DES

Code

```
binary1 = {0:'0000', 1:'0001', 2:'0010', 3:'0011', 4:'0100', 5:'0101', 6:'0110', 7:'0111', 8:'1000', 9:'1001', 10:'1010', 11:'1011', 12:'1100', 13:'1101', 14:'1110', 15:'1111'}

binary = {'0':'0000', '1':'0001', '2':'0010', '3':'0011', '4':'0100', '5':'0101', '6':'0110', '7':'0111', '8':'1000', '9':'1001', 'A':'1010', 'B':'1011', 'C':'1100', 'D':'1101', 'E':'1110', 'F':'1111'}

shift =  {1:1, 2:1, 3:2, 4:2, 5:2, 6:2, 7:2, 8:2, 9:1, 10:2, 11:2, 12:2, 13:2, 14:2, 15:2, 16:1}

q = {'00':0, '01':1, '10':2, '11':3}

q1 ={'0000':'0', '0001':'1', '0010':'2', '0011':'3', '0100':'4', '0101':'5', '0110':'6', '0111':'7', '1000':'8', '1001':'9', '1010':'A', '1011':'B', '1100':'C', '1101':'D', '1110':'E', '1111':'F'}

q11 ={'0000':0, '0001':1, '0010':2, '0011':3, '0100':4, '0101':5, '0110':6, '0111':7, '1000':8, '1001':9, '1010':10, '1011':11, '1100':12, '1101':13, '1110':14, '1111':15}


PC1 = [57,49,41,33,25,17,9,1,58,50,42,34,26,18,10,2,59,51,43,35,27,19,11,3,60,52,44,36,63,55,47,39,31,23,15,7,62,54,46,38,30,22,14,6,61,53,45,37,29,21,13,5,28,20,12,4]

PC2 = [14,17,11,24,1,5,3,28,15,6,21,10,23,19,12,4,26,8,16,7,27,20,13,2,41,52,31,37,47,55,30,40,51,45,33,48,44,49,39,56,34,53,46,42,50,36,29,32]

IP1 = [58,50,42,34,26,18,10,2,60,52,44,36,28,20,12,4,62,54,46,38,30,22,14,6,64,56,48,40,32,24,16,8,57,49,41,33,25,17,9,1,59,51,43,35,27,19,11,3,61,53,45,37,29,21,13,5,63,55,47,39,31,23,15,7]

EBIT =[32,1,2,3,4,5,4,5,6,7,8,9,8,9,10,11,12,13,12,13,14,15,16,17,16,17,18,19,20,21,20,21,22,23,24,25,24,25,26,27,28,29,28,29,30,31,32,1]

P =   [16,7,20,21,29,12,28,17,1,15,23,26,5,18,31,10,2,8,24,14,32,27,3,9,19,13,30,6,22,11,4,25]

IPP = [40,8,48,16,56,24,64,32,39,7,47,15,55,23,63,31,38,6,46,14,54,22,62,30,37,5,45,13,53,21,61,29,36,4,44,12,52,20,60,28,35,3,43,11,51,19,59,27,34,2,42,10,50,18,58,26,33,1,41,9,49,17,57,25]



s1 = [14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7,0,15,7,4,14,2,13,1,10,6,12,11,9,5,3,8,4,1,14,8,13,6,2,11,15,12,9,7,3,10,5,0,15,12,8,2,4,9,1,7,5,11,3,14,10,0,6,13]

s2 = [15,1,8,14,6,11,3,4,9,7,2,13,12,0,5,10,3,13,4,7,15,2,8,14,12,0,1,10,6,9,11,5,0,14,7,11,10,4,13,1,5,8,12,6,9,3,2,15,13,8,10,1,3,15,4,2,11,6,7,12,0,5,14,9]
```

