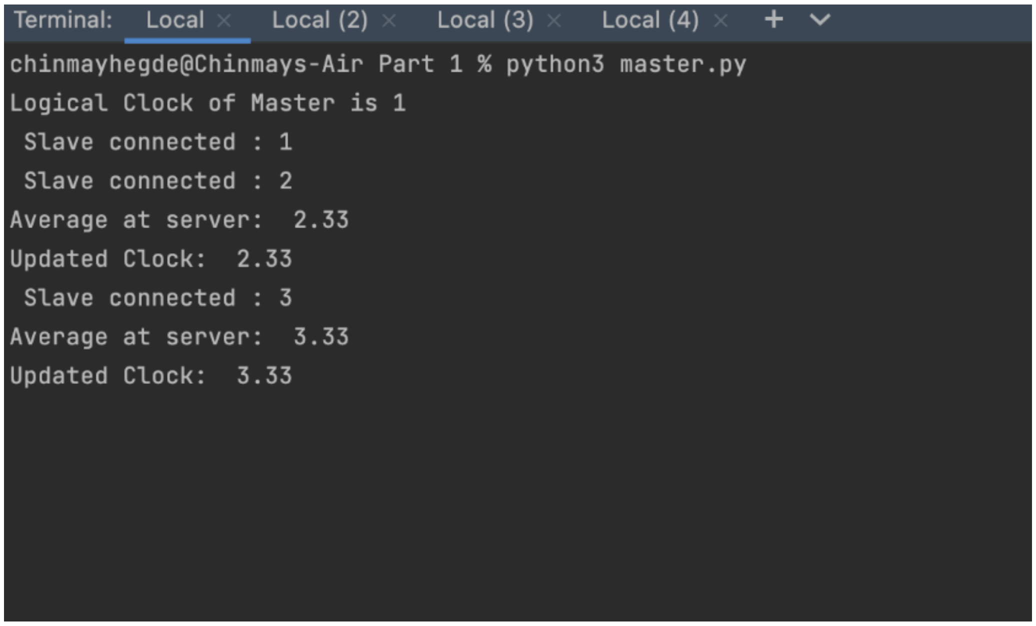
## STEPS TO IMPLEMENT :

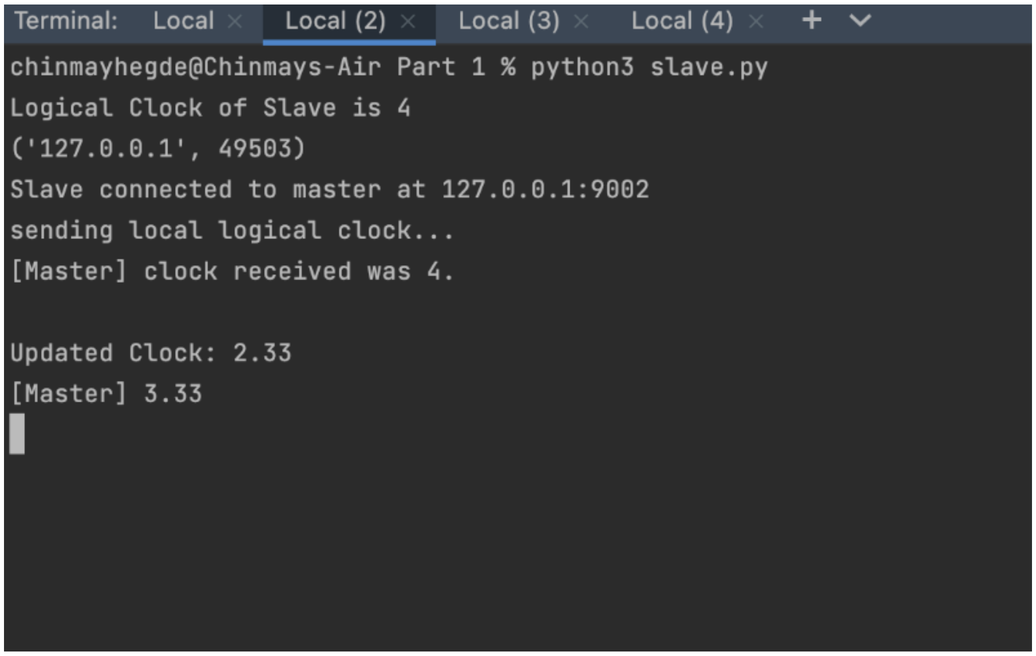
### PART 1

Below are the screen-shots of the running of the program



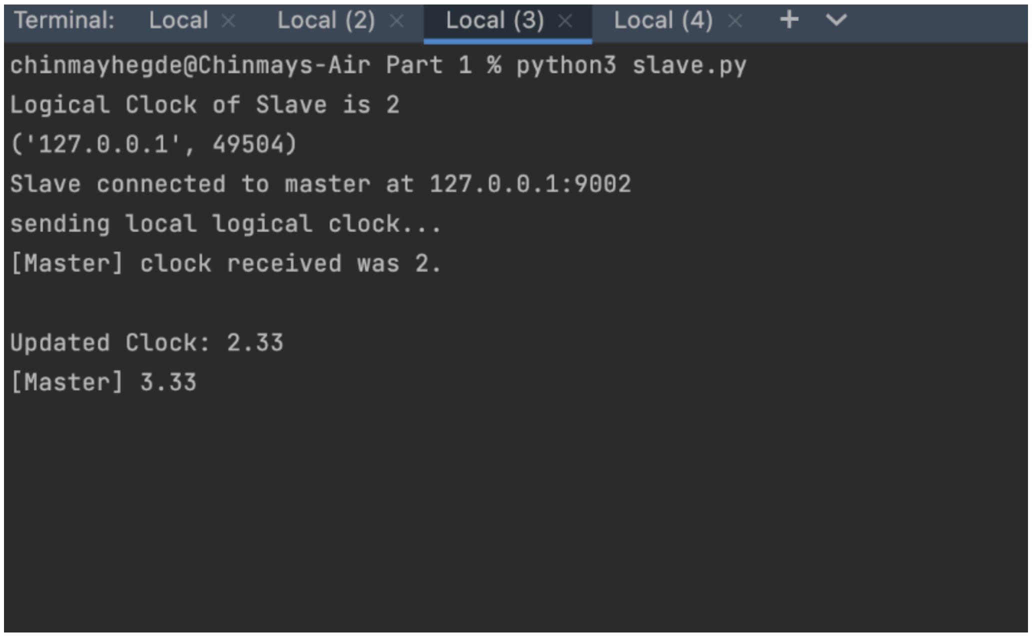
```
chinmayhegde@Chinmays-Air Part 1 % python3 master.py
Logical Clock of Master is 1
 Slave connected : 1
 Slave connected : 2
Average at server:  2.33
Updated Clock:  2.33
 Slave connected : 3
Average at server:  3.33
Updated Clock:  3.33
```

Fig 1: Terminal running Master displays the average clock time



```
chinmayhegde@Chinmays-Air Part 1 % python3 slave.py
Logical Clock of Slave is 4
('127.0.0.1', 49503)
Slave connected to master at 127.0.0.1:9002
sending local logical clock...
[Master] clock received was 4.

Updated Clock: 2.33
[Master] 3.33
```

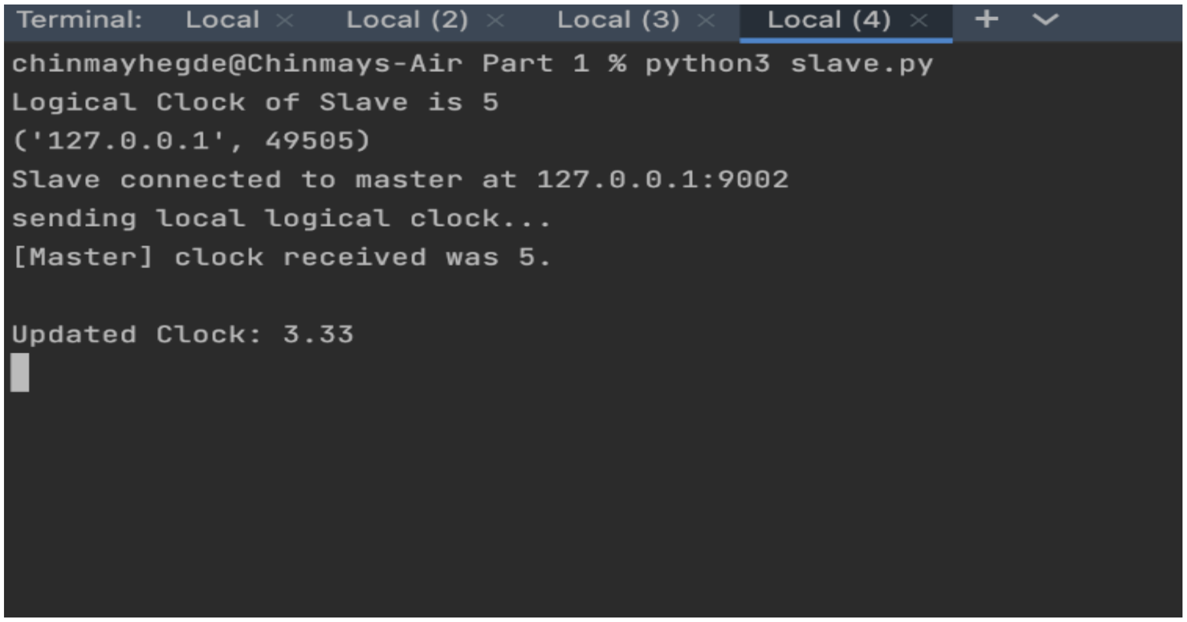
Fig 2: Terminal running Slave(1) displaying its own logical time and the average time after sync



Terminal: Local Local (2) Local (3) Local (4)

```
chinmayhegde@Chinmays-Air Part 1 % python3 slave.py
Logical Clock of Slave is 2
('127.0.0.1', 49504)
Slave connected to master at 127.0.0.1:9002
sending local logical clock...
[Master] clock received was 2.

Updated Clock: 2.33
[Master] 3.33
```

Fig 3: Terminal running Slave(2) displaying its own logical time and the average time after sync



Terminal: Local Local (2) Local (3) Local (4)

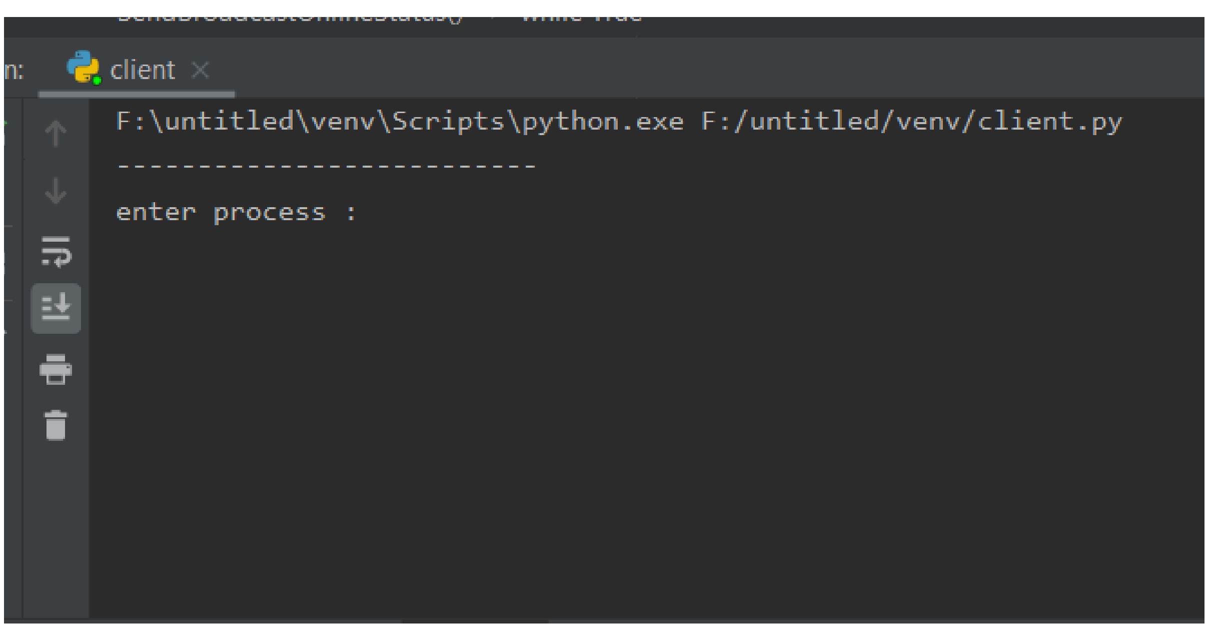
```
chinmayhegde@Chinmays-Air Part 1 % python3 slave.py
Logical Clock of Slave is 5
('127.0.0.1', 49505)
Slave connected to master at 127.0.0.1:9002
sending local logical clock...
[Master] clock received was 5.

Updated Clock: 3.33
```

Fig 4: Terminal running Slave(3) displaying its own logical time and the average time after sync.

## PART 2

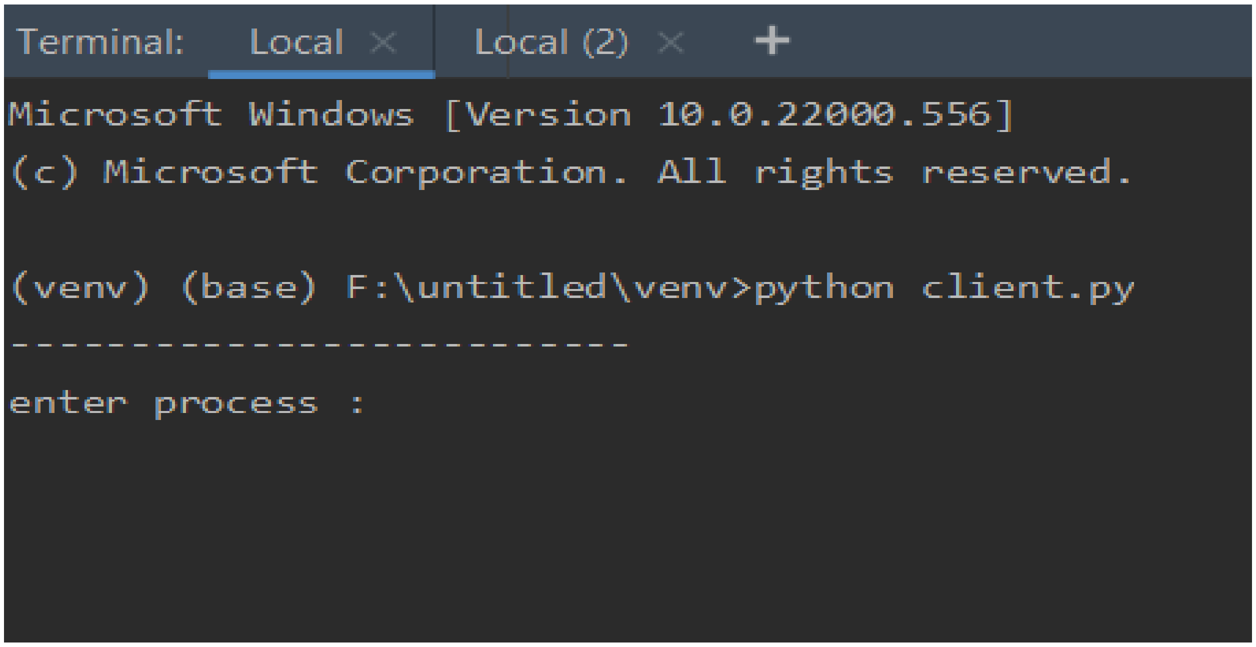
**Implementation:** At first open the file client.py which is present in the folder part2.

- Open the terminal and navigate to the folder part2 in the terminal.
- Now try to run the given client.py folder by executing "**python client.py**" in the terminal.
- Now open two more terminals and execute the same command "**python client.py**".



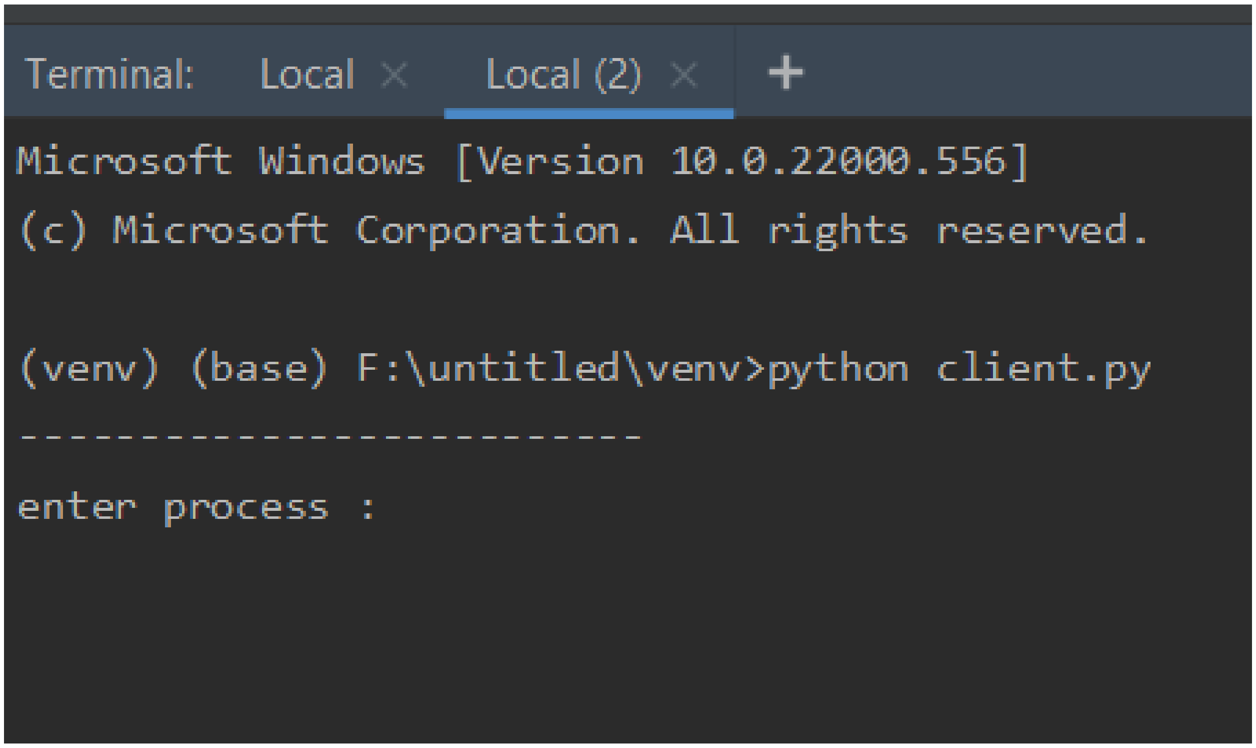
```
F:\untitled\venv\Scripts\python.exe F:/untitled/venv/client.py
--------------------------
enter process :
```

Fig 5: Executing the client.py in terminal one



```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.

(venv) (base) F:\untitled\venv>python client.py
---------------------------
enter process :
```

Fig 6: Executing the client.py in terminal two

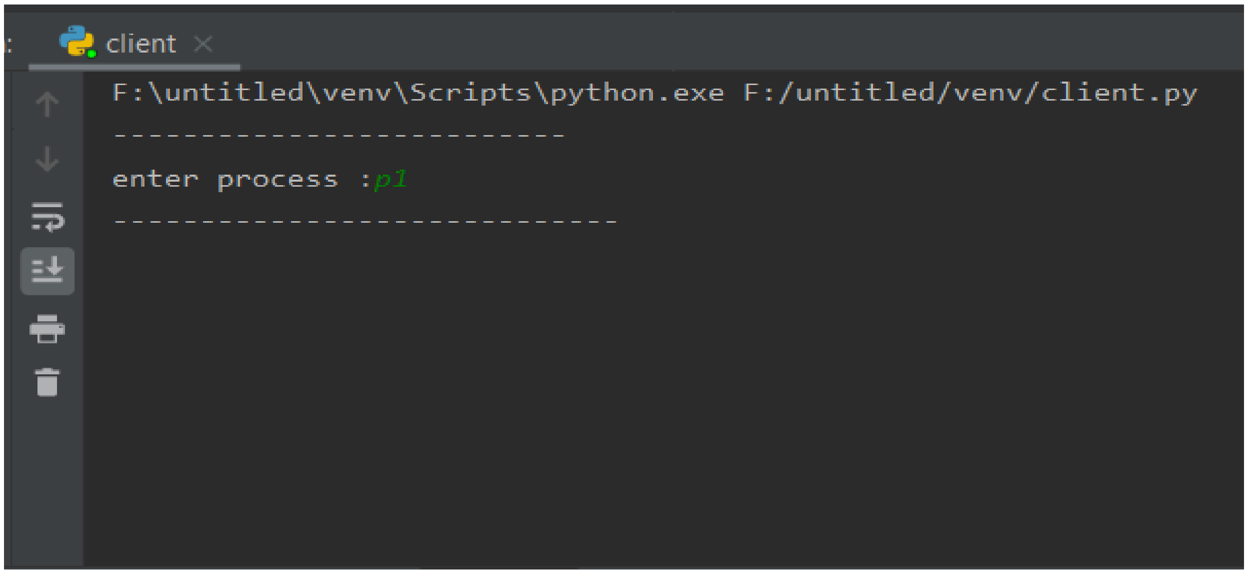


```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.

(venv) (base) F:\untitled\venv>python client.py
---------------------------
enter process :
```

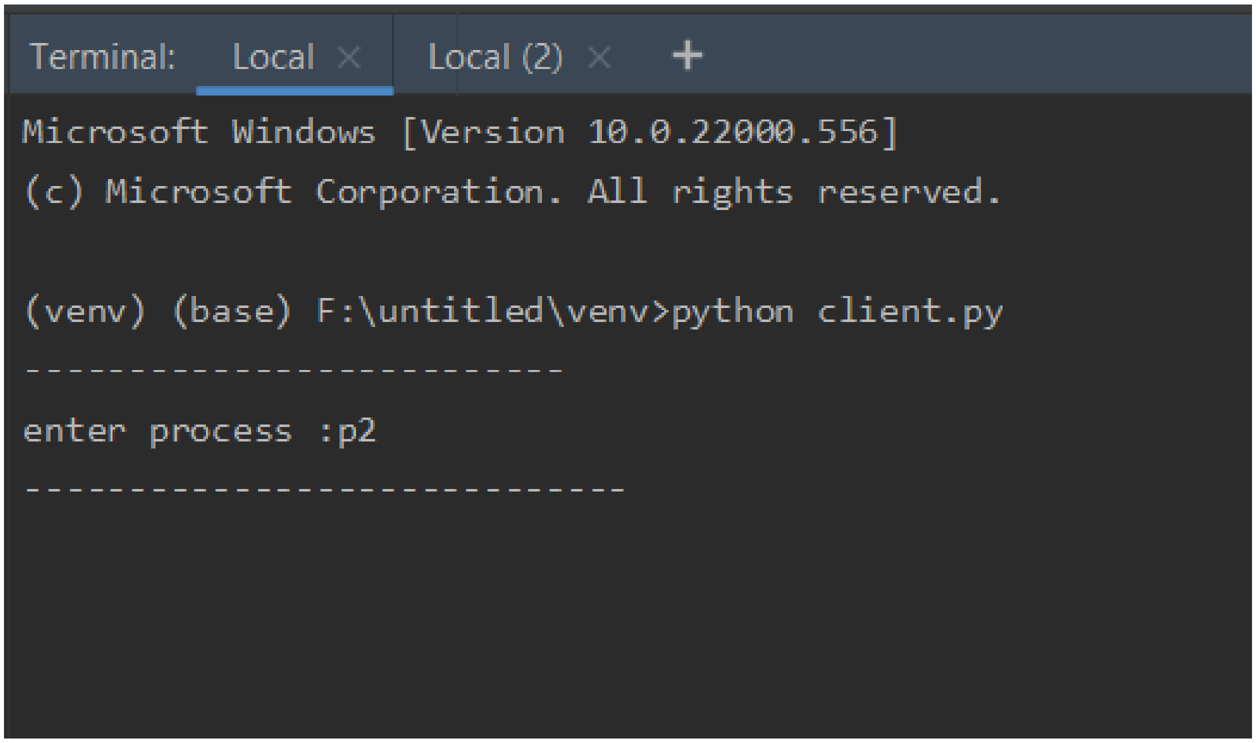
Fig 7: Executing the client.py in terminal three

The main reason behind executing client.py in three different terminals is to create three different process to create a communication between each node.( you can execute as many process as you want)

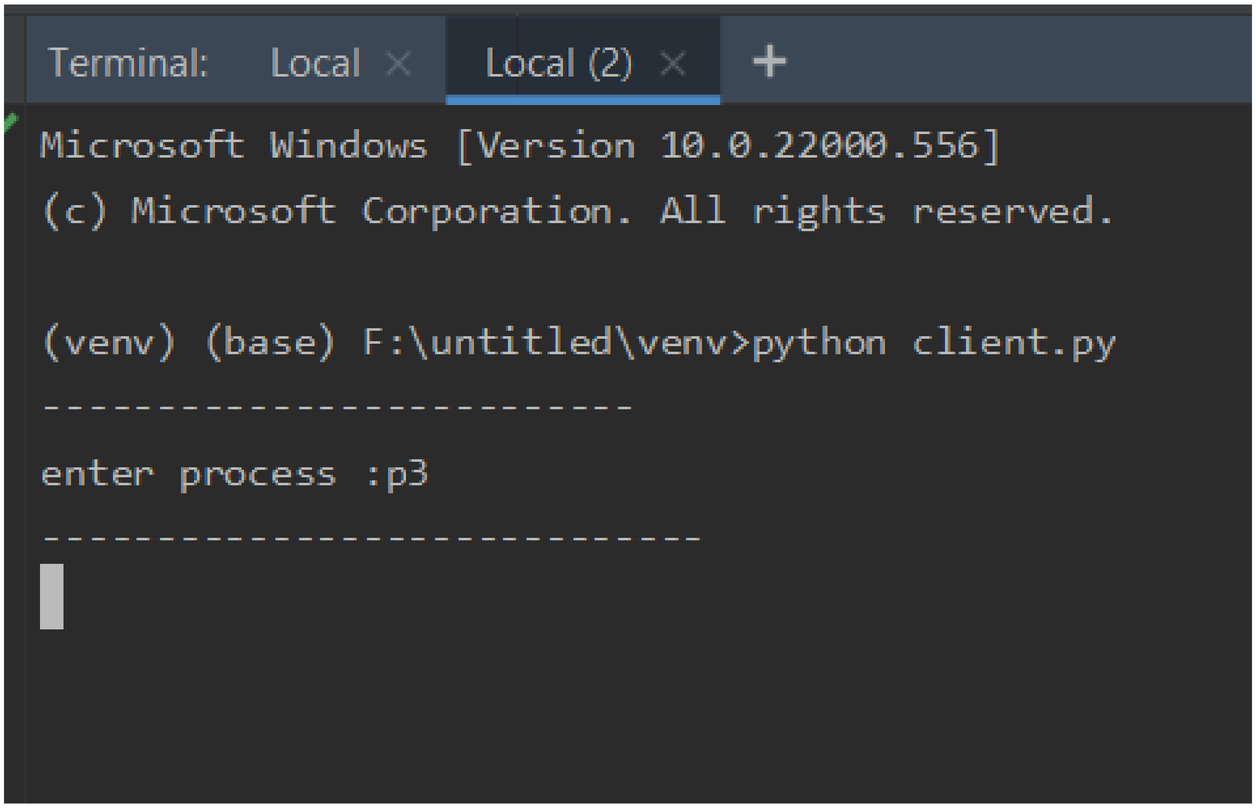
- Here comes the main part after you execute the given file in terminal one , Give a name to your process in terminal one (for eg: p1)
- Similar with cases terminal 2 (for eg: p2) and terminal 3(for eg: p3)



```
F:\untitled\venv\Scripts\python.exe F:/untitled/venv/client.py
--------------------------
enter process :p1
-----------------------------
```

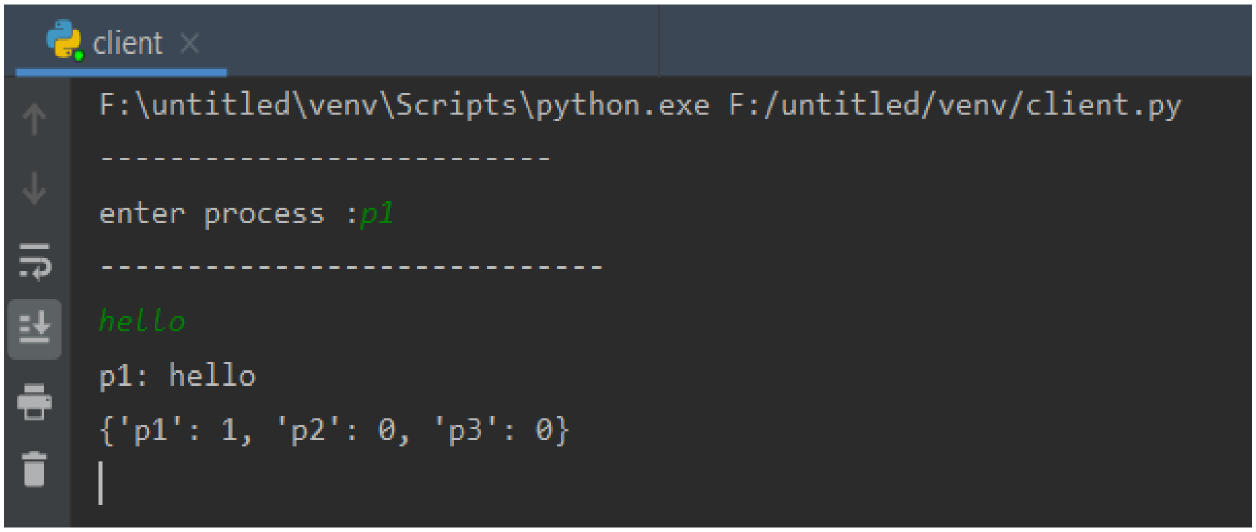


```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.

(venv) (base) F:\untitled\venv>python client.py
--------------------------
enter process :p2
-----------------------------
```



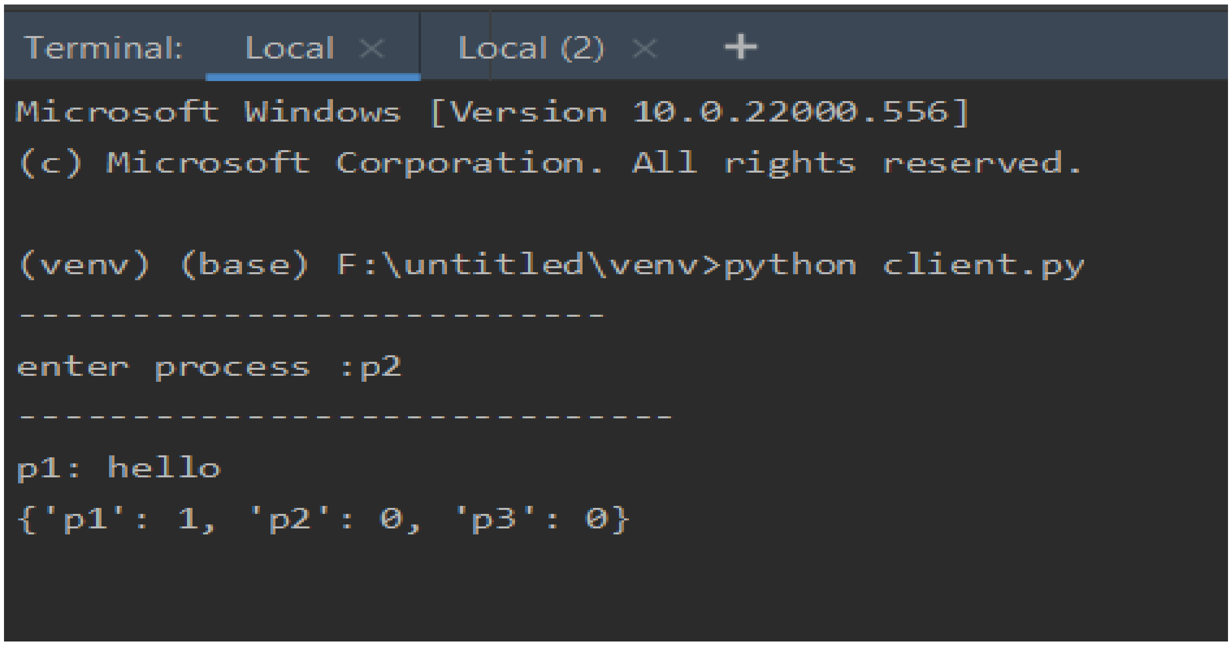
```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.

(venv) (base) F:\untitled\venv>python client.py
--------------------------
enter process :p3
-----------------------------
```

- Now p1 tries to send a message to rest of the peers in the network ( for example: “ hello “  )



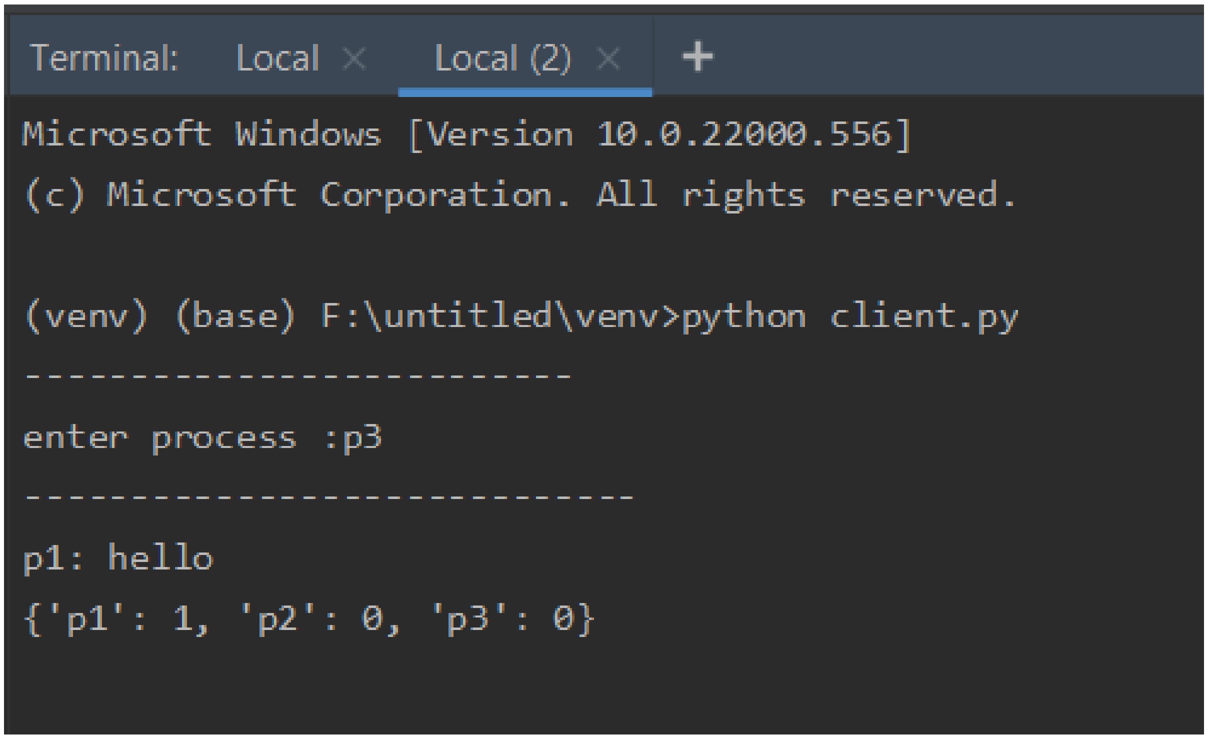
```
F:\untitled\venv\Scripts\python.exe F:/untitled/venv/client.py
--------------------------
enter process :p1
-----------------------------
hello
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
```

- **{‘p1’: 1, ’p2’ : 0, ‘p3’ : 0}** states that p1 has sent the message to other peers, then the vector clock implements the vector p1 to 1 and the rest of the other two remains 0 as the process didn't send any message .
- Now let's check what is happening with the other two terminals.



```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.

(venv) (base) F:\untitled\venv>python client.py
--------------------------
enter process :p2
-----------------------------
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
```

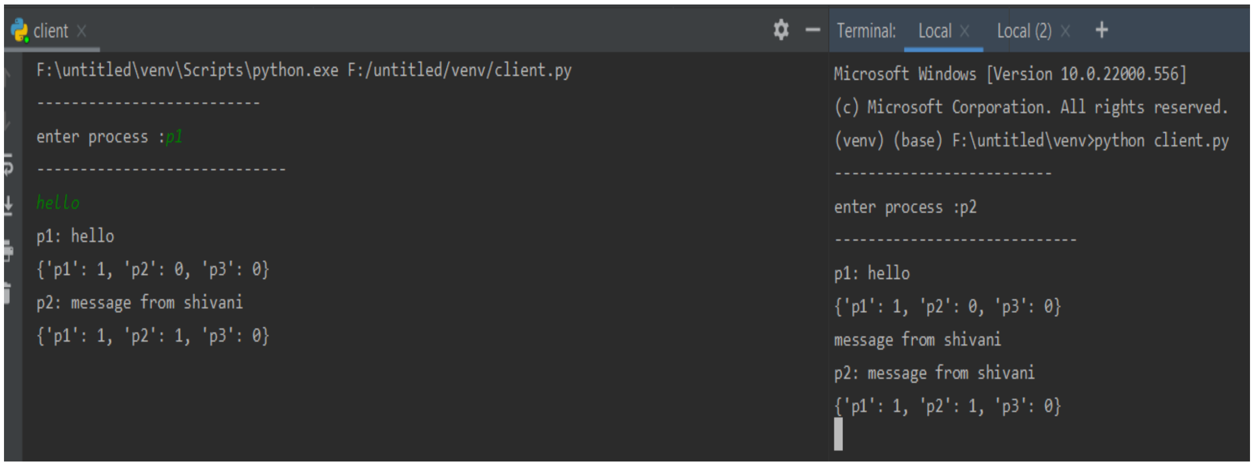
Fig 8: vector clock in terminal 2



```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.

(venv) (base) F:\untitled\venv>python client.py
--------------------------
enter process :p3
-----------------------------
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
```

Fig 9: vector clock in terminal 3

Now let's try to send a message from p2 to other peers (for example: message from shivani).

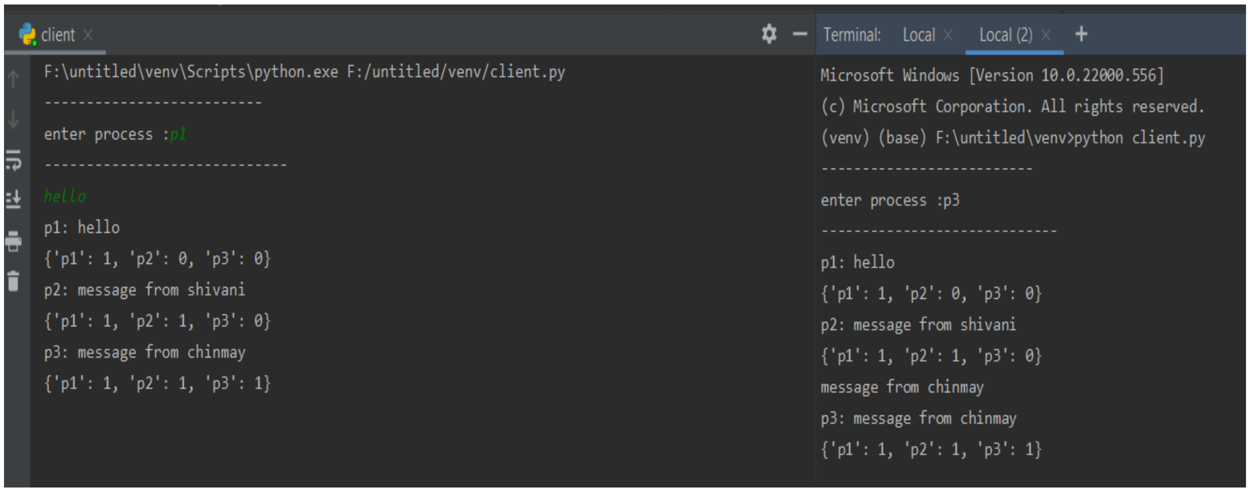


```
F:\untitled\venv\Scripts\python.exe F:/untitled/venv/client.py
------------------------
enter process :p1
---------------------------
hello
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
p2: message from shivani
{'p1': 1, 'p2': 1, 'p3': 0}
```

```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.
(venv) (base) F:\untitled\venv>python client.py
------------------------
enter process :p2
---------------------------
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
message from shivani
p2: message from shivani
{'p1': 1, 'p2': 1, 'p3': 0}
```

If you observe the two terminals the value of p2 is getting incremented. Hence displays you the new vector clock.

Similarly sending message from p3 (for example: message from chinmay)



```
F:\untitled\venv\Scripts\python.exe F:/untitled/venv/client.py
-------------------------
enter process :p1
----------------------------
hello
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
p2: message from shivani
{'p1': 1, 'p2': 1, 'p3': 0}
p3: message from chinmay
{'p1': 1, 'p2': 1, 'p3': 1}
```

```
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. All rights reserved.
(venv) (base) F:\untitled\venv>python client.py
-------------------------
enter process :p3
----------------------------
p1: hello
{'p1': 1, 'p2': 0, 'p3': 0}
p2: message from shivani
{'p1': 1, 'p2': 1, 'p3': 0}
message from chinmay
p3: message from chinmay
{'p1': 1, 'p2': 1, 'p3': 1}
```

If you observe the two terminals the value of p3 is getting incremented. Hence displays you the new vector clock.

Hence you can keep on sending messages from different processes and try to communicate with other peers. The vector clock keeps on getting updates as per the communication.