

AIM:

To implement Socket Programming and establish a connection between client and server.

THEORY:

- 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 are the real backbones behind web browsing. In simpler terms there is a server and a client.
- Python provides two levels of access to network services. At a low level, you can access the basic socket support in the underlying operating system, which allows you to implement clients and servers for both connection-oriented and connectionless protocols.
- Python also has libraries that provide higher-level access to specific application-level network protocols, such as FTP, HTTP, and so on.
- Sockets have their own vocabulary:

Sr.No.	Term & Description
1	Domain The family of protocols that is used as the transport mechanism. These values are constants such as AF_INET, PF_INET, PF_UNIX, PF_X25, and so on.
2	type The type of communications between the two endpoints, typically SOCK_STREAM for connection-oriented protocols and SOCK_DGRAM for connectionless protocols.
3	protocol Typically zero, this may be used to identify a variant of a protocol within a domain and type.
4	hostname The identifier of a network interface – <ul style="list-style-type: none">• A string, which can be a host name, a dotted-quad address, or an IPV6 address in colon (and possibly dot) notation• A string "<broadcast>", which specifies an INADDR_BROADCAST address.• A zero-length string, which specifies INADDR_ANY, or• An Integer, interpreted as a binary address in host byte order.

5	<p>port</p> <p>Each server listens for clients calling on one or more ports. A port may be a Fixnum port number, a string containing a port number, or the name of a service.</p>
---	--

The `socket` Module

- To create a socket, you must use the `socket.socket()` function available in `socket` module, which has the general syntax –

```
s = socket.socket (socket_family, socket_type, protocol=0)
```
- Here is the description of the parameters –
 - socket_family** – This is either `AF_UNIX` or `AF_INET`, as explained earlier.
 - socket_type** – This is either `SOCK_STREAM` or `SOCK_DGRAM`.
 - protocol** – This is usually left out, defaulting to 0.
- Once you have `socket` object, then you can use required functions to create your client or server program.

Server Socket Methods

Sr.No.	Method & Description
1	<p>s.bind()</p> <p>This method binds address (hostname, port number pair) to socket.</p>
2	<p>s.listen()</p> <p>This method sets up and start TCP listener.</p>
3	<p>s.accept()</p> <p>This passively accept TCP client connection, waiting until connection arrives (blocking).</p>

Client Socket Methods

Sr.No.	Method & Description
1	s.connect() This method actively initiates TCP server connection.

General Socket Methods

Sr.No.	Method & Description
1	s.recv() This method receives TCP message
2	s.send() This method transmits TCP message
3	s.recvfrom() This method receives UDP message
4	s.sendto() This method transmits UDP message
5	s.close() This method closes socket
6	socket.gethostname() Returns the hostname.

Code:

Server.py

```
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((socket.gethostname(), 3000))
s.listen(5)

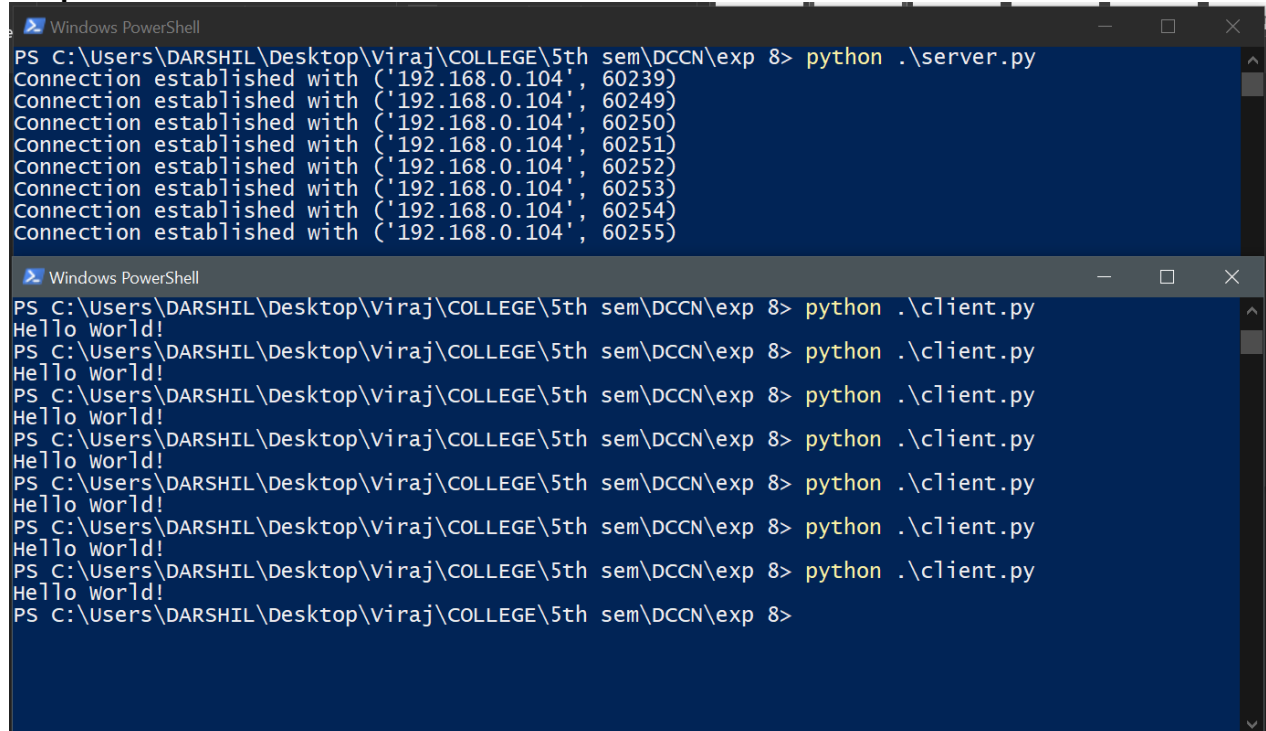
while True:
    clientsocket, address = s.accept()
    print(f'Connection established with {address}')
    clientsocket.send(bytes('Hello World!', 'utf-8'))
    clientsocket.close()
```

Client.py

```
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((socket.gethostname(), 3000))
msg = s.recv(1024)
print(msg.decode('utf-8'))
```

Output



The screenshot shows two Windows PowerShell windows. The top window, titled 'Windows PowerShell', shows the execution of 'python .\server.py'. It displays seven lines of output: 'Connection established with ('192.168.0.104', 60239)', 'Connection established with ('192.168.0.104', 60249)', 'Connection established with ('192.168.0.104', 60250)', 'Connection established with ('192.168.0.104', 60251)', 'Connection established with ('192.168.0.104', 60252)', 'Connection established with ('192.168.0.104', 60253)', and 'Connection established with ('192.168.0.104', 60254)'. The bottom window, also titled 'Windows PowerShell', shows the execution of 'python .\client.py' multiple times. Each execution results in the output 'Hello world!'.

```
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\server.py
Connection established with ('192.168.0.104', 60239)
Connection established with ('192.168.0.104', 60249)
Connection established with ('192.168.0.104', 60250)
Connection established with ('192.168.0.104', 60251)
Connection established with ('192.168.0.104', 60252)
Connection established with ('192.168.0.104', 60253)
Connection established with ('192.168.0.104', 60254)

PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8> python .\client.py
Hello world!
PS C:\Users\DARSHIL\Desktop\Viraj\COLLEGE\5th sem\DCCN\exp 8>
```

CONCLUSION:

I understood how to successfully establish a connection between client and server using socket programming.

REFEERENCES:

1. [geeksforgeeks.org/socket-programming-python/](https://www.geeksforgeeks.org/socket-programming-python/)
2. https://www.tutorialspoint.com/python_network_programming/python_sockets_programming.htm