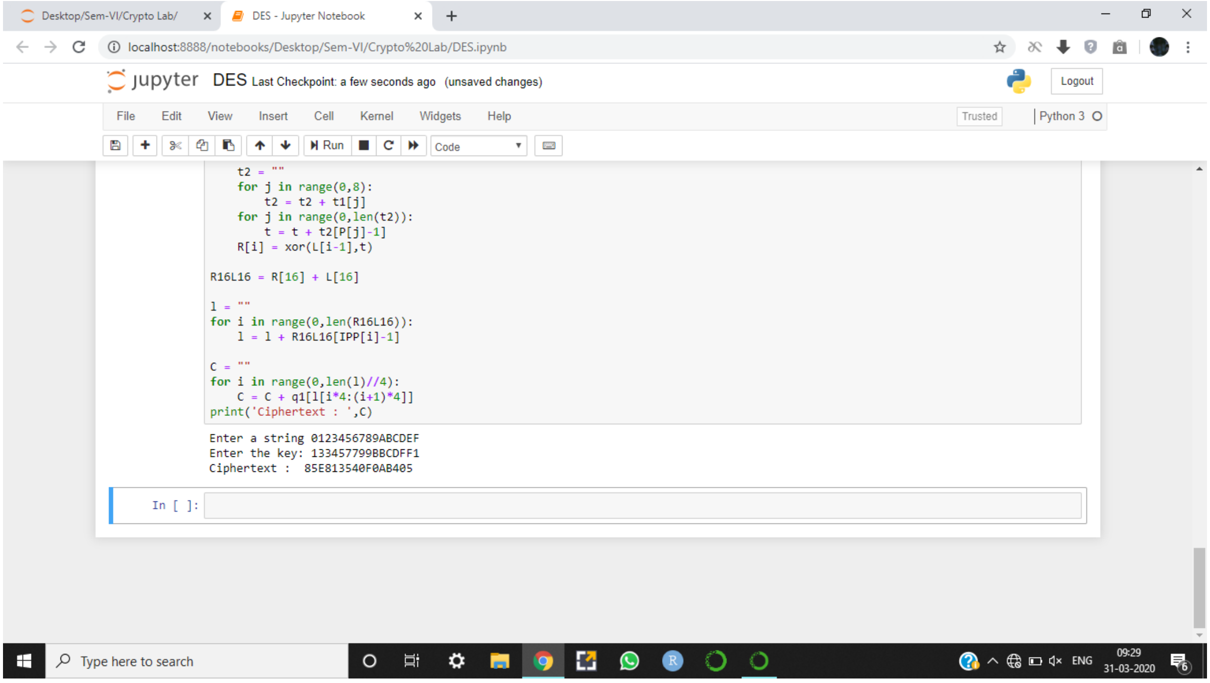
```
s3 =
[10,0,9,14,6,3,15,5,1,13,12,7,11,4,2,8,13,7,0,9,3,4,6,10,2,8,5,14,12,11,15,1,13,6,4,9,8,15,3,0,11,1,2,12,5,10,14,7,1,10,13,0,6,9,8,7,4,15,14,3,11,5,2,12]

s4 =
[7,13,14,3,0,6,9,10,1,2,8,5,11,12,4,15,13,8,11,5,6,15,0,3,4,7,3,12,1,10,14,9,10,6,9,0,12,11,7,13,15,1,3,14,5,2,8,4,3,15,0,6,10,1,13,8,9,4,5,11,12,7,2,14]

s5 =
[2,12,4,1,7,10,11,6,8,5,3,15,13,0,14,9,14,11,2,12,4,7,13,1,5,0,15,10,3,9,8,6,4,2,1,11,10,13,7,8,15,9,12,5,6,3,0,14,11,8,12,7,1,14,2,13,6,15,0,9,10,4,5,3]

s6 =
[12,1,10,15,9,2,6,8,0,13,3,4,14,7,5,11,10,15,4,2,7,12,9,5,6,1,13,14,0,11,3,8,9,14,15,5,2,8,12,3,7,0,4,10,1,13,11,6,4,3,2,12,9,5,15,10,11,14,1,7,6,0,8,13]

s7 =
[4,11,2,14,15,0,8,13,3,12,9,7,5,10,6,1,13,0,11,7,4,9,1,10,14,3,5,12,2,15,8,6,1,4,11,13,12,3,7,14,10,15,6,8,0,5,9,2,6,11,13,8,1,4,10,7,9,5,0,15,14,2,3,12]

s8 =
[13,2,8,4,6,15,11,1,10,9,3,14,5,0,12,7,1,15,13,8,10,3,7,4,12,5,6,11,0,14,9,2,7,11,4,1,9,12,14,2,0,6,10,13,15,3,5,8,2,1,14,7,4,10,8,13,15,12,9,0,3,5,6,11]

s  = [s1] + [s2] + [s3] + [s4] + [s5] + [s6] + [s7] + [s8]


def EE(R):
    R_ = ""
    for i in range(0,48):
        R_ = R_ + str(R[EBIT[i]-1])
    return R_


def xor(a1,a2):
    empty = ""
    x = len(a1)
    for i in range(0,x):
        if(a1[i] == a2[i]):
            empty = empty + '0'
        else:
            empty = empty + '1'
    return empty
```

```
def spfunc(temp):
    B = [0] * 8
    S = [0] * 8
    S1 =[0] * 8
    SB =[0] * 8
    B[0] = temp[0:6]
    B[1] = temp[6:12]
    B[2] = temp[12:18]
    B[3] = temp[18:24]
    B[4] = temp[24:30]
    B[5] = temp[30:36]
    B[6] = temp[36:42]
    B[7] = temp[42:48]

    for i in range(0,8):
        S[i] = q[B[i][0] + B[i][len(B[i])-1]]
        S1[i]= q11[B[i][1:5]]

    for i in range(0,8):
        SB[i] = binary1[s[i][int(S[i]*16) + int(S1[i])]]
    return SB

x = input("Enter a string ")
L = ""
R = ""

y = len(x)
for i in range(0,y//2):
    L = L + binary[x[i]]
```

```
for i in range(y//2,y):
    R = R + binary[x[i]]

K = ""
K__ = input("Enter the key: ")
for i in range(0,len(K__)):
    if(K__[i] in binary):
        K = K + binary[K__[i]]

K_ = ""
IP = ""
IP_ = ""
IP_ = L + R
L0 = R0 = ""
C0 = D0 = ""
for i in range(0,len(PC1)):
    K_ = K_ + K[PC1[i]-1]

for i in range(0,len(IP1)):
    IP = IP + IP_[IP1[i]-1]

y = len(K_)
for i in range(0,y//2):
    C0 = C0 + K_[i]
for i in range(y//2,y):
    D0 = D0 + K_[i]

y1 = len(IP)
for i in range(0,y1//2):
    L0 = L0 + IP[i]
for i in range(y1//2,y1):
    R0 = R0 + IP[i]


C = [0] * 17

D = [0] * 17

C[0] = C0

D[0] = D0


for i in range(1,17):

    if shift[i] == 1:

        C[i] = C[i-1][1:] + C[i-1][0]

        D[i] = D[i-1][1:] + D[i-1][0]

    else:

        C[i] = C[i-1][2:] + C[i-1][0] + C[i-1][1]

        D[i] = D[i-1][2:] + D[i-1][0] + D[i-1][1]


K = [0] * 16

K_=""


for i in range(0,16):

    K[i] = C[i+1] + D[i+1]


for j in range(0,16):

    for i in range(0,len(PC2)):

        K_ = K_ + K[j][PC2[i]-1]

    K[j] = K_

    K_ = ""


L = [0] * 17

R = [0] * 17


L[0] = L0
```

```
R[0] = R0


for i in range(1,17):
    L[i] = R[i-1]
    temp = EE(R[i-1])
    temp1 = xor(temp,K[i-1])
    t1 = spfunc(temp1)
    t = ""
    t2 = ""
    for j in range(0,8):
        t2 = t2 + t1[j]
    for j in range(0,len(t2)):
        t = t + t2[P[j]-1]
    R[i] = xor(L[i-1],t)


R16L16 = R[16] + L[16]


l = ""
for i in range(0,len(R16L16)):
    l = l + R16L16[IPP[i]-1]


C = ""
for i in range(0,len(l)//4):
    C = C + q1[l[i*4:(i+1)*4]]
print('Ciphertext : ',C)
```

Output (DES)



