

REPORT

[시스템프로그래밍
Assignment 2-2]



학과	컴퓨터정보공학부
----	----------

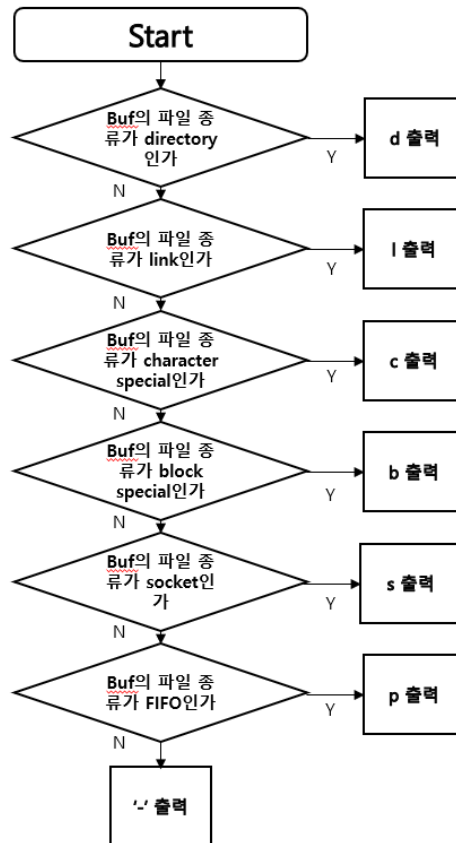
학번	2018202018
이름	유승재
담당 교수	김태석 교수님
제출일	2023.05.03

1. Introduction

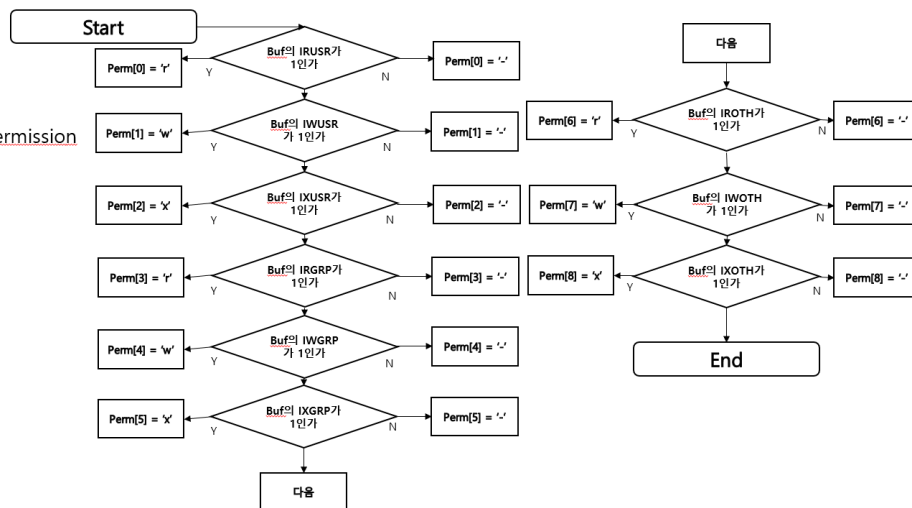
이번 과제는 Web server 에서 Socket 을 생성하고, 웹 브라우저에서 접속 시에 웹 브라우저가 보낸 HTTP request message 를 처리하고, 이전 과제에서 구현했던 HTML_1s 의 결과를 HTTP response message 에 담아서 웹 브라우저에 전송하면, 이를 client 가 웹 브라우저에서 결과를 확인하는 코드를 구현하는 것이 이번 과제입니다.

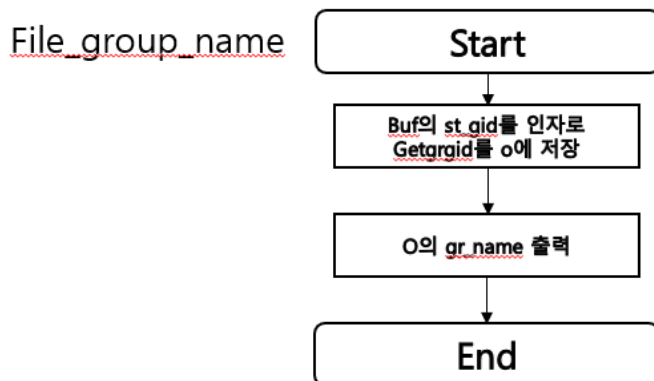
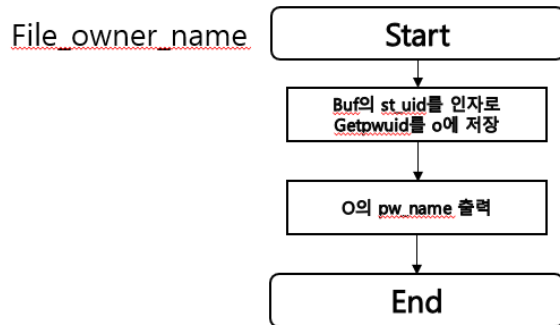
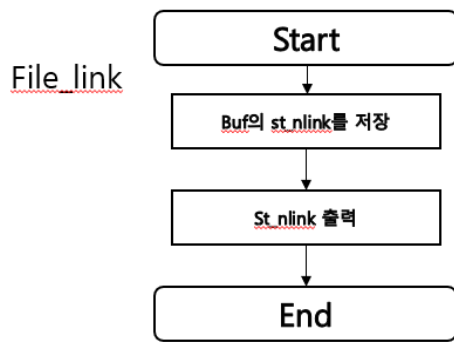
2. Flow Chart

