COMSATS University Islamabad, Attock Campus

# Name: Uzair Shah

# Reg No. SP24-BSE-028

# IS Lab Mid term

# Date: 20-Oct-25

**Question 1**

**Caesar Cipher**

def caesar\_cipher\_encrypt(message, shift):

    ciphertext = ""

    shift = shift % 26

    for char in message:

        if 'A' <= char <= 'Z':

            start\_point = ord('A')

            original\_pos = ord(char) - start\_point

            new\_pos = (original\_pos + shift) % 26

            new\_char = chr(new\_pos + start\_point)

            ciphertext += new\_char

        elif 'a' <= char <= 'z':

            start\_point = ord('a')

            original\_pos = ord(char) - start\_point

            new\_pos = (original\_pos + shift) % 26

            new\_char = chr(new\_pos + start\_point)

            ciphertext += new\_char

        else:

            ciphertext += char

    return ciphertext

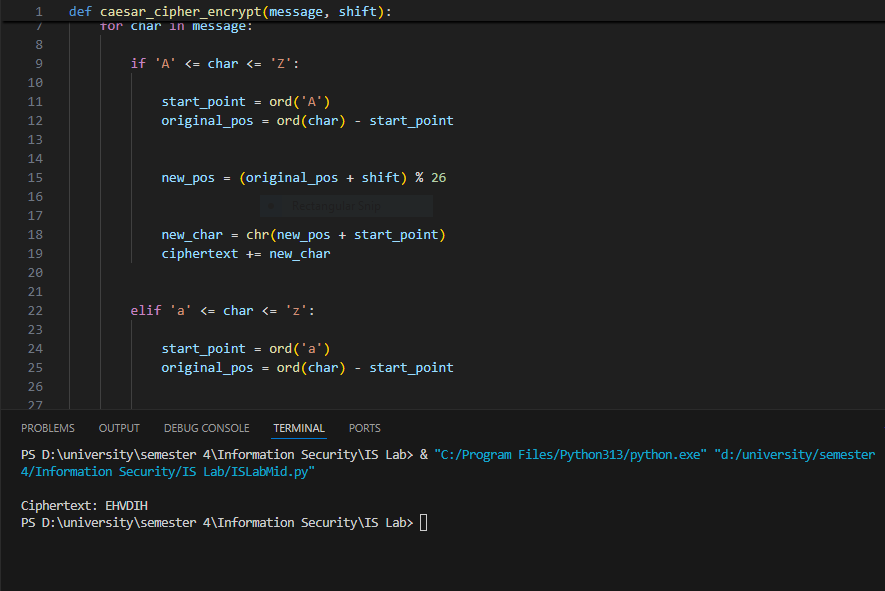
user\_message = "BESAFE"

user\_shift = 3

encrypted\_text = caesar\_cipher\_encrypt(user\_message, user\_shift)

print(f"\nCiphertext: {encrypted\_text}")

# output



**Question 2**

**Vigenère Cipher (Encryption Only)**

def vigenere\_encrypt(plaintext, key):

    plaintext = "".join(filter(str.isalpha, plaintext.upper()))

    key = "".join(filter(str.isalpha, key.upper()))

    if not key:

        print("Error: Key must contain at least one letter.")

        return ""

    ciphertext = []

    key\_len = len(key)

    key\_index = 0

    for p\_char in plaintext:

        p\_val = ord(p\_char) - ord('A')

        k\_char = key[key\_index % key\_len]

        k\_val = ord(k\_char) - ord('A')

        c\_val = (p\_val + k\_val) % 26

        c\_char = chr(c\_val + ord('A'))

        ciphertext.append(c\_char)

        key\_index += 1

    return "".join(ciphertext)

if \_\_name\_\_ == "\_\_main\_\_":

    print("--- Vigenère Cipher Encryptor ---")

    user\_plaintext = input("Enter plaintext: ")

    user\_key = input("Enter key: ")

    ciphertext\_result = vigenere\_encrypt(user\_plaintext, user\_key)

    if ciphertext\_result:

        print(f"\nCiphertext: {ciphertext\_result}")

