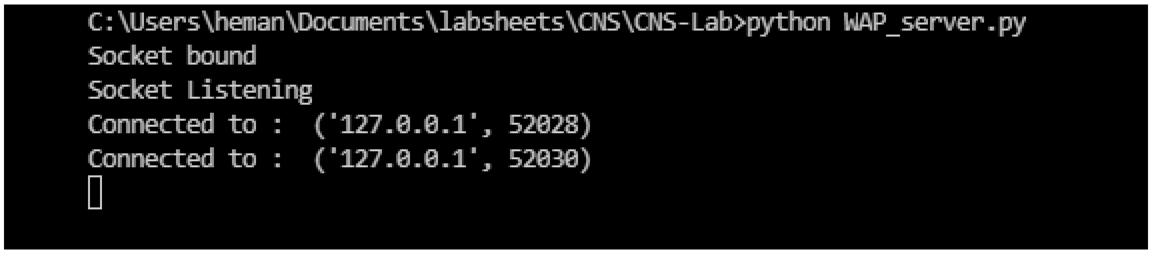
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, NAGPUR
CNS LAB ASSIGNMENT 1

Group Members:
Ruchika Pandharikar BT19CSE009
Aditi Vishwakarma BT19CSE031
Neha Kalbande BT19CSE047
Hemanshu Chaudhari BT19CSE056

Q.1 WAP using python to connect client and server.

```python
import socket
s = socket.socket()
port = 4563
s.connect(('127.0.0.1', port))
print('The following message was received from the server : \n>> ', s.recv(1024).decode())
print('Closing connection..')
s.close()
```

Output:

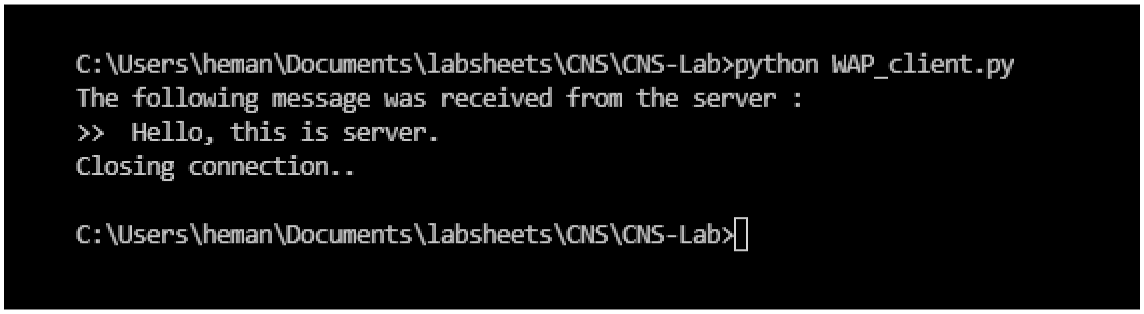


```
C:\Users\heman\Documents\labsheets\CNS\CNS-Lab>python WAP_server.py
Socket bound
Socket Listening
Connected to :  ('127.0.0.1', 52028)
Connected to :  ('127.0.0.1', 52030)
```

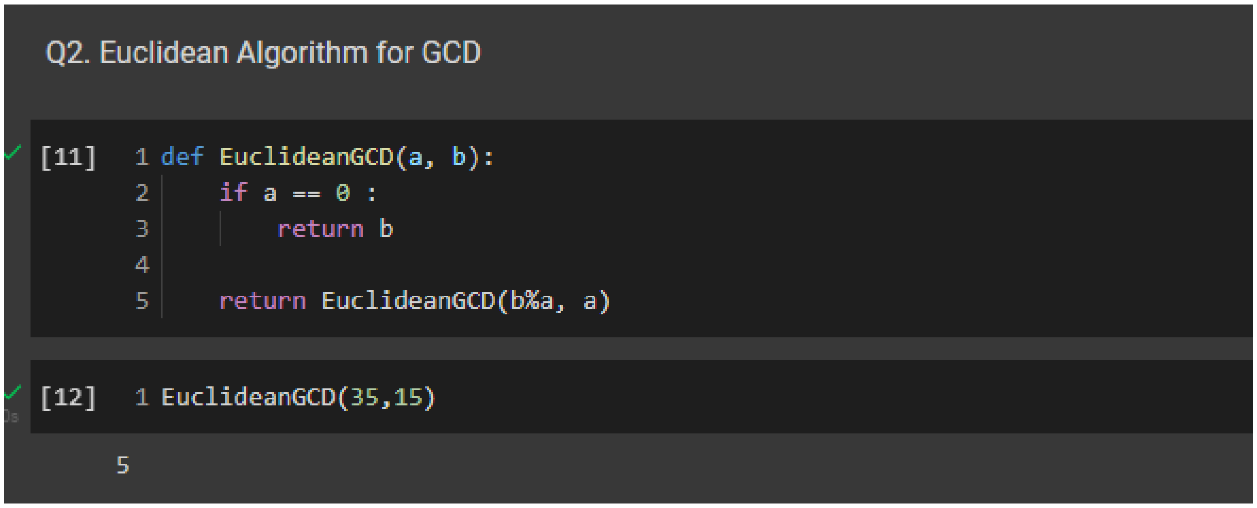
```python
import socket
s = socket.socket()
port = 4563
```

```
s.bind(('', port))
print('Socket bound')
s.listen()
print('Socket Listening')
while True:
    c, addr = s.accept()
    print('Connected to : ', addr)
    c.send('Hello, this is server.'.encode())
    c.close()
```

