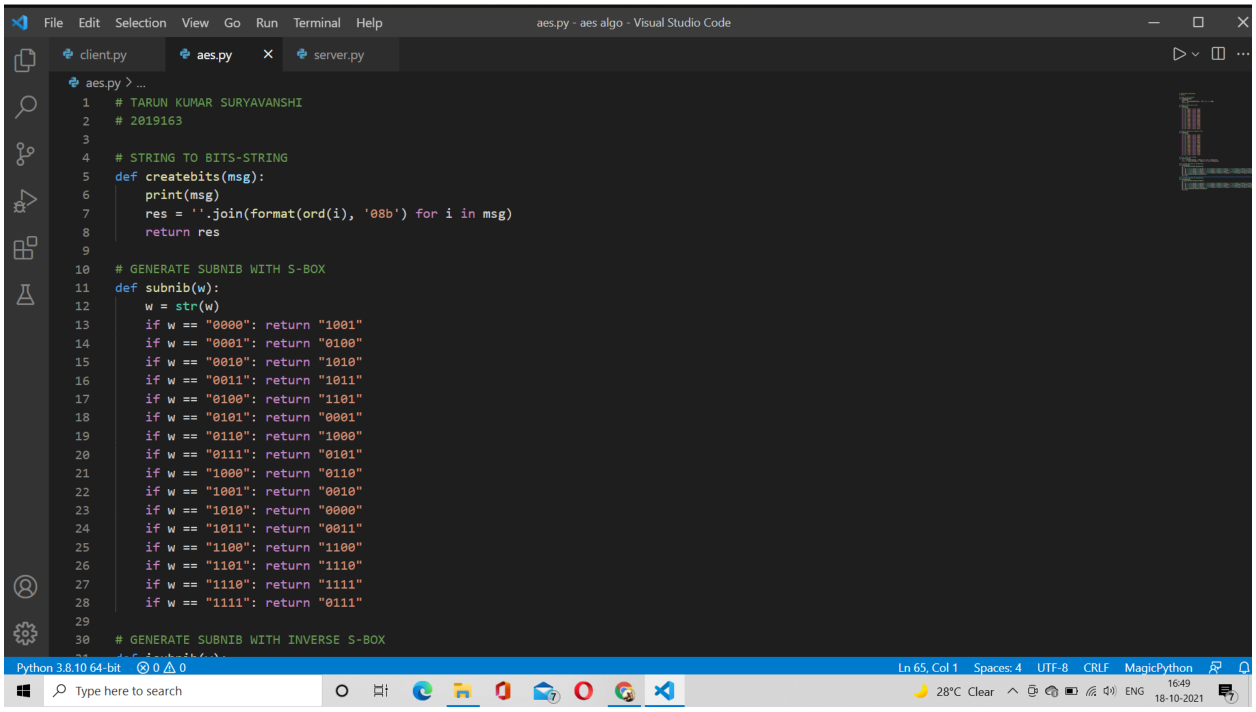
TARUN KUMAR SURYAVANSHI
2019163

Aes.py:

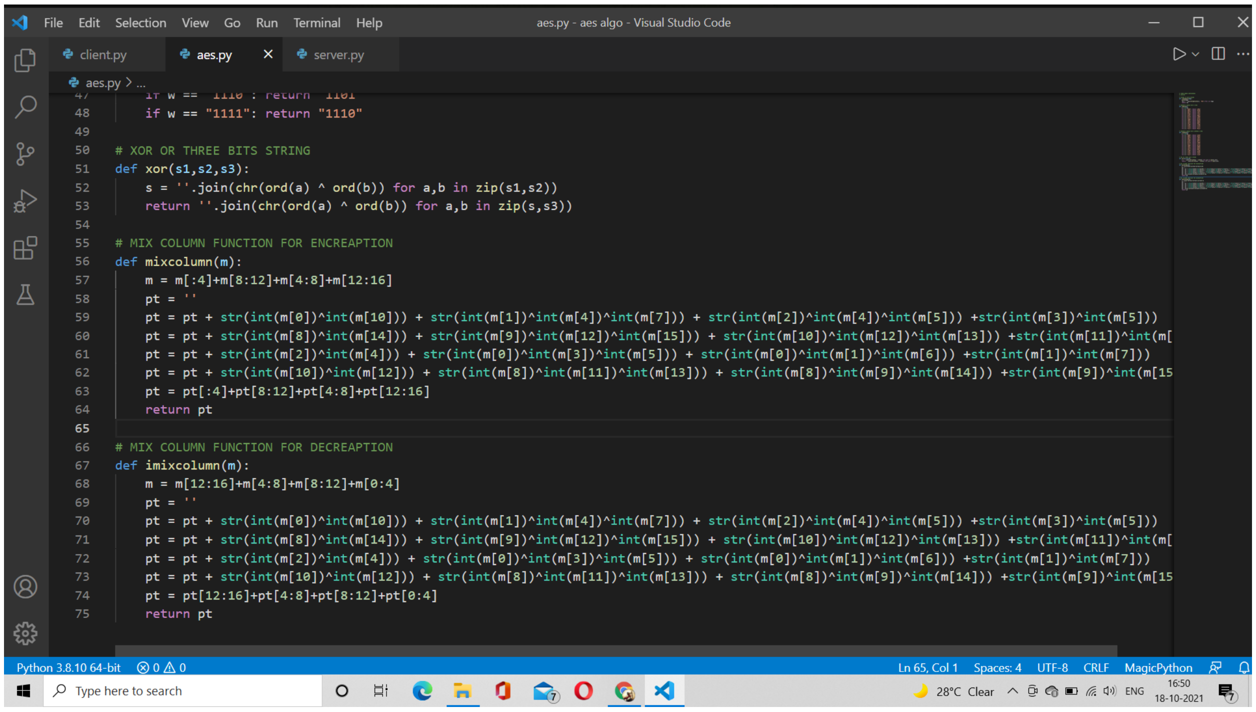


```python
# TARUN KUMAR SURYAVANSHI
# 2019163

# STRING TO BITS-STRING
def createbits(msg):
    print(msg)
    res = ''.join(format(ord(i), '08b') for i in msg)
    return res

# GENERATE SUBNIB WITH S-BOX
def subnib(w):
    w = str(w)
    if w == "0000": return "1001"
    if w == "0001": return "0100"
    if w == "0010": return "1010"
    if w == "0011": return "1011"
    if w == "0100": return "1101"
    if w == "0101": return "0001"
    if w == "0110": return "1000"
    if w == "0111": return "0101"
    if w == "1000": return "0110"
    if w == "1001": return "0010"
    if w == "1010": return "0000"
    if w == "1011": return "0011"
    if w == "1100": return "1100"
    if w == "1101": return "1110"
    if w == "1110": return "1111"
    if w == "1111": return "0111"

# GENERATE SUBNIB WITH INVERSE S-BOX
def isubnib(w):
```

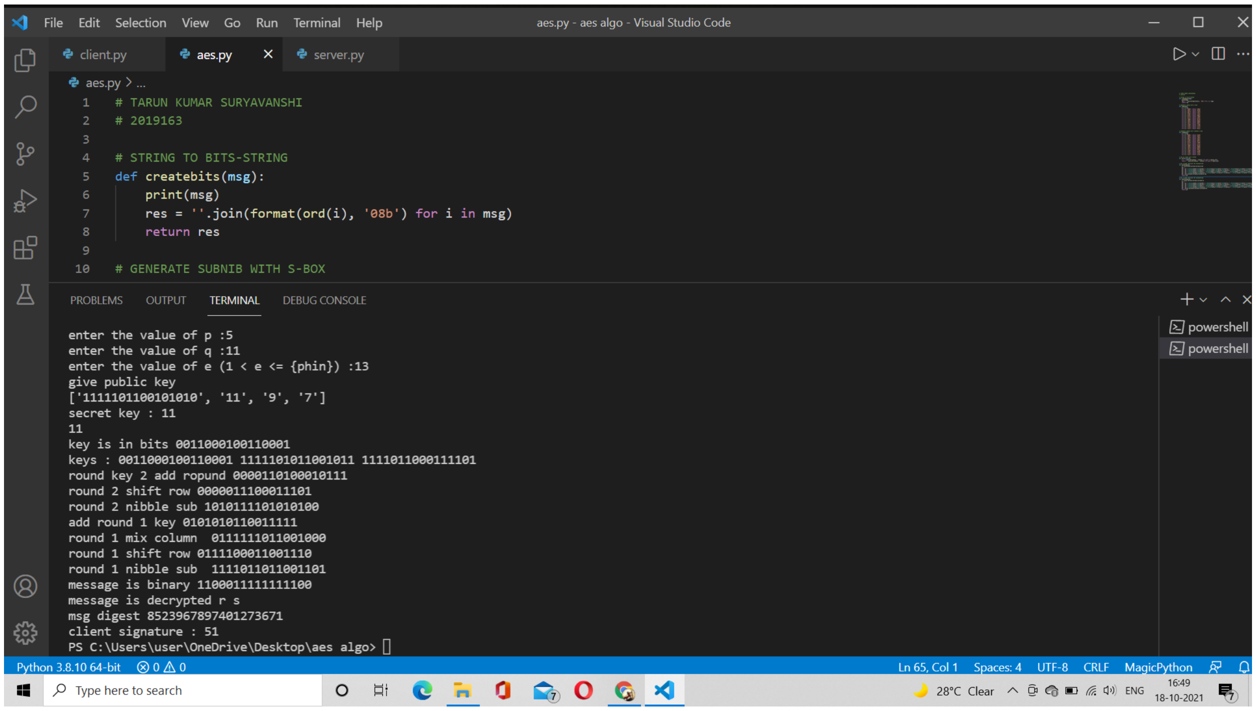


```python
    if w == "1110": return "1101"
    if w == "1111": return "1110"

# XOR OR THREE BITS STRING
def xor(s1,s2,s3):
    s = ''.join(chr(ord(a) ^ ord(b)) for a,b in zip(s1,s2))