Desktop/Sem-VI/Crypto Lab/
DES - Jupyter Notebook
localhost:8888/notebooks/Desktop/Sem-VI/Crypto%20Lab/DES.ipynb
jupyter DES Last Checkpoint: a few seconds ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG
09:29
31-03-2020

```
    t2 = ""
    for j in range(0,8):
        t2 = t2 + t1[j]
    for j in range(0,len(t2)):
        t = t + t2[P[j]-1]
    R[i] = xor(L[i-1],t)

R16L16 = R[16] + L[16]

l = ""
for i in range(0,len(R16L16)):
    l = l + R16L16[IPP[i]-1]

C = ""
for i in range(0,len(l)//4):
    C = C + q1[l[i*4:(i+1)*4]]
print('Ciphertext : ',C)
```

```
Enter a string 0123456789ABCDEF
Enter the key: 133457799BBCDFF1
Ciphertext :  85E813540F0AB405
```

```
In [ ]:
```

RSA CRYPTOSYSTEM

User 1 Code (Has to be run first)

```
alpha = {'A':'1', 'B':'2', 'C':'3', 'D':'4', 'E':'5', 'F':'6', 'G':'7', 'H':'8', 'I':'9', 'J':'10', 'K':'11', 'L':'12', 'M':'13',
'N':'14', 'O':'15', 'P':'16', 'Q':'17', 'R':'18', 'S':'19', 'T':'20', 'U':'21', 'V':'22', 'W':'23', 'X':'24', 'Y':'25',
'Z':'26', 'a':'27', 'b':'28', 'c':'29', 'd':'30', 'e':'31', 'f':'32', 'g':'33', 'h':'34', 'i':'35', 'j':'36', 'k':'37', 'l':'38',
'm':'39', 'n':'40', 'o':'41', 'p':'42', 'q':'43', 'r':'44', 's':'45', 't':'46', 'u':'47', 'v':'48', 'w':'49', 'x':'50', 'y':'51',
'z':'52', ' ':'53', '!':'54', '@':'55', '#':'56', '$':'57', '%':'58', '&':'59', '*':'60', '(':'61', ')':'62', '+':'63', '-':'64',
'/':'65', '=':'66', '0':'67', '1':'68', '2':'69', '3':'70', '4':'71', '5':'72', '6':'73', '7':'74', '8':'75', '9':'76'}

import socket,time


def Tcp_connect(HostIp,Port):
    global s
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((HostIp,Port))
    return


def Tcp_server_wait(numofclientwait,port):
    global s2
    s2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s2.bind(('',port))
    s2.listen(numofclientwait)



def Tcp_server_next():
    global s
    s = s2.accept()[0]


def Tcp_Write(D):
    for i in range(0,len(D)):
        D[i] = D[i].encode()
        s.send(D[i] + b'#')
    s.send(b'\r')
    return

def Tcp_Read():
    a = ' '
    b = ''
    while a != '\r':
        a = s.recv(1)
        a = a.decode()
        b = b + a
    return b

def Tcp_Close():
    s.close()
    return

def gcd(a,b):
    if b == 0:
        return a
    else:
        return gcd(b,a%b)

def power(a1,a2,a3):
    x = 1
    for i in range(0,a2):
        x = x * a1
    return x%a3

p1 = 83
p2 = 89
x = list(alpha.keys())
y = list(alpha.values())
```

```
n = p1 * p2
t = (p1 - 1) * (p2 - 1)

for e in range(2,t):
    if gcd(e,t) == 1 and e!=p1 and e!=p2:
        break

while True:
    k = 8
    d = (1 + (k * t)) / e
    if d == int(d):
        d = int(d)
        break

Tcp_server_wait(1,17046)
Tcp_server_next()
for i in range(0,5):
    t1 = []
    C = []
    temp = ''
    b = Tcp_Read()
    for j in range(0,len(b)):
        if b[j] != '#':
            temp = temp + b[j]
        else:
            t1.append(temp)
            temp = ''
    for j in range(0,len(t1)):
        D = power(int(t1[j]),d,n)
        temp = temp + x[D-1]
    print(temp)
    a = input('Enter string  : ')
    t1 = []
    for l in range(0,len(a)):
        t1.append(alpha[a[l]])
        c = power(int(t1[l]),5,10961)
        C.append(str(c))
    Tcp_Write(C)

Tcp_Close()
```

