로고이(가) 표시된 사진

자동 생성된 설명

시스템프로그래밍실습

Assignment2-2 과제

수업 명 : 시스템프로그래밍실습

과제 이름 : assignment2-2

담당 교수님 : 김태석 교수님

학 번 : 2019202005

이 름 : 남종식

**과제 소개**

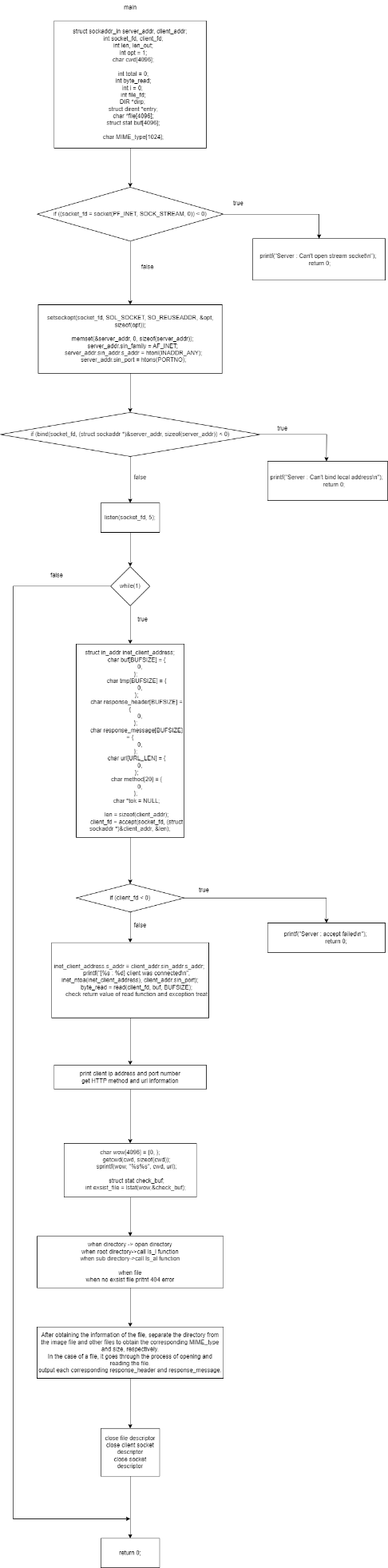
이번 과제는 저번 과제에서 진행한 HTML\_ls의 결과를 다른 장치의 웹 브라우저에서 확인할 수 있도록 지원하는 서버 프로그램을 작성하는 과제입니다. 이번 과제에서부터 네트워크 쪽을 다루기 때문에 이전 과제들과는 조금 다른 방식으로 진행했습니다.

먼저 과제를 진행하기 위해서는 HTTP 프로토콜을 사용하는데 HTTP(HyperText Transfer Protocol)는 인터넷에서 데이터를 주고받기 위한 프로토콜 중 하나입니다. HTTP는 클라이언트/서버 모델을 따릅니다. 클라이언트는 Request를 보내고, 서버는 Request를 처리하고 그에 대한 Response를 보냅니다. Request와 Response는 각각 Header와 Body로 구성됩니다. 헤더에는 Request와 Response에 대한 정보가 들어 있으며, 바디에는 Request와 Response의 본문이 들어 있습니다.