    return ''.join(chr(ord(a) ^ ord(b)) for a,b in zip(s,s3))

# MIX COLUMN FUNCTION FOR ENCREAPTION
def mixcolumn(m):
    m = m[:4]+m[8:12]+m[4:8]+m[12:16]
    pt = ''
    pt = pt + str(int(m[0])^int(m[10])) + str(int(m[1])^int(m[4])^int(m[7])) + str(int(m[2])^int(m[4])^int(m[5])) +str(int(m[3])^int(m[5]))
    pt = pt + str(int(m[8])^int(m[14])) + str(int(m[9])^int(m[12])^int(m[15])) + str(int(m[10])^int(m[12])^int(m[13])) +str(int(m[11])^int(m[
    pt = pt + str(int(m[2])^int(m[4])) + str(int(m[0])^int(m[3])^int(m[5])) + str(int(m[0])^int(m[1])^int(m[6])) +str(int(m[1])^int(m[7]))
    pt = pt + str(int(m[10])^int(m[12])) + str(int(m[8])^int(m[11])^int(m[13])) + str(int(m[8])^int(m[9])^int(m[14])) +str(int(m[9])^int(m[15
    pt = pt[:4]+pt[8:12]+pt[4:8]+pt[12:16]
    return pt

# MIX COLUMN FUNCTION FOR DECREAPTION
def imixcolumn(m):
    m = m[12:16]+m[4:8]+m[8:12]+m[0:4]
    pt = ''
    pt = pt + str(int(m[0])^int(m[10])) + str(int(m[1])^int(m[4])^int(m[7])) + str(int(m[2])^int(m[4])^int(m[5])) +str(int(m[3])^int(m[5]))
    pt = pt + str(int(m[8])^int(m[14])) + str(int(m[9])^int(m[12])^int(m[15])) + str(int(m[10])^int(m[12])^int(m[13])) +str(int(m[11])^int(m[
