Computer Network
Project 2
Simple Multi-user TCP Chat
(Difficulty ★☆☆☆☆)

CSI4106-01

Fall, 2020

Prelim.

Before you do this homework, you must be fully aware of "**Project Policy Notice**"

Goal (you are expected to)

1. Learn a basic socket programming
   - TCP only
   - Sockets in Linux Environment
     - **Do not write a code in Windows.**
     - **The two OSs have different socket APIs.**
     - **That means, they are not compatible.**
2. Understand the “Client-Server” architecture
3. **Write a simple TCP-based multi-user chat application in multi-thread.**

Steps to do this project

1. Google or read your textbook to follow up **the key implement** we provide.
   - Check the slide #7.
2. Understand what **Multi-Thread** does exactly
   - (hint) Think about the difference between single thread program and multi thread program.
3. **Write your own codes for multi-user chatting application running in real-time.**

Objectives

- You must write your own code **`cli.[py/c]`**
- You must write your own code **`srv.[py/c]`**

- The two codes must take the two parameters
  - IP address and Port number
  - e.g.) `python srv.py 0.0.0.0 5000`
  - e.g.) `python cli.py 127.0.0.1 5000`

What are these functions?

- **socket**(socket.AF_INET, socket.SOCK_STREAM)
- **setsockopt**(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
- **bind**(("0.0.0.0", 7777))
- **listen**(5)
- **connect**((host, port))
- **accept**()
- **close**()
- **Thread**()

In case of python

Key implement

- bind, listen, connect, accept, close
- Send msg from the socket
- Receive msg from the socket
- Use threads for sending and receiving msg respectively in client
- Use threads for meeting requirements of each client in server

Guidelines – Server

```
[root@localhost p1]# python3 srv.py 127.0.0.1 8888
Chat Server started on port 8888.
> New user 127.0.0.1:31335 entered (1 user online)
[127.0.0.1:31335] Hello
> New user 127.0.0.1:31341 entered (2 users online)
[127.0.0.1:31341] World
< The user 127.0.0.1:31341 left (1 user online)
< The user 127.0.0.1:31335 left (0 user online)
^C
exit
[root@localhost p1]#
```

Guidelines – Client 1

```
[root@localhost p1]# python3 cli.py 127.0.0.1 8888
> Connected to the chat server (1 user online)
[You] Hello
> New user 127.0.0.1:31341 entered (2 users online)
[127.0.0.1:31341] World
< The user 127.0.0.1:31341 left (1 user online)
^C
exit
[root@localhost p1]#
```

Guidelines – Client 2

```
[root@localhost p1]# python3 cli.py 127.0.0.1 8888
> Connected to the chat server (2 users online)
[You] World
^C
exit
[root@localhost p1]#
```

Your report must include

- **Introduction/Reference** (3pts)
  - Software environment, programming language you used, version, reference and so on.
- **Flow chart or Diagram** (10pts)
  - Must show the logic of your program
  - Focus on describing how your client and server work.
- **At least 3 snapshots** which prove your codes are working well. (5pts)

Your report must include (cont'd)

- **Logical explanations block by block in detail**. (20pts)
  - It is different with brief comments in your source code!!!
  - In your report, write what the blocks do and why you implemented those functions.
- **All explanations of the 8 functions** in "What are these functions?" slide. (5pts)
- **Difference between using `multi-thread` and the function `select()`?** (7pts)
  - Explanation + Pros and Cons of each

We will test your code as follows.

- OS: Ubuntu Linux
- Language: Python3 / C language
- Your codes can
  - Run with **custom IP and Port** (10pts)
  - Work perfectly during evaluation **with no error** (30pts)
  - Be **terminated** by Ctrl+C (5pts)
  - **Close the sockets** (We will check this by **`netstat`**) (5pts)

You will get 0 points if you…

- Copy your friend's codes
  + Change a little bit of them.
  + Wish that TAs don't catch that.

- Use a 3rd-party API or codes.

Deliverable

- **Only one zip file of "YourID_p2.zip"**
  - **If your ID is 2018147123, 2018147123_p2.zip should be your deliverable file name.**

- **In the zip file only the three files must be included without any folder**
  - **report.pdf**
  - **cli.py or cli.c**
  - **srv.py or srv.c**
    - **if you use C language, include compile.sh as well**
    - **Files with different format/name will be a deduction [2pts each]**

- **DUE DATE**

  31/Oct/2020 23:59:59 KST

  No exception for exceeding deadline

- **Delay Policy**

  -33%pts for ~1/Nov 23:59:59

  -66%pts for ~2/Nov 23:59:59

  -100%pts for 3/Nov 00:00:00~

Score Policy: *Max. 100pts*

1	Not submitted / not working / missing files	**0 pts**
2	Overdue ➔ Delay	**-33% pts/day**
3	**The rules or directions whose scores are not specified/ are not followed**	-10 pts/rule
4	**Any 3rd party framework is used**	**0 pts**
5	**Plagiarizing / Over-implementation (Any kinds of Suspicion of Code-copy)**	**0 pts**
6	**Impolite Report / Lack of Comments**	**0 pts / -50% pts**