NAME : BHUSHAN KOYANDE
CLASS : TE COMPUTER
BATCH : B
UID : 2018130021
ROLL NO : 24

CEL 51, DCCN, Monsoon 2020
Lab 8: Socket Programming

AIM -

To establish connection between server client using sockets.

THEORY -

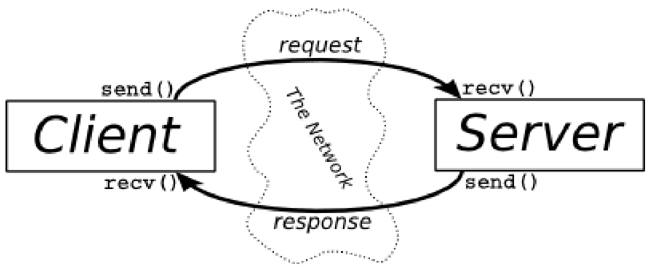
Background
Sockets have a long history. Their use originated with ARPANETin 1971 and later became an API in the Berkeley Software Distribution (BSD) operating system released in 1983 called Berkeley sockets.

When the Internet took off in the 1990s with the World Wide Web, so did network programming. Web servers and browsers weren't the only applications taking advantage of newly connected networks and using sockets. Client-server applications of all types and sizes came into widespread use[1].

Today, although the underlying protocols used by the socket API have evolved over the years, and we've seen new ones, the low-level API has remained the same.

The most common type of socket applications are client-server applications, where one side acts as the server and waits for connections from clients. This is the type of application that I'll be covering in this tutorial. More specifically, we'll look at the socket API for Internet sockets, sometimes called Berkeley or BSD sockets. There are also Unix domain sockets, which can only be used to communicate between processes on the same host.

What is socket programming?
Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket (node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server. They form the backbones of web browsing.[2]



The exchange of information between client and server is summarized in the above diagram.A server has a bind() method which binds it to a specific ip and port so that it can listen to incoming requests on that ip and port. A server has a listen() method which puts the server into listen mode. This allows the server to listen to incoming connections. And last a server has an accept() and close() method. The accept method initiates a connection with the client and the close method closes the connection with the client.

CODE -

Server

server.py

```
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((socket.gethostname(), 8000))
s.listen(5)

while True:
        clientsocket, address = s.accept()
        print(f'Connection established with {address}')
        clientsocket.send(bytes('Hello World! This is Bhushan', 'utf-8'))
        clientsocket.close()
```

Client

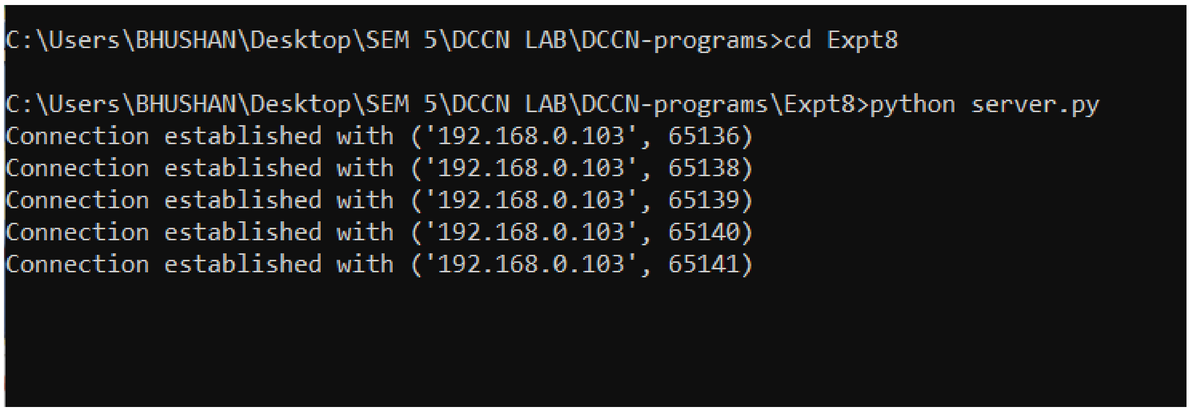
client.py

```
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((socket.gethostname(), 8000))
msg = s.recv(1024)
print(msg.decode('utf-8'))
```

OUTPUT -

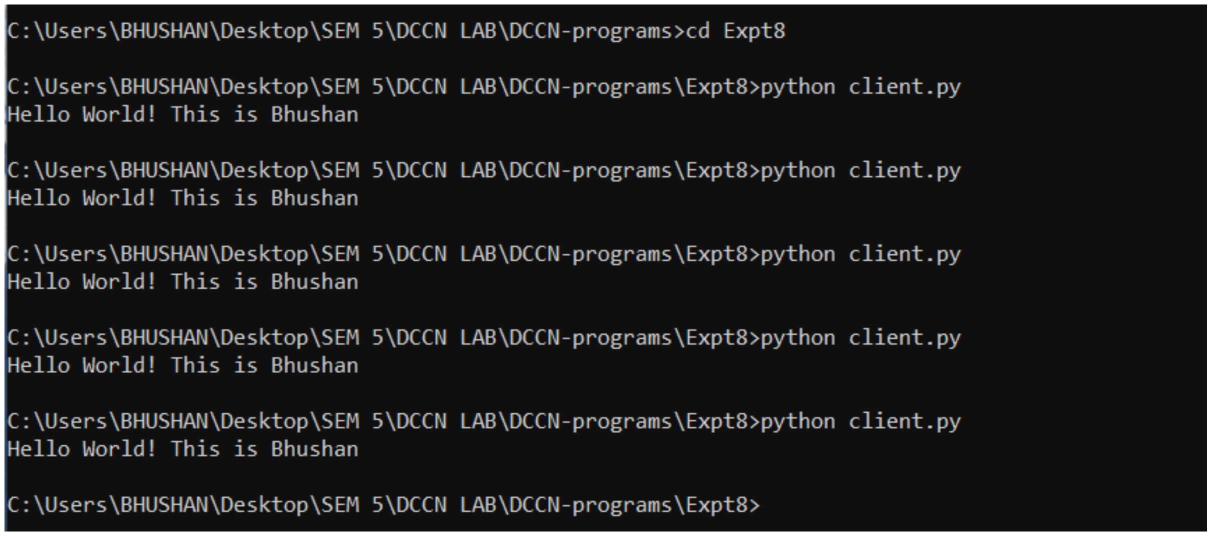
Server



```
C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs>cd Expt8

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>python server.py
Connection established with ('192.168.0.103', 65136)
Connection established with ('192.168.0.103', 65138)
Connection established with ('192.168.0.103', 65139)
Connection established with ('192.168.0.103', 65140)
Connection established with ('192.168.0.103', 65141)
```

Client



```
C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs>cd Expt8

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>python client.py
Hello World! This is Bhushan

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>python client.py
Hello World! This is Bhushan

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>python client.py
Hello World! This is Bhushan

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>python client.py
Hello World! This is Bhushan

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>python client.py
Hello World! This is Bhushan

C:\Users\BHUSHAN\Desktop\SEM 5\DCCN LAB\DCCN-programs\Expt8>
```

CONCLUSION -

I understood the basics of socket programming and established a simple connection between client and server using the socket programming with the help of Python language.

REFERENCES -

[1] https://realpython.com/python-sockets/
[2] https://www.geeksforgeeks.org/socket-programming-python/