1. Using TCP/IP sockets, write a client-server program to make the client send the file name and the server to send back the contents of the requested file if present.

**tcp\_server.py**

from socket import \*

# Server configuration

serverPort = 12000

serverSocket = socket(AF\_INET, SOCK\_STREAM)

# Bind the server to the specified IP and port

serverSocket.bind(("127.0.0.1", serverPort))

serverSocket.listen(1) # Server listens for incoming connections

print("The server is ready to receive")

while True:

# Accept a connection from the client

connectionSocket, clientAddress = serverSocket.accept()

print(f"Connection established with {clientAddress}")

# Receive the filename from the client

filename = connectionSocket.recv(1024).decode()

print(f"Requested file: {filename}")

try:

# Open the file and read its contents

with open(filename, "r") as file:

fileContents = file.read()

# Send the file contents to the client

connectionSocket.send(fileContents.encode())

print(f"Sent file contents to {clientAddress}")

except FileNotFoundError:

# Handle case when the file is not found

errorMessage = "File not found."

connectionSocket.send(errorMessage.encode())

print(f"File '{filename}' not found. Sent error message to client.")

# Close the connection with the client

connectionSocket.close()

**tcp\_client.py**

from socket import \*

# Client configuration

serverName = "127.0.0.1"

serverPort = 12000

# Create a TCP socket and connect to the server

clientSocket = socket(AF\_INET, SOCK\_STREAM)

clientSocket.connect((serverName, serverPort))

# Input the filename from the user

filename = input("Enter file name: ")

# Send the filename to the server

clientSocket.send(filename.encode())

# Receive the file contents from the server

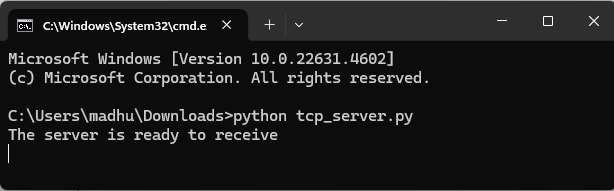
fileContents = clientSocket.recv(1024).decode()

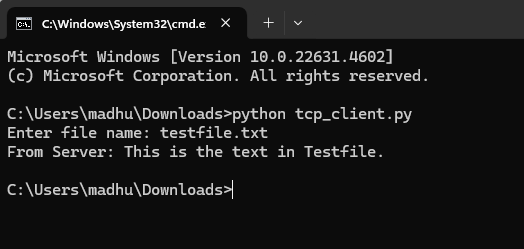
print("From Server:", fileContents)

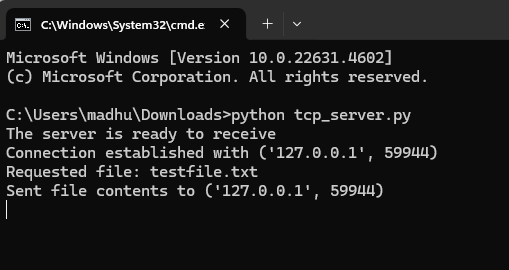
# Close the socket

clientSocket.close()

**OUTPUT:**

****

****

****

1. Using UDP sockets, write a client-server program to make the client send the file name and the server to send back the contents of the requested file if present.

**udp\_server.py**

from socket import \*

# Server configuration

serverPort = 12000

serverSocket = socket(AF\_INET, SOCK\_DGRAM)

# Bind the server to the specified IP address and port

serverSocket.bind(("127.0.0.1", serverPort))

print("The server is ready to receive")

while True:

# Receive the filename from the client

sentence, clientAddress = serverSocket.recvfrom(2048)

filename = sentence.decode("utf-8") # Decode the filename

print(f"Requested file: {filename}")

try:

# Open the file and read its contents

with open(filename, "r") as file:

fileContents = file.read(2048)

# Send the file contents back to the client

serverSocket.sendto(fileContents.encode("utf-8"), clientAddress)

print(f"Sent back to client: {fileContents}")

except FileNotFoundError:

# Handle case when the file is not found

errorMessage = "File not found."

serverSocket.sendto(errorMessage.encode("utf-8"), clientAddress)

print(f"File '{filename}' not found. Sent error message to client.")

except Exception as e:

# Handle other exceptions

errorMessage = f"An error occurred: {str(e)}"

serverSocket.sendto(errorMessage.encode("utf-8"), clientAddress)

print(f"An unexpected error occurred: {str(e)}")

**udp\_client.py**

from socket import \*

# Client configuration

serverName = "127.0.0.1"

serverPort = 12000

# Create a UDP socket

clientSocket = socket(AF\_INET, SOCK\_DGRAM)

# Input the filename from the user

filename = input("Enter file name: ")

# Send the filename to the server

clientSocket.sendto(filename.encode("utf-8"), (serverName, serverPort))

# Receive the response from the server

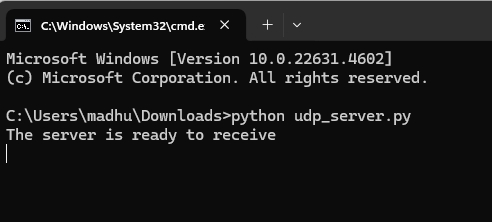
fileContents, serverAddress = clientSocket.recvfrom(2048)

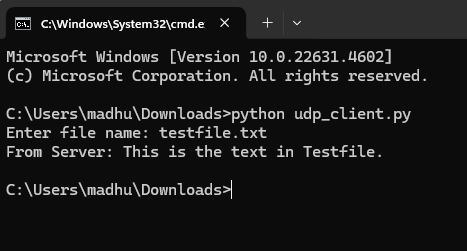
print("From Server:", fileContents.decode("utf-8"))

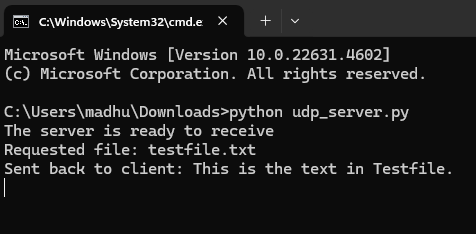
# Close the socket

clientSocket.close()

**OUTPUT:**

****

****

****