User 2 Code

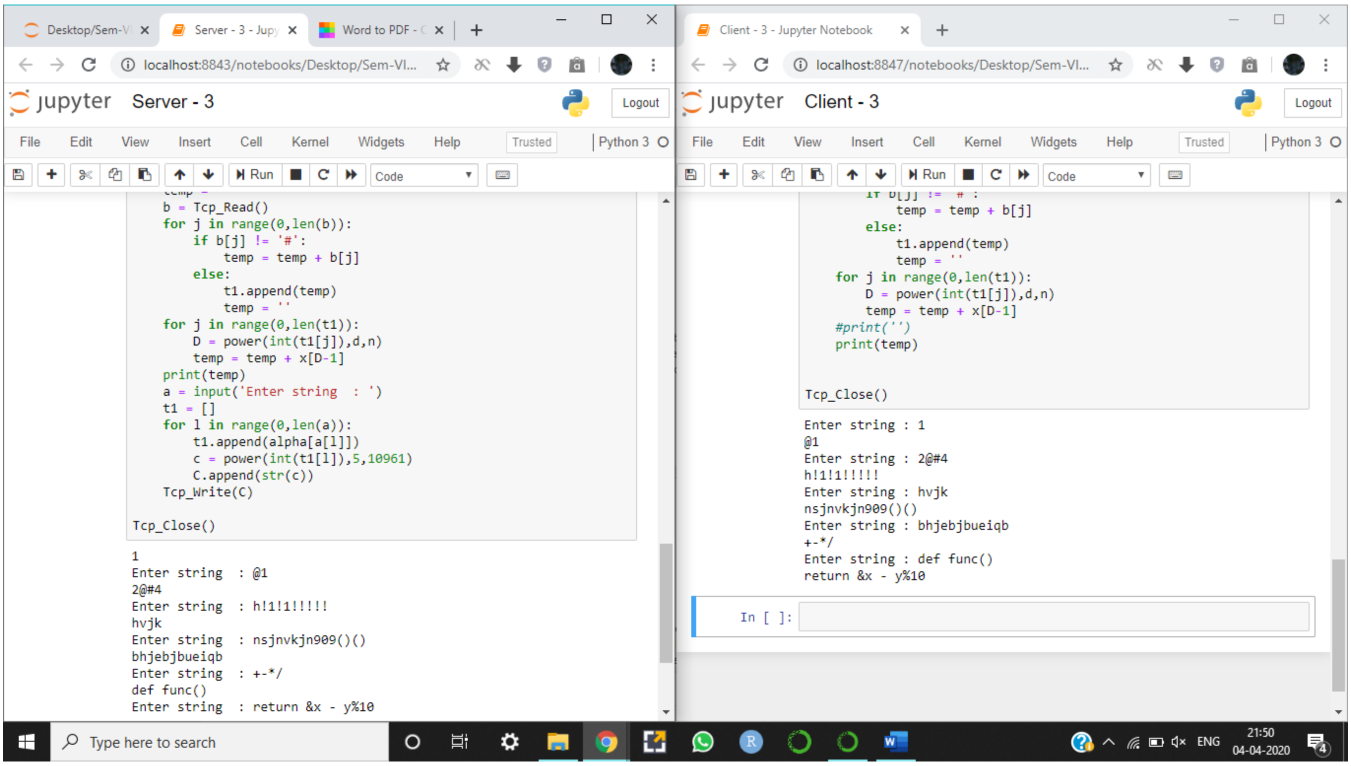
```
alpha = {'A':'1', 'B':'2', 'C':'3', 'D':'4', 'E':'5', 'F':'6', 'G':'7', 'H':'8', 'I':'9', 'J':'10', 'K':'11', 'L':'12', 'M':'13',
'N':'14', 'O':'15', 'P':'16', 'Q':'17', 'R':'18', 'S':'19', 'T':'20', 'U':'21', 'V':'22', 'W':'23', 'X':'24', 'Y':'25',
'Z':'26', 'a':'27', 'b':'28', 'c':'29', 'd':'30', 'e':'31', 'f':'32', 'g':'33', 'h':'34', 'i':'35', 'j':'36', 'k':'37', 'l':'38',
'm':'39', 'n':'40', 'o':'41', 'p':'42', 'q':'43', 'r':'44', 's':'45', 't':'46', 'u':'47', 'v':'48', 'w':'49', 'x':'50', 'y':'51',
'z':'52', ' ':'53', '!':'54', '@':'55', '#':'56', '$':'57', '%':'58', '&':'59', '*':'60', '(':'61', ')':'62', '+':'63', '-':'64',
'/':'65', '=':'66', '0':'67', '1':'68', '2':'69', '3':'70', '4':'71', '5':'72', '6':'73', '7':'74', '8':'75', '9':'76'}

import socket,time

def Tcp_connect(HostIp,Port):
    global s
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((HostIp,Port))
    return

def Tcp_Write(D):
    for i in range(0,len(D)):
        D[i] = D[i].encode()
        s.send(D[i] + b'#')
    s.send(b'\r')
    return
```

```
def Tcp_Read():
    t = []
    a = ' '
    b = ''
    while a != '\r':
        a = s.recv(1)
        a = a.decode()
        b = b + a
    return b

def Tcp_Close():
    s.close()
    return

def gcd(a,b):
    if b == 0:
        return a
    else:
        return gcd(b,a%b)

def power(a1,a2,a3):
    x = 1
    for i in range(0,a2):
        x = x * a1
    return x%a3

p1 = 97
p2 = 113
x = list(alpha.keys())
y = list(alpha.values())
```

```
n = p1 * p2
t = (p1 - 1) * (p2 - 1)

for e in range(2,t):
    if gcd(e,t) == 1 and e!=p1 and e!=p2:
        break

while True:
    k = 7
    d = (1 + (k * t)) / e
    if d == int(d):
        d = int(d)
        break

Tcp_connect('127.0.0.1',17046)
for i in range(0,5):
    t1 = []
    C = []
    temp = ''
    a = input('Enter string : ')
    for l in range(0,len(a)):
        t1.append(alpha[a[l]])
        c = power(int(t1[l]),3,7387)
        C.append(str(c))
    Tcp_Write(C)
    b = Tcp_Read()
    t1 = []
    for j in range(0,len(b)):
        if b[j] != '#':
            temp = temp + b[j]
        else:
```

```
        t1.append(temp)
        temp = ''
    for j in range(0,len(t1)):
        D = power(int(t1[j]),d,n)
        temp = temp + x[D-1]
    #print('')
    print(temp)


Tcp_Close()
```

Output (RSA)



```
    b = Tcp_Read()
    for j in range(0,len(b)):
        if b[j] != '#':
            temp = temp + b[j]
        else:
            t1.append(temp)
            temp = ''
    for j in range(0,len(t1)):
        D = power(int(t1[j]),d,n)
        temp = temp + x[D-1]
    print(temp)
    a = input('Enter string  : ')
    t1 = []
    for l in range(0,len(a)):
        t1.append(alpha[a[l]])
        c = power(int(t1[l]),5,10961)
        C.append(str(c))
    Tcp_Write(C)

Tcp_Close()
```

```
1
Enter string  : @1
2@#4
Enter string  : h!1!1!!!!!
hvjk
Enter string  : nsjnvkjn909()()
bhjebjbueiqb
Enter string  : +-*/
def func()
Enter string  : return &x - y%10
```

```
        if b[j] != '#':
            temp = temp + b[j]
        else:
            t1.append(temp)
            temp = ''
    for j in range(0,len(t1)):
        D = power(int(t1[j]),d,n)
        temp = temp + x[D-1]
    #print('')
    print(temp)


Tcp_Close()
```

```
Enter string : 1
@1
Enter string : 2@#4
h!1!1!!!!!
Enter string : hvjk
nsjnvkjn909()()
Enter string : bhjebjbueiqb
+-*/
Enter string : def func()
return &x - y%10
```

In [ ]:

RSA DIGITAL SIGNATURE

User 1 Code

```
import socket,time
from random import randint

def Tcp_connect(HostIp,Port):
    global s
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((HostIp,Port))
    return

def Tcp_server_wait(numofclientwait,port):
    global s2
    s2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s2.bind(('',port))
    s2.listen(numofclientwait)



def Tcp_server_next():
    global s
    s = s2.accept()[0]

def Tcp_Write(D):
    D = str(D)
    D = D.encode()
    s.send(D)
    s.send(b'\r')
    return

def Tcp_Read():
    a = ' '
    b = ''
    while a != '\r':
        a = s.recv(1)
        a = a.decode()
        b = b + a
    return int(b)

def Tcp_Close():
    s.close()
    return

def gcd(a,b):
    if b == 0:
        return a
    else:
        return gcd(b,a%b)

def power(a1,a2,a3):
    x = 1
    for i in range(0,a2):
        x = x * a1
    return x%a3

def exteuclid(a,b):
    r1 = a
    r2 = b
    s1 = 1
    s2 = 0
    t1 = 0
    t2 = 1
    while r2 > 0:
        q = r1 // r2
        r = r1 - q * r2
        r1 = r2
        r2 = r
        s = s1 - q * s2
        s1 = s2
        s2 = s
        t = t1 - q * t2
        t1 = t2
        t2 = t

    if t1 < 0:
        t1 = t1 % a

    return (r1,t1)

p1 = 97
p2 = 113

n = p1 * p2
print(n)
t = (p1 - 1) * (p2 - 1)

key = []

for i in range(2,t):
    g = gcd(t,i)
    if g == 1:
        key.append(i)
e = key[2]
```

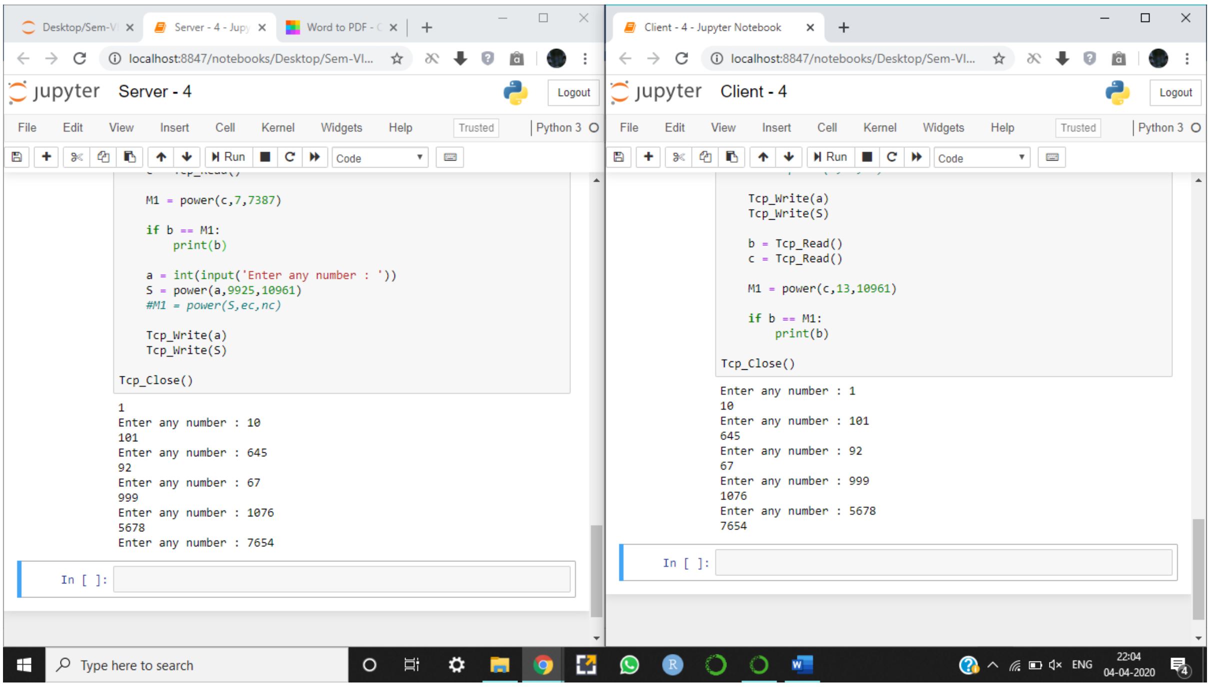
```
print(e)
r,d = exteuclid(t,e)

if r == 1:
    d = int(d)
    print("Decryption Key   :",d)

Tcp_server_wait(1,1707)
Tcp_server_next()
for i in range(0,5):
    t1 = []
    C = []
    #temp = ''
    b = Tcp_Read()
    c = Tcp_Read()

    M1 = power(c,7,7387)

    if b == M1:
        print(b)

    a = int(input('Enter any number : '))
    S = power(a,9925,10961)
    #M1 = power(S,ec,nc)

    Tcp_Write(a)
    Tcp_Write(S)

Tcp_Close()
```

User 2 Code

```
import socket,time
from random import randint

def Tcp_connect(HostIp,Port):
   global s
   s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   s.connect((HostIp,Port))
   return

def Tcp_Write(D):
   D = str(D)
   D = D.encode()
   s.send(D)
   s.send(b'\r')
   return

def Tcp_Read():
   a = ' '
   b = ''
   while a != '\r':
      a = s.recv(1)
      a = a.decode()
      b = b + a
   return int(b)

def Tcp_Close():
   s.close()
   return
```

```
def gcd(a,b):
    if b == 0:
        return a
    else:
        return gcd(b,a%b)

def power(a1,a2,a3):
    x = 1
    for i in range(0,a2):
        x = x * a1
    return x%a3

def exteuclid(a,b):
    r1 = a
    r2 = b
    s1 = 1
    s2 = 0
    t1 = 0
    t2 = 1

    while r2 > 0:
        q = r1 // r2
        r = r1 - q * r2
        r1 = r2
        r2 = r
        s = s1 - q * s2
        s1 = s2
        s2 = s
        t = t1 - q * t2
        t1 = t2
        t2 = t
    if t1 < 0:
        t1 = t1 % a

    return (r1,t1)

p1 = 83
p2 = 89
n = p1 * p2
print(n)
t = (p1 - 1) * (p2 - 1)

key = []

for i in range(2,t):
    g = gcd(t,i)
    if g == 1:
        key.append(i)
e = key[2]
print(e)
r,d = exteuclid(t,e)

if r == 1:
    d = int(d)
    print("Decryption Key   :",d)

Tcp_connect('127.0.0.1',1707)
for i in range(0,5):
    t1 = []
    C = []
    #temp = ''
    a = int(input('Enter any number : '))
    S = power(a,1031,7387)
    #M1 = power(S,ec,nc)

    Tcp_Write(a)
    Tcp_Write(S)

    b = Tcp_Read()
    c = Tcp_Read()

    M1 = power(c,13,10961)

    if b == M1:
        print(b)

Tcp_Close()
```