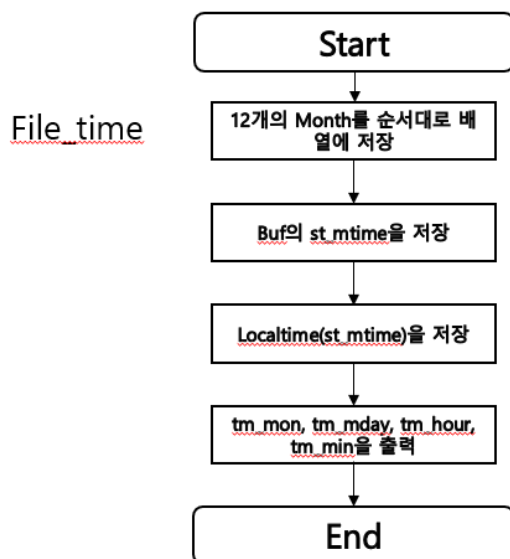
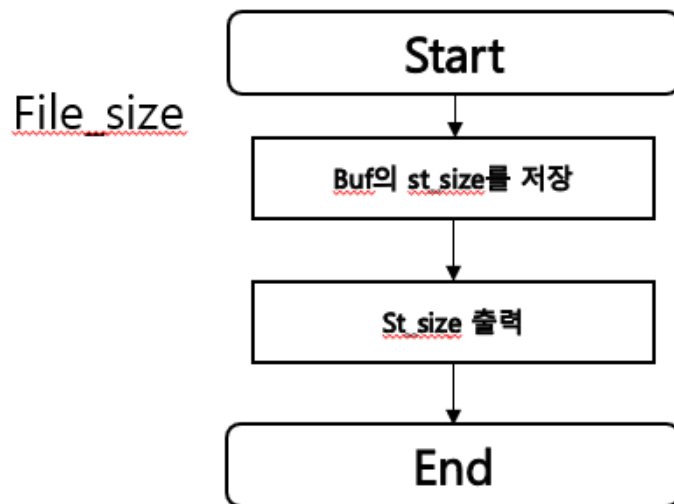
File_check

