COMSATS University Islamabad, Attock campus
Department of Computer Science

Program: BS-SE

Sp24-BSE-051

Brekhna Gul

21st-October-2025

Lab mid term

Information security

Ma’am Ambreen GUL

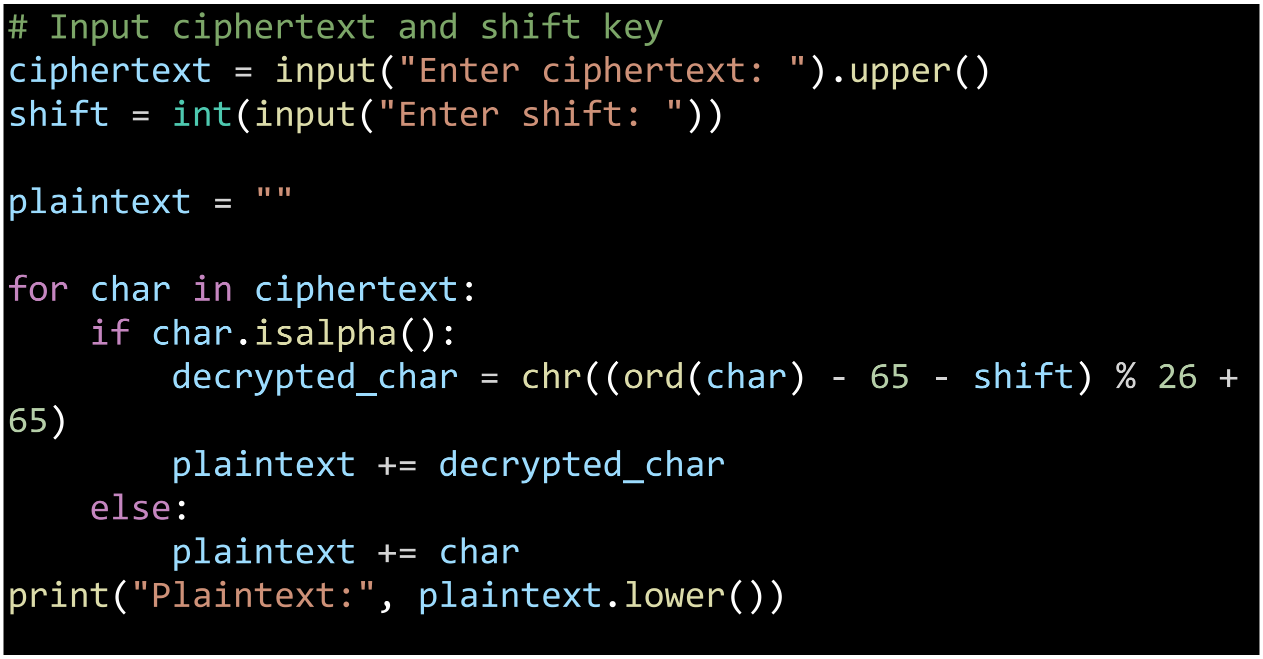
Question 2 – Caesar Cipher (Decryption)

Write a Python program to decrypt a message that was encrypted using the Caesar Cipher. The program should take ciphertext (LXFOPVEFRNHR) and key (5) as input and display the plaintext.

Answer:

