

컴퓨터네트워크

HW #4 (30 Points)

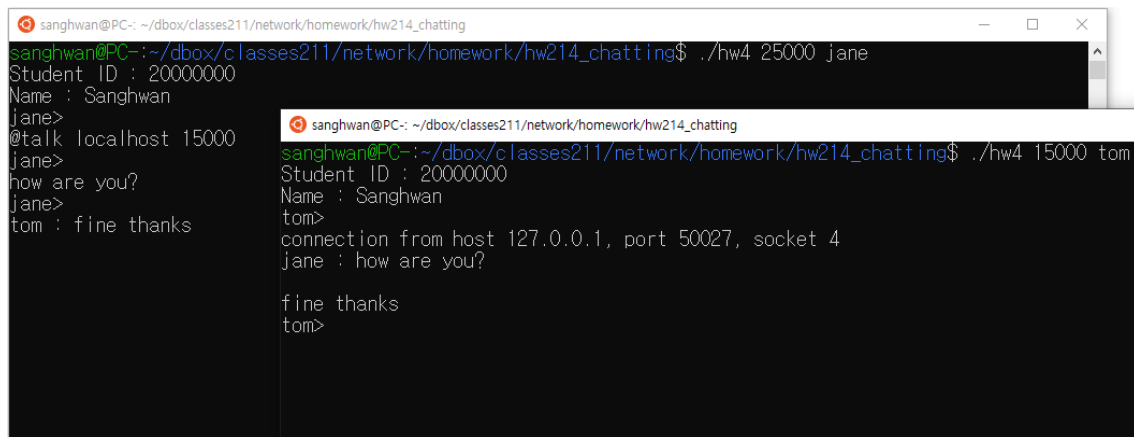
Due date : 2021/6/2 (eCampus)

제출물 : 학번.c, 학번.cpp, 학번.py (하나의 파일에 모든 것을 다 구현함)

2개 이상의 파일을 제출하는 경우 0점 처리될 수 있음.

이 과제는 간단한 2인 채팅 프로그램을 구현하는 것이다. Hw3에서 구현한 채팅 서버를 활용하면 3인 이상의 채팅도 가능하게 된다.

아래는 실행 예이다. 소스를 컴파일해서 hw4라는 실행파일을 만들어 냈다.



```
sanghwan@PC: ~/dbox/classes211/network/homework/hw214_chatting
sanghwan@PC:~/dbox/classes211/network/homework/hw214_chatting$ ./hw4 25000 jane
Student ID : 200000000
Name : Sanghwan
jane>
@talk localhost 15000
jane>
how are you?
jane>
tom : fine thanks

sanghwan@PC: ~/dbox/classes211/network/homework/hw214_chatting
sanghwan@PC:~/dbox/classes211/network/homework/hw214_chatting$ ./hw4 15000 tom
Student ID : 200000000
Name : Sanghwan
tom>
connection from host 127.0.0.1, port 50027, socket 4
jane : how are you?

fine thanks
tom>
```

이 응용은 사용자로부터 포트 번호와 아이디를 입력받아서 시작한다. 문법은 아래와 같다.

`$ hw4 tcpport userid`

- tcpport : 상대방으로부터 연결을 받는 포트 번호
- userid : 이 사용자의 아이디 (그냥 간단한 문자열이면 됨)

프로그램을 실행하면 학번과 이름을 출력하게 되어 있는데, 각자의 학번과 이름으로 변경하면 된다.

위 그림에서는 jane이 tom에게 아래의 명령을 이용하여 연결 요청을 한다.

@talk hostname tcpport

- hostname : 상대방 컴퓨터의 이름
- tcpport : 상대방 프로그램의 포트 번호

위 예제에서는 tom의 포트번호는 15000이고, jane의 포트 번호는 25000 이다. 따라서

jane은 “@talk localhost 15000” 이라고 연결 명령을 입력하였다.

Tom의 화면에서는 연결 요청이 들어오면 아래와 같이 client의 IP 주소와 port 번호를 출력한다. 맨 마지막은 이 연결로 만들어진 소켓 번호이다.

```
connection from host 127.0.0.1, port 1741, socket 4
```

연결이 설정된 다음에는 한 사용자가 메시지를 보내면 다른 사용자의 화면에 출력되어야 한다. 위에서는 jane이 하나의 메시지를 tom에게 보내고, tom이 연달아서 2개의 메시지를 보내는 예제를 보여준다. **즉 서로간에 순서에 상관없이 메시지를 보낼 수 있어야 한다.**

예를 들어 한 사용자가 여러 개의 메시지를 보낼 수도 있고, 둘이 동시에 보낼 수도 있어야 한다.

사용자는 이 채팅 프로그램을 끝내기 위해서 아래 명령을 사용한다.

@quit

한 사용자가 프로그램을 끝내면, 다른 사용자는 이 연결이 끊어진 것을 파악하고, 아래와 같은 메시지를 보여준다.

```
Connection Closed sd
```

- sd : 그 연결의 소켓 번호.

또한 중단한 사용자나 아니면 새로운 사용자가 다시 연결 요청하면 연결을 받아들일 수 있어야 한다. 아래는 그런 예를 보여준다.

```
sanghwan@PC: ~/dbox/classes211/network/homework/hw214_chatting
sanghwan@PC:~:/dbox/classes211/network/homework/hw214_chatting$ ./hw4 25000 jane
Student ID : 20000000
Name : Sanghwan
jane>
@talk localhost 15000
jane>
how are you?
jane>
tom : fine thanks

connection from host 192.81.208.51, port 61953, socket 5
GET / HTTP/1.0

GET / HTTP/1.0

GET / HTTP/1.0

Connection Closed 5
@quit
sanghwan@PC:~:/dbox/classes211/network/homework/hw214_chatting$ ./hw4 25000 jane
Student ID : 20000000
Name : Sanghwan
jane>
@talk localhost 15000
jane>
hi
jane>
tom : good
```

```
선택 sanghwan@PC: ~/dbox/classes211/network/homework/hw214_chatting
sanghwan@PC:~:/dbox/classes211/network/homework/hw214_chatting$ ./hw4 15000 tom
Student ID : 20000000
Name : Sanghwan
tom>
connection from host 127.0.0.1, port 50027, socket 4
jane : how are you?

fine thanks
tom>
Connection Closed 4
connection from host 127.0.0.1, port 65354, socket 4
jane : hi

good
tom>
```

참고

- hw4.c 에 기본적인 코드가 제공되어 있으니, 확장하면 된다..
- 이 숙제에서는 **select()** system call을 적절하게 활용하면 키보드와 소켓에서 들어오는 입력을 동시에 처리 가능하다.
- **serverorg.c**, **clientorg.c** 의 예제가 select() 시스템콜을 사용하기 때문에 이 파일들의 코드를 분석하면 많은 도움이 될 것이다.

추가. 파이썬 사용자를 위하여...

이 과제는 파이썬으로 구현하여도 된다. 파이썬 3.0 이상을 사용하도록.

필요시 nonblocking IO에 관한 내용을 공부할 것.

<https://medium.com/vaidik Kapoor/understanding-non-blocking-i-o-with-python-part-1-ec31a2e2db9b>

프로그램 테스트

```
$ bash labtest.sh hw4sol.c 15000 25000
```

아래와 같이 15가 나오면 성공임.

Result: 15 hw4sol.c

테스트는 1분 이상 걸릴 수 있으니 주의하기 바람.