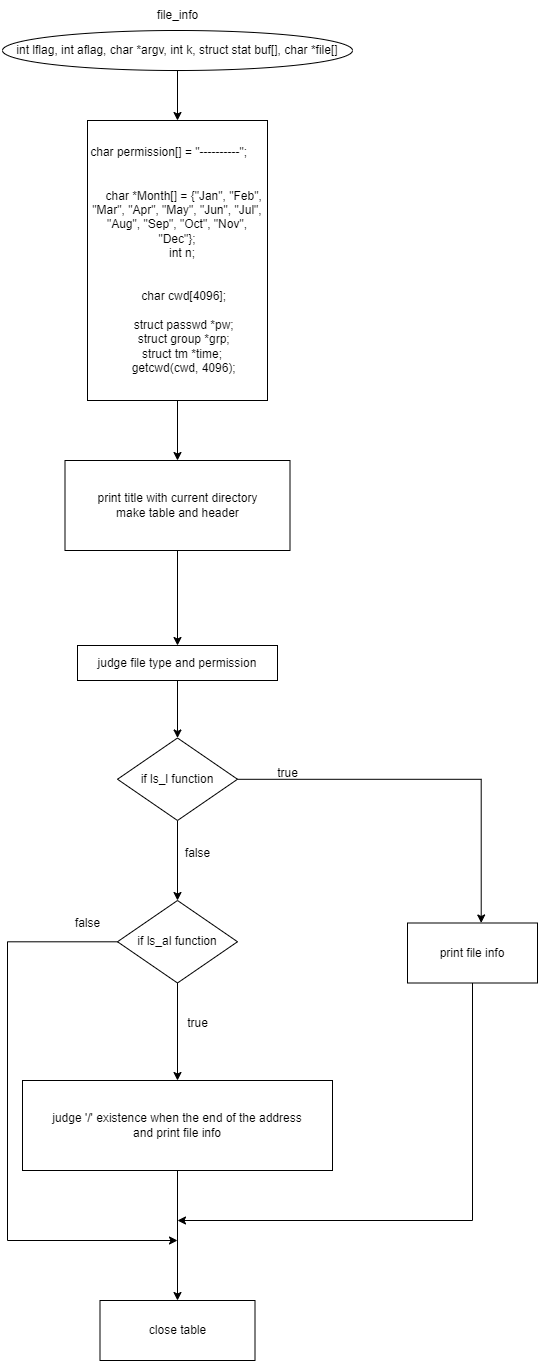
위 내용을 바탕으로 과제를 진행하는데 처음 실행 디렉토리를 root디렉토리로 설정하고 이때 -l옵션을 통해 히든파일 없이 출력합니다. 하위 디렉토리로 접속할 때는 -al옵션을 통해 히든파일 포함 파일 리스트를 출력합니다. 접속하기 위해서는 하이퍼링크를 통해야만 하는데 이때 하이퍼링크는 상대경로로만 생성하면 상위 디렉토리로는 이동할 수 없습니다. 그리고 파일 접속 시에는 파일의 내용을 출력해줘야 하며 이미지 파일 클릭 시에는 파일 다운로드 후 이미지가 출력되어야 합니다. 존재하지 않는 파일에 대해서는 404 error를 출력하면 됩니다.

**Flow Chart**

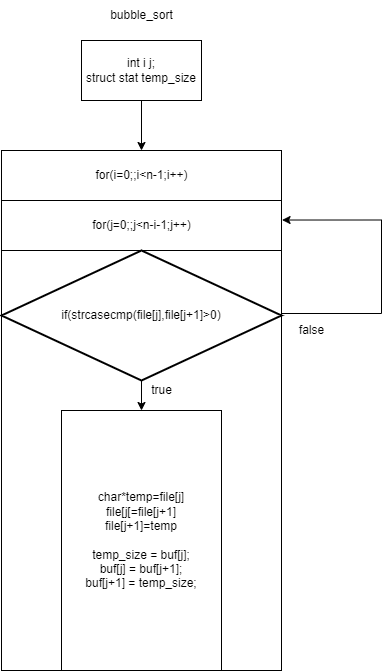
**Main fuction**



**File\_info function**



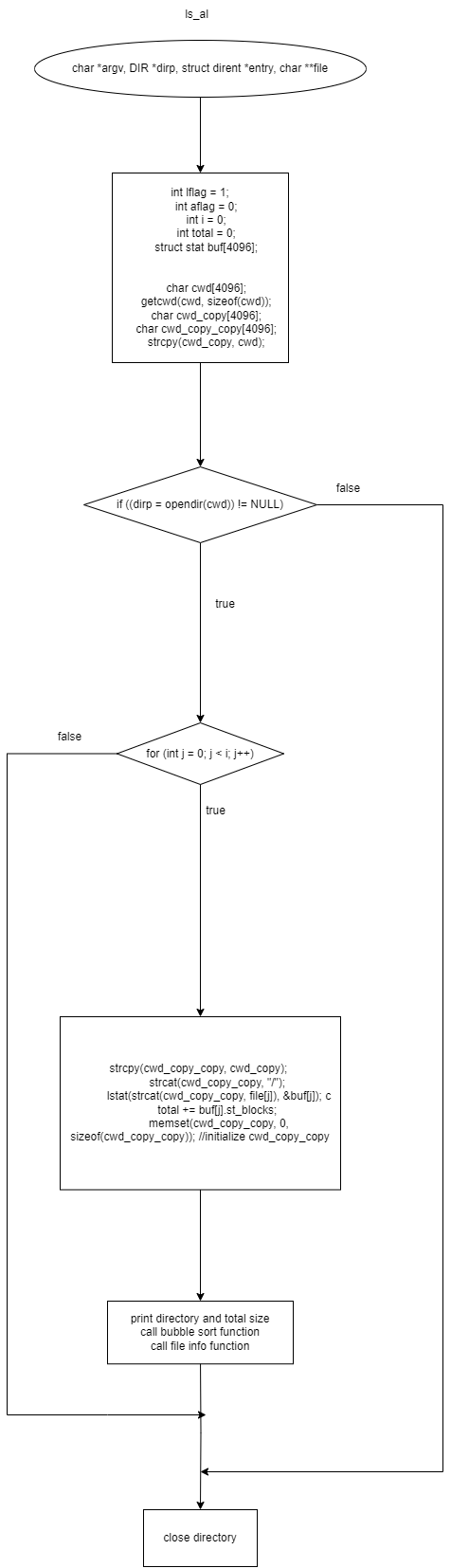
**Bubble sort function**



**Ls\_l function**



**Ls\_al function**



**Pseudo Code**

Main function

{

declare server address, client address

declare socket descriptor, client descriptor

declare pointer to directory stream and Pointer to a dirent structure

declare file list stat structure that stores information about a file or director

