|  |  |
| --- | --- |
| Name: | Khushi M Appannavar |
| Roll No.: | 524 |
| USN: | 01FE21BCS126 |

IS Algorithm Implementation

# Question:

Implement each round of DES encryption and decryption and demonstrate its usage with a provided plaintext and key.

# Solution:

from Crypto.Cipher import DES

from Crypto.Util.Padding import pad, unpad

from Crypto.Random import get\_random\_bytes

import hashlib

def generate\_des\_key(password):

# Generate an 8-byte DES key from the password

key = hashlib.sha256(password.encode()).digest()[:8]

return key

def des\_encrypt(data, key):

# Encrypt data using DES with CBC mode

cipher = DES.new(key, DES.MODE\_CBC)

padded\_data = pad(data.encode('utf-8'), DES.block\_size)

encrypted\_data = cipher.encrypt(padded\_data)

return cipher.iv, encrypted\_data

def des\_decrypt(iv, encrypted\_data, key):

# Decrypt data using DES with CBC mode

cipher = DES.new(key, DES.MODE\_CBC, iv)

decrypted\_data = cipher.decrypt(encrypted\_data)

return unpad(decrypted\_data, DES.block\_size)

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

# Main execution: get password and data from user, then encrypt and decrypt the data

password = input("Enter password to generate DES key: ")

key = generate\_des\_key(password)

print("Given Key:" + password)

data = input("Enter data to encrypt: ")

print("Original Text:" + data)

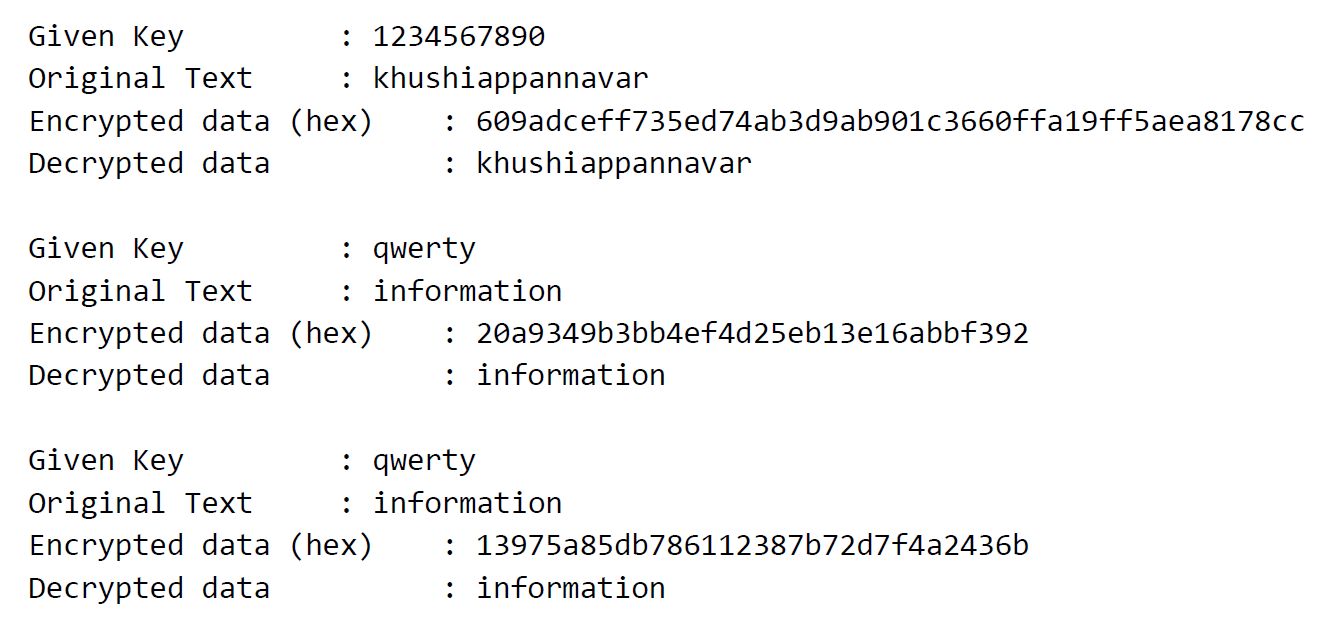
iv, encrypted\_data = des\_encrypt(data, key)

print("\nEncrypted data (hex):", encrypted\_data.hex())

decrypted\_data = des\_decrypt(iv, encrypted\_data, key)

print("Decrypted data:", decrypted\_data.decode('utf-8'))

# Test Cases



# Additional

## Question:

Can we implement the same code in ECB? justify your reason. If yes, state the difference

## Answer:

Yes, it is possible.

## Justification:

|  |  |
| --- | --- |
| **ECB (Electronic Code Book):** | **CBC (Cipher Block Chaining):** |
| Encrypts each block independently. | Encrypts each block by XORing it with the previous ciphertext block. |
| Identical plaintext blocks result in identical ciphertext blocks, revealing patterns. | Identical plaintext blocks can result in different ciphertext blocks. |
| Less secure for large datasets. | More secure for large datasets. |
| Suitable for very short, single-block data. | Requires an IV for randomness, More complex and slower than ECB. |
| ECB mode should be avoided for large datasets or sensitive information. | CBC mode is generally preferred for most applications due to its enhanced security and resistance to pattern analysis. |