Socket-programming

Socket programming in C and Python. Computer Networks Assignment-1.

Question-1

There are 2 files `client.c` and `server.c` . The type of connection used in this question is non-perisitant i.e every time client downloads some file the connection is closed and established again on the request. The following error handling have been taken care of:

- Socket creation Error
- Connection Error
- File opening Error
- If file is avaiable on server or not

To run the code

```
cd Server
gcc server.c
./a.out
# Open new terminal
cd Client
gcc client.c
./a.out test.md 1.pdf 2.pdf 3.pdf 4.pdf 5.pdf
```

On running the code the client side will ask you to Enter the filename, the server will try to search your file in `Server/Data` folder. If the file is found in the path it wil be downloaded, otherwise it will display an error message, showing that such file doesn't exist on server.

Question-2

PART 1: PERSISTANT CONNECTION

Connection between client and server is established only once, that is connection is maintained between different file transfers. The connection is closed when all files are tranferred

- Multiple objects can be sent over single TCP connection between client, server
- Server leaves connection open after sending response, subsequent HTTP messages between same client/server sent over open connection
- As little as one RTT is needed for all the referenced objects

Server.py

```
conn, addr = s.accept()
while True: #breaks when all the files are delivered
        # filename is received from client
        filename = conn.recv(1024)
        # file is transfered and closed
        f.close()
# After the files requested by client are send the connection is closed.
print('Connection Closed')
conn.close()
```

Client.py

```
# socket is made only once
s = socket.socket()
#Connection is made only once
s.connect((host, port))

while True:
    # Iterate over the list of files given by the client
    fname = list_files[i]
    s.send(fname)
    # receives data from server writes in the file
# After all the files have been transferred , we close the socket
s.close()
print('connection closed')
```

To run the code

```
cd Server
python server_persistant.py
# Open another terminal for client
cd Client
# python client_persistant.py <file-names>
python client_persistant.py test.md 1.pdf 2.pdf 3.pdf 4.pdf 5.pdf
# To check time
time python client_persistant.py test.md 1.pdf 2.pdf 3.pdf 4.pdf 5.pdf
#0.03sec
```

PART 2: NON-PERSISTANT CONNECTION

Connection between client and server is established every time client needs to transfer a new file. The connection is closed when a file is transferred and then is established again if client sends another request.

- At most one object is send over TCP connection, connection is then closed
- Downloading mutiple objects require mutiple connections
- Requires 2 RTTs per object , For each object Non-persistent HTTP response time = 2RTT+ file transmission time

Server.py

```
# Loop breaks when all files are transferred, with 1 file being trnsferred in every
iteration
while True:
        # Establish from client
        conn, addr = s.accept()
        # filename is received from client
        filename = conn.recv(1024)
        # file is transfered and closed
        f.close()
        # After the requested files is sent by server, the connection is closed.
        print('Connection Closed')
        conn.close()
```

Client.py

```
# Loop breaks when all files are transferred, with 1 file being trnsferred in every
iteration
while True:
    # Making new socket for every file transfer
    s = socket.socket()
    # Tries to connect to the server after every file transfer
    s.connect((host, port))
    fname = list_files[i]
    s.send(fname)
    # receives data from server writes in the file and closes the connection
    s.close()
    print('Connection Closed\n')
```

To run the code

```
cd Server
python server_nonpersistant.py
# Open another terminal for client
cd Client
# python client_nonpersistant.py <file-names>
python client_nonpersistant.py test.md 1.pdf 2.pdf 3.pdf 4.pdf 5.pdf
# To check time
time python client_nonpersistant.py test.md 1.pdf 2.pdf 3.pdf 4.pdf 5.pdf
#0.02sec
```

Ideally time taken by **Non Persistant** should be more than time taken by **Persistant** connection as the server and client as not on the same machine, so there is large overhead for each TCP connection.

Time taken by **Persistant** connection is more than time taken by **Non Persistant** connection. This is because client and server are on the same machine, so there is no overhead in making new socket every time, we have to send a file. But this means that persistant and non-persistant should have almost same time. But the time in persistant is much more than non-persistant due to **Nagle algorithm**.

Nagle's algorithm ensures that small packets are not sent. The data is accumulated on the sender's buffer and is sent only when the data is sufficiently big. This is to ensure that the headers for the TCP packet etc are not bigger than the data itself. **This means in this context is that when you have files on a persistent connection, they do not get sent immediately. Whereas in a non-persistent connection, closing the connection would ensure that the buffer would get flushed.**