Output (RSA Digital Signature)



```
    M1 = power(c,7,7387)

    if b == M1:
        print(b)

    a = int(input('Enter any number : '))
    S = power(a,9925,10961)
    #M1 = power(S,ec,nc)

    Tcp_Write(a)
    Tcp_Write(S)

Tcp_Close()
```

```
1
Enter any number : 10
101
Enter any number : 645
92
Enter any number : 67
999
Enter any number : 1076
5678
Enter any number : 7654
```

```
In [ ]:
```

```
    Tcp_Write(a)
    Tcp_Write(S)

    b = Tcp_Read()
    c = Tcp_Read()

    M1 = power(c,13,10961)

    if b == M1:
        print(b)

Tcp_Close()
```

```
Enter any number : 1
10
Enter any number : 101
645
Enter any number : 92
67
Enter any number : 999
1076
Enter any number : 5678
7654
```

```
In [ ]:
```

AES

Code

```
binary = {'0':'0000', '1':'0001', '2':'0010', '3':'0011', '4':'0100', '5':'0101', '6':'0110', '7':'0111',
'8':'1000', '9':'1001', 'A':'1010', 'B':'1011', 'C':'1100', 'D':'1101', 'E':'1110', 'F':'1111'}

q1     = {'0000':'0', '0001':'1', '0010':'2', '0011':'3', '0100':'4', '0101':'5', '0110':'6', '0111':'7', '1000':'8',
'1001':'9', '1010':'A', '1011':'B', '1100':'C', '1101':'D', '1110':'E', '1111':'F'}

letters =
{'a':'61','b':'62','c':'63','d':'64','e':'65','f':'66','g':'67','h':'68','i':'69','j':'6A','k':'6B','l':'6C','m':'6D','n':'6E','
o':'6F','p':'70','q':'71','r':'72','s':'73','t':'74','u':'75','v':'76','w':'77','x':'78','y':'79','z':'7A','A':'41','B':'42','
C':'43','D':'44','E':'45','F':'46','G':'47','H':'48','I':'49','J':'4A','K':'4B','L':'4C','M':'4D','N':'4E','O':'4F','P':'50
','Q':'51','R':'52','S':'53','T':'54','U':'55','V':'56','W':'57','X':'58','Y':'59','Z':'5A',' ':'20'}

digits =  {'0':'0000', '1':'0001', '2':'0010', '3':'0011', '4':'0100', '5':'0101', '6':'0110', '7':'0111', '8':'1000',
'9':'1001', 'A':'1010', 'B':'1011', 'C':'1100', 'D':'1101', 'E':'1110', 'F':'1111'}

sbox = {'00':'63', '01':'7C', '02':'77', '03':'7B', '04':'F2', '05':'6B', '06':'6F', '07':'C5', '08':'30', '09':'01',
'0A':'67', '0B':'2B', '0C':'FE', '0D':'D7', '0E':'AB', '0F':'76', '10':'CA', '11':'82', '12':'C9', '13':'7D', '14':'FA',
'15':'59', '16':'47', '17':'F0', '18':'AD', '19':'D4', '1A':'A2', '1B':'AF', '1C':'9C', '1D':'A4', '1E':'72', '1F':'C0',
'20':'B7', '21':'FD', '22':'93', '23':'26', '24':'36', '25':'3F', '26':'F7', '27':'CC', '28':'34', '29':'A5', '2A':'E5',
'2B':'F1', '2C':'71', '2D':'D8', '2E':'31', '2F':'15', '30':'04', '31':'C7', '32':'23', '33':'C3', '34':'18', '35':'96',
'36':'05', '37':'9A', '38':'07', '39':'12', '3A':'80', '3B':'E2', '3C':'EB', '3D':'27', '3E':'B2', '3F':'75', '40':'09',
'41':'83', '42':'2C', '43':'1A', '44':'1B', '45':'6E', '46':'5A', '47':'A0', '48':'52', '49':'3B', '4A':'D6', '4B':'B3',
'4C':'29', '4D':'E3', '4E':'2F', '4F':'84', '50':'53', '51':'D1', '52':'00', '53':'ED', '54':'20', '55':'FC', '56':'B1',
'57':'5B', '58':'6A', '59':'CB', '5A':'BE', '5B':'39', '5C':'4A', '5D':'4C', '5E':'58', '5F':'CF', '60':'D0', '61':'EF',
'62':'AA', '63':'FB', '64':'43', '65':'4D', '66':'33', '67':'85', '68':'45', '69':'F9', '6A':'02', '6B':'7F', '6C':'50',
'6D':'3C', '6E':'9F', '6F':'A8', '70':'51', '71':'A3', '72':'40', '73':'8F', '74':'92', '75':'9D', '76':'38', '77':'F5',
'78':'BC', '79':'B6', '7A':'DA', '7B':'21', '7C':'10', '7D':'FF', '7E':'F3', '7F':'D2', '80':'CD', '81':'0C', '82':'13',
'83':'EC', '84':'5F', '85':'97', '86':'44', '87':'17', '88':'C4', '89':'A7', '8A':'7E', '8B':'3D', '8C':'64', '8D':'5D',
'8E':'19', '8F':'73', '90':'60', '91':'81', '92':'4F', '93':'DC', '94':'22', '95':'2A', '96':'90', '97':'88', '98':'46',
'99':'EE', '9A':'B8', '9B':'14', '9C':'DE', '9D':'5E', '9E':'0B', '9F':'DB', 'A0':'E0', 'A1':'32', 'A2':'3A', 'A3':'0A',
'A4':'49', 'A5':'06', 'A6':'24', 'A7':'5C', 'A8':'C2', 'A9':'D3', 'AA':'AC', 'AB':'62', 'AC':'91', 'AD':'95',
'AE':'E4', 'AF':'79', 'B0':'E7', 'B1':'C8', 'B2':'37', 'B3':'6D', 'B4':'8D', 'B5':'D5', 'B6':'4E', 'B7':'A9', 'B8':'6C',
'B9':'56', 'BA':'F4', 'BB':'EA', 'BC':'65', 'BD':'7A', 'BE':'AE', 'BF':'08', 'C0':'BA', 'C1':'78', 'C2':'25', 'C3':'2E',
'C4':'1C', 'C5':'A6', 'C6':'B4', 'C7':'C6', 'C8':'E8', 'C9':'DD', 'CA':'74', 'CB':'1F', 'CC':'4B', 'CD':'BD', 'CE':'8B',
'CF':'8A', 'D0':'70', 'D1':'3E', 'D2':'B5', 'D3':'66', 'D4':'48', 'D5':'03', 'D6':'F6', 'D7':'0E', 'D8':'61',
'D9':'35', 'DA':'57', 'DB':'B9', 'DC':'86', 'DD':'C1', 'DE':'1D', 'DF':'9E', 'E0':'E1', 'E1':'F8', 'E2':'98', 'E3':'11',
'E4':'69', 'E5':'D9', 'E6':'8E', 'E7':'94', 'E8':'9B', 'E9':'1E', 'EA':'87', 'EB':'E9', 'EC':'CE', 'ED':'55', 'EE':'28',
'EF':'DF', 'F0':'8C', 'F1':'A1', 'F2':'89', 'F3':'0D', 'F4':'BF', 'F5':'E6', 'F6':'42', 'F7':'68', 'F8':'41', 'F9':'99',
'FA':'2D', 'FB':'0F', 'FC':'B0', 'FD':'54', 'FE':'BB', 'FF':'16'}


def rot(w):

    wk = w.copy()
    temp = wk[0]
    wk[0] = wk[1]
    wk[1] = wk[2]
    wk[2] = wk[3]
    wk[3] = temp
    return wk

def reg(empty):
    a = empty[0:4]
    b = empty[4:8]
    t  = ""
    t = q1[a] + q1[b]
    return t

def xor(a1,a2):
    empty = ""
    x = len(a1)
    for z in range(0,x):
        if(a1[z] == a2[z]):
            empty = empty + '0'
        else:
            empty = empty + '1'
    return empty

def spfunc(num):
    x = len(num)
    s = ""
    t =  ""
    for i in range(0,x):
        s = s + binary[num[i]]
    return s
```

```
def app(s):
    s = s[::-1]
    for i in range(0,16-len(s)):
        s = s + '0'
    s = s[::-1]
    return s

def deapp(res):
    for k in range(0,16):
        if(res[k] == '1'):
            break
        else:
            continue
    res = res[k:]
    if(len(res) < 8):
        res = res[::-1]
        for i in range(0,8-len(res)):
            res = res + '0'
        res = res[::-1]
    return(res)

x = input("Enter the plaintext : ")
y = input("Enter the key : ")
rk = [0] * 11
x0 = []
plain = ""
w0 = []
w1 = []
w2 = []
w3 = []
```

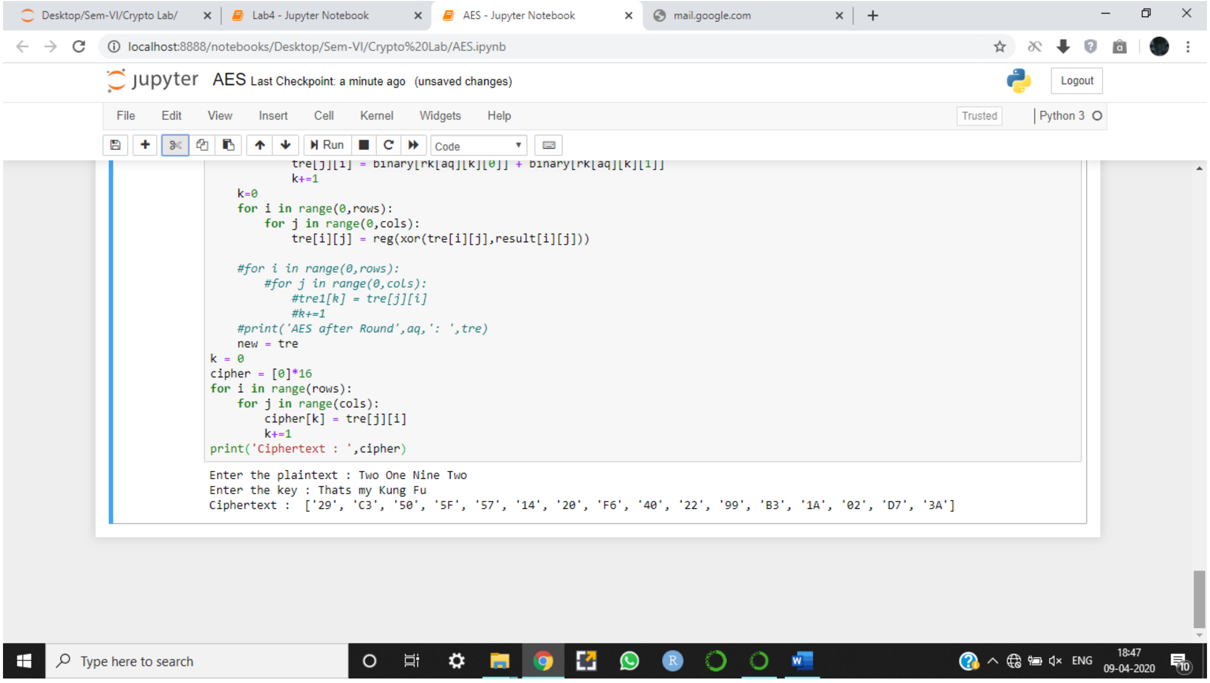
```
for i in range(0,len(x)):
   if x[i] in letters:
      plain = plain + str(letters[x[i]])


for i in range(0,len(y)):
   if y[i] in letters:
      x0 = x0 + [letters[y[i]]]
      if len(w0) < 4 :
         w0 = w0 + [letters[y[i]]]
      elif len(w1) < 4 :
         w1 = w1 + [letters[y[i]]]
      elif len(w2) < 4 :
         w2 = w2 + [letters[y[i]]]
      else:
         w3 = w3 + [letters[y[i]]]
rk[0] = x0
rcon = ['00000001', '00000010', '00000100', '00001000', '00010000', '00100000', '01000000',
'10000000', '00011011', '00110110']


for aq in range(0,10):
   wk = rot(w3)
   ww = []
   for i in range(0,len(wk)):
      ww = ww + [sbox[str(wk[i])]]


   ep = digits[ww[0][0]] + digits[ww[0][1]]
   t = xor(ep, rcon[aq])
   x = list(digits.keys())
   y = list(digits.values())
```

```
lo = x[y.index(t[0:4])] +  x[y.index(t[4:8])]
ww[0] = lo

w4 = [0] * 4
w5 = [0] * 4
w6 = [0] * 4
w7 = [0] * 4
w  = [0] * 16
x1 = []

for i in range(0,4):
   k1 = xor(digits[w0[i][0]],digits[ww[i][0]])
   k2 = xor(digits[w0[i][1]],digits[ww[i][1]])
   w4[i] = x[y.index(k1)]+x[y.index(k2)]

for i in range(0,4):
   k1 = xor(digits[w1[i][0]],digits[w4[i][0]])
   k2 = xor(digits[w1[i][1]],digits[w4[i][1]])
   w5[i] = x[y.index(k1)]+x[y.index(k2)]

for i in range(0,4):
   k1 = xor(digits[w2[i][:1]],digits[w5[i][:1]])
   k2 = xor(digits[w2[i][1:2]],digits[w5[i][1:2]])
   w6[i] = x[y.index(k1)]+x[y.index(k2)]
for i in range(0,4):
   k1 = xor(digits[w3[i][:1]],digits[w6[i][:1]])
   k2 = xor(digits[w3[i][1:2]],digits[w6[i][1:2]])
   w7[i] = x[y.index(k1)]+x[y.index(k2)]

x1 = w4 + w5 + w6 + w7
rk[aq+1] = x1
```

```
    w0 = w4

    w1 = w5

    w2 = w6

    w3 = w7


rows,cols = (4,4)

arr  = [[0 for i in range(cols)]for j in range(rows)]

arrk = [[0 for i in range(cols)]for j in range(rows)]


k = 0

for i in range(0,cols):

    for j in range(0,rows):

        arr[j][i] = plain[k:k+2]

        k+=2


k = 0

for i in range(0,cols):

    for j in range(0,rows):

        arrk[j][i] = rk[0][k]

        k+=1


new = [[0 for i in range(cols)]for j in range(rows)]

n   = [[0 for i in range(cols)]for j in range(rows)]

n1  = [[0 for i in range(cols)]for j in range(rows)]


for i in range(0,rows):

    for j in range(0,cols):

        new[i][j] = xor(spfunc(arr[i][j]),spfunc(arrk[i][j]))

        new[i][j] = reg(new[i][j])
```

```
for aq in range(1,11):
    for i in range(0,rows):
        for j in range(0,cols):
            new[i][j] = sbox[new[i][j]]
            #print(new[i][j])

    n = new
    n[1] = rot(new[1])
    n[2] = rot(rot(new[2]))
    n[3] = rot(rot(rot(new[3])))

    #print(n)

    new = [['02','03','01','01'],['01','02','03','01'],['01','01','02','03'],['03','01','01','02']]
    #print(new)

    result = [['00000000' for i in range(cols)]for j in range(rows)]
    if aq != 10:
        for i in range(len(n)):
