cipher(text, shift, mode):
    result = ""
    for char in text:
        if char.isalpha():
            # Adjust base depending on case
            base = ord('A') if char.isupper() else ord('a')
            if mode == 'encrypt':
                result += chr((ord(char) - base + shift) % 26 + base)
            elif mode == 'decrypt':
                result += chr((ord(char) - base - shift) % 26 + base)
            else:
                result += char # Keep non-alphabet characters unchanged
    return result

def main():
    print("=== Caesar Cipher Tool ===")
    mode = input("Do you want to encrypt or decrypt? ").strip().lower()
    if mode not in ['encrypt', 'decrypt']:
        print("Invalid mode. Choose 'encrypt' or 'decrypt'.")
        return

    message = input("Enter your message: ")
    try:
        shift = int(input("Enter shift value (number): "))
    except ValueError:
        print("Shift value must be a number.")
        return

    result = caesar_cipher(message, shift, mode)
    print(f"\nResult ({mode}ed): {result}")

if __name__ == "__main__":
    main()
```

The right window shows the execution of the program. It prompts the user to choose between 'encrypt' and 'decrypt'. The user enters 'encrypt', and the program asks for a message and a shift value. The user enters 'Hi I am Arya!' and '4'. The program outputs the encrypted message: 'Lm M eq Evce!'. The user then chooses 'decrypt', enters the same message and shift value, and the program outputs the decrypted message: 'Hi I am Arya!'.

Conclusion

This assignment allowed me to combine cryptography and programming in a practical way. I was able to create a fully functional Caesar Cipher tool that is interactive, user-friendly, and logically sound. It helped me develop my coding skills, reinforced my understanding of encryption techniques, and gave me insight into how even simple ciphers require precise planning and logic to implement.