Output:



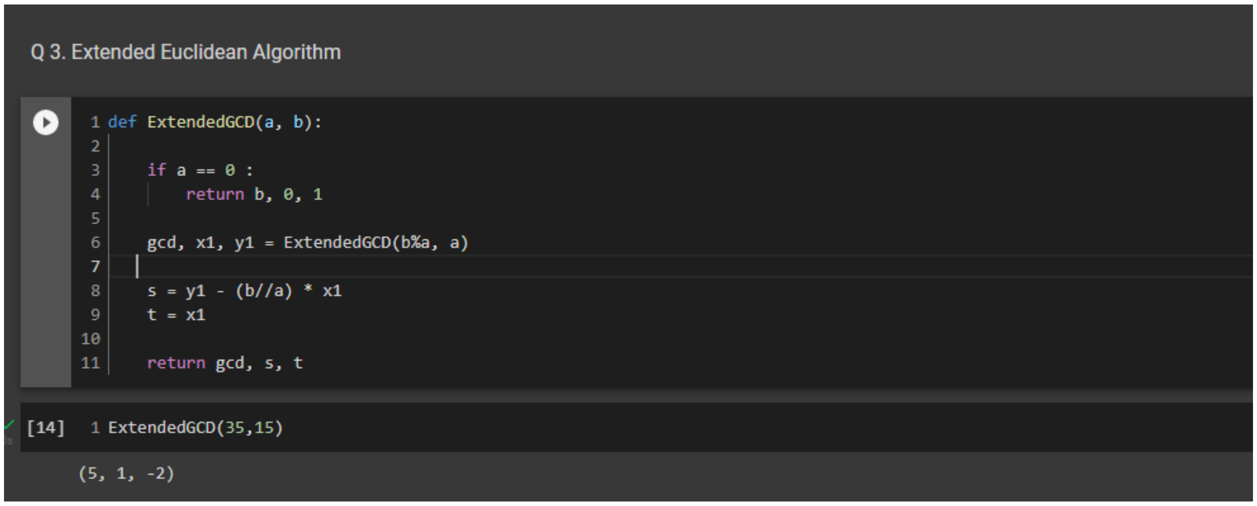
```
C:\Users\heman\Documents\labsheets\CNS\CNS-Lab>python WAP_client.py
The following message was received from the server :
>>  Hello, this is server.
Closing connection..

C:\Users\heman\Documents\labsheets\CNS\CNS-Lab>
```



Q2. Euclidean Algorithm for GCD

```
[11]  def EuclideanGCD(a, b):
          if a == 0 :
              return b

          return EuclideanGCD(b%a, a)
```

```
[12]  EuclideanGCD(35,15)

      5
```

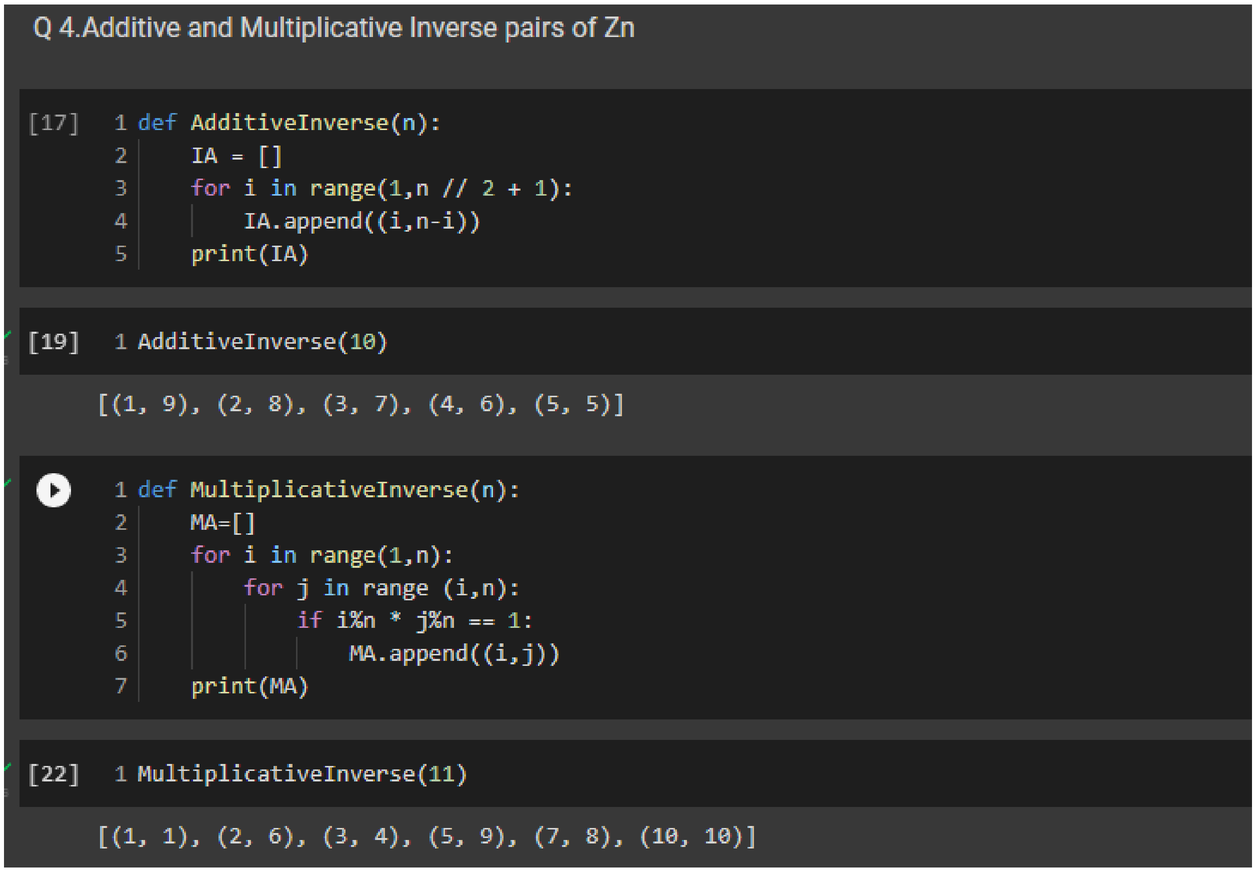


Q 3. Extended Euclidean Algorithm

```python
def ExtendedGCD(a, b):

    if a == 0 :
        return b, 0, 1

    gcd, x1, y1 = ExtendedGCD(b%a, a)

    s = y1 - (b//a) * x1
    t = x1

    return gcd, s, t
```

```
[14] ExtendedGCD(35,15)
```

```
(5, 1, -2)
```



Q 4.Additive and Multiplicative Inverse pairs of Zn

```python
[17] def AdditiveInverse(n):
         IA = []
         for i in range(1,n // 2 + 1):
             IA.append((i,n-i))
         print(IA)
```

```
[19] AdditiveInverse(10)
```

```
[(1, 9), (2, 8), (3, 7), (4, 6), (5, 5)]
```

```python
def MultiplicativeInverse(n):
    MA=[]
    for i in range(1,n):
        for j in range (i,n):
            if i%n * j%n == 1:
                MA.append((i,j))
    print(MA)
```

```
[22] MultiplicativeInverse(11)
```

```
[(1, 1), (2, 6), (3, 4), (5, 9), (7, 8), (10, 10)]
```