1.클라이언트에서 서버(host,port)로 호출

```
// (서버쪽 주석을 살펴보면 감이 더 잘 올 것이다.)
Socket sock = new Socket(hostname, port);
```

5. 왜 OutputStream out = sock.getOutPutStream() 을 양쪽에서 해줘야 하나?

서버(192.168.XX)

클라이언트

ServerSocket accept

서비스 번호 port 33333

3.client에서

```
OutputStream out =
sock.getOutputStream ();
```

수신하기 위해 설정 및 Hello Network Programming 전송

```
// 실제 sock은 클라이언트 소켓을 의미하게
// 그래서 좀 더 가독성이 좋은 코드는 Sock
// (전화 왔을때 통화하기 슬라이드가 accep
Socket sock = servSock.accept();

// 접속한 클라이언트의 IP를 확인하는 코드
```

2.serversocket에서 accept 호출

4.accept후 "들어온 host, client connect" 출력

6.getOutPutStream인 'out'을 writer(date.문자열)에 호출

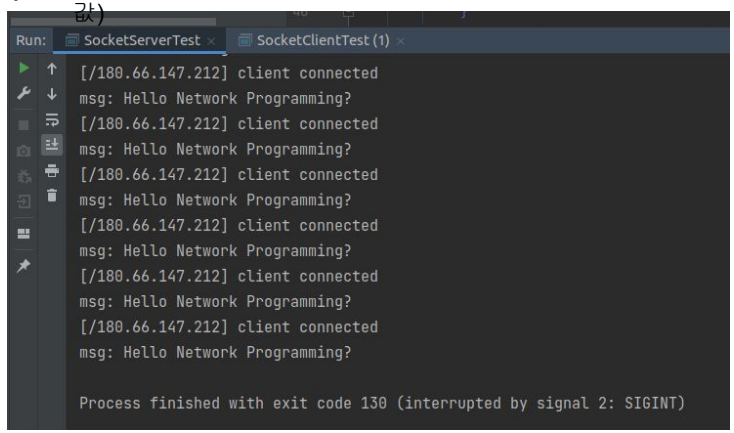
8.BufferedReader 생성자로 클라이언트로 부터 전송된 값: Hello Network Programming 'readline' 출력
(msg : readline)

7.서버에서 보낸 date값 수신 및 출력

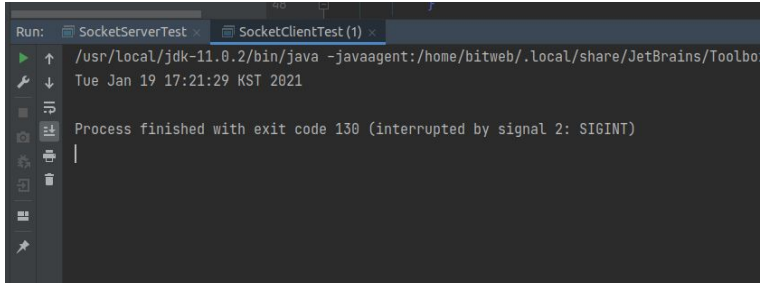
```
InputStream in = sock.getInputStream();
BufferedReader reader =
        new BufferedReader(new InputStreamReader(in));

// 서버가 보낸 내용을 time에 저장하고 출력한다.
String time = reader.readLine();
System.out.println(time);
```

inputstream-5번과 마찬가지로 양쪽 호출하는거랑 같은 원리일듯합니다.

IntelliJ에서 나타내는 server Run창 (클라이언트가 요청했던 값)



IntelliJ에서 나타내는 client Run창 (서버로부터 받은 시간값)



추가 질문
SocketClient에서 for문은 왜 도는지 궁금합니다

```
for(int i = 0; i < 10; i++) {
    try {
        // Socket 객체를 할당해서
        // 서버의 IP, 포트 번호를 가지고 접속을 요청합니다.
        // 서버에 대한 소켓을 획득하게 됩니다.
        // 이 요청이 들어갈때 서버의 accept()가 동작하게 됩니다.
        // 예를 들자면 이 행위는 전화를 거는것과 같다.
        // (서버쪽 주석을 살펴보면 감이 더 잘 올 것이다.)
        Socket sock = new Socket(hostname, port);
```