Call the socket function to create a socket file descriptor

Call setsocketopt to enable reuse of previously used port numbers

Initialize server\_addr structure

Save address information for the server socket

Bind a socket with socket\_fd to the IP address and port number specified in the server\_addr structure

Socket wait to accept incoming connections from client sockets.

while (1)

{

Define variables required to process client requests

define size of client address

get the socket file descriptor associated with the client

store ip address

Make an exception when read function return 0

Copy HTTP method information and url information

get current working directory

get path of wow (wow=cwd+url)

get information of files

open directory

{

when root directory

{

call ls -l function

}

when sub directory

{

call ls -al function

}

}

whwen not directory

{

no exsist file

{

print 404 error

}

}

get file info

when directory

{

MIME\_type=text/html

get size of response message

print response header

write response header

write response message

print the client's IP address and port number

close file descriptor

close client socket descriptor

}

when image file

{

MIME\_type=image/\*

get file size

file open

file read

print response header

write response header

write response message

print the client's IP address and port number

close file descriptor

close client socket descriptor

}

when file

{

MIME\_type=text/plain

get file size

file open

file read

print response header

write response header

write response message

print the client's IP address and port number

close file descriptor

close client socket descriptor

}

}

close socket descriptor

exit

}

define file info function about directory

{

get current directory

print title current directory

make table

make header

for (n = 0; n < k; n++)

{

judge file mode and permission

get File Owner's Information

get the information of the file's owning group

get the last modified time of that file

function for Parsing Time Information

/when root directory

{

when directory hyperlink with blue color :file[n]

when link file hyperlink with green color :file[n]

when file hyperlink with red color :file[n]

}

when sub directory

{

When the '/' is at the end of the address

{

when directory hyperlink with blue color :url file[n]

when link file hyperlink with green color :url file[n]

when file hyperlink with red color :url file[n]

}

When there is no '/' at the end of the address

{

when directory hyperlink with blue color :url/file[n]

when link file hyperlink with green color :url/file[n]

when file hyperlink with red color :url/file[n]

}

}

}

close table

}

define bubble sort function

{

initialize value

Bubble sort the file names alphabetically

}

define ls -l function

{

get current directory

get current directory+url

when open directory

{

Repeat read directory

{

store file names without hidden files

}

for (int j = 0; j < i; j++)

{

get current directory+url+/

get info about current directory+url+/+file

get block size

initialize current directory+url+/+file

}

print directory path & total size

call bubble\_sort function

print file info

}

close directory

}

define ls -al function

{

get current directory

get current directory+url

when open directory+url

{

Repeat read directory

{

store file list with hidden file

}

for (int j = 0; j < i; j++)