    pt = pt + str(int(m[2])^int(m[4])) + str(int(m[0])^int(m[3])^int(m[5])) + str(int(m[0])^int(m[1])^int(m[6])) +str(int(m[1])^int(m[7]))
    pt = pt + str(int(m[10])^int(m[12])) + str(int(m[8])^int(m[11])^int(m[13])) + str(int(m[8])^int(m[9])^int(m[14])) +str(int(m[9])^int(m[15
    pt = pt[12:16]+pt[4:8]+pt[8:12]+pt[0:4]
    return pt
```



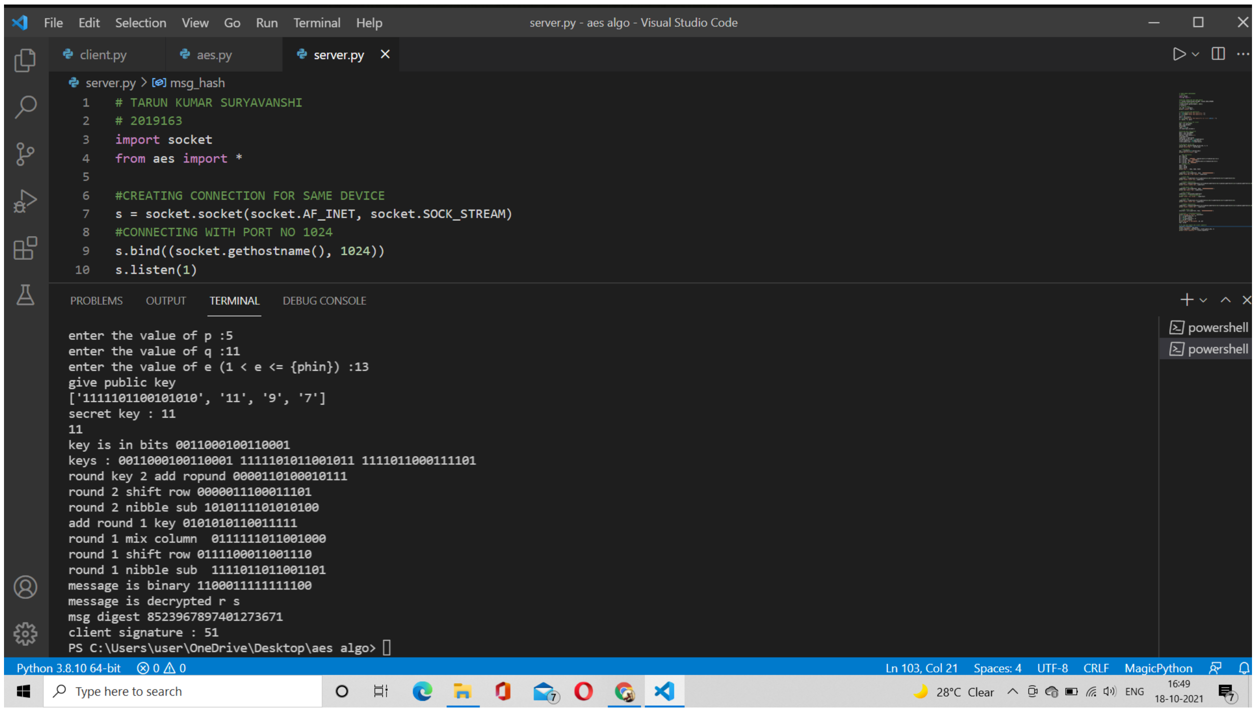
```python
# TARUN KUMAR SURYAVANSHI
# 2019163

# STRING TO BITS-STRING
def createbits(msg):
    print(msg)
    res = ''.join(format(ord(i), '08b') for i in msg)
    return res

# GENERATE SUBNIB WITH S-BOX
```

```
enter the value of p :5
enter the value of q :11
enter the value of e (1 < e <= {phin}) :13
give public key
['1111101100101010', '11', '9', '7']
secret key : 11
11
key is in bits 0011000100110001
keys : 0011000100110001 1111101011001011 1111011000111101
round key 2 add ropund 0000110100010111
round 2 shift row 0000011100011101
round 2 nibble sub 1010111101010100
add round 1 key 0101010110011111
round 1 mix column  0111111011001000
round 1 shift row 0111100011001110
round 1 nibble sub  1111011011001101
message is binary 1100011111111100
message is decrypted r s