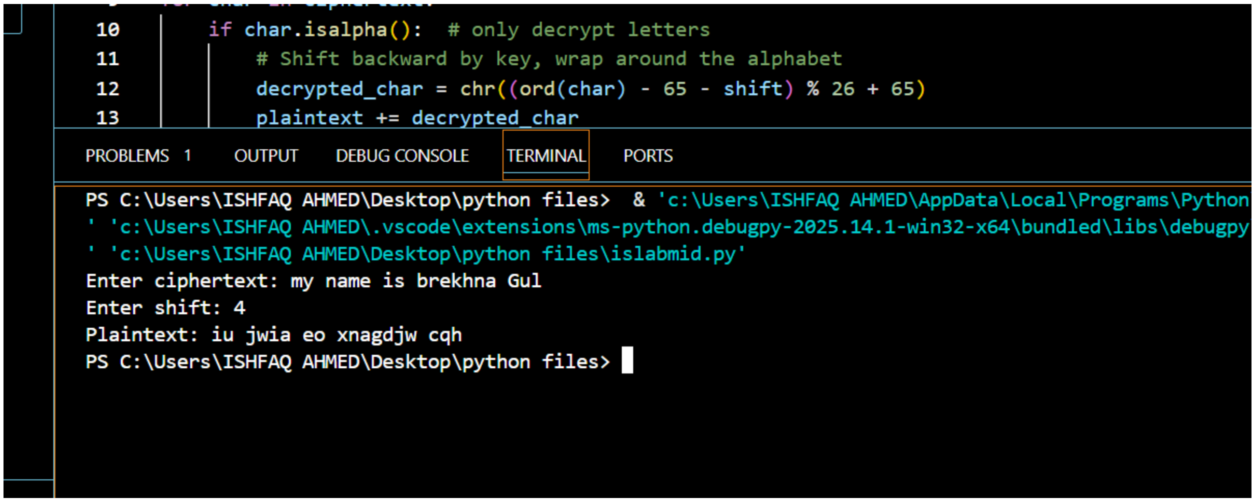
Caesar Cipher Decryption Program

The code is:



```python
# Input ciphertext and shift key
ciphertext = input("Enter ciphertext: ").upper()
shift = int(input("Enter shift: "))

plaintext = ""

for char in ciphertext:
    if char.isalpha():
        decrypted_char = chr((ord(char) - 65 - shift) % 26 + 65)
        plaintext += decrypted_char
    else:
        plaintext += char
print("Plaintext:", plaintext.lower())
```

Expected output:



```
    for char in ciphertext:
        if char.isalpha():  # only decrypt letters
            # Shift backward by key, wrap around the alphabet
            decrypted_char = chr((ord(char) - 65 - shift) % 26 + 65)
            plaintext += decrypted_char
```

```
PS C:\Users\ISHFAQ AHMED\Desktop\python files>  & 'c:\Users\ISHFAQ AHMED\AppData\Local\Programs\Python
' 'c:\Users\ISHFAQ AHMED\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy
' 'c:\Users\ISHFAQ AHMED\Desktop\python files\islabmid.py'
Enter ciphertext: my name is brekhna Gul
Enter shift: 4
Plaintext: iu jwia eo xnagdjw cqh
PS C:\Users\ISHFAQ AHMED\Desktop\python files> 
```

Question 3 – Vigenère Cipher (Decryption Only):

Write a Python program to decrypt a ciphertext using the Vigenère Cipher. Ask the user for ciphertext and key, and display the decrypted plaintext.

Example:

Enter ciphertext: LXFOPVEFRNHR

Enter key: LEMON

Plaintext: ATTACKATDAWN

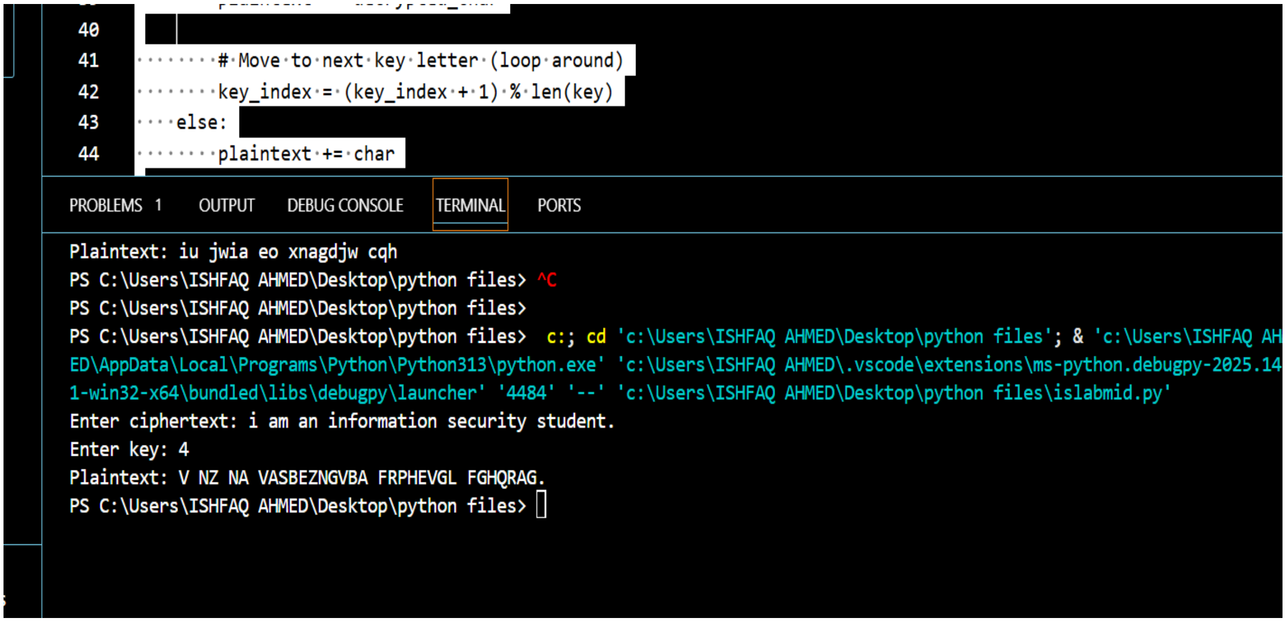
Answer:

Vigenère Cipher Decryption Program

The code is:

```
# Get inputs
ciphertext = input("Enter ciphertext: ").upper()
key = input("Enter key: ").upper()

plaintext = ""
key_index = 0

for char in ciphertext:
    if char.isalpha():
        shift = ord(key[key_index]) - 65
        decrypted_char = chr((ord(char) - 65 - shift) % 26 + 
65)
        plaintext += decrypted_char

        key_index = (key_index + 1) % len(key)
    else:
        plaintext += char

print("Plaintext:", plaintext)
```

Expected output:



```
        # Move to next key letter (loop around)
        key_index = (key_index + 1) % len(key)
    else:
        plaintext += char
```

```
Plaintext: iu jwia eo xnagdjw cqh
PS C:\Users\ISHFAQ AHMED\Desktop\python files> ^C
PS C:\Users\ISHFAQ AHMED\Desktop\python files>
PS C:\Users\ISHFAQ AHMED\Desktop\python files>  c:; cd 'c:\Users\ISHFAQ AHMED\Desktop\python files'; & 'c:\Users\ISHFAQ AH
ED\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\ISHFAQ AHMED\.vscode\extensions\ms-python.debugpy-2025.14
1-win32-x64\bundled\libs\debugpy\launcher' '4484' '--' 'c:\Users\ISHFAQ AHMED\Desktop\python files\islabmid.py'
Enter ciphertext: i am an information security student.
Enter key: 4
Plaintext: V NZ NA VASBEZNGVBA FRPHEVGL FGHQRAG.
PS C:\Users\ISHFAQ AHMED\Desktop\python files> 
```

Question 4 – Debugging Task (Caesar Cipher Code)

The following program is intended to encrypt text using the Caesar Cipher, but it contains an error. Fix the mistake so that it runs correctly and gives the right output.

●
```
def caesar_encrypt(text, shift):
    result = ""
    for char in text:
        if char.isalpha():
            result += chr(ord(char) + shift)
        else:
            result += char
            return result

  msg = input("Enter message: ")
  s = int(input("Enter shift: "))
  print("Ciphertext:", caesar_encrypt(msg, s))
```

