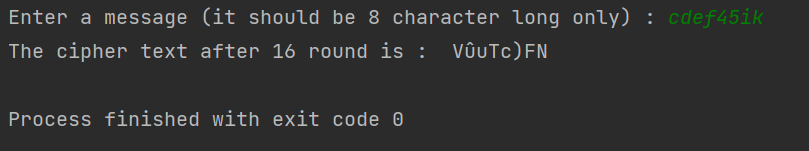
**Source Code :**

# DES

import random  
  
def DESWorking(binary\_input: str):  
  
 left = binary\_input[:32]  
 right = binary\_input[32:]  
 # print(left + right)  
 k = random.randint(1, 42949672)  
 k = format(k, '032b')  
 xor = []  
 for i in range(32):  
 xor.append(int(right[i]) ^ int(k[i]))  
 new\_xor = []  
 for i in range(32):  
 new\_xor.append(int(xor[i]) ^ int(left[i]))  
 right = "".join(str(i) for i in new\_xor)  
 return right, left  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 message = input("Enter a message (it should be 8 character long only) : ")  
 binary\_of\_message = "".join(format(ord(i), '08b') for i in message)  
 k = 0  
 left = 0  
 right = 0  
 while k < 16:  
 left, right = DESWorking(binary\_of\_message)  
 binary\_of\_message = left + right  
 k += 1  
 total = left + right  
 print("The cipher text after 16 round is : ", end=" ")  
 for i in range(0, len(total), 8):  
 print(chr(int(total[i:i+8], 2)), end="")  
 print()