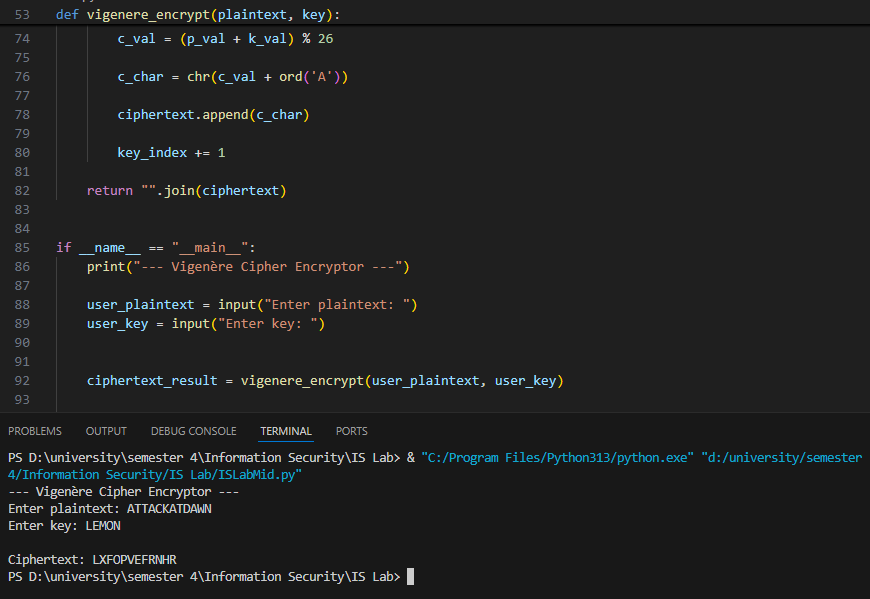
# Example from your prompt:

# Enter plaintext: ATTACKATDAWN

# Enter key: LEMON

# Ciphertext: LXFOPVEFRNHR

# Output



**Question 3**

**Debugging Task (Simple XOR Encryption)**

def xor\_encrypt(text, key):

    if not key:

        print("Error: Key cannot be empty.")

        return ""

    result = ""

    key\_len = len(key)

    for i in range(len(text)):

        key\_char = key[i % key\_len]

        xor\_value = ord(text[i]) ^ ord(key\_char)

        result += chr(xor\_value)

    return result

if \_\_name\_\_ == "\_\_main\_\_":

    print("--- Simple XOR Encryption Debugged ---")

    text = "secret message"

    key = "cipher"

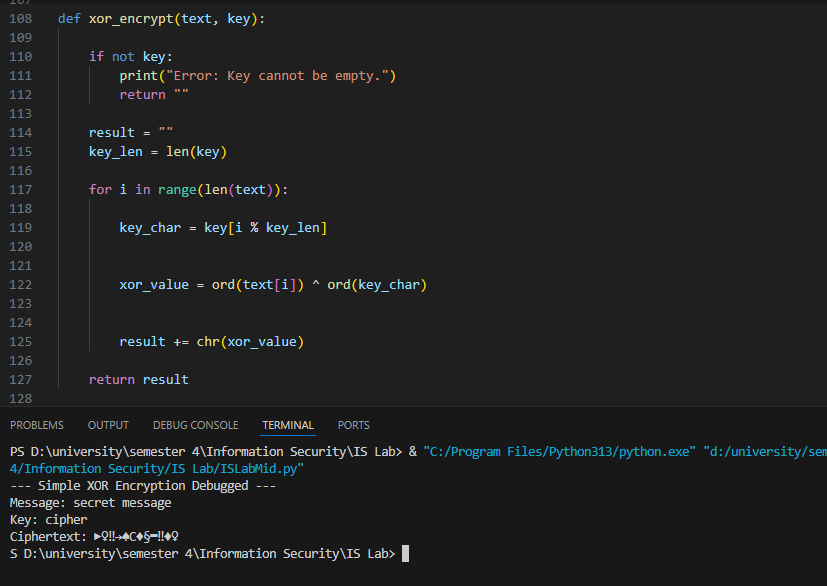
    print(f"Message: {text}")

    print(f"Key: {key}")

    ciphertext = xor\_encrypt(text, key)

    print("Ciphertext:", ciphertext)

# output



**Question 4**

**DES and AES**

a) **One Difference Between DES and AES:**

The main difference is in their **core design structure**. DES uses a **Feistel network** (a complex, iterative structure), while AES uses a much simpler and more modern **Substitution-Permutation Network (SPN)**.

b) **AES Block Size and One Key Size:**

The **block size** for AES is fixed at **128 bits**. A common **key size** used is **128 bits** (though 192 and 256 bits are also standard).

c) **One Reason Why AES is More Secure Than DES:**

AES uses a significantly longer **key length**. DES's key is effectively only 56 bits, which is weak and easily cracked by modern computers, whereas AES's minimum key size of 128 bits is still considered highly secure against brute-force attacks.