msg digest 8523967897401273671
client signature : 51
PS C:\Users\user\OneDrive\Desktop\aes algo> 
```

Server.py :

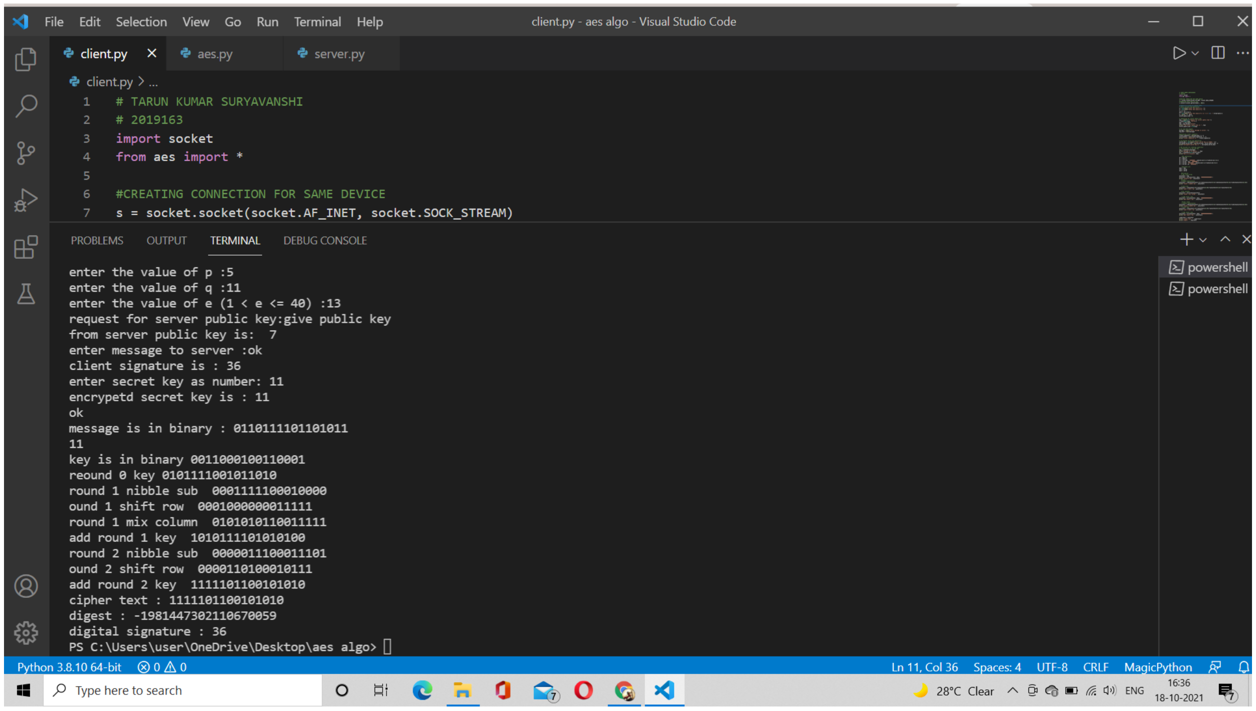


```python
# TARUN KUMAR SURYAVANSHI
# 2019163
import socket
from aes import *

#CREATING CONNECTION FOR SAME DEVICE
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
#CONNECTING WITH PORT NO 1024
s.bind((socket.gethostname(), 1024))
s.listen(1)
```

```
enter the value of p :5
enter the value of q :11
enter the value of e (1 < e <= {phin}) :13
give public key
['1111101100101010', '11', '9', '7']
secret key : 11
11
key is in bits 0011000100110001
keys : 0011000100110001 1111101011001011 1111011000111101
round key 2 add ropund 0000110100010111
round 2 shift row 0000011100011101
round 2 nibble sub 1010111101010100
add round 1 key 0101010110011111
round 1 mix column  0111111011001000
round 1 shift row 0111100011001110
round 1 nibble sub  1111011011001101
message is binary 1100011111111100
message is decrypted r s
msg digest 8523967897401273671
client signature : 51
PS C:\Users\user\OneDrive\Desktop\aes algo> 
```

Client.py



File Edit Selection View Go Run Terminal Help
client.py - aes algo - Visual Studio Code
client.py
aes.py
server.py
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
powershell
powershell
Python 3.8.10 64-bit
Ln 11, Col 36 Spaces: 4 UTF-8 CRLF MagicPython
Type here to search
28°C Clear
ENG
16:36
18-10-2021

```python
# TARUN KUMAR SURYAVANSHI
# 2019163
import socket
from aes import *

#CREATING CONNECTION FOR SAME DEVICE
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
enter the value of p :5
enter the value of q :11
enter the value of e (1 < e <= 40) :13
request for server public key:give public key
from server public key is:  7
enter message to server :ok
client signature is : 36
enter secret key as number: 11
encrypetd secret key is : 11
ok
message is in binary : 0110111101101011
11
key is in binary 0011000100110001
reound 0 key 0101111001011010
round 1 nibble sub  0001111100010000
ound 1 shift row  0001000000011111
round 1 mix column  0101010110011111
add round 1 key  1010111101010100
round 2 nibble sub  0000011100011101
ound 2 shift row  0000110100010111
add round 2 key  1111101100101010
cipher text : 1111101100101010
digest : -1981447302110670059
digital signature : 36
PS C:\Users\user\OneDrive\Desktop\aes algo> 
```