

[506489] System Programming (F'18)

Term Project (Week 3, Final) Report

팀 정보

No.	이름	학번
1	진재형	20155257
2	송호준	20125146
3	최용순	20167154
팀 이름: 마지막조		

<목차>

팀 정보	1
결과물 제출	2
보고서 작성 가이드	2
Part I: 최종 보고서	3
Part II: 데모/시연 시나리오 및 결과물	7
Part III: 프로젝트 진행중 발생한 문제점 및 해결방안	14
Part IV: 개선방향	15

* 보고서 작성이 완료되면 <목차>를 업데이트 해 주세요. (방법: “목차” 클릭>”목차 업데이트”>”목차 전체 업데이트”)

결과물 제출

- SmartCampus 에 업로드 해야 하는 제출물 목록은 다음과 같습니다.
 - Report(프로젝트 보고서), PDF 형식
 - 프로젝트 발표자료 (ppt, pptx, pdf 등)
 - GitHub 프로젝트를 다운로드 한 zip 파일
- 제출 기한을 넘기면 자동으로 0 점 처리됩니다.
- 팀 과제이며, 팀별로 한 명만 제출하면 됩니다.

* 보고서 및 발표자료도 GitHub 에서 관리해야 합니다!!!

보고서 작성 가이드

- Week 3 시작: 2018. 12. 12. (Wed) 1:00pm
- Week 3 마감: 2018. 12. 17. (Mon) 12:59pm
- 매 주차별로 반드시 1 회 이상 GitHub 에 commit 한 내역이 있어야 합니다. (팀별로 1 회 이상, 개인별 1 회 아님). GitHub 에 commit 한 내역이 없을 경우, 큰 감점을 받게 됩니다.

본 보고서는 총 3 개의 파트로 구성되어 있습니다.

- Part I: 최종 보고서
 - 1 주차 보고서 내용 전체를 포함하여 작성하고, 그 동안 추가/수정된 내용이 있다면 그에 맞게 보고서를 수정해야 합니다.
- Part II: 데모/시연 결과물
 - 개발한 프로그램이 정상적으로 구동된다는 것을 보여주세요.
 - 데모/시연을 위해 사용한 시나리오를 자세하게 설명하고, 해당 시나리오에 대한 단계별 데모 결과물(예: 스크린샷)을 첨부하거나 또는 데모 영상을 촬영해서 첨부파일로 제출해도 됩니다.
- Part III: 프로젝트 진행중 발생한 문제점 및 해결방안
 - 2 주차 보고서 내용중 “프로젝트 진행 중 발생하는 문제점 및 해결방안”에 대한 내용을 포함하는 내용으로 작성하세요.
 - 그 외에 추가로 발생한 문제점이 있었다면 해당 문제점 및 해결방안을 기술하세요.
- Part IV: 개선방향
 - 개발 결과물을 앞으로 어떻게 개선할 수 있을지에 대한 의견을 서술하세요.

Part I: 최종 보고서

[Q 0][GitHub URL] 팀 프로젝트를 위해 사용하는 GitHub 주소는? [10pts]

<https://github.com/wtty37/System-Programing-work.git>

[Q 1][프로젝트 제목] 프로젝트 제목은 무엇인가요? [10pts]

파일 공유 시스템

[Q 2][프로젝트 개요]

- 프로젝트 선정 동기? [10pts]
- 프로젝트 내용? (프로젝트 내용에 대한 간단한 소개) [10pts]
- 기대효과 및 활용방안? // 답변 생략. [Q 5] 에서 답하세요

요즘 파일 공유 프로그램을 많이 쓰는 추세인데, 추세에 맞게 평소 많이 쓰는 파일 공유 프로그램의 원리도 알아가면서 이것을 더욱 유용하게 사용하기 위해 어떤 방향으로 발전해나가야되는지 한번쯤 생각해보면서 직접 구현하는 데에서 의의가 있다고 판단하였다.

클라이언트는 자신만의 파일 공유 프로그램에 자신이 소중한게 또는 중요시 여기는 데이터를 보관할 수 있어 만약 컴퓨터나 스마트폰 오류로 데이터가 날라가도 파일 공유 시스템에 자신이 저장했던 정보들이 있어 안정성이 확보되고 또한 파일 공유 시스템에 파일들을 저장함으로 저장 공간 확보가 유용하다.

이처럼 21 세기에 파일공유시스템을 사용하면 많은 이익을 누릴 수 있다.

[Q 3][프로젝트 개발목표 및 개발/구현 내용]

- 프로젝트 개발목표 [10pts]
- 프로젝트 개발/구현 내용 [10pts]

목표는 개인 사용자 전용 파일 공유 시스템 구축이다.

개인이 중요하다고 생각하는 데이터를 자신만의 공간에 따로 보관함으로 데이터 안정성도 높힐 수 있고 그만큼 저장 공간 확보도 유용하다.

서버는 메인, 클라이언트는 고객. 즉, 개인 사용자이다.

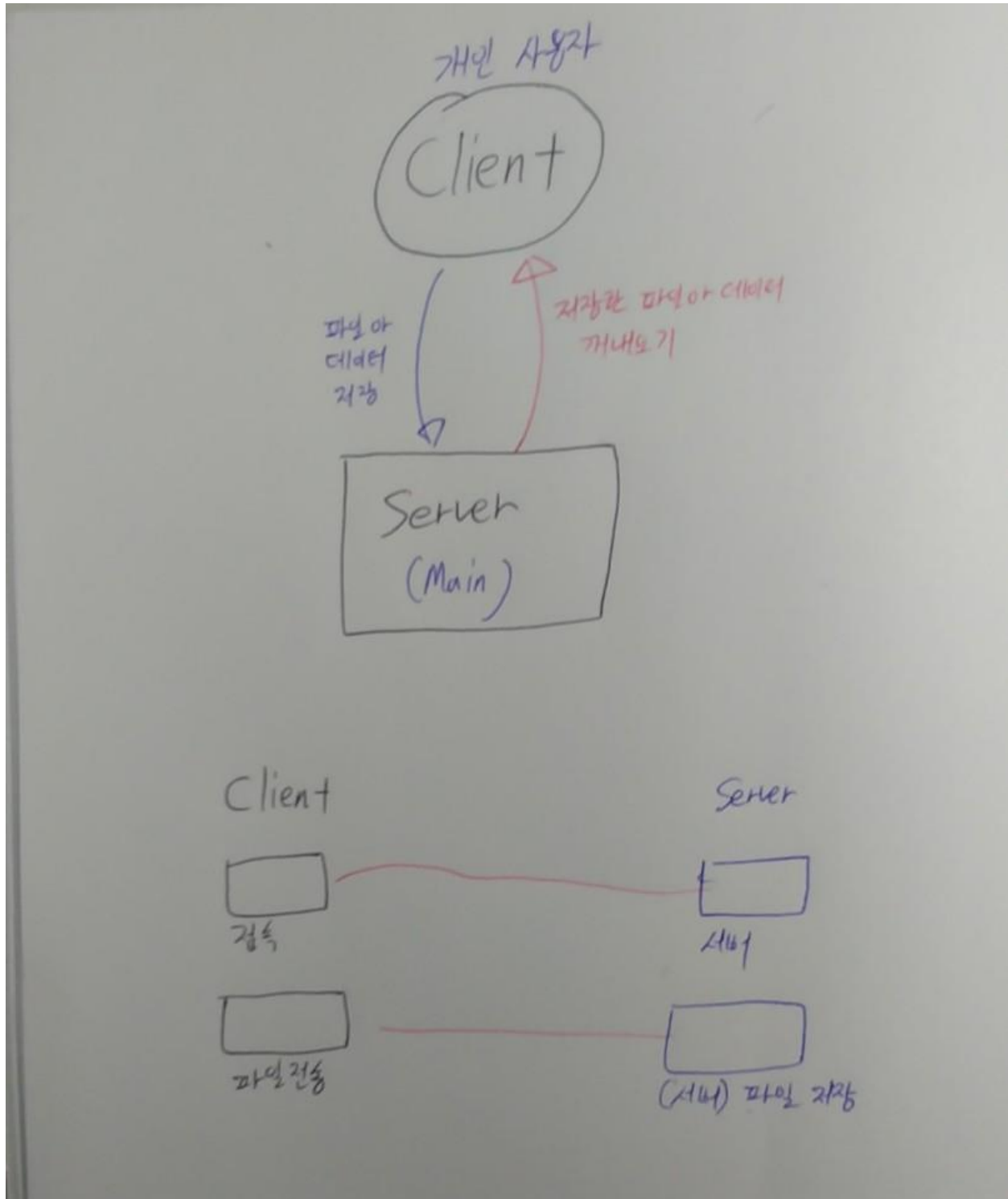
개인 사용자는 유니캐스트 기반의 파일 공유 시스템에 자신이 중요하다고 생각하는 데이터를 마음껏 저장할 수 있고 파일 공유 시스템의 서버와 클라이언트 간 1:1 통신이므로 제 3 자의

열람 및 공유가 안된다는 점에서 개인의 프라이버시를 보호받을 수 있고 그만큼 데이터 안정성은 확보될 수 있다.

1:1 통신의 파일 공유 시스템은 또 다른 일종의 가상 저장 공간을 제공하는 역할도 한다.

자신이 필요로 하는 데이터가 너무 많아서 PC의 용량 부족을 초래할 때 파일 공유 시스템에 그러한 데이터들을 모아서 저장할 수 있어서 개인 PC의 용량 확보가 가능하다는 점에서 또 다른 저장 공간 역할도 수행한다.

[Q 4][어플리케이션 구성도] 어플리케이션 구성도를 자세하게 그림으로 나타내세요. [20pts]



[Q 5][기대효과 및 활용방안] 프로젝트 결과물이 어떤 응용분야에서 어떻게 활용될 수 있을지 구체적으로 설명하세요. [20pts]

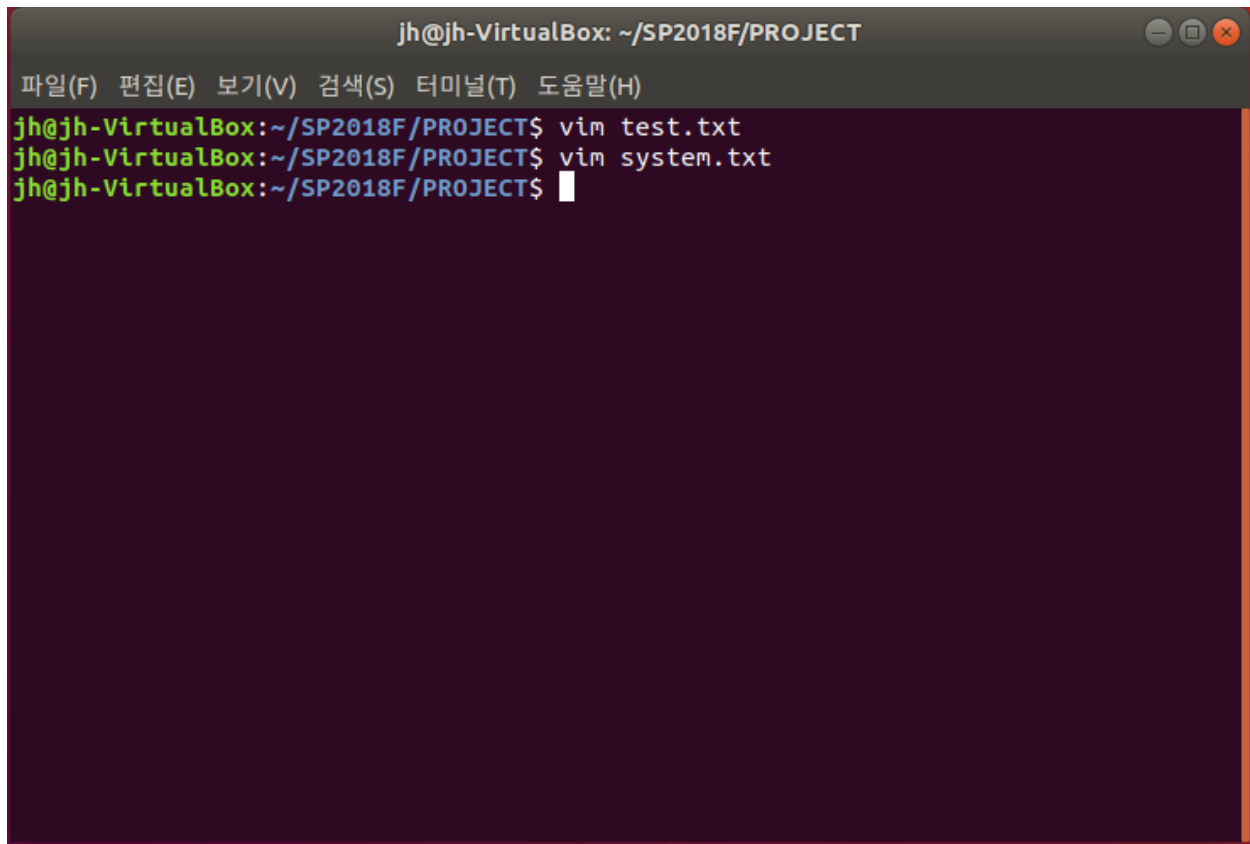
앞에서 언급한 바와 같이, 클라이언트는 파일 공유 시스템 서버와 1:1 통신을 통해서 자신의 중요한 데이터나 파일들을 부담 없이 저장할 수 있고 제 3 자의 접근이 없어서 보안 면에서도 훌륭하고 자신만의 공간이라는 점에서 자유도가 높다. PC 에 중요한 파일이나 데이터를 파일 공유 시스템에 추가로 보관함으로써 PC 의 문제로 유실된다 하더라도 복구가 용이하다. 이것은 결국 데이터의 안정성을 보장해줄 수 있다.

또한, 개인의 PC 의 저장 공간 확보 면에서도 좋은 영향을 준다. PC 에 중요한 파일이나 데이터를 모두 가지고 있기에는 용량 면에서 부담스러울 때 파일 공유 시스템을 통해 따로 보관이 가능하다는 면에서 저장 공간 확보가 용이하다.

1:1 유니캐스트 기반 파일 공유 시스템이면 USB 나 하드디스크와 비슷한 기능을 한다고 생각할 수도 있다. 비슷한 기능은 분명 맞지만 차이점이 있다. 하드디스크나 USB 는 매번 가지고 다니기 부담스러울 뿐만 아니라 가지고 다니지 않을 경우도 분명 있다.

그럴 때 파일 공유 시스템을 통해 어디서든 인터넷만 있으면 손쉽게 자신의 데이터에 접근이 가능하다는 점에서 클라이언트의 삶의 질을 높힐 수 있다.

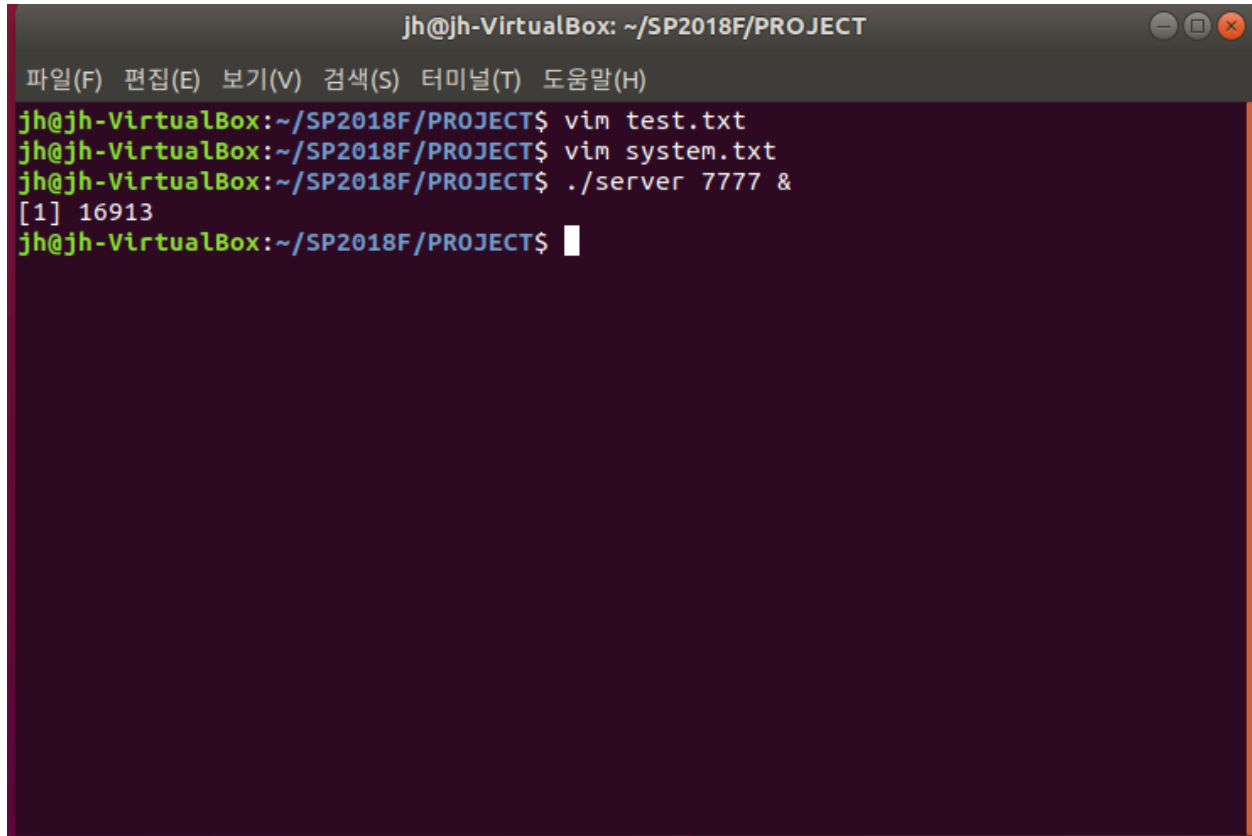
Part II: 데모/시연 시나리오 및 결과물



```
jh@jh-VirtualBox: ~/SP2018F/PROJECT
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
jh@jh-VirtualBox:~/SP2018F/PROJECT$ vim test.txt
jh@jh-VirtualBox:~/SP2018F/PROJECT$ vim system.txt
jh@jh-VirtualBox:~/SP2018F/PROJECT$
```

➔ 먼저, 데모를 위해서 vim 에디터를 사용하여 test.txt 와 system.txt 파일을 생성하였다.

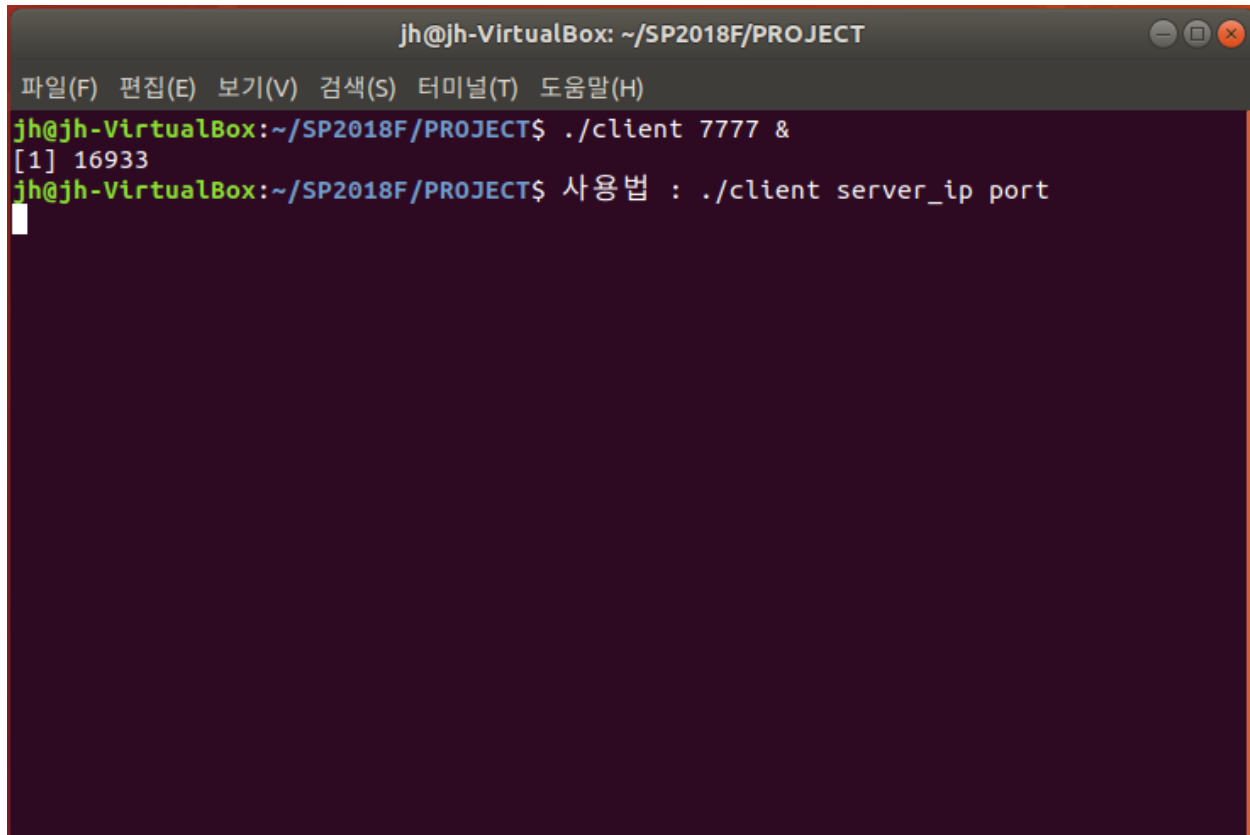
<서버 가동>



```
jh@jh-VirtualBox: ~/SP2018F/PROJECT
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
jh@jh-VirtualBox:~/SP2018F/PROJECT$ vim test.txt
jh@jh-VirtualBox:~/SP2018F/PROJECT$ vim system.txt
jh@jh-VirtualBox:~/SP2018F/PROJECT$ ./server 7777 &
[1] 16913
jh@jh-VirtualBox:~/SP2018F/PROJECT$
```

- ➔ 포트 번호 7777 을 선택해서 서버 구동. 즉, 파일 공유 시스템에서 메인을 담당하는 서버가 가동 상태.

<클라이언트 가동>

A terminal window titled 'jh@jh-VirtualBox: ~/SP2018F/PROJECT'. The window has a menu bar with '파일(F)', '편집(E)', '보기(V)', '검색(S)', '터미널(T)', and '도움말(H)'. The terminal shows the following commands and output:

```
jh@jh-VirtualBox:~/SP2018F/PROJECT$ ./client 7777 &  
[1] 16933  
jh@jh-VirtualBox:~/SP2018F/PROJECT$ 사용법 : ./client server_ip port
```

➔ 이 때 클라이언트도 서버와 마찬가지로 같은 포트번호를 선택해서 가동.

<클라이언트 메인(파일 공유 시스템)에 접속>

```
jh@jh-VirtualBox:~/SP2018F/PROJECT$ 사용법 : ./client server_ip port
./client 127.0.0.1 7777
명령어 : get, put, pwd, ls, cd, quit
client> 
```

- ➔ 사용법에 따라 서버 ip 인 127.0.0.1 과 선택한 포트번호 7777 을 입력.
여기서 127.0.0.1 은 로컬호스트. 즉 자기 자신을 가리키는 ip 이다. 이것은 동일한 기계에서만 액세스할 수 있고 포트가 인터넷이나 네트워크가 아닌 PC 자체의 연결만을 수신한다. 유니캐스트 기반의 파일 공유 시스템을 개인 PC 에서 구현하기 위해 127.0.0.1 사용

<파일 업로드할 때>

```
명령어 : get, put, pwd, ls, cd, quit
client> put
업로드할 파일 : a
파일이 없습니다.
명령어 : get, put, pwd, ls, cd, quit
client> put
업로드할 파일 : test.txt
업로드 완료
명령어 : get, put, pwd, ls, cd, quit
client> put
업로드할 파일 : system.txt
업로드 완료
명령어 : get, put, pwd, ls, cd, quit
client> 
```

- ➔ 먼저 생성한 파일들을 파일 공유 시스템에 업로드 해보자.
a 라는 파일을 업로드하려고 하니 파일이 없다는 메시지를 띄운다.
최근에 생성한 test.txt, system.txt 를 업로드하니 완료라는 메시지를 띄운다.
정상적으로 업로드가 된 것이다.

<다운로드 할 때>

```
client> test.txt
명령어 : get, put, pwd, ls, cd, quit
client> get
다운로드할 파일 : a
파일이 없습니다
명령어 : get, put, pwd, ls, cd, quit
client> get
다운로드할 파일 : test.txt
다운로드 완료
명령어 : get, put, pwd, ls, cd, quit
client> get
다운로드할 파일 : system.txt
다운로드 완료
명령어 : get, put, pwd, ls, cd, quit
client> 
```

- ➔ 다음으로는 파일 공유 시스템에서 개인 pc 로 파일을 다운로드 할 때의 상황이다.
파일 공유 시스템에 없는 a 라는 파일을 다운로드 받으려고 하니 파일이 없다는 메시지를 띄운다.

방금 업로드한 test.txt 와 system.txt 를 다운로드하려고 하니 “다운로드 완료”라는 메시지를 띄운다. 이는 정상적으로 실행이 된 것이다.

<현재까지 업로드한 파일들을 열람할 때>

```
명령어 : get, put, pwd, ls, cd, quit
client> ls
--The Remote Directory List--
a.out
a.out_1
a.out_1_1
client
client.c
server
server.c
sns.txt
sns.txt_1
sns.txt_1_1
system.txt
system.txt_1
system.txt_1_1
system.txt_1_1_1
system.txt_1_1_1_1
temp.txt
temps.txt
test.txt
```

- ➔ 다음은 파일 공유 시스템에 업로드한 목록을 보기 위해 ls 명령어를 사용한 경우이다. 그동안 업로드한 목록들이 나열되는데 이때 방금 업로드한 system.txt 와 test.txt 도 확인할 수 있다.

<저장한 파일들의 경로를 볼 때>

```
명령어 : get, put, pwd, ls, cd, quit
client> pwd
--The path of the Remote Directory--
/home/jh/SP2018F/PROJECT
```

- ➔ 다음과 같이 pwd 명령어를 검색했을 때 파일들이 업로드 된 경로가 나오게 된다.

<경로 변경을 할 때>

```
명령어 : get, put, pwd, ls, cd, quit
client> cd
이동할 경로 이름 : /home/jh/SP2018F
경로 변경 완료
```

- ➔ cd 를 통해서 원하는 경로로 이동이 가능하다. 예를 들어서 파일 공유 시스템에 기존의 파일들이 A 폴더에 저장되었다면 cd 명령어를 통해서 B 폴더로 한번에 이동이 가능하다.

```
명령어 : get, put, pwd, ls, cd, quit
client> pwd
--The path of the Remote Directory--
/home/jh/SP2018F
```

- ➔ 그래서 최종적으로 출력을 해보면 원하는 경로로 이동된 것을 볼 수 있다.

<종료하였을 때>

```
jh@jh-VirtualBox:~/SP2018F/PROJECT$ ./server 7777 &
[2] 17606
jh@jh-VirtualBox:~/SP2018F/PROJECT$ FTP server quitting..
jh@jh-VirtualBox:~/SP2018F/PROJECT$

명령어 : get, put, pwd, ls, cd, quit
client> quit
서버를 닫는 중..
jh@jh-VirtualBox:~/SP2018F/PROJECT$
```

- ➔ 마지막으로 quit 을 했을 때 클라이언트와 서버 모두 닫히는 것을 볼 수 있다.

Part III: 프로젝트 진행중 발생한 문제점 및 해결방안

프로젝트를 진행하면서 업로드, 다운로드, 파일목록 출력, 경로 변경, 현재경로 출력, 종료 등 등 전체적인 기능들을 구현하기는 하였다. 하지만 헛점이 하나 있다.

바로 ls 명령어의 결함이다. 이것은 put 명령어를 통한 업로드만이 아니라,

에디터로 인한 파일 생성에도 ls 리스트에 출력이 된다.

예를 들어, vim 에디터를 통해 test.txt 를 생성한 후 서버와 클라이언트 구동을 하였는데 이때 test.txt 파일을 put 명령어를 통해 파일 공유 시스템에 업로드를 해야지만 ls 명령어를 쳤을 때 파일 목록에 나와야 하는 것이 정석이다.

그러나 vim 에디터로 생성하자마자 파일 목록에 포함되는 것이 최대 문제이다. 이 문제에 대해 생각해본 결과, 서버가 현재 위치에서 ls >> temp.txt 를 하는데 그걸 클라이언트가 받아서 cat temp.txt 로 출력해서 문제가 생기는 것 같다. fopen 으로 읽는다면 temp 를 제외한 업로드한 파일명만 보일 것으로 예상된다. 추가적으로 서버쪽 소스에서 파일 보내주고나서 temp 삭제하는 시스템 명령어를 추가해야 ls 해줄때마다 이전에 만들었던 temp 파일을 목록에 못넣을 것이다.

이것 외에는 전체적으로 잘 실행되었다. 하지만 이번 프로젝트의 최대 난제는 멀티캐스트이다.

본 프로젝트는 유니캐스트를 기반으로 하는 파일 공유 시스템을 구현하였는데 좀 더 확대된 기능과 편리성을 위해서는 다중 클라이언트를 대상으로 하는 1:다 통신의 멀티캐스트를 기반으로 하는 파일 공유 시스템을 구현했어야 한다.

그 부분은 실력 부족으로 인해 구현하지 못해서 아쉽고 대신에 유니캐스트 기반의 파일 공유 시스템이라는 새로운 방식을 도입하였고 완성도 있게 제작하였다.

Part IV: 개선방향

현재 유니캐스트 기반의 파일 공유 시스템이 앞으로 개선할 방향은 먼저 서버-클라이언트 간 1:다 통신을 할 수 있는 멀티캐스트 기반의 파일 공유 시스템을 구현하는 것이다.

개개인이 서로 파일을 업로드 하고 서로 자신이 필요하는 정보를 열람하면서 필요에 따라 다운로드할 수 있는 상호작용도 할 수 있게끔 한다. 다만 클라이언트가 업로드 하기 전 자신의 파일을 제 3자가 열람 및 비공개 설정을 할 수 있게 하고 다운로드 또한 가능 여부를 체크하게 한 다음 업로드를 시키게 한다. 그래서 개인의 프라이버시도 존중하는 그런 멀티캐스트 기반 파일 공유 시스템을 구현하는 것이다.

다음으로는, 클라이언트가 서버에 업로드한 파일의 정보와 자신이 다운로드한 파일의 정보를 서버측에서 뿌려주어 쉽게 열람할 수 있게 하는 기능도 추가하는 것이다.

이러한 모든 기능들이 추가되었을 때 완벽한 파일 공유 시스템이 될 것이다.

[끝]