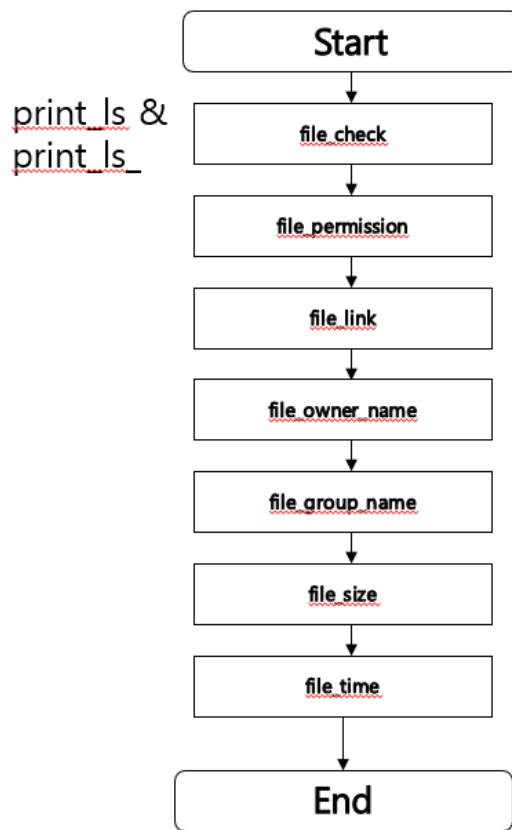


File_permission

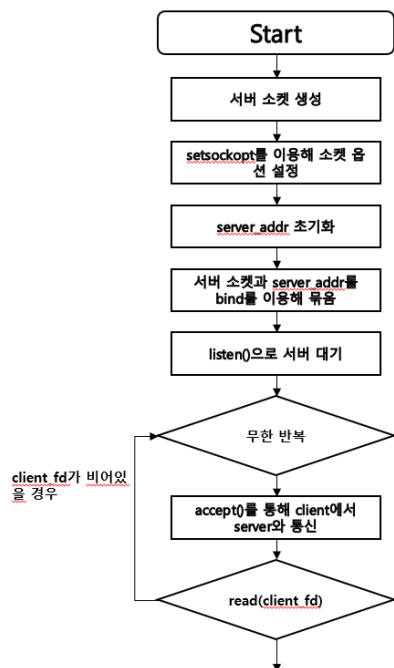


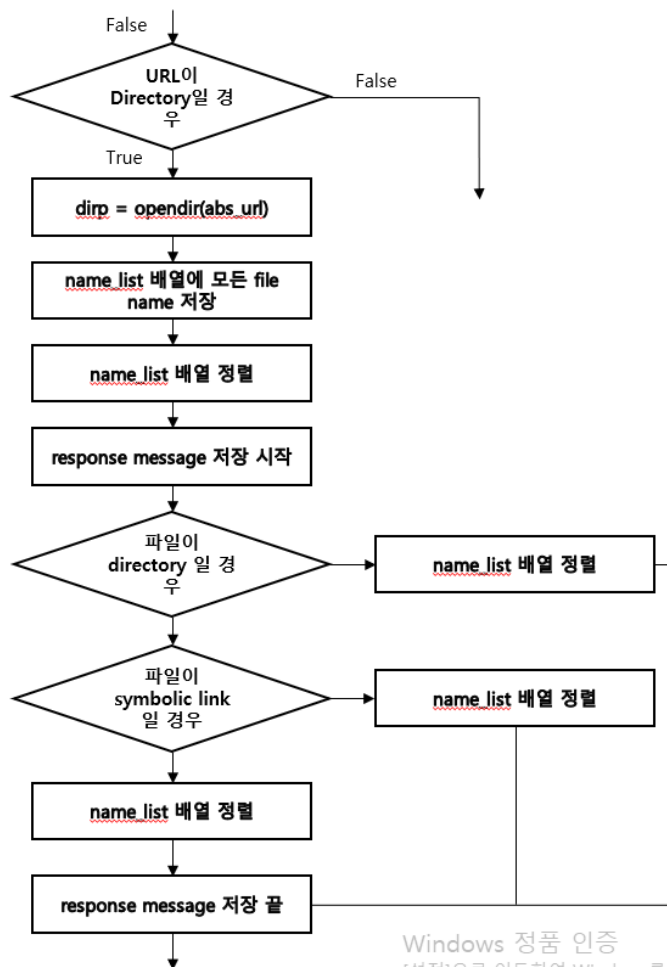
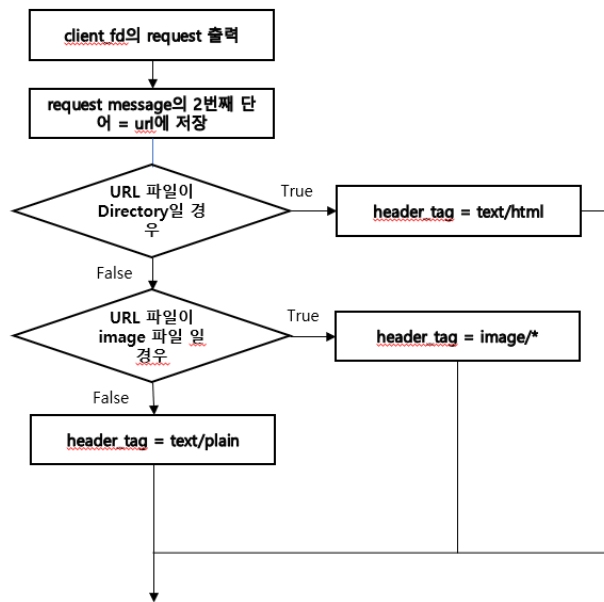




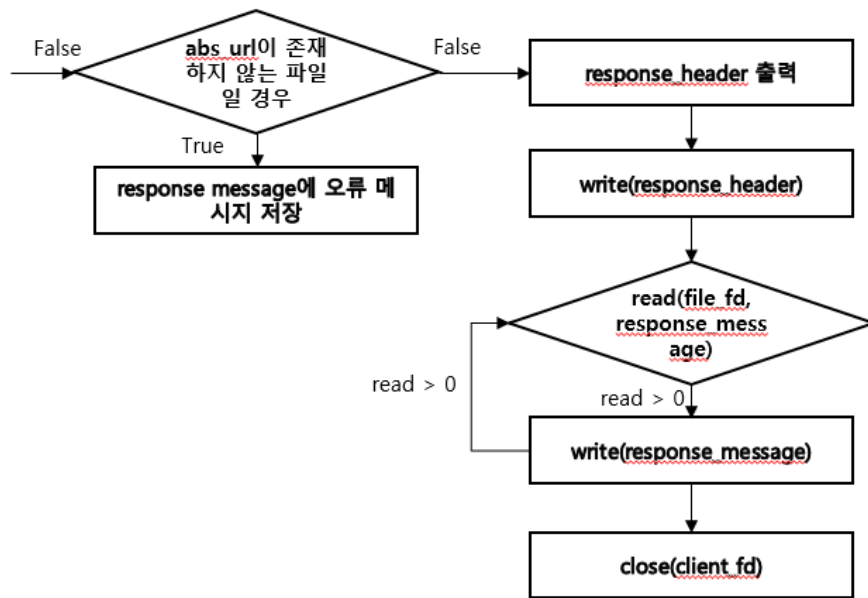


Main 함수





Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 !