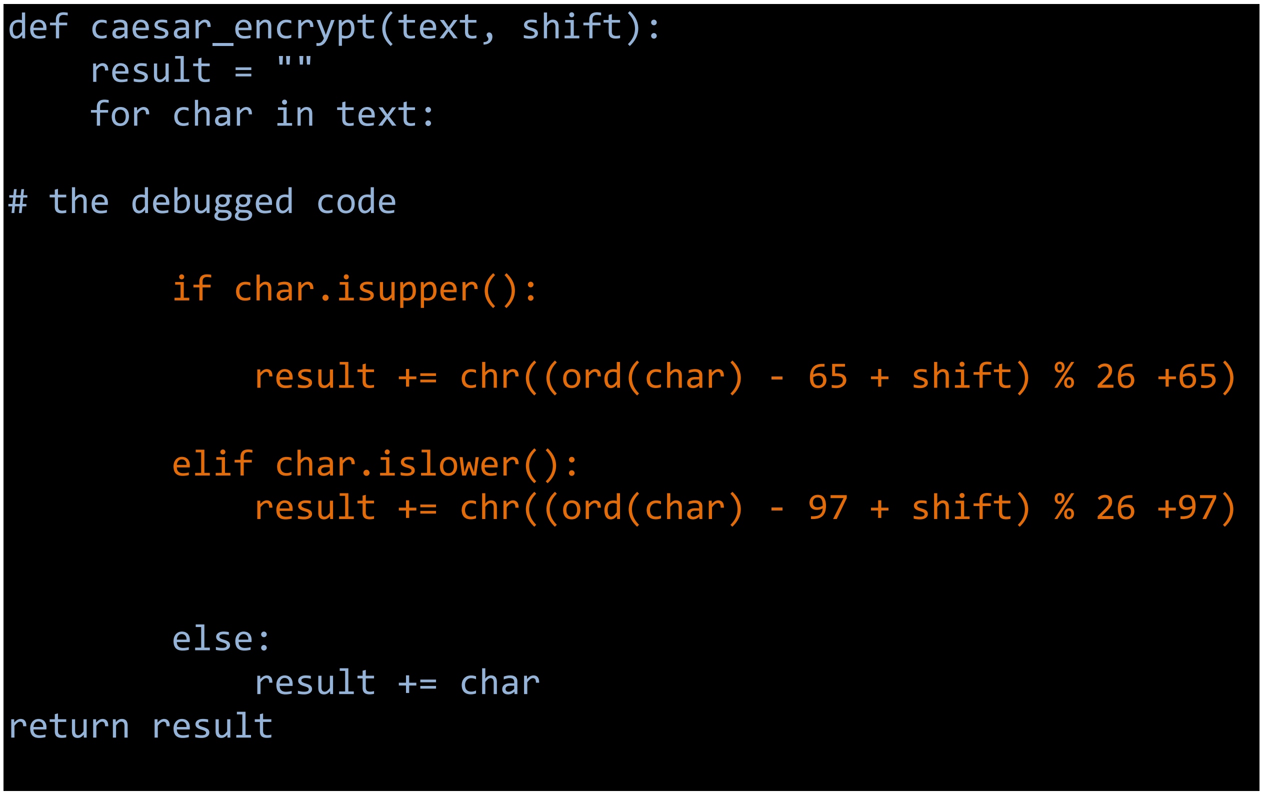
Hint: The code doesn't wrap around alphabets (A–Z or a–z). Use modular arithmetic to fix the shifting logic.

Answer:

Caesar Cipher Encryption:

- ➢ Added wrap-around logic for uppercase letters (A–Z)
- ➢ Added wrap-around logic for lowercase letters (a–z)

