



# Computer Networks Lab 4

CS F303

Susmit Wani

2018A7PS0116G

## Lab 4 : Named Pipes following FIFO

Dasgaonkar Yogesh Namdeo • Feb 13

Labs • 32 points

Due Tomorrow, 7:00 PM

Lab 4 evaluation: Named Pipes following FIFO

Total 32 marks


Q 1. Write a C program for client-server text messaging app using named PIPE(FIFO).

Steps of the app:

- 1 - The text messages sent by the client to the server or vice-versa will be written to the named pipes.
- 2 - Either side reading the messages will read it from the names pipe.
- 3 - You have to write two programs separately for a client and a server. You will run them in different terminals, concurrently, to show the output.
- 4- There is no constraint on the message size. You are free to use any number of pipes.
- 5- When the user types 'exit' in the client or the server terminal, the program must exit without blocking the other end. All pipes must be closed and memory freed before you close.

Files to Submit:

- 1) The client and the server .c files (Correct execution of each of the following carry 4 marks each, amounting to 24 marks)
  - The text message typed from the client terminal to be sent to the server
  - The display of client's sent messages on the server's terminal
  - The same two screenshots above must be repeated for the opposite direction message example, i.e, server to client message transfer and termination
  - Terminate the client program by typing 'exit' at the client terminal
  - Terminate the server by typing 'exit' at the server terminal
- 2) Screenshots of output showing: (Total 6 screenshots X 1 mark = 6 marks) in a PDF
- 3) A README instruction file to compile and run your code and produce the output. (Total - 2 marks)

- 
1. The text message typed from the client terminal to be sent to the server


```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o c client.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./c
Client: Hello
```

```
█
```

2. The display of client's sent messages on the server's terminal

```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o s server.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./s
Client: Hello
```

```
Server: □
```

- 
3. The text message typed from the server terminal to be sent to the client

```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o s server.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./s
Client: Hello

Server: Hey!


█
```

4. The display of server's sent messages on the client's terminal

```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o c client.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./c
Client: Hello

Server: Hey!

Client: █
```



## 5. Terminate the client program by typing 'exit' at the client terminal

Client Side Terminal:

```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o c client.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./c
Client: Hello

Server: Hey!

Client: exit
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ █
```

Server Side Terminal:


```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o s server.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./s
Client: Hello

Server: Hey!

Client has gone offline

Server: hello?
Message not delivered. Client is offline

Server: █
```



## 6. Terminate the server by typing 'exit' at the server terminal

Server Side Terminal:

```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o s server.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./s
Client: Hello

Server: exit
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ █
```

Client Side Terminal:

```
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ gcc -o c client.c
lenovo@susmits-lenovo:~/Desktop/CN Lab 4$ ./c
Client: Hello

Server has gone offline

Client: why
Message not delivered. Client is offline

Client: okay
Message not delivered. Client is offline

Client: █
```