3. Pseudo Code

```
void file_check(struct stat ){
```

```
    char 선언;
```

```
    table 행 시작;
```

```
    if (디렉토리){
```

```
        d 출력
```

```
    }
```



```
else if (링크){  
    | 출력  
}  
  
else if (character special){  
    c 출력  
}  
  
else if (block special){  
    b 출력  
}  
  
else if (socket){  
    s 출력  
}  
  
else if (FIFO){  
    p 출력  
}  
  
else if (regular){  
    - 출력  
}  
  
}
```

```
Void file_permission(struct stat ){
```

```
Perm[10];
```

Stat 의 st_mode 를 가져와 권한에 맞게 perm 의 자리에 char 배치

Perm 출력

```
}
```

```
Void file_link(struct stat ){
```

Stat 의 st_nlink 를 가져와 link 개수 확인

Link 개수 출력

```
}
```

```
Void file_owner_name(struct stat ) {
```

Stat 의 getpwuid 로 st_uid 를 가져와 struct passwd 형태에 저장

Struct passwd 의 pw_name 출력

```
}
```

```
Void file_group_name(struct stat ){
```

Stat 의 getgrgid 로 st_gid 를 가져와 struct group 형태에 저장

Struct group 의 gr_name 출력

```
}
```

```
Void file_size(struct stat ){
```

Off_t 변수 0 으로 초기화;

변수에 st_size 로 대입;

St_size 출력

}

Void file_time(struct stat){

12 개의 month 를 배열 크기 12 개의 문자열 배열에 1 월부터
12 월까지 저장;

Struct stat 의 st_mtime 을 localtime 함수를 이용해 struct tm 에 저장;

Struct tm 형식의 tm_mon, tm_mday, tm_hour, tm_min 출력

Table 행 끝

}

Void print_ls(struct stat){

File_check, file_permission, file_link, file_owner_name, file_group_name,

File_size, file_time 한번에 struct stat 을 넣어줘 출력

}

Int main(){

Socket_fd 생성;

Setsockopt 로 socket_fd 설정;

Struct sockaddr_in Server_addr 초기화;

Server_addr 과 socket_fd 바인딩;

Listen(socket_fd);

While (1){

Client_fd = Accept(socket_fd);

read(client_fd 를 읽을 경우 잘못 읽었을 때){

while 문 continue;

}

Client_fd 를 read 해서 request message 출력;

Request message 에서 url 확인;

url 이 directory 일 경우

header_tag = text/html;

url 이 이미지파일 일 경우

header_tag = image/*

그 외일 경우

Header_tag = text/plain

If (url 이 "/" (cwd)일 경우)

cwd 의 non-hidden 파일들 배열에 저장;

파일 배열 정렬;

파일배열 출력 형식을 HTML_Is 형식에 맞추어 response_message 에
저장;

Else if (url 이 cwd 가 아닌 경우)

 If (url 이 directory 인 경우)

 url 에 위치한 파일들 전부 배열에 저장;

 파일 배열 정렬;

 파일 배열 출력 형식을 HTML_Is 형식에 맞추어
 response_message 에 저장;

 Else if (url 이 파일인 경우)

 url 의 파일 정보 전부 read;

 read 한 데이터 전부 response_message 에 저장;

 response_header client_fd 에 출력;

 response_message client_fd 에 출력;

 else (존재하지 않는 파일인 경우)

 404 Error 양식에 맞추어 출력;

url 이 directory 이거나 image 파일 인 경우

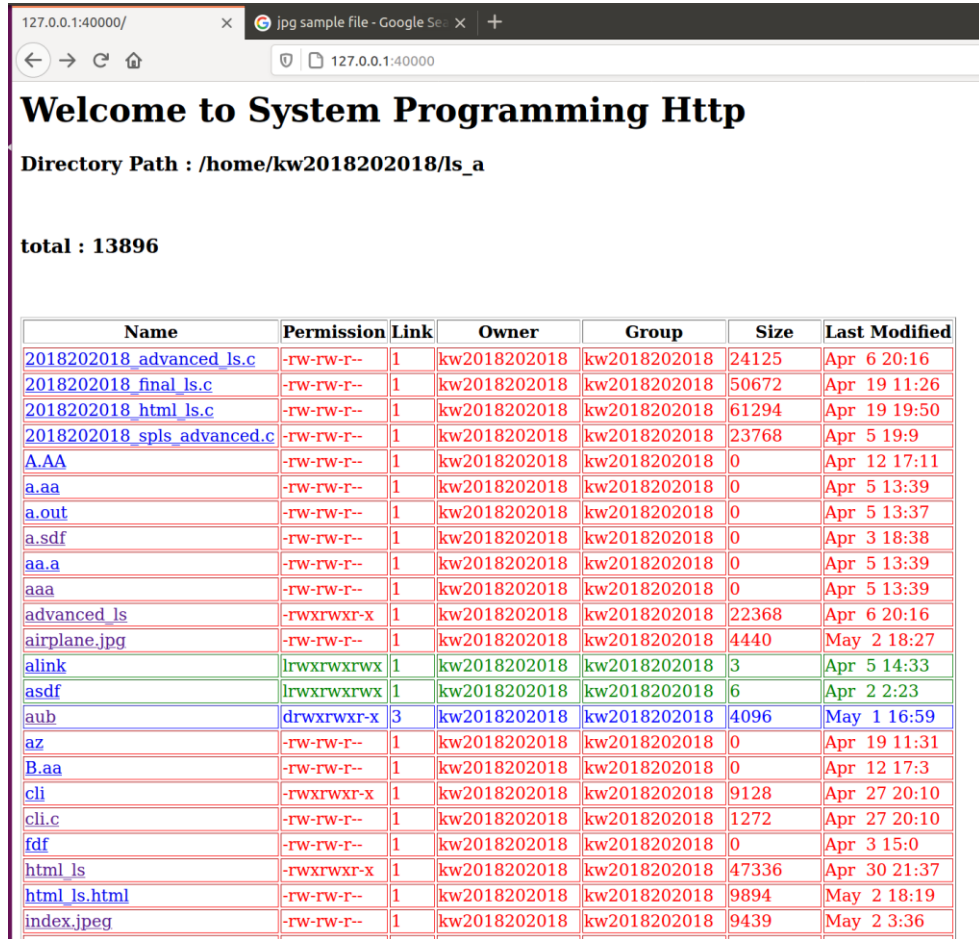
response_header 출력;

이전에 저장한 response_message 출력;

}

}

4. 결과화면



127.0.0.1:40000/ x jpg sample file - Google Se: x +

← → ↺ ↻ 127.0.0.1:40000

Welcome to System Programming Http

Directory Path : /home/kw2018202018/ls_a

total : 13896

Name	Permission	Link	Owner	Group	Size	Last Modified
2018202018_advanced_ls.c	-rw-rw-r--	1	kw2018202018	kw2018202018	24125	Apr 6 20:16
2018202018_final_ls.c	-rw-rw-r--	1	kw2018202018	kw2018202018	50672	Apr 19 11:26
2018202018_html_ls.c	-rw-rw-r--	1	kw2018202018	kw2018202018	61294	Apr 19 19:50
2018202018_spls_advanced.c	-rw-rw-r--	1	kw2018202018	kw2018202018	23768	Apr 5 19:9
A.AA	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 12 17:11
a.aa	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 5 13:39
a.out	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 5 13:37
a.sdf	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 3 18:38
aa.a	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 5 13:39
aaa	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 5 13:39
advanced_ls	-rwxrwxr-x	1	kw2018202018	kw2018202018	22368	Apr 6 20:16
airplane.jpg	-rw-rw-r--	1	kw2018202018	kw2018202018	4440	May 2 18:27
alink	lrwxrwxrwx	1	kw2018202018	kw2018202018	3	Apr 5 14:33
asdf	lrwxrwxrwx	1	kw2018202018	kw2018202018	6	Apr 2 2:23
aub	drwxrwxr-x	3	kw2018202018	kw2018202018	4096	May 1 16:59
az	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 19 11:31
B.aa	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 12 17:3
cli	-rwxrwxr-x	1	kw2018202018	kw2018202018	9128	Apr 27 20:10
cli.c	-rw-rw-r--	1	kw2018202018	kw2018202018	1272	Apr 27 20:10
fdf	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 3 15:0
html_ls	-rwxrwxr-x	1	kw2018202018	kw2018202018	47336	Apr 30 21:37
html_ls.html	-rw-rw-r--	1	kw2018202018	kw2018202018	9894	May 2 18:19
index.jpeg	-rw-rw-r--	1	kw2018202018	kw2018202018	9439	May 2 3:36

기본적인 경로 출력입니다.

127.0.0.1:40000/aub
 jpg sample file - Google Search

127.0.0.1:40000/aub

System Programming Http

Directory Path : /home/kw2018202018/ls_a/aub

total : 56

Name	Permission	Link	Owner	Group	Size	Last Modified
.	drwxrwxr-x	3	kw2018202018	kw2018202018	4096	May 1 16:59
..	drwxrwxr-x	4	kw2018202018	kw2018202018	4096	May 3 2:45
dir1	drwxrwxr-x	2	kw2018202018	kw2018202018	4096	May 1 16:59
file1	-rw-rw-r--	1	kw2018202018	kw2018202018	0	Apr 3 12:47
file2	-rw-rw-r--	1	kw2018202018	kw2018202018	14	May 1 9:32
link1	lrwxrwxrwx	1	kw2018202018	kw2018202018	5	Apr 19 15:45
link2	lrwxrwxrwx	1	kw2018202018	kw2018202018	5	May 1 9:32
slime.png	-rw-rw-r--	1	kw2018202018	kw2018202018	37551	Apr 30 22:23
temp.txt	-rw-rw-r--	1	kw2018202018	kw2018202018	0	May 1 16:58

127.0.0.1:40000/aub/dir1
 jpg sample file - Google Search

127.0.0.1:40000/aub/dir1

System Programming Http

Directory Path : /home/kw2018202018/ls_a/aub/dir1

total : 8

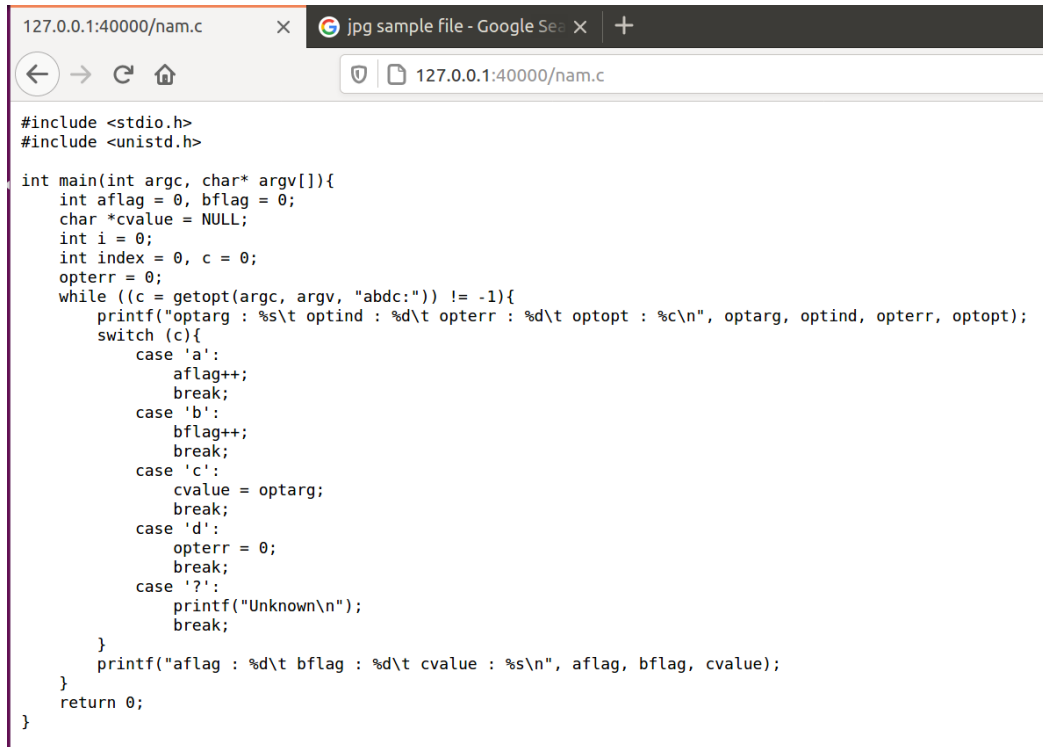
Name	Permission	Link	Owner	Group	Size	Last Modified
.	drwxrwxr-x	2	kw2018202018	kw2018202018	4096	May 1 16:59
..	drwxrwxr-x	3	kw2018202018	kw2018202018	4096	May 1 16:59

하위 디렉토리를 들어간 모습입니다.

127.0.0.1:40000/aub/file2
 jpg sample file - Google Search

127.0.0.1:40000/aub/file2

This is file2

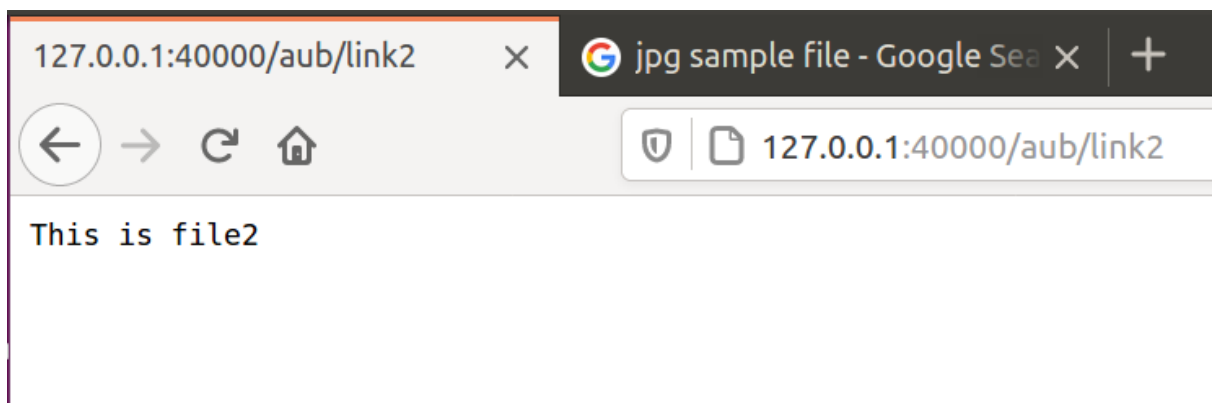


```
#include <stdio.h>
#include <unistd.h>

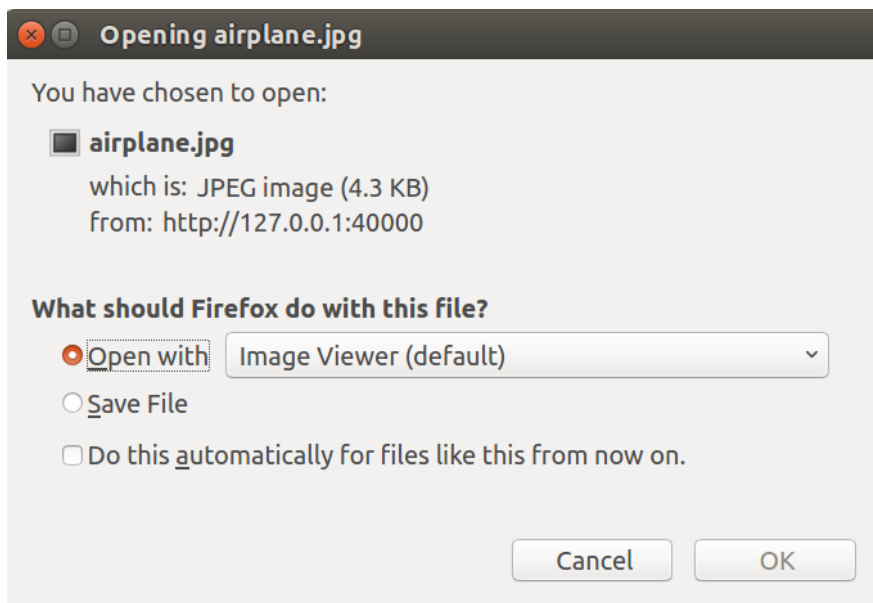
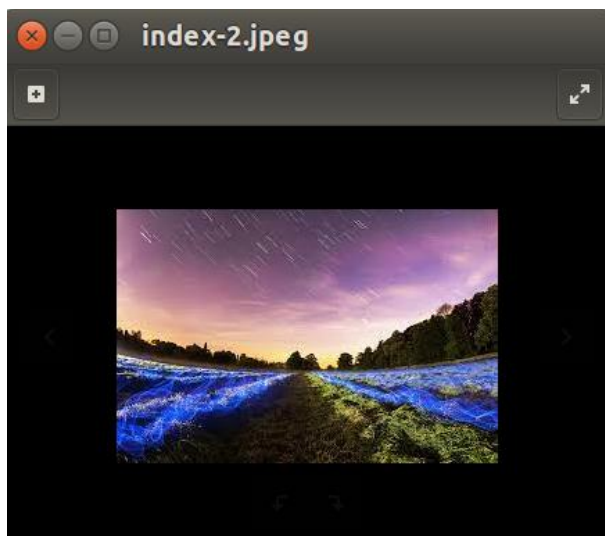
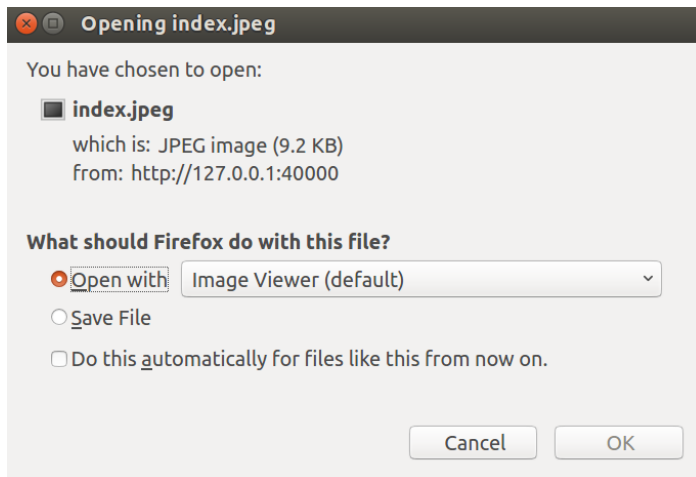
int main(int argc, char* argv[]){
    int aflag = 0, bflag = 0;
    char *cvalue = NULL;
    int i = 0;
    int index = 0, c = 0;
    opterr = 0;
    while ((c = getopt(argc, argv, "abdc:")) != -1){
        printf("optarg : %s\t optind : %d\t opterr : %d\t optopt : %c\n", optarg, optind, opterr, optopt);
        switch (c){
            case 'a':
                aflag++;
                break;
            case 'b':
                bflag++;
                break;
            case 'c':
                cvalue = optarg;
                break;
            case 'd':
                opterr = 0;
                break;
            case '?':
                printf("Unknown\n");
                break;
        }
        printf("aflag : %d\t bflag : %d\t cvalue : %s\n", aflag, bflag, cvalue);
    }
    return 0;
}
```

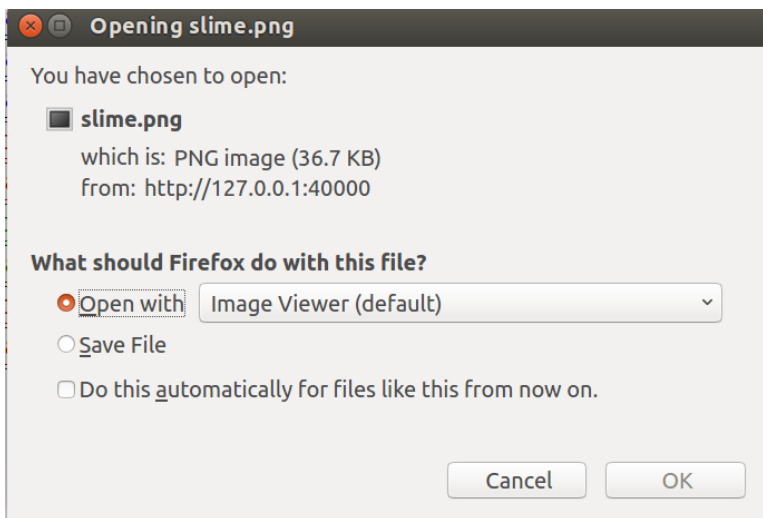
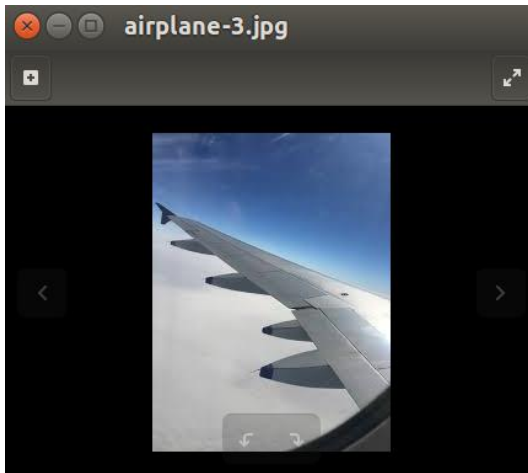
파일을 출력한 모습입니다.

C 언어 파일과 다른 일반적인 파일들을 코드 형태로 출력되는 것을 확인할 수 있었습니다.



File2 의 link 파일인 link2 를 접속할 경우 file2 의 내용이 출력되는 것을 확인할 수 있습니다.





모든 image 파일이 정확하게 출력되는 것을 확인할 수 있습니다.

5. 고찰

이번 과제는 HTTP 메시지에 대해서 이해하고 client 의 요청을 server 에서 해결하여 이를 html 형식으로 웹에 출력해주는 간단한 코드를 작성했습니다. 이번 과제에서는 HTTP request 와 response message 에 대한 이해가 중요했다고 생각합니다. 과제를 진행함에 있어서 response message 에 대한 이해도가 부족하여 과제 진행에 어려움을 겪었습니다. 하지만 HTTP message 의 이해와 이번에 새로 알게 된 open, read, write 등의 함수들을 적절히 사용함으로써 이번 과제를 해결하게 되었습니다. 또한 이번 과제는 이전까지와는 조금 다르게 서버와 클라이언트 간의 통신을 확인하는 것이 주된 목표였기 때문에 코드적인 부분이 아닌 통신이 느려지거나, 예상 외의 오류가 발생하는 일이 생각보다 많았던 것 같습니다. 아직 이러한 문제가 발생하긴 하지만 제가 해결할 수 있는 정도의 문제는 전부 해결하였다고 생각합니다. 시간이 조금 더 있었다라면 더 깔끔한 코드를 구현하는 것이 가능했을 것이라고 생각합니다.

6. Reference