EXPERIMENT-9

*K.CHARAN: **2023000608***

AIM:-

Write a program to create a server that listens to port 5005 using stream sockets. Write a simple client program to connect to the server. The client should request for a text file and the server should return the file before terminating the connection. The client should display the file contents at the terminal. If the server does not have the file, it should return a message "Sorry, file not found." and then close the connection. Now, modify the client program so that a new file is opened at the client and the received content is saved to it.

SERVER CODE:-

```
import socket import os

# Create a server socket server_socket = socket.socket(socket.AF_INET,

socket.SOCK_STREAM)

# Bind the socket to localhost and port 5005 server_socket.bind(('localhost', 5005))

# Start listening for incoming connections (backlog of 1)

server_socket.listen(1) print("Server is listening on port 5005...")

# Accept a connection from the client conn, addr =

server_socket.accept() print(f"Connection established with

{addr}")

# Receive the filename request from the client filename =

conn.recv(1024).decode()

# Check if the file exists if

os.path.isfile(filename):     # Send the file
```

content to the client   with

open(filename, 'rb') as file: file_data =

file.read(1024)   while file_data:

            conn.send(file_data)

file_data = file.read(1024) else:

    # Send a message saying the file was not found   conn.send(b"Sorry,

file not found.")

# Close the connection conn.close()

server_socket.close()

CLIENT CODE:-

import socket # Create a client socket client_socket =

socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect to the server at localhost and port 5005 client_socket.connect(('localhost', 5005)) # Request a

file from the server filename = input("Enter the filename you want to request: ")

client_socket.send(filename.encode()) # Receive the file content or error message file_content =

client_socket.recv(1024) received_data = b"" while file_content:

    received_data += file_content   file_content =

client_socket.recv(1024)

# Check if the file was found or if there was an error message if

received_data == b"Sorry, file not found.":

    print(received_data.decode()) else:

# Display the received file contents on the terminal

print("\nFile contents received:")

print(received_data.decode())   # Save the received content

into a new file   with open(f"received_{filename}", 'wb') as

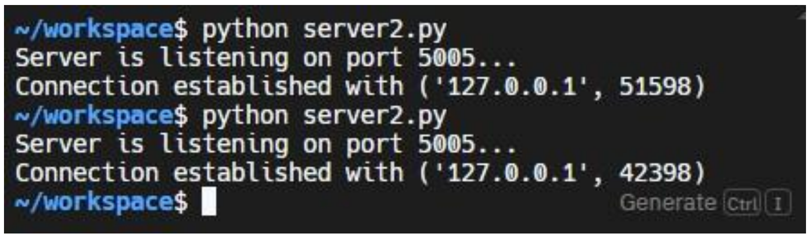
file:

    file.write(received_data) #

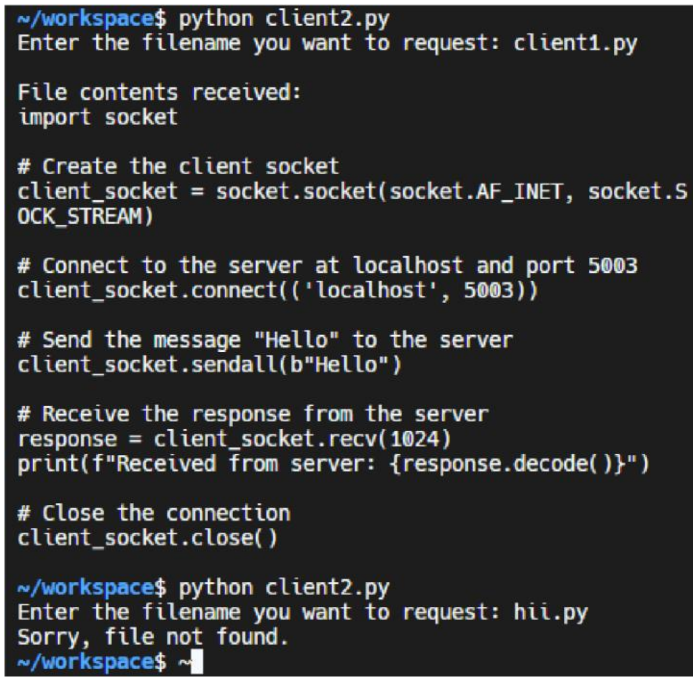
Close the connection

client_socket.close()

OUTPUT:-



```
~/workspace$ python server2.py
Server is listening on port 5005...
Connection established with ('127.0.0.1', 51598)
~/workspace$ python server2.py
Server is listening on port 5005...
Connection established with ('127.0.0.1', 42398)
~/workspace$
```



```
~/workspace$ python client2.py
Enter the filename you want to request: client1.py

File contents received:
import socket

# Create the client socket
client_socket = socket.socket(socket.AF_INET, socket.S
OCK_STREAM)

# Connect to the server at localhost and port 5003
client_socket.connect(('localhost', 5003))

# Send the message "Hello" to the server
client_socket.sendall(b"Hello")

# Receive the response from the server
response = client_socket.recv(1024)
print(f"Received from server: {response.decode()}")

# Close the connection
client_socket.close()

~/workspace$ python client2.py
Enter the filename you want to request: hii.py
Sorry, file not found.
~/workspace$ ~
```

The server waits for a connection on **port 5005**. The client connects and requests a file. If the file exists, the server sends it in small parts. If not, it sends a "file not found" message.

The client receives the data, saves the file, or shows an error message.

CONCLUSION:-

The program successfully transfers files between a server and a client using **TCP sockets**. It works well but can be improved by allowing multiple clients and handling errors better.