In [1]:

```python
# Permutation boxes and S-boxes
InputPerm = [2, 6, 3, 1, 4, 8, 5, 7]
FinalPerm = [4, 1, 3, 5, 7, 2, 8, 6]

EPtable = [4, 1, 2, 3, 2, 3, 4, 1]
S0 = [[1, 0, 3, 2], [3, 2, 1, 0], [0, 2, 1, 3], [3, 1, 3, 2]]
S1 = [[0, 1, 2, 3], [2, 0, 1, 3], [3, 0, 1, 0], [2, 1, 0, 3]]
P4table = [2, 4, 3, 1]

P10Table = [3, 5, 2, 7, 4, 10, 1, 9, 8, 6]
P8Table = [6, 3, 7, 4, 8, 5, 10, 9]


def perm(B, table):
    output = 0
    for i, e in enumerate(table):
        if i >= e:
            output |= (B & (128 >> (e - 1))) >> (i - (e - 1))
        else:
            output |= (B & (128 >> (e - 1))) << ((e - 1) - i)
    return output


def keyGen(key):
    def leftShift(pk):
        ls, rs = pk >> 5, pk & 0b0000011111
        ls = ((ls >> 4) & 1) | (ls << 1) & 0b0000011110
        rs = ((rs >> 4) & 1) | (rs << 1) & 0b0000011110
        return ((ls << 5) & 0b1111100000) | rs

    k = perm(key, P10Table)
    shiftOne = leftShift(k)
    shiftTwo = leftShift(leftShift(shiftOne))
    subkey1 = perm(shiftOne, P8Table)
    subkey2 = perm(shiftTwo, P8Table)
    return subkey1, subkey2


def InitialPermutation(B):
    return perm(B, InputPerm)


def swapper(B):
    return (B << 4 | B >> 4) & 0xff


def mixer(key, B):
    def f(skey, rNib):
        temp = skey ^ perm(swapper(rNib), EPtable)
        l, r = temp & 0xf0, temp & 0x0f
        lr, lc, rr, rc = ((l >> 2) & 0x2) | (l & 0x1), (l >> 1) & 0x3, ((r >> 2) & 0x2)
| (r & 0x1), (r >> 1) & 0x3
        sboxout = swapper((S0[lr][lc] << 2) + S1[rr][rc])
        return perm(sboxout, P4table)

    lNib, rNib = B & 0xf0, B & 0x0f
    return lNib ^ f(key, rNib) | rNib


def FinalPermutation(B):
    return perm(B, FinalPerm)


def encrypt(text, key):
    t = mixer(keyGen(key)[0], InitialPermutation(text))
    return FinalPermutation(mixer(keyGen(key)[1], swapper(t)))
```

```
def decrypt(cipher, key):
    t = mixer(keyGen(key)[1], InitialPermutation(cipher))
    return FinalPermutation(mixer(keyGen(key)[0], swapper(t)))


if __name__ == "__main__":
    plaintext = input("Enter the plaintext (Please enter text in inverted commas, eg : \"
abcde\"): ")
    key = int(input("Enter Key: "))
    encipher = ""
    decipher = ""
    for i in plaintext:
        encipher += chr(encrypt(ord(i), key))
    print("Cipher Text: " + encipher)
    for i in encipher:
        decipher += chr(decrypt(ord(i), key))
    print("Decipher Text: " + decipher)
```

```
Enter the plaintext (Please enter text in inverted commas, eg : "abcde"): Hello I am Jess
ica
Enter Key: 1010
Cipher Text: EIHHäqLqlAqàI¬¬dÉl
Decipher Text: Hello I am Jessica
```

In [ ]: