# LAB ASSIGNMENT V

**COMPUTER NETWORKS LAB (BCSE308P)**

**CLASS ID - VL2024250102387**

**MAXIMUM MARKS**: 10 **DUE DATE**: 15 NOVEMBER 2024

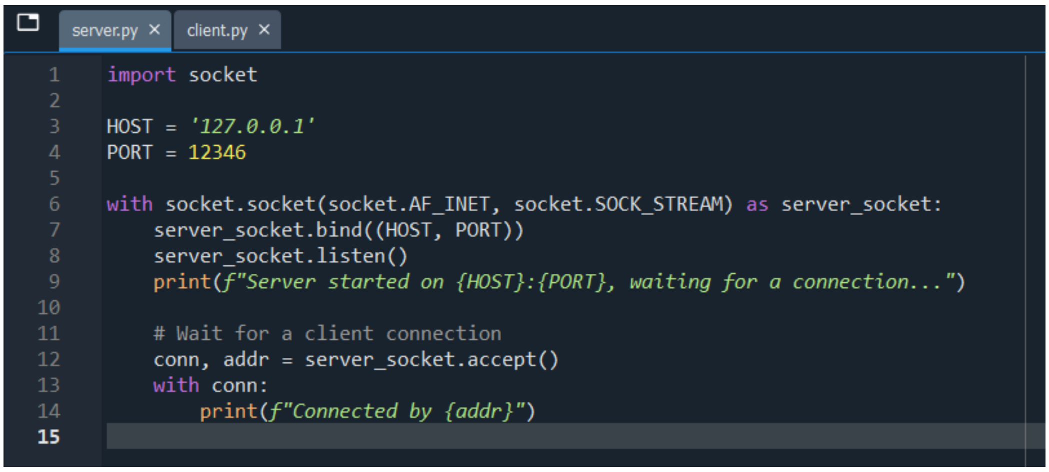
**NAME: R HEMESH** **REG NO:22BCT0328**

**Using the Spyder IDE, perform the following tasks utilizing Socket Programming library in Python. Include the screenshot of each step.**

1. **The server is first started on a known port.**

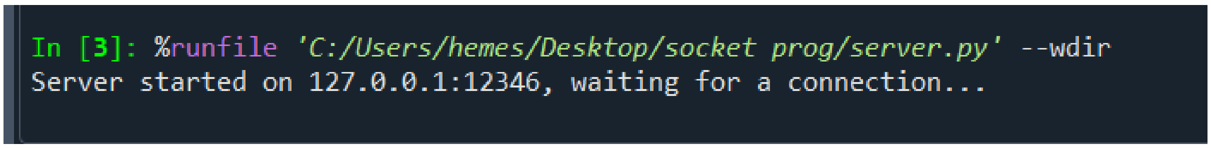
   **server.py :**

```python
import socket
HOST = '127.0.0.1'
PORT = 12345
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"Server started on {HOST}:{PORT}, waiting for a connection...")
    # Wait for a client connection
    conn, addr = server_socket.accept()
    with conn:
        print(f"Connected by {addr}")
```



```python
import socket

HOST = '127.0.0.1'
PORT = 12346

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"Server started on {HOST}:{PORT}, waiting for a connection...")

    # Wait for a client connection
    conn, addr = server_socket.accept()
    with conn:
        print(f"Connected by {addr}")
```

**OUTPUT in the server.py terminal:**

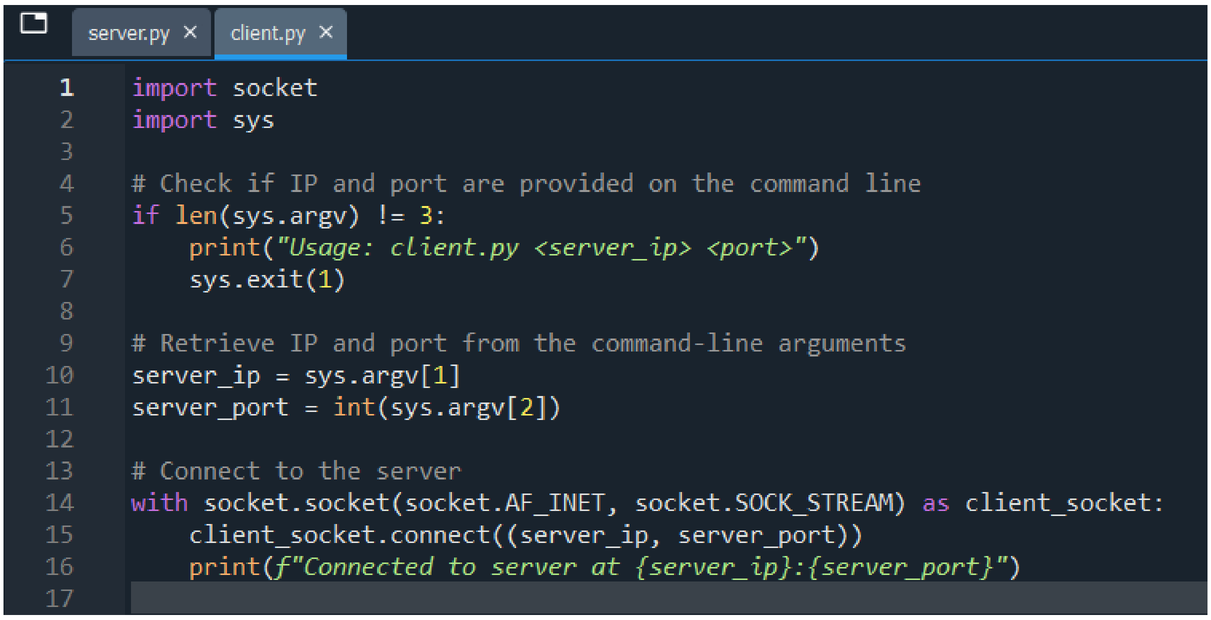


```
In [3]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
```

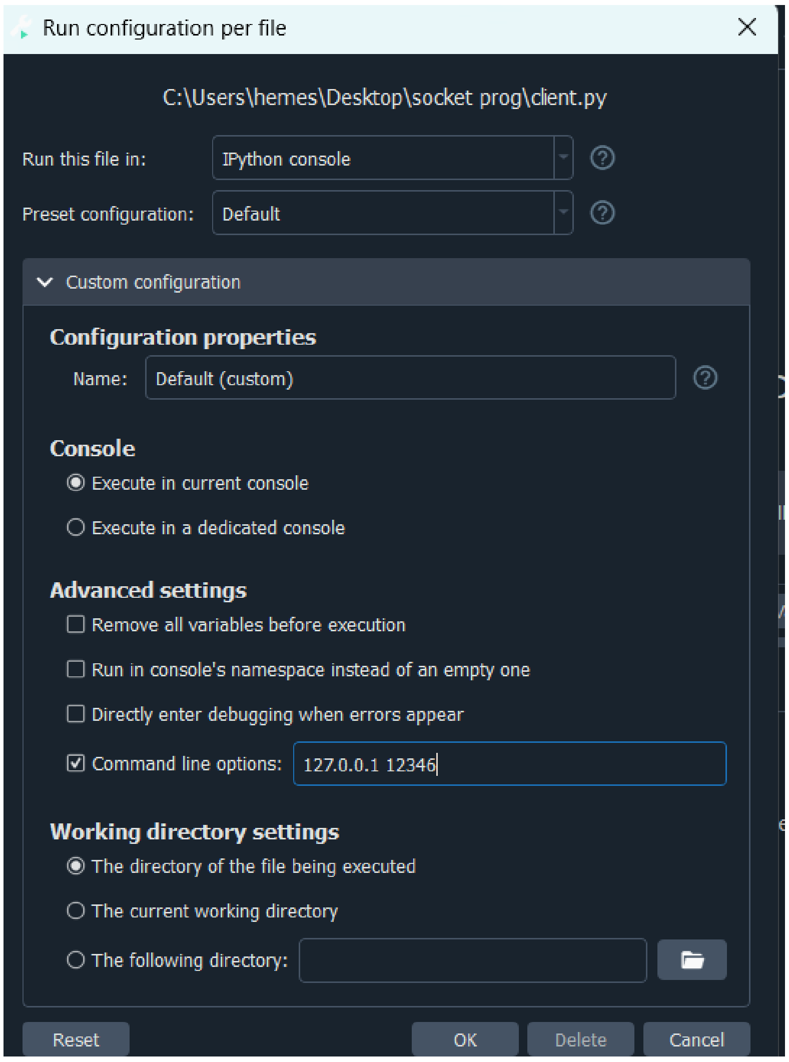
2. **The client program is started (server IP & port are provided on the commandline).**

**client.py :**

```
import socket
import sys

# Check if IP and port are provided on the command line
if len(sys.argv) != 3:
    print("Usage: client.py <server_ip> <port>")
    sys.exit(1)

# Retrieve IP and port from the command-line arguments
server_ip = sys.argv[1]
server_port = int(sys.argv[2])

# Connect to the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
    client_socket.connect((server_ip, server_port))
    print(f"Connected to server at {server_ip}:{server_port}")
```



```
import socket
import sys

# Check if IP and port are provided on the command line
if len(sys.argv) != 3:
    print("Usage: client.py <server_ip> <port>")
    sys.exit(1)

# Retrieve IP and port from the command-line arguments
server_ip = sys.argv[1]
server_port = int(sys.argv[2])

# Connect to the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
    client_socket.connect((server_ip, server_port))
    print(f"Connected to server at {server_ip}:{server_port}")
```


Run configuration per file

C:\Users\hemes\Desktop\socket prog\client.py

Run this file in: IPython console

Preset configuration: Default

Custom configuration

**Configuration properties**

Name: Default (custom)

**Console**

- [✓] Execute in current console
- [ ] Execute in a dedicated console

**Advanced settings**

- [ ] Remove all variables before execution
- [ ] Run in console's namespace instead of an empty one
- [ ] Directly enter debugging when errors appear
- [✓] Command line options: 127.0.0.1 12346

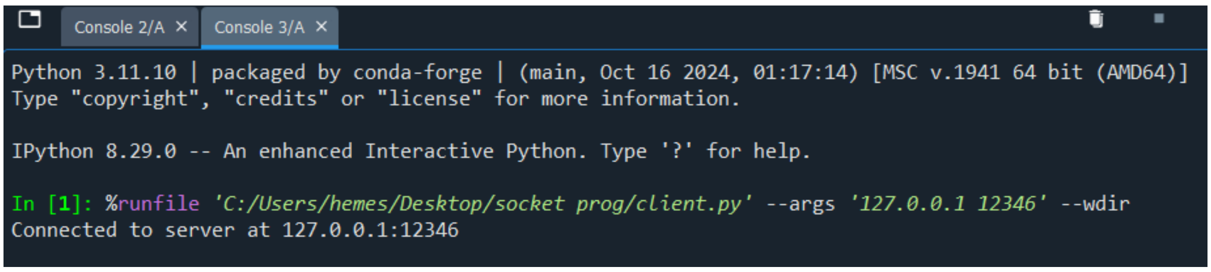
**Working directory settings**

- [✓] The directory of the file being executed
- [ ] The current working directory
- [ ] The following directory:

**server IP & port are provided on the commandline as specified.**

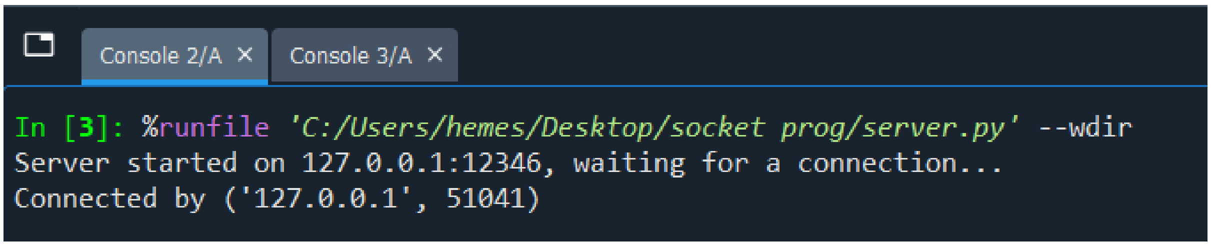
**In the configuration file of client.py , we provide server IP and port in the command line “ 127.0.0.1 12346”**

**OUTPUT in the client.py terminal :**



```
Python 3.11.10 | packaged by conda-forge | (main, Oct 16 2024, 01:17:14) [MSC v.1941 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.29.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
```

**OUTPUT in the server.py terminal:**

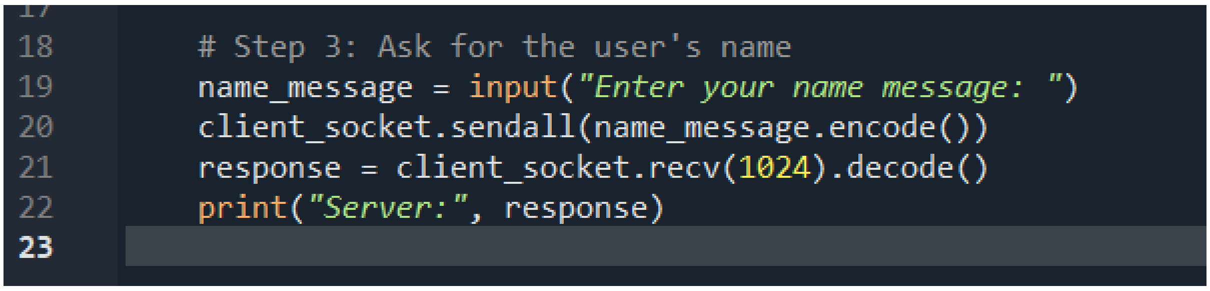


```
In [3]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
Connected by ('127.0.0.1', 51041)
```

3. **The client connects to the server, and then asks the user for input. The user types the message on the terminal “My name is ”. The user's input is sent to the server via the connected socket.**

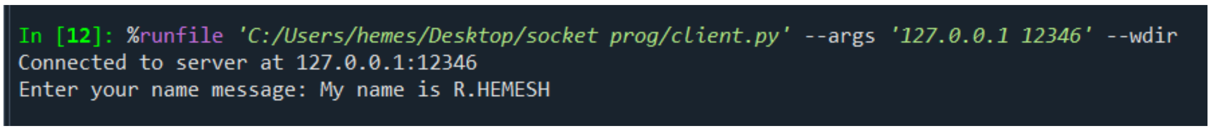
**client.py:**

```
    # Step 3: Ask for the user's name
    name_message = input("Enter your name message: ")
    client_socket.sendall(name_message.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)
```



```
    # Step 3: Ask for the user's name
    name_message = input("Enter your name message: ")
    client_socket.sendall(name_message.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)
```

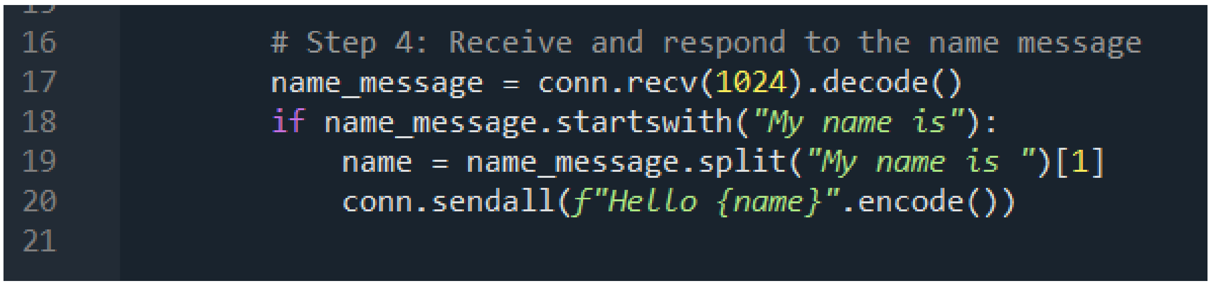
**OUTPUT in the client.py terminal :**



```
In [12]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
```

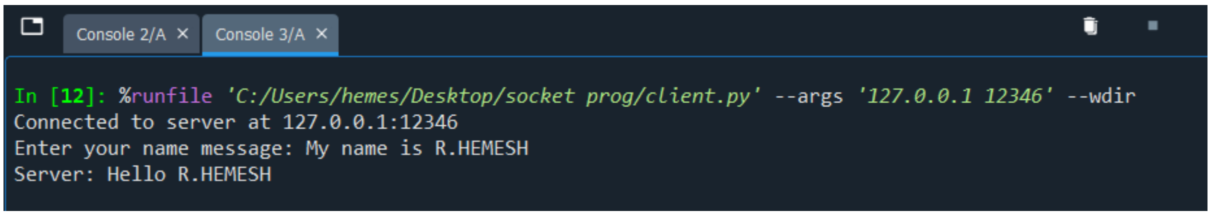
**4. The server reads the user's input from the client socket and replies with "Hello ".**
**server.py:**

```
        # Step 4: Receive and respond to the name message
        name_message = conn.recv(1024).decode()
        if name_message.startswith("My name is"):
            name = name_message.split("My name is ")[1]
            conn.sendall(f"Hello {name}".encode())
```



```
        # Step 4: Receive and respond to the name message
        name_message = conn.recv(1024).decode()
        if name_message.startswith("My name is"):
            name = name_message.split("My name is ")[1]
            conn.sendall(f"Hello {name}".encode())
```

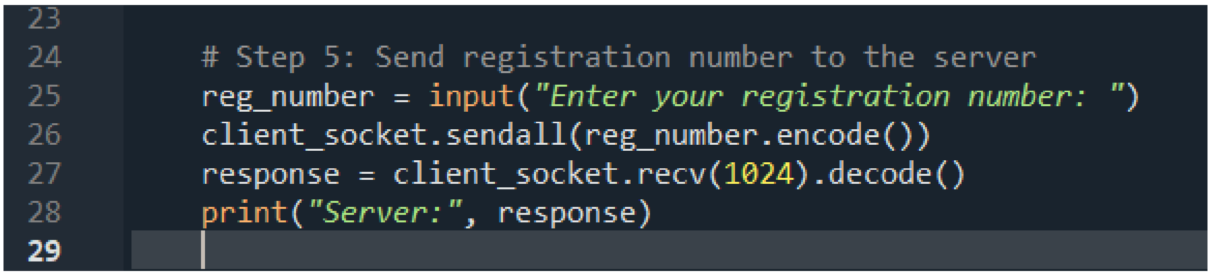
**OUTPUT in the client.py terminal :**



```
In [12]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
```

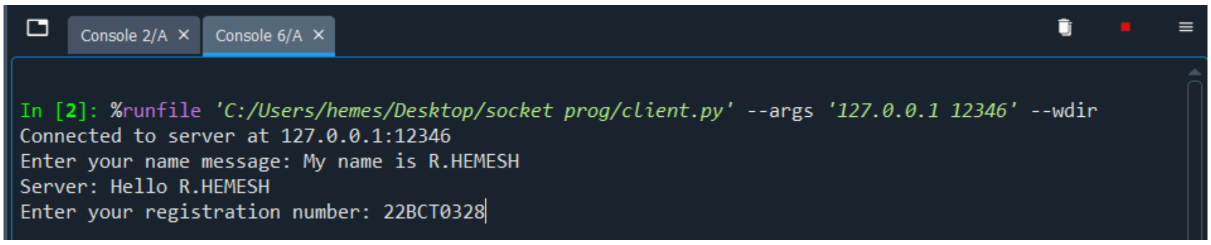
**5. Then, the client asks the user for the registration number to send to the server.**
**client.py:**

```
    # Step 5: Send registration number to the server
    reg_number = input("Enter your registration number: ")
    client_socket.sendall(reg_number.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)
```



```

    # Step 5: Send registration number to the server
    reg_number = input("Enter your registration number: ")
    client_socket.sendall(reg_number.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

```

**OUTPUT in the client.py terminal :**

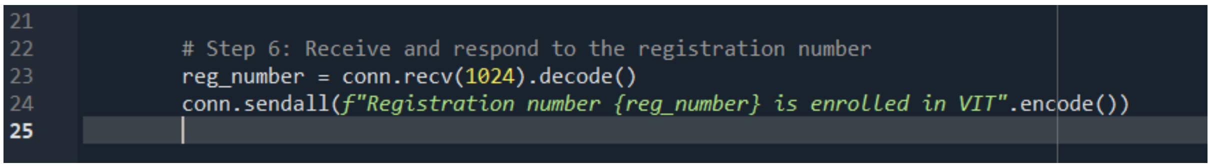


```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
```

**6. The server must reply with "Registration number is enrolled in VIT".**

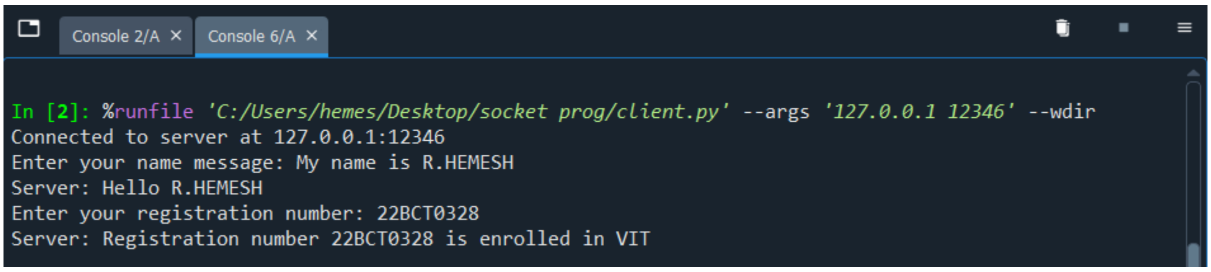
**server.py:**

```
# Step 6: Receive and respond to the registration number
reg_number = conn.recv(1024).decode()
conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())
```



```

    # Step 6: Receive and respond to the registration number
    reg_number = conn.recv(1024).decode()
    conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())

```

**OUTPUT in the client.py terminal :**

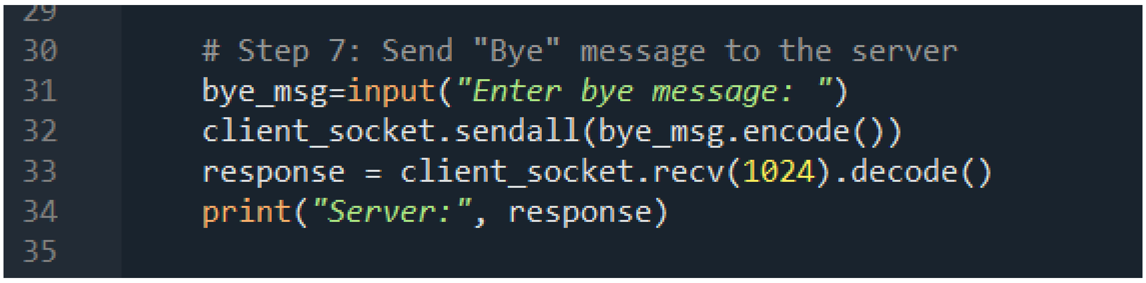


```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
```

**7. Finally, the user types “Bye” and sends to the server.**

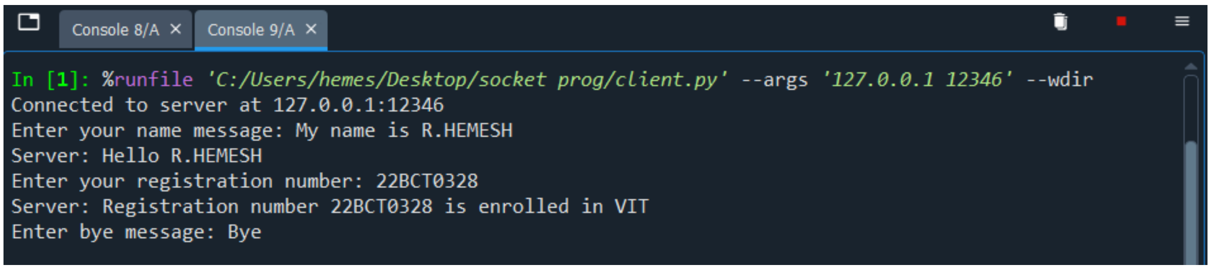
**client.py:**

```
    # Step 7: Send "Bye" message to the server
    bye_msg=input("Enter bye message: ")
    client_socket.sendall(bye_msg.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)
```



```
    # Step 7: Send "Bye" message to the server
    bye_msg=input("Enter bye message: ")
    client_socket.sendall(bye_msg.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)
```

**OUTPUT in the client.py terminal :**

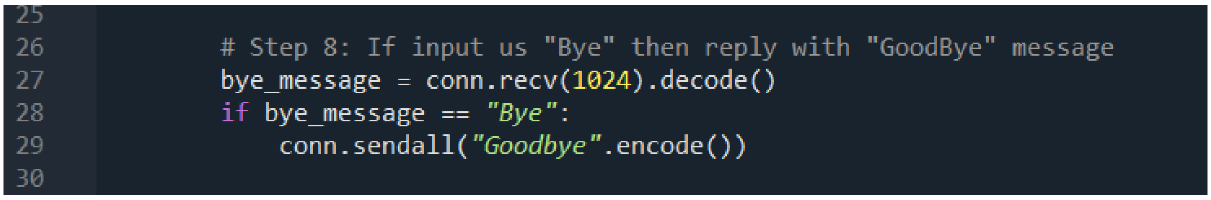


```
In [1]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
```

**8. If the user has typed "Bye", the server must reply with "Goodbye".**

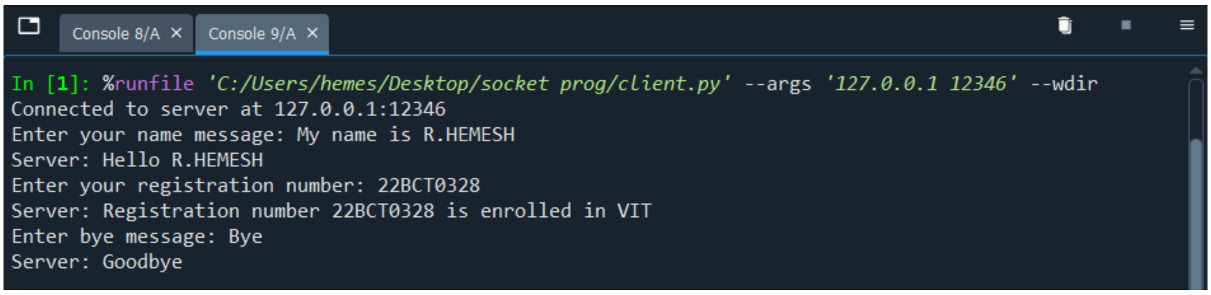
**server.py**

```
        # Step 8: If input us "Bye" then reply with "GoodBye" message
        bye_message = conn.recv(1024).decode()
        if bye_message == "Bye":
            conn.sendall("Goodbye".encode())
```



```
        # Step 8: If input us "Bye" then reply with "GoodBye" message
        bye_message = conn.recv(1024).decode()
        if bye_message == "Bye":
            conn.sendall("Goodbye".encode())
```

**OUTPUT in the client.py terminal :**



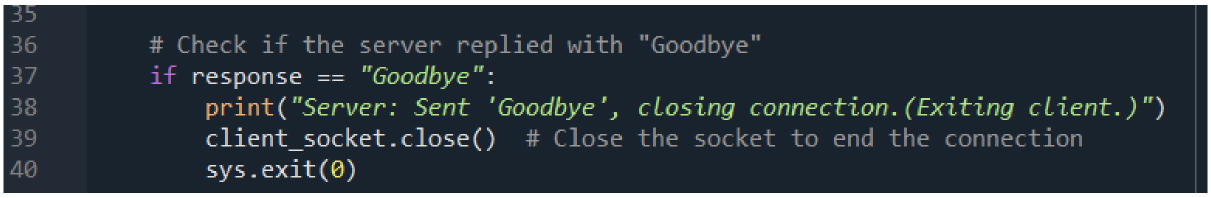
Console 8/A ✕ Console 9/A ✕

```
In [1]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
```

**9. If the server replied with a "Goodbye", the client quits.**

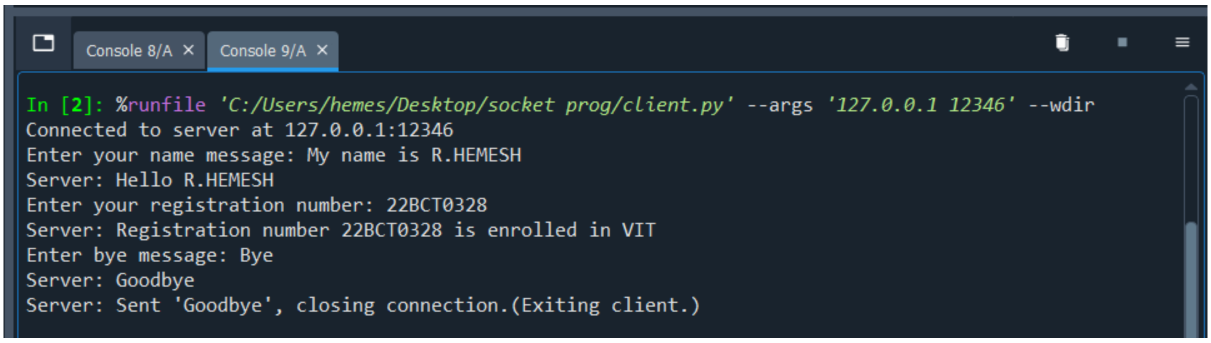
**client.py:**

```
    # Check if the server replied with "Goodbye"
    if response == "Goodbye":
        print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
        client_socket.close()  # Close the socket to end the connection
        sys.exit(0)
```



```

    # Check if the server replied with "Goodbye"
    if response == "Goodbye":
        print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
        client_socket.close()  # Close the socket to end the connection
        sys.exit(0)
```

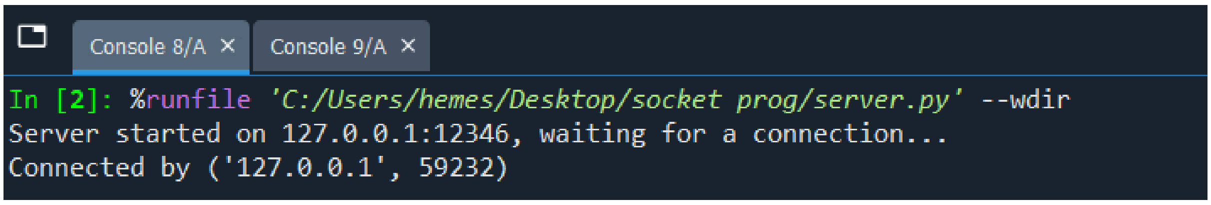
**OUTPUT in the client.py terminal :**



Console 8/A ✕ Console 9/A ✕

```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
Server: Sent 'Goodbye', closing connection.(Exiting client.)
```

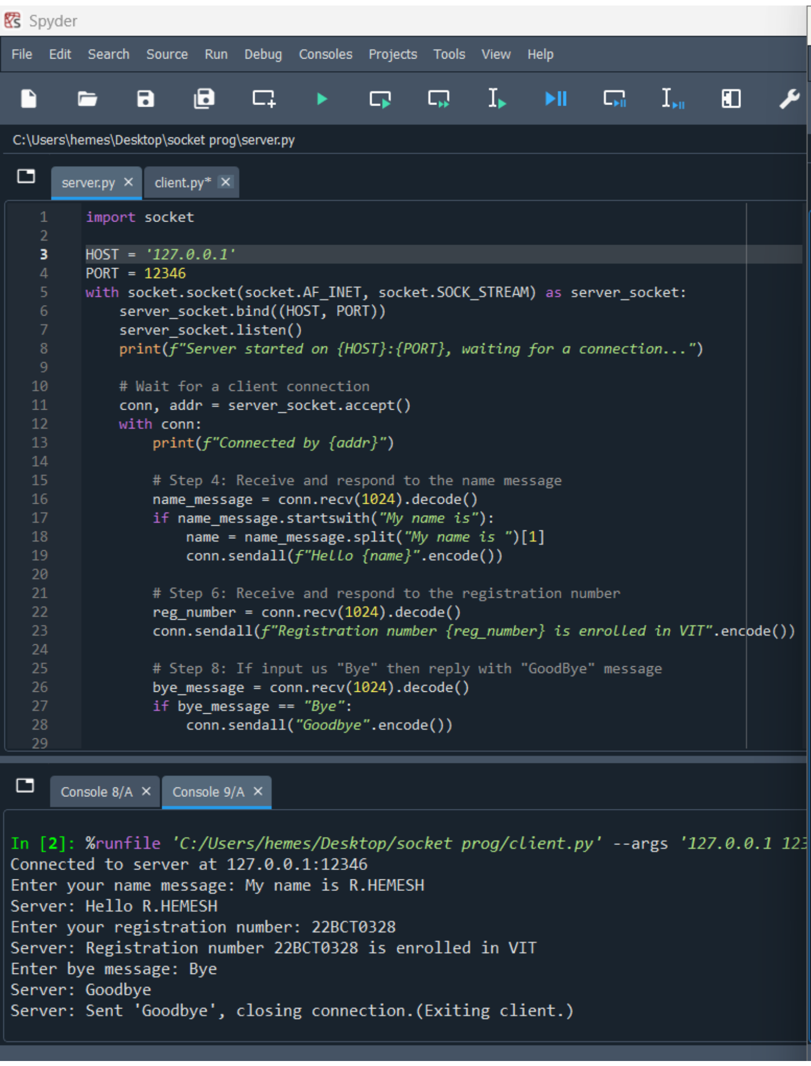
**OUTPUT in the server.py terminal:**



Console 8/A ✕ Console 9/A ✕

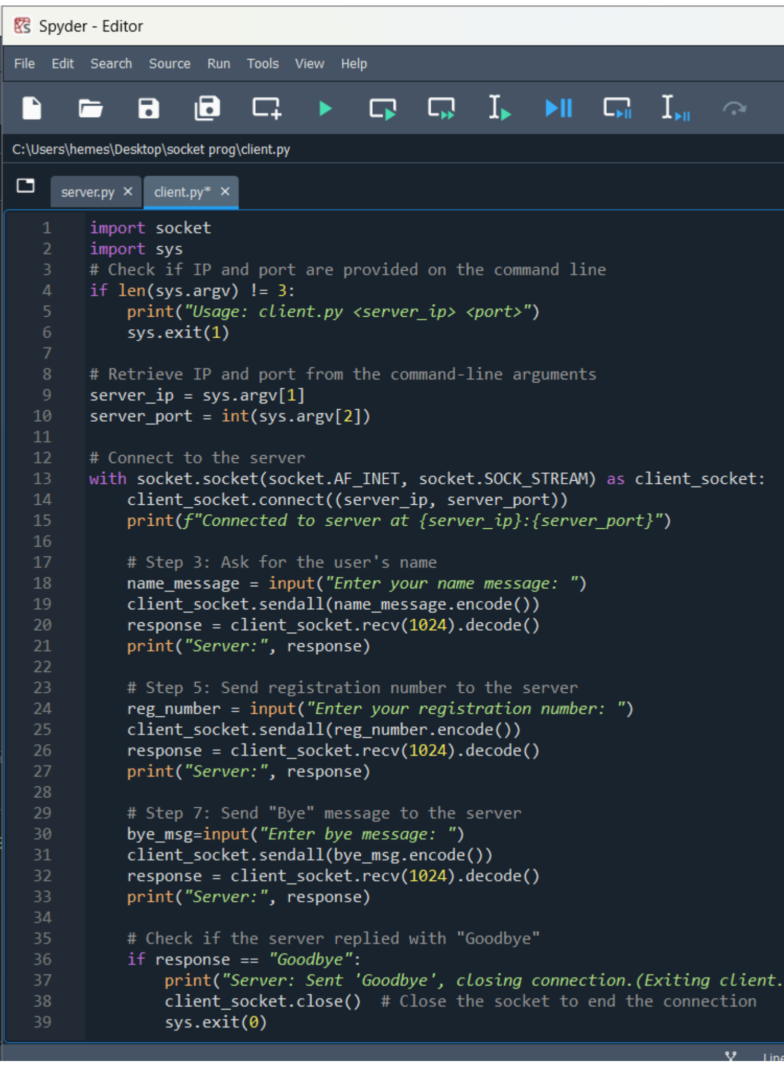
```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
Connected by ('127.0.0.1', 59232)
```

**FULL CODE FOR ALL 9 STEPS (server.py, client.py, output) :**


Spyder
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hemes\Desktop\socket prog\server.py
server.py
client.py*
Console 8/A
Console 9/A

```python
import socket

HOST = '127.0.0.1'
PORT = 12346
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"Server started on {HOST}:{PORT}, waiting for a connection...")

    # Wait for a client connection
    conn, addr = server_socket.accept()
    with conn:
        print(f"Connected by {addr}")

        # Step 4: Receive and respond to the name message
        name_message = conn.recv(1024).decode()
        if name_message.startswith("My name is"):
            name = name_message.split("My name is ")[1]
            conn.sendall(f"Hello {name}".encode())

        # Step 6: Receive and respond to the registration number
        reg_number = conn.recv(1024).decode()
        conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())

        # Step 8: If input us "Bye" then reply with "GoodBye" message
        bye_message = conn.recv(1024).decode()
        if bye_message == "Bye":
            conn.sendall("Goodbye".encode())

```

```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 123
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
Server: Sent 'Goodbye', closing connection.(Exiting client.)
```


Spyder - Editor
File Edit Search Source Run Tools View Help
C:\Users\hemes\Desktop\socket prog\client.py
server.py
client.py*

```python
import socket
import sys
# Check if IP and port are provided on the command line
if len(sys.argv) != 3:
    print("Usage: client.py <server_ip> <port>")
    sys.exit(1)

# Retrieve IP and port from the command-line arguments
server_ip = sys.argv[1]
server_port = int(sys.argv[2])

# Connect to the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
    client_socket.connect((server_ip, server_port))
    print(f"Connected to server at {server_ip}:{server_port}")

    # Step 3: Ask for the user's name
    name_message = input("Enter your name message: ")
    client_socket.sendall(name_message.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

    # Step 5: Send registration number to the server
    reg_number = input("Enter your registration number: ")
    client_socket.sendall(reg_number.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

    # Step 7: Send "Bye" message to the server
    bye_msg=input("Enter bye message: ")
    client_socket.sendall(bye_msg.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

    # Check if the server replied with "Goodbye"
    if response == "Goodbye":
        print("Server: Sent 'Goodbye', closing connection.(Exiting client.
        client_socket.close()  # Close the socket to end the connection
        sys.exit(0)
```

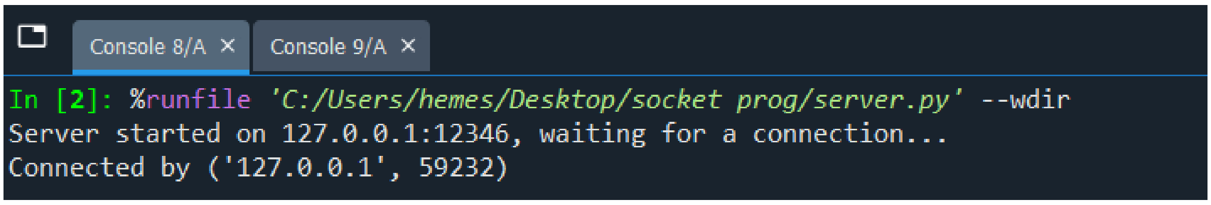
**Server.py**

```python
import socket

HOST = '127.0.0.1'
PORT = 12346
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"Server started on {HOST}:{PORT}, waiting for a connection...")

    # Wait for a client connection
    conn, addr = server_socket.accept()
    with conn:
        print(f"Connected by {addr}")

        # Step 4: Receive and respond to the name message
        name_message = conn.recv(1024).decode()
        if name_message.startswith("My name is"):
            name = name_message.split("My name is ")[1]
            conn.sendall(f"Hello {name}".encode())

        # Step 6: Receive and respond to the registration number
        reg_number = conn.recv(1024).decode()
        conn.sendall(f"Registration number {reg_number} is enrolled in 
VIT".encode())

        # Step 8: If input us "Bye" then reply with "GoodBye" message
        bye_message = conn.recv(1024).decode()
        if bye_message == "Bye":
            conn.sendall("Goodbye".encode())
```

**client.py**

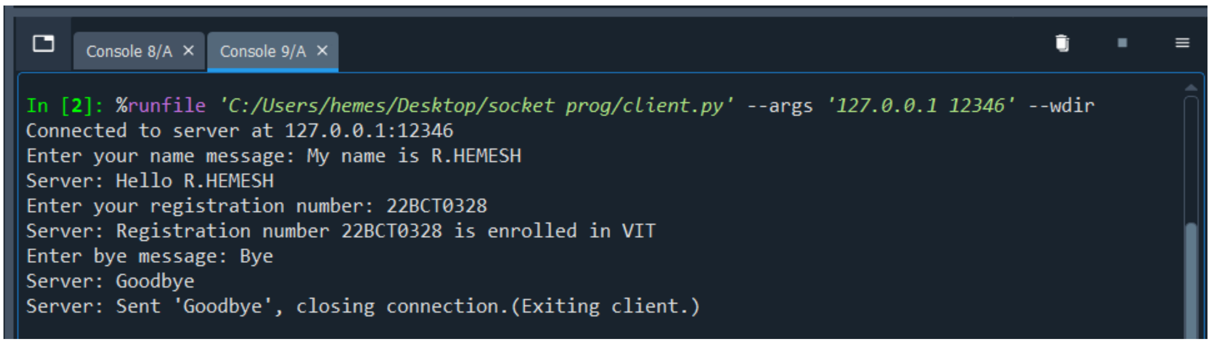
```python
import socket
import sys
# Check if IP and port are provided on the command line
if len(sys.argv) != 3:
    print("Usage: client.py <server_ip> <port>")
    sys.exit(1)

# Retrieve IP and port from the command-line arguments
server_ip = sys.argv[1]
server_port = int(sys.argv[2])
# Connect to the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
    client_socket.connect((server_ip, server_port))
    print(f"Connected to server at {server_ip}:{server_port}")

    # Step 3: Ask for the user's name
    name_message = input("Enter your name message: ")
    client_socket.sendall(name_message.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

    # Step 5: Send registration number to the server
    reg_number = input("Enter your registration number: ")
    client_socket.sendall(reg_number.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

    # Step 7: Send "Bye" message to the server
    bye_msg=input("Enter bye message: ")
    client_socket.sendall(bye_msg.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)
    # Check if the server replied with "Goodbye"
    if response == "Goodbye":
        print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
        client_socket.close()  # Close the socket to end the connection
        sys.exit(0)
```

**OUTPUT in the server.py terminal :**



```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
Connected by ('127.0.0.1', 59232)
```

**OUTPUT in the client.py terminal:**



```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
Server: Sent 'Goodbye', closing connection.(Exiting client.)
```