|  |  |
| --- | --- |
| **Ex. No. 2** | **Implement Substitution Cipher** |
| **Date of Exercise** | **19.08.2024** |

**Aim**

To implement the simple substitution technique named Caesar cipher using python language.

**Description**

To encrypt a message with a Caesar cipher, each letter in the message is changed using a simple rule: shift by three. Each letter is replaced by the letter three letters ahead in the alphabet. A becomes D, B becomes E, and so on. For the last letters, we can think of the alphabet as a circle and "wrap around". W becomes Z, X becomes A, Y becomes B, and Z becomes C. To change a message back, each letter is replaced by the one three before it.

1. **Perform Caesar Cipher encryption**

**Algorithm**

STEP-1: Read the plain text from the user.

STEP-2: Read the key value from the user.

STEP-3: If the key is positive then encrypt the text by adding the key with

each character in the plain text.

STEP-4: Else subtract the key from the plain text.

STEP-5: Display the cipher text obtained above.

**Program**

print("URK21CS1128")

plaintxt = input("Enter the plain text: ")

key = int(input("Enter the key: "))

ct = ""

for i in plaintxt:

if i.islower():

ct += chr((ord(i)-97+key)%26+97)

elif i.isupper():

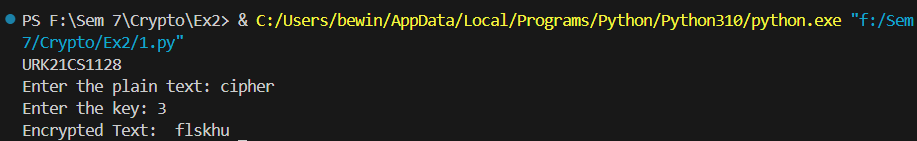
ct += chr((ord(i)-65+key)%26+65)

else:

ct += " "

print("Encrypted Text: ",ct)

**Output Screenshot**



1. **Perform encryption and decryption using affine cipher.**

**Algorithm**

STEP 1: Read the problem statement

STEP 2: Get the plaintext, value of ‘a’ and ‘b’ from the user

STEP 3: Create an empty string ‘ct’ to store the cipher text

STEP 4: Perform the encryption operation.

STEP 5: Print the encrypted text

STEP 6: Create an empty string ‘pt’ to store the plain text

STEP 7: Perform the decryption operation.

STEP 8: Print the decrypted text.

**Program:**

print("URK21CS1128")

plaintext = input("Enter the plaintext: ")

a = int(input("Enter a value: "))

b = int(input("Enter b value: "))

ct = ""

for char in plaintext:

if char.isalpha():

base = ord('A') if char.isupper() else ord('a')

ct += chr(((a\*(ord(char)-base)+b)%26)+base)

else:

ct += char

print("After Encryption: ",ct)

pt = ""

inverse = pow(a,-1,26)

for char in ct:

if char.isalpha():

base = ord('A') if char.isupper() else ord('a')

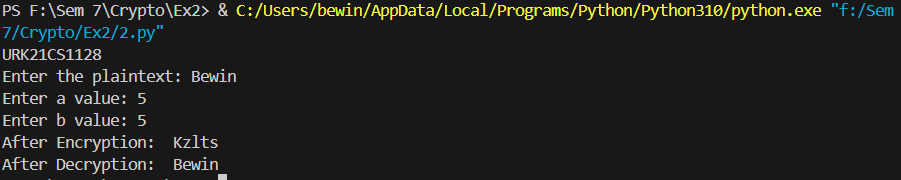
pt += chr(((inverse \* (ord(char)-base-b))%26)+base)

else:

pt += char

print("After Decryption: ",pt)

**Output:**



1. **Perform encryption and decryption using vigenere cipher.**

**Algorithm:**

STEP 1: Read the problem statement.

STEP 2: Get the plaintext and key from the user

STEP 3: Convert the key into a list of characters.

STEP 4: If the key length is less than the plaintext length, repeat the key until it matches it.

STEP 5: Do the Encryption operation and print the ciphertext.

STEP 6: Do the Decryption operation and print the decrypted text.

**Program:**

print("URK21CS1128")

plaintext = input("Enter the plain text: ")

key = input("Enter the key: ")

key = list(key)

if len(plaintext) == len(key):

key = "".join(key)

else:

for i in range(len(plaintext) - len(key)):

key.append(key[i % len(key)])

key = "".join(key)

ct = ""

for i in range(len(plaintext)):

if plaintext[i].isalpha():

base = ord('A') if plaintext[i].isupper() else ord('a')

ct += chr((ord(plaintext[i])- base + ord(key[i])-ord('A')) %26 +base)

else:

ct += plaintext[i]

print("After Encryption: ",ct)

pt = ""

for i in range(len(ct)):

if ct[i].isalpha():

base = ord('A') if ct[i].isupper() else ord('a')

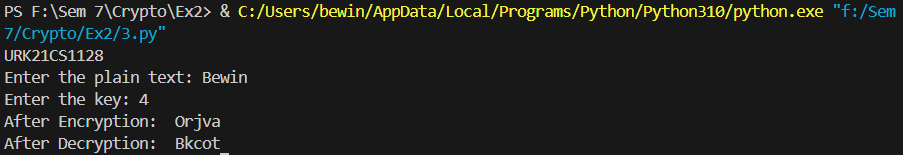
pt += chr((ord(ct[i]) - ord(key[i])+26)%26+base)

else:

pt += ct[i]

print("After Decryption: ",pt)

**Output:**



**Result**

The program has executed successfully and the output is displayed in the console.