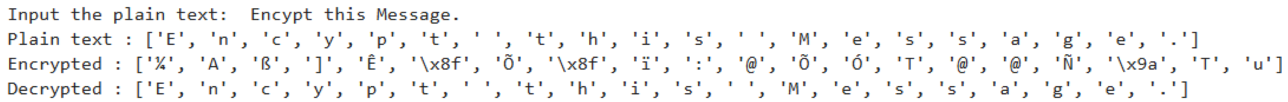
Name: Gaurang Vaghela
Rollno: TEAD-22561
Mini Project Lab
Practical 1
Problem Statement: Implementation of S-DES (Data Encryption Standard)

Code:

```
import numpy as np

IP = (2, 6, 3, 1, 4, 8, 5, 7)
IP_INVERSE = (4, 1, 3, 5, 7, 2, 8, 6)
E = (4, 1, 2, 3, 2, 3, 4, 1)
P10 = (3, 5, 2, 7, 4, 10, 1, 9, 8, 6)
P8 = (6, 3, 7, 4, 8, 5, 10, 9)
P4 = (2, 4, 3, 1)
S0 = np.asarray([1,0,3,2,3,2,1,0,0,2,1,3,3,1,3,2]).reshape(4,4)
S1 = np.asarray([0,1,2,3,2,0,1,3,3,0,1,0,2,1,0,3]).reshape(4,4)

KEY = "0111111101"
left = lambda x : x[:len(x)//2]
right = lambda x : x[len(x)//2:]

def permutation(original, key):
    return "".join(original[i-1] for i in key)

def shift(bit):
    return left(bit)[1:] + left(bit)[0] + right(bit)[1:] + right(bit)[0]

def genfirstKey():
    rotated = shift(permutation(KEY,P10))
    return permutation(rotated, P8)

def gensecondKey():
    rotated = shift(shift(shift(permutation(KEY,P10))))
    return permutation(rotated, P8)

def xor(bit, key):
    bit = map(int,bit)
    key = map(int,key)
    return "".join( str((i+j) % 2) for i,j in zip(bit,key))
```

```
def SBox(bit, sbox):
    row = int(bit[0] + bit[3], 2)
    col = int(bit[1] + bit[2], 2)
    return "{0:02b}".format(sbox[row][col])

def Fk(bit, key):
    L,R = left(bit), right(bit)
    bit = xor(permutation(R,E),key)
    bit = SBox(left(bit), S0) + SBox(right(bit), S1)
    bit = permutation(bit, P4)
    return xor(bit, L)

def encrypt(plain):
    bit = permutation(plain, IP)
    tmp = Fk(bit, genfirstKey())
    bit = right(bit) + tmp
    bit = Fk(bit, gensecondKey())
    return permutation(bit + tmp, IP_INVERSE)

def decrypt(enc):
    bit = permutation(enc, IP)
    tmp = Fk(bit, gensecondKey())
    bit = right(bit) + tmp
    bit = Fk(bit, genfirstKey())
    return permutation(bit + tmp, IP_INVERSE)

def main():
    _plain = input("Input the plain text: ")
    _plain = list(_plain)
    print("Plain text : {}".format(_plain))

    _encry = map(lambda x : encrypt(bin(ord(x)).lstrip("-0b").zfill(8)), list(_plain))
    _encry = list(_encry)

    _decry = map(lambda x : chr(int(decrypt(x),2)), list(_encry))

    _encry = map(lambda x : chr(int("0b"+x,2)), _encry)

    _encry = list(_encry)
    print("Encrypted : {}".format(_encry))

    _decry = list(_decry)
    print("Decrypted : {}".format(_decry))

if __name__ == "__main__":
    main()
```

Output:



```
Input the plain text:  Encypt this Message.
Plain text : ['E', 'n', 'c', 'y', 'p', 't', ' ', 't', 'h', 'i', 's', ' ', 'M', 'e', 's', 's', 'a', 'g', 'e', '.']
Encrypted : ['¼', 'A', 'ß', ']', 'Ê', '\x8f', 'Õ', '\x8f', 'ï', ':', '@', 'Õ', 'Ó', 'T', '@', '@', 'Ñ', '\x9a', 'T', 'u']
Decrypted : ['E', 'n', 'c', 'y', 'p', 't', ' ', 't', 'h', 'i', 's', ' ', 'M', 'e', 's', 's', 'a', 'g', 'e', '.']
```