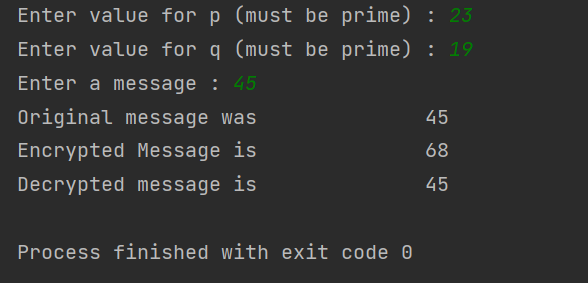
**Output:**



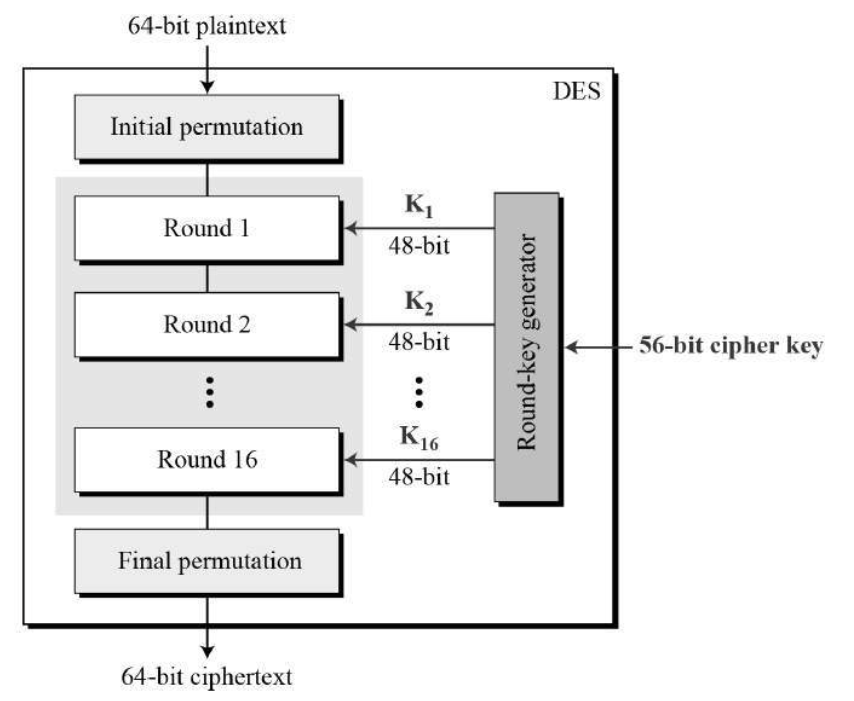
# RSA

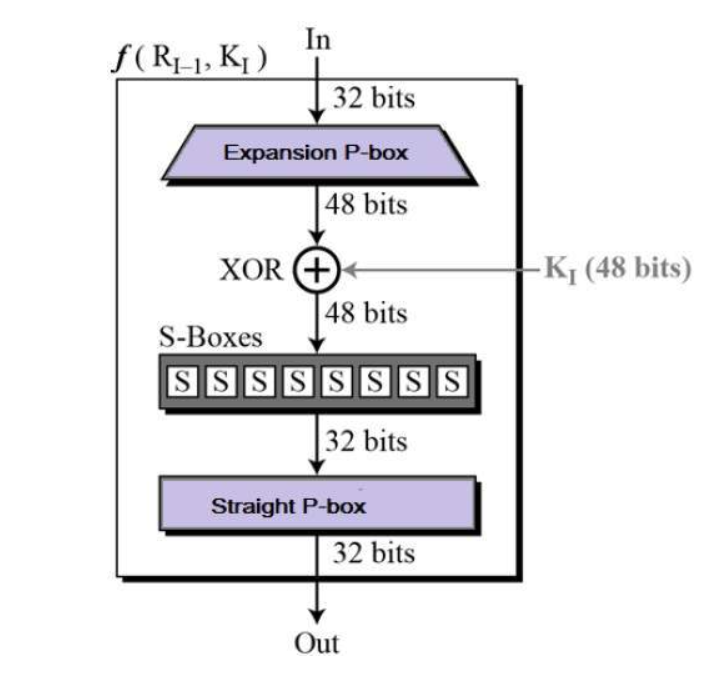
import random   
  
def gcdByEuclideanMethod(a, b):  
 return a if b == 0 else gcdByEuclideanMethod(b, a%b)  
  
def encryptMessage(message, e, n):  
 cipher = (message \*\* e) % n  
 return cipher  
  
def decryptMessage(cipher, d, n):  
 plain = (cipher \*\* d) % n  
 return plain   
  
if \_\_name\_\_ == '\_\_main\_\_':  
 p = int(input("Enter value for p (must be prime) : "))  
 q = int(input("Enter value for p (must be prime) : "))  
 n = p \* q  
 phi\_n = (p-1) \* (q-1)  
 e = [i for i in range(3, phi\_n, 2) if gcdByEuclideanMethod(phi\_n, i) == 1]  
 e = e[random.randint(0, len(e))]  
 d = [i for i in range(3, phi\_n) if (i \* e) % phi\_n == 1]  
 d = d[random.randint(0, len(d)-1)]  
 # print(e, d)  
 message = int(input("Enter a message : "))  
 cipher = encryptMessage(message, e, n)  
 decrypted = decryptMessage(cipher, d, n)  
 # print("Public key <{}, {}> \nPrivate key <{}, {}>".format(e, n, d, n))  
 print("Original message was {:15}\nEncrypted Message is {:15}\nDecrypted message is {:15}".format(message, cipher, decrypted))

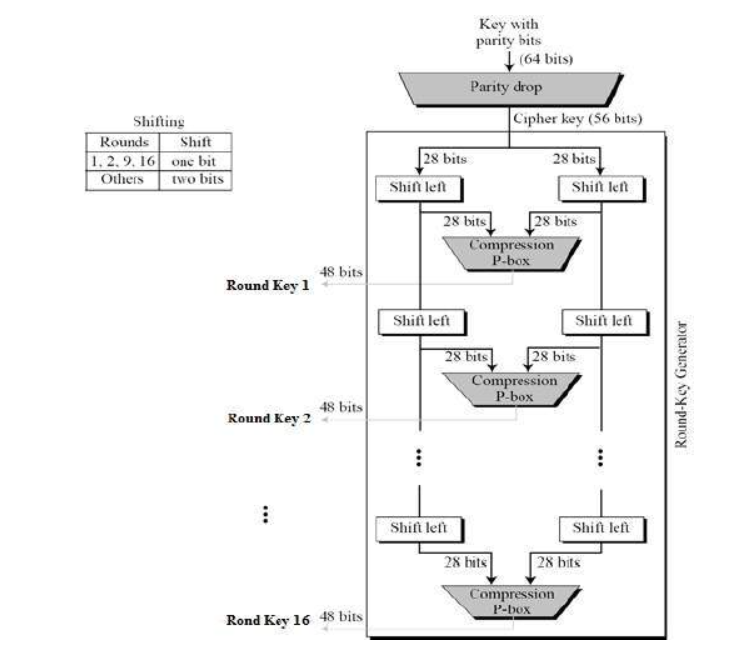
**Output :**



***DES:***

******

******

******

***Key generation in DES***