{

get current directory+url+/

get info about current directory+url+/+file

get block size

initialize current directory+url+/+file

}

print directory path & toatl size

call bubble\_sort function

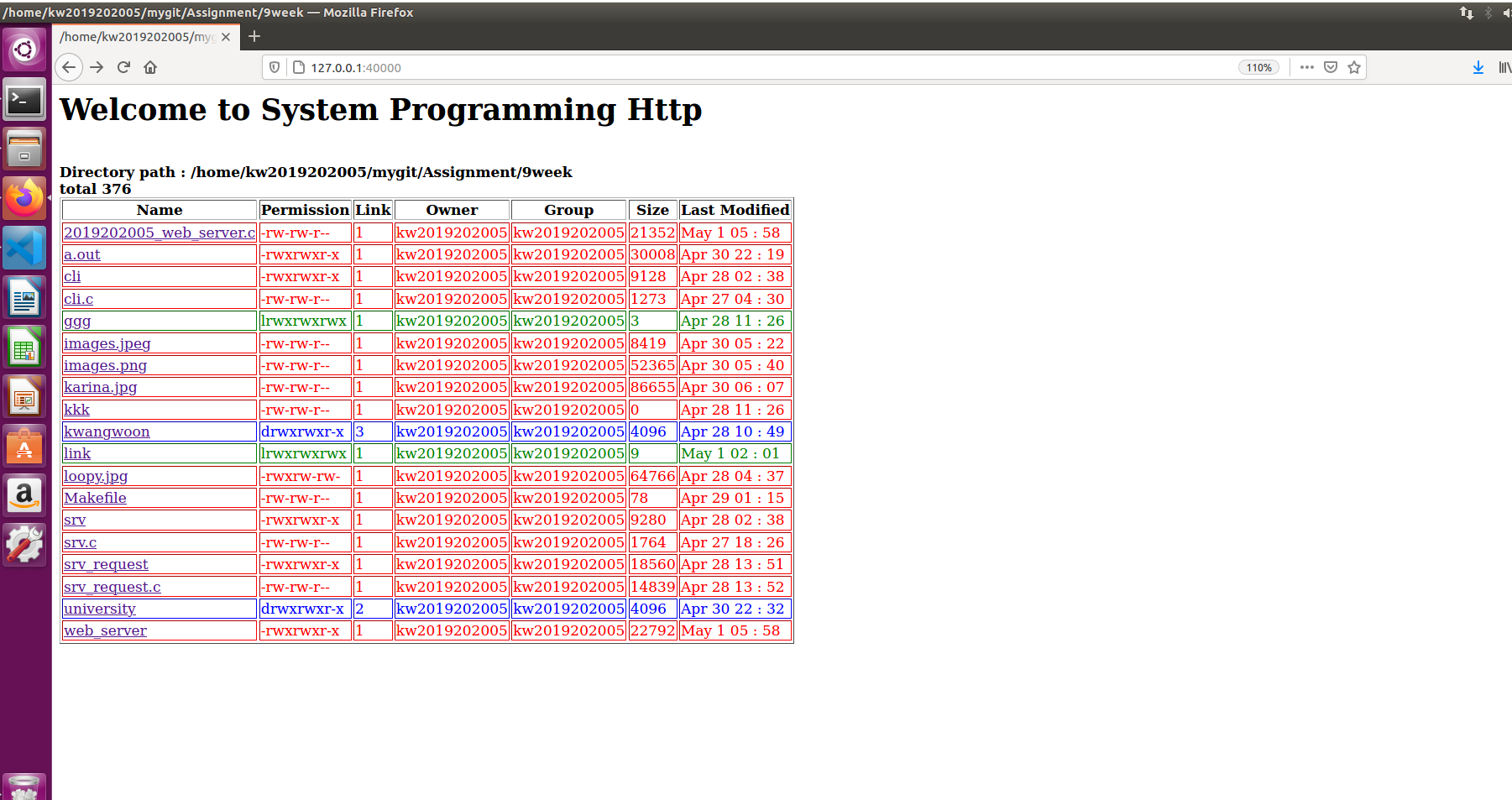
print file info

}

close directory

}

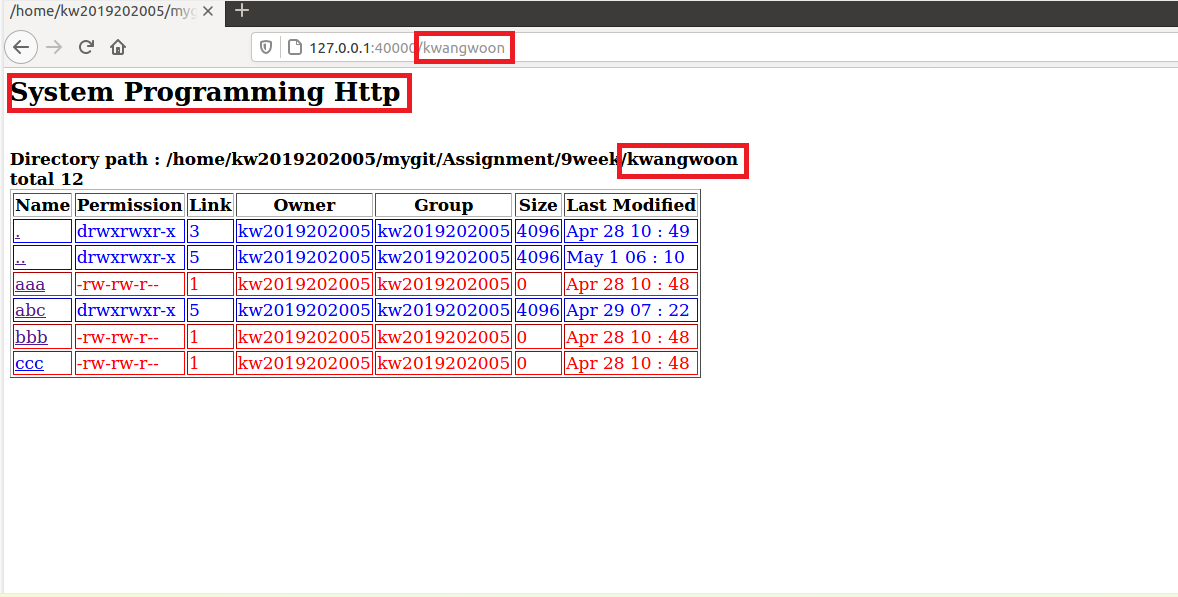
**결과화면**



