**PANDIT DEENDAYAL ENERGY UNIVERSITY**

**SCHOOL OF TECHNOLOGY**

****

**Course: Information Security**

**Course Code: 20CP304P**

**LAB MANUAL**

**Semester 5**

**Name: Ronak Panchal**

**Roll Number: 22BCP192**

**Class: Division 3, G5**

**Submitted To: Dr. Rutvij Jhaveri**

**Practical 1:** **Study and implement a program for Caesar Cipher and modify it.**

**Theory:** In cryptography, the Caesar cipher, also known as Caesar's cipher, the shift cipher, Caesar's code, or Caesar shift, is one of the simplest and most well-known encryption techniques. This substitution cipher replaces each letter in the plaintext with a letter a fixed number of positions down the alphabet. For instance, with a left shift of 3, D is replaced by A, E becomes B, and so forth. The method is named after Julius Caesar, who used it in his private correspondence.

**Example:** The transformation can be represented by aligning two alphabets; the cipher alphabet is the plain alphabet rotated left or right by some number of positions. For instance, here is a Caesar cipher using a left rotation of three places, equivalent to a right shift of 23 (the shift parameter is used as the [key](https://en.wikipedia.org/wiki/Key_(cryptography))):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Plain** | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| **Cipher** | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |

When encrypting, a person looks up each letter of the message in the "plain" line and writes down the corresponding letter in the "cipher" line.

Plaintext: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Ciphertext: QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD

Deciphering is done in reverse, with a right shift of 3.

The encryption can also be represented using [modular arithmetic](https://en.wikipedia.org/wiki/Modular_arithmetic) by first transforming the letters into numbers, according to the scheme, A → 0, B → 1, ..., Z → 25.

**Source Code:**

**alphabet = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']**

**should\_continue = True**

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

**end\_text = ""**

**for char in start\_text:**

**if char in alphabet:**

**position = alphabet.index(char)**

**if cipher\_direction == "e":**

**new\_position = (position + shift\_amount) % len(alphabet)**

**elif cipher\_direction == "d":**

**new\_position = (position - shift\_amount) % len(alphabet)**

**end\_text += alphabet[new\_position]**

**else:**

**end\_text += char**

**return end\_text**

**while should\_continue:**

**direction = input("Type 'e' to encrypt, type 'd' to decrypt:\n").lower()**

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

**shift = int(input("Type the shift number:\n")) % 26**

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

**if direction == 'e':**

**print(f"The encoded text is {result}")**

**elif direction == 'd':**

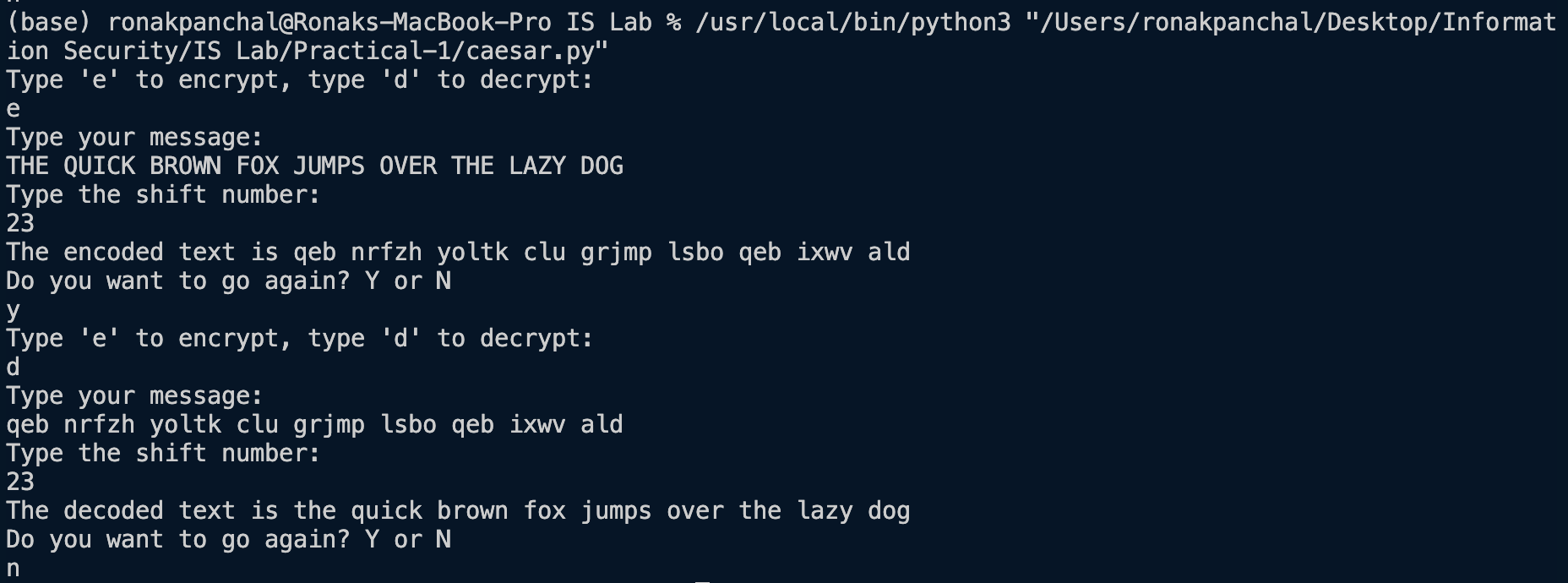
**print(f"The decoded text is {result}")**

**repeat = input("Do you want to go again? Y or N\n").lower()**

**if repeat == 'n':**

**should\_continue = False**

**Output:**



**Time Complexity:**  **O(n) for both Encryption and Decryption (n is length of plaintext).**

**Space Complexity:**  **O(1) for both Encryption and Decryption.**

**Revised**

**Logic:**

**Create a list of fibonacci series of size of text and shift left or right the character with the corresponding** **number in fibonacci series.**

**Source Code:**

**#Create a list of fibonacci number of size of text**

**def generate\_fibonacci(n):**

**fib\_sequence = [0, 1]**

**while len(fib\_sequence) < n:**

**fib\_sequence.append(fib\_sequence[-1] + fib\_sequence[-2])**

**return fib\_sequence[:n]**

**def caesar\_with\_fibonacci(start\_text, cipher\_direction):**

**n = len(start\_text)**

**fib\_sequence = generate\_fibonacci(n)**

**end\_text = ""**

**for i in range(n):**

**char = start\_text[i]**

**shift\_amount = fib\_sequence[i] % 26**

**if char in alphabet:**

**position = alphabet.index(char)**

**if cipher\_direction == "e":**

**new\_position = (position + shift\_amount) % len(alphabet)**

**elif cipher\_direction == "d":**

**new\_position = (position - shift\_amount) % len(alphabet)**

**end\_text += alphabet[new\_position]**

**else:**

**end\_text += char**

**return end\_text**

**alphabet = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']**

**should\_continue = True**

**while should\_continue:**

**direction = input("Type 'e' to encrypt, type 'd' to decrypt:\n").lower()**

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

**result = caesar\_with\_fibonacci(start\_text = text, cipher\_direction = direction)**

**if direction == 'e':**

**print(f"The encoded text is {result}")**

**elif direction == 'd':**

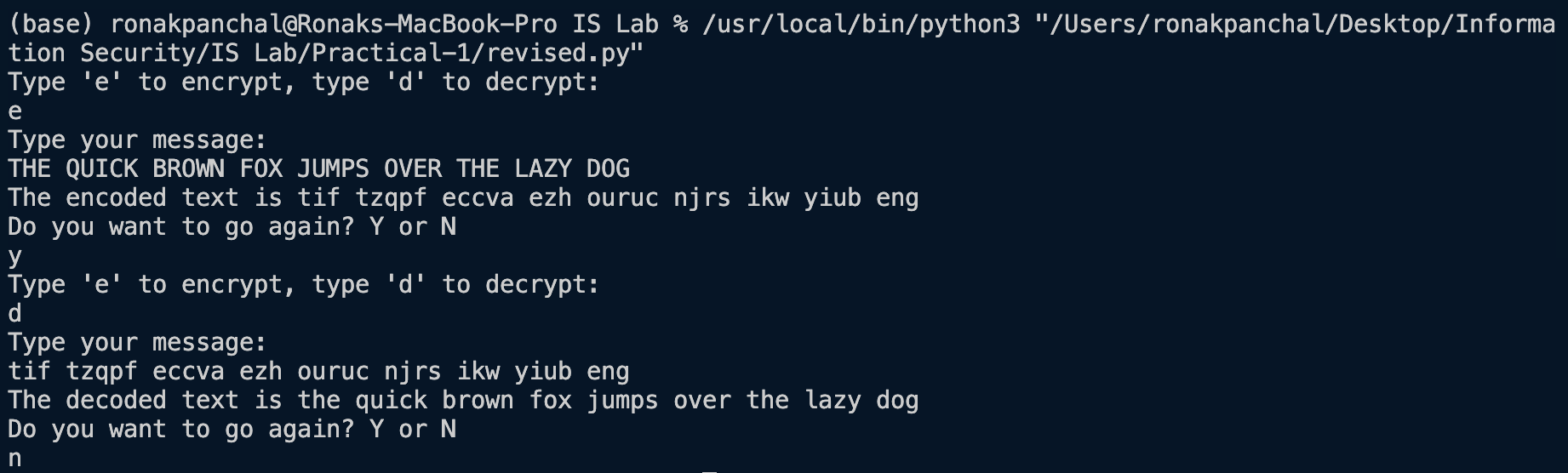
**print(f"The decoded text is {result}")**

**repeat = input("Do you want to go again? Y or N\n").lower()**

**if repeat == 'n':**

**should\_continue = False**

**Output:**



**Time Complexity:**  **O(n) for both Encryption and Decryption (n is length of plaintext).**

**Space Complexity:**  **O(n) for both Encryption and Decryption.**