            for j in range(len(new[0])):
                for k in range(len(new)):
                    if(new[i][k] == '03'):
                        temp = '02'
                        t = (bin(int(spfunc(temp),2)*int(spfunc(n[k][j]),2)))
                        t = app(t[2:])
                        tt = app(spfunc(n[k][j]))
                        t = app(xor(t,tt))
                        result[i][j] = app(result[i][j])
                        result[i][j] = app(xor(result[i][j], t))
                    else:
                        t = (bin(int(spfunc(new[i][k]),2)*int(spfunc(n[k][j]),2)))
                        t = app(t[2:])
                        result[i][j] = app(result[i][j])
                        result[i][j] = app(xor(result[i][j],t))

        for i in range(4):
            for j in range(4):
                result[i][j] = deapp(result[i][j])
                if len(result[i][j]) == 9:
                    result[i][j] = deapp(xor(result[i][j],'100011011'))
                #result[i][j] = reg(result[i][j])
        #print(result)
    if aq == 10:
        result = n
        for li in range(rows):
            for zi in range(cols):
                result[li][zi] = spfunc(result[li][zi][0]) + spfunc(result[li][zi][1])
    k=0
    tre = [['00000000' for i in range(cols)]for j in range(rows)]
    #tre1= [0]*16
    for i in range(0,rows):
        for j in range(0,cols):
            tre[j][i] = binary[rk[aq][k][0]] + binary[rk[aq][k][1]]
            k+=1
    k=0
    for i in range(0,rows):
        for j in range(0,cols):
            tre[i][j] = reg(xor(tre[i][j],result[i][j]))
        new = tre
    k = 0
    cipher = [0]*16
    for i in range(rows):
        for j in range(cols):
            cipher[k] = tre[j][i]
            k+=1
    print('Ciphertext : ',cipher)
```

Output (AES):



Desktop/Sem-VI/Crypto Lab/
Lab4 - Jupyter Notebook
AES - Jupyter Notebook
mail.google.com
localhost:8888/notebooks/Desktop/Sem-VI/Crypto%20Lab/AES.ipynb
jupyter AES Last Checkpoint: a minute ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG
18:47
09-04-2020

```
                tre[j][i] = binary[rk[aq][k][0]] + binary[rk[aq][k][1]]
                k+=1
        k=0
        for i in range(0,rows):
            for j in range(0,cols):
                tre[i][j] = reg(xor(tre[i][j],result[i][j]))

        #for i in range(0,rows):
            #for j in range(0,cols):
                #tre1[k] = tre[j][i]
                #k+=1
        #print('AES after Round',aq,': ',tre)
        new = tre
k = 0
cipher = [0]*16
for i in range(rows):
    for j in range(cols):
        cipher[k] = tre[j][i]
        k+=1
print('Ciphertext : ',cipher)
```

```
Enter the plaintext : Two One Nine Two
Enter the key : Thats my Kung Fu
Ciphertext :  ['29', 'C3', '50', '5F', '57', '14', '20', 'F6', '40', '22', '99', 'B3', '1A', '02', 'D7', '3A']
```