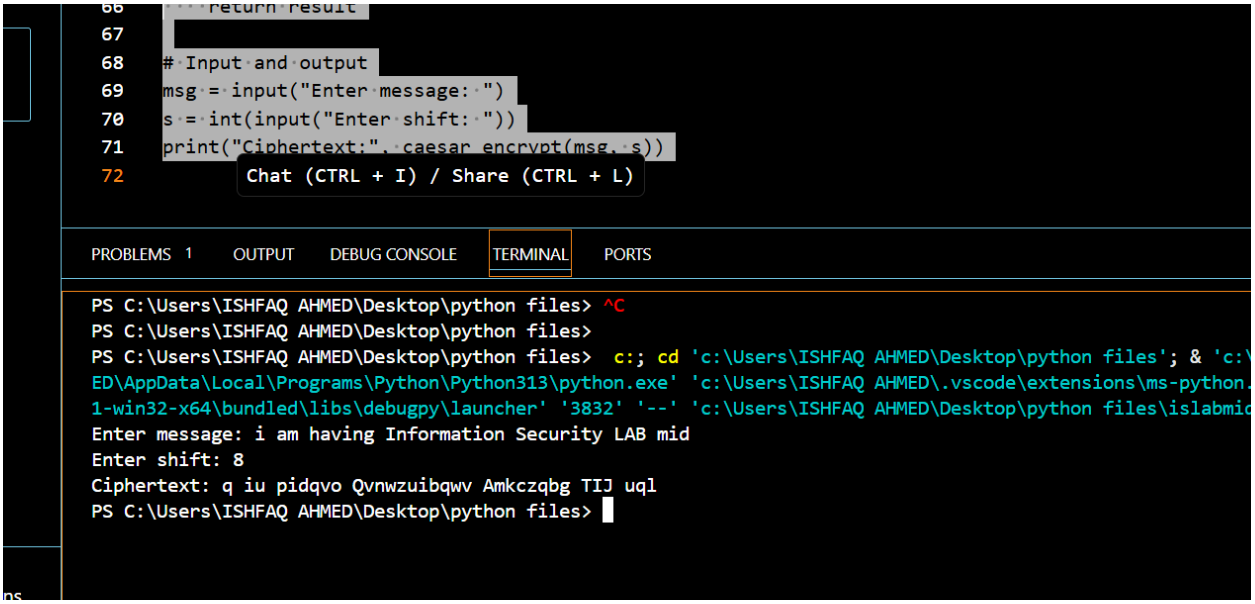
The debugged encrypted code:



```
def caesar_encrypt(text, shift):
    result = ""
    for char in text:

# the debugged code

        if char.isupper():

            result += chr((ord(char) - 65 + shift) % 26 +65)

        elif char.islower():
            result += chr((ord(char) - 97 + shift) % 26 +97)


        else:
            result += char
return result
```

```
# Input and output
msg = input("Enter message: ")
s = int(input("Enter shift: "))
print("Ciphertext:", caesar_encrypt(msg, s))
```

Expected output:



```
    return result

# Input and output
msg = input("Enter message: ")
s = int(input("Enter shift: "))
print("Ciphertext:", caesar_encrypt(msg, s))
```

```
PS C:\Users\ISHFAQ AHMED\Desktop\python files> ^C
PS C:\Users\ISHFAQ AHMED\Desktop\python files>
PS C:\Users\ISHFAQ AHMED\Desktop\python files>  c:; cd 'c:\Users\ISHFAQ AHMED\Desktop\python files'; & 'c:\
ED\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\ISHFAQ AHMED\.vscode\extensions\ms-python.
1-win32-x64\bundled\libs\debugpy\launcher' '3832' '--' 'c:\Users\ISHFAQ AHMED\Desktop\python files\islabmic
Enter message: i am having Information Security LAB mid
Enter shift: 8
Ciphertext: q iu pidqvo Qvnwzuibqwv Amkczqbg TIJ uql
PS C:\Users\ISHFAQ AHMED\Desktop\python files>
```

Question 5 – Conceptual: DES and AES

Answer briefly:

a) Write one similarity between DES and AES.

Answer:

Both **DES** (Data Encryption Standard) and **AES** (Advanced Encryption Standard) are **symmetric key block cipher algorithms**.

This means the **same secret key** is used for both **encryption** and **decryption** of data. They both also divide the plaintext into **fixed-size blocks** and apply multiple rounds of substitution and permutation to secure the data.

b) What does CBC mode stand for in block ciphers

Answer:

CBC stands for **Cipher Block Chaining** mode.
In CBC mode, before each plaintext block is encrypted, it is **XORed (**combined**)** with the **previous ciphertext block**. This chaining process ensures that **identical plaintext blocks produce different ciphertext blocks**, making the encryption more secure. The first block uses an **Initialization Vector (IV)** instead of a previous block.

c) Why is AES faster than DES

Answer:

AES is faster than DES because it operates on **larger 128-bit data blocks** and is designed to work efficiently on **modern hardware and processors**.
It also uses **simpler mathematical operations** (like substitution and permutation on bytes) compared to DES's **bit-level operations**, making AES both **faster and more secure**.

Ended: