Indian Institute of Technology Jodhpur

CSL6010 – Cyber Security
LAB-1 Report

Name- Aman Srivastava
Roll No.- B20CS100

Date-12th Jan 2023

- **AIM:**

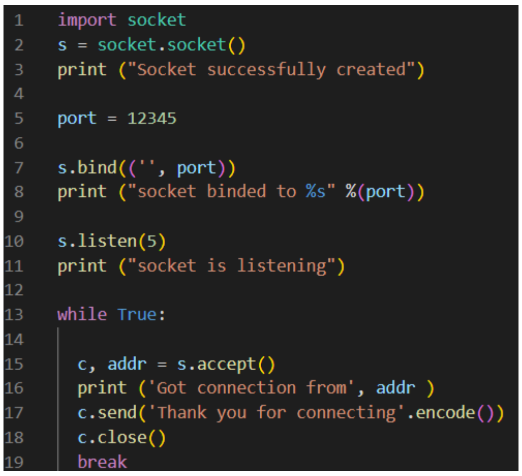
  Design and implementation of a simple client/server model and running application using sockets and TCP/IP.

In this lab, we attempted to create several programs using socket programming. The goal of these programs was to demonstrate the basic functionality of socket programming and how it can be used to create simple client-server interactions.

- **Problem Statement 1:**

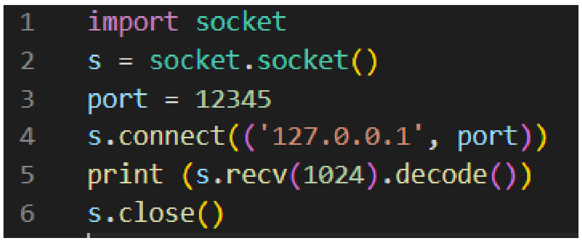
  Create a client server simple handshaking program using socket programming.

server.py



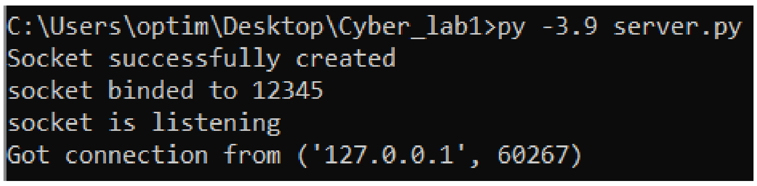
```python
import socket
s = socket.socket()
print ("Socket successfully created")

port = 12345

s.bind(('', port))
print ("socket binded to %s" %(port))

s.listen(5)
print ("socket is listening")

while True:

    c, addr = s.accept()
    print ('Got connection from', addr )
    c.send('Thank you for connecting'.encode())
    c.close()
    break
```

client.py

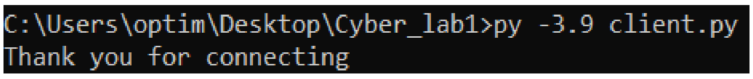


```python
import socket
s = socket.socket()
port = 12345
s.connect(('127.0.0.1', port))
print (s.recv(1024).decode())
s.close()
```

Output:



```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 server.py
Socket successfully created
socket binded to 12345
socket is listening
Got connection from ('127.0.0.1', 60267)
```



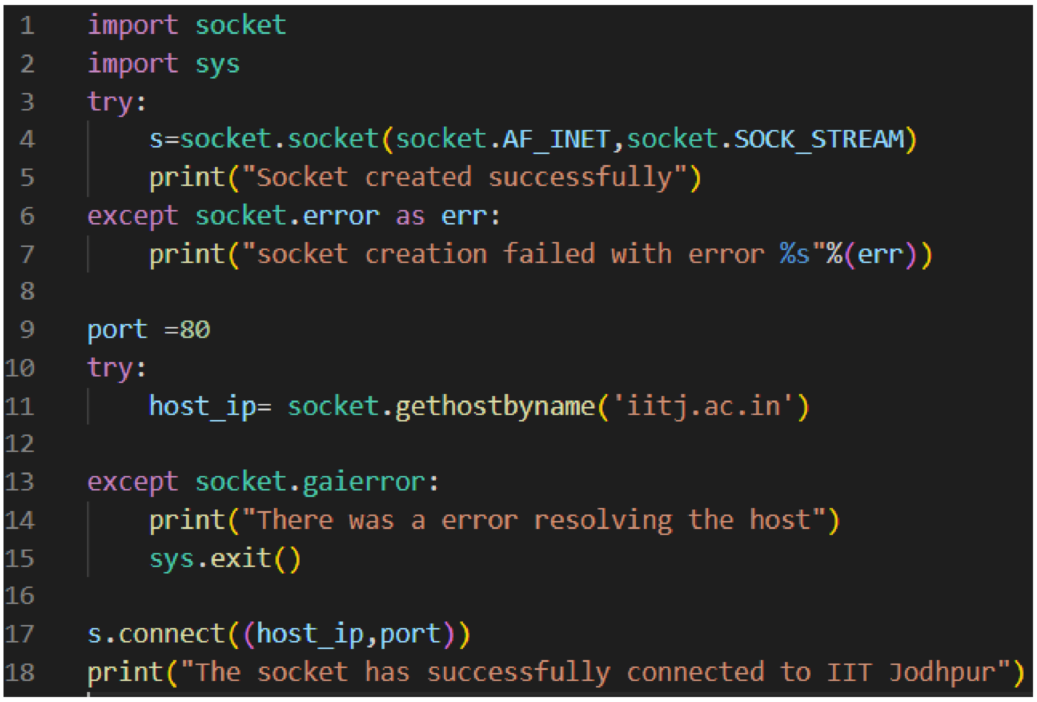
```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 client.py
Thank you for connecting
```

The first program we created was a simple handshaking program. The server side of the program used the socket module to create a socket, bind it to a specific IP address and port, and listen for incoming connections. Once a connection was established, the server and client were able to exchange data through the socket.

- **Problem Statement 2:**

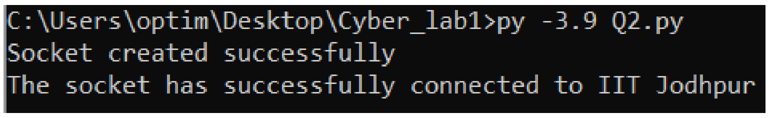
  Create a socket program to connect to a predefined server.

Q2.py



```python
import socket
import sys
try:
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    print("Socket created successfully")
except socket.error as err:
    print("socket creation failed with error %s"%(err))

port =80
try:
    host_ip= socket.gethostbyname('iitj.ac.in')

except socket.gaierror:
    print("There was a error resolving the host")
    sys.exit()

s.connect((host_ip,port))
print("The socket has successfully connected to IIT Jodhpur")
```

Output:



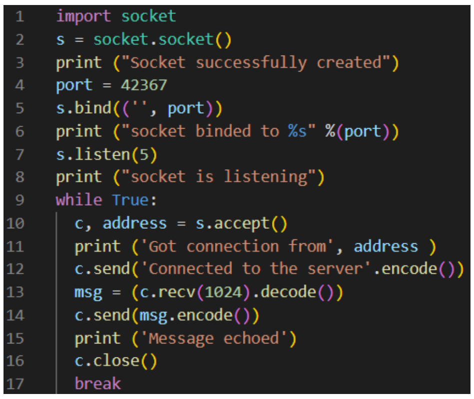
```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 Q2.py
Socket created successfully
The socket has successfully connected to IIT Jodhpur
```

This program was a client that connected to a predefined server (https://iitj.ac.in/). This program demonstrated how a client can use a socket to connect to a server and exchange data with it.

- **Problem Statement 3:**

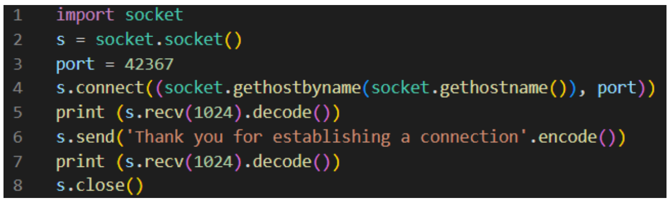
  Create a client server program using socket programming, Where the server will simply echo whatever it receives back to the client.

3_server.py



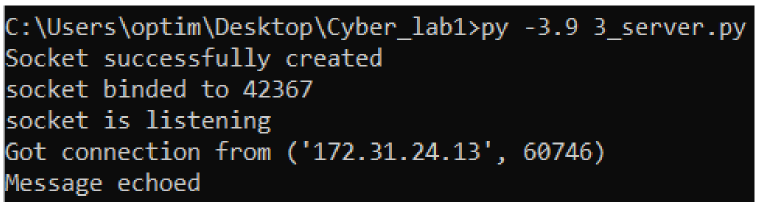
```python
import socket
s = socket.socket()
print ("Socket successfully created")
port = 42367
s.bind(('', port))
print ("socket binded to %s" %(port))
s.listen(5)
print ("socket is listening")
while True:
  c, address = s.accept()
  print ('Got connection from', address )
  c.send('Connected to the server'.encode())
  msg = (c.recv(1024).decode())
  c.send(msg.encode())
  print ('Message echoed')
  c.close()
  break
```

3_client.py

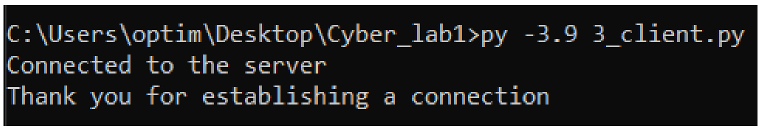


```python
import socket
s = socket.socket()
port = 42367
s.connect((socket.gethostbyname(socket.gethostname()), port))
print (s.recv(1024).decode())
s.send('Thank you for establishing a connection'.encode())
print (s.recv(1024).decode())
s.close()
```

Output:



```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 3_server.py
Socket successfully created
socket binded to 42367
socket is listening
Got connection from ('172.31.24.13', 60746)
Message echoed
```



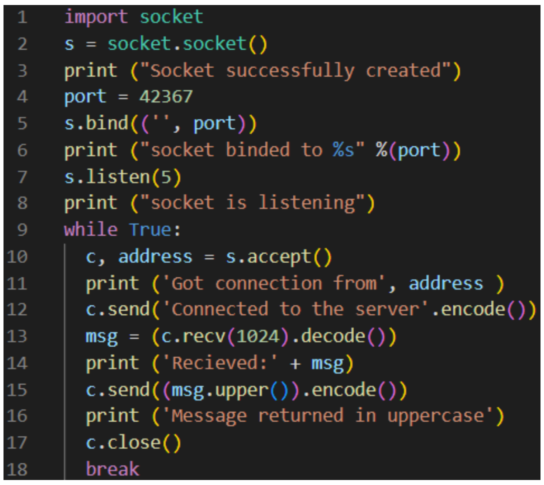
```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 3_client.py
Connected to the server
Thank you for establishing a connection
```

This program was a simple echo server. In this program, the server simply echoed whatever it received back to the client. This program was a good demonstration of how data can be transmitted between a client and a server over a socket.

- **Problem Statement 4:**

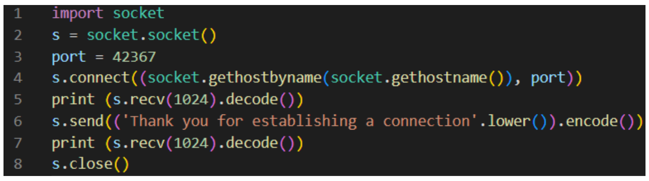
  Create a client server program using socket programming, Where the client sends a text message in Lowercase to the server and the server returns it in uppercase to the client.

4_server.py



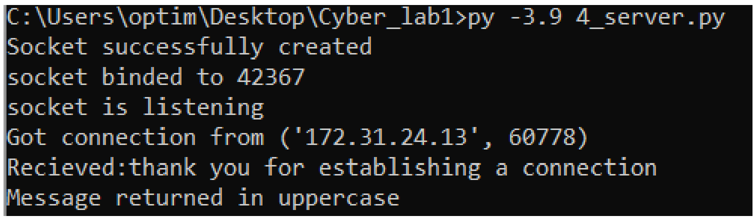
```python
import socket
s = socket.socket()
print ("Socket successfully created")
port = 42367
s.bind(('', port))
print ("socket binded to %s" %(port))
s.listen(5)
print ("socket is listening")
while True:
  c, address = s.accept()
  print ('Got connection from', address )
  c.send('Connected to the server'.encode())
  msg = (c.recv(1024).decode())
  print ('Recieved:' + msg)
  c.send((msg.upper()).encode())
  print ('Message returned in uppercase')
  c.close()
  break
```

4_client.py

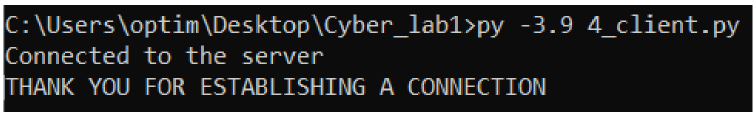


```python
import socket
s = socket.socket()
port = 42367
s.connect((socket.gethostbyname(socket.gethostname()), port))
print (s.recv(1024).decode())
s.send(('Thank you for establishing a connection'.lower()).encode())
print (s.recv(1024).decode())
s.close()
```

Output:



```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 4_server.py
Socket successfully created
socket binded to 42367
socket is listening
Got connection from ('172.31.24.13', 60778)
Recieved:thank you for establishing a connection
Message returned in uppercase
```



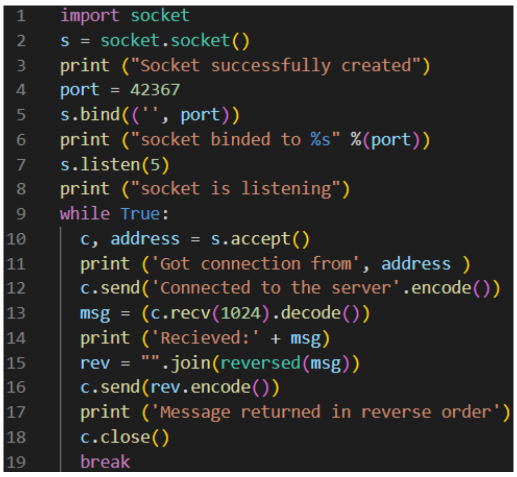
```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 4_client.py
Connected to the server
THANK YOU FOR ESTABLISHING A CONNECTION
```

This program was a client-server program where the client sent a text message in lowercase to the server and the server returned it in uppercase. This program demonstrated how the server can process data received from the client and return a modified version of that data to the client.

- **Problem Statement 5:**

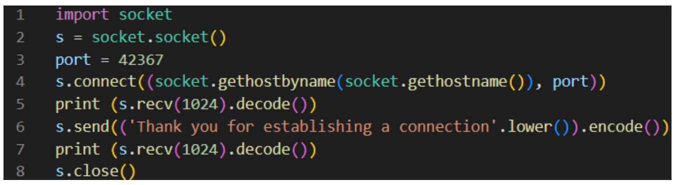
  Create a client server program using socket programming, Where the client sends a text message to the server and the server returns it in reverse order to the client.

server.py



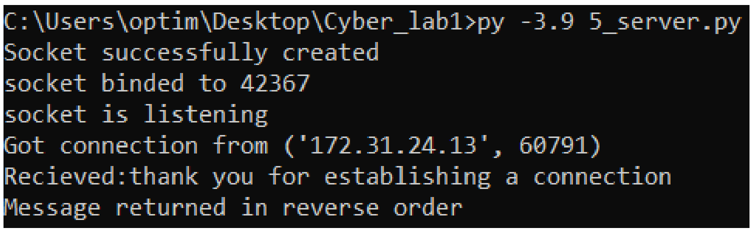
```python
import socket
s = socket.socket()
print ("Socket successfully created")
port = 42367
s.bind(('', port))
print ("socket binded to %s" %(port))
s.listen(5)
print ("socket is listening")
while True:
    c, address = s.accept()
    print ('Got connection from', address )
    c.send('Connected to the server'.encode())
    msg = (c.recv(1024).decode())
    print ('Recieved:' + msg)
    rev = "".join(reversed(msg))
    c.send(rev.encode())
    print ('Message returned in reverse order')
    c.close()
    break
```

client.py

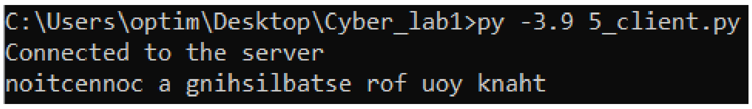


```python
import socket
s = socket.socket()
port = 42367
s.connect((socket.gethostbyname(socket.gethostname()), port))
print (s.recv(1024).decode())
s.send(('Thank you for establishing a connection'.lower()).encode())
print (s.recv(1024).decode())
s.close()
```

Output:



```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 5_server.py
Socket successfully created
socket binded to 42367
socket is listening
Got connection from ('172.31.24.13', 60791)
Recieved:thank you for establishing a connection
Message returned in reverse order
```



```
C:\Users\optim\Desktop\Cyber_lab1>py -3.9 5_client.py
Connected to the server
noitcennoc a gnihsilbatse rof uoy knaht
```

This program was a client-server program where the client sent a text message to the server and the server returned it in reverse order. This program demonstrated how the server can process data received from the client and return a modified version of that data to the client.

- **CONCLUSION:**

  I have successfully implemented all the five lab questions, it provided a good introduction to the basics of socket programming and how it can be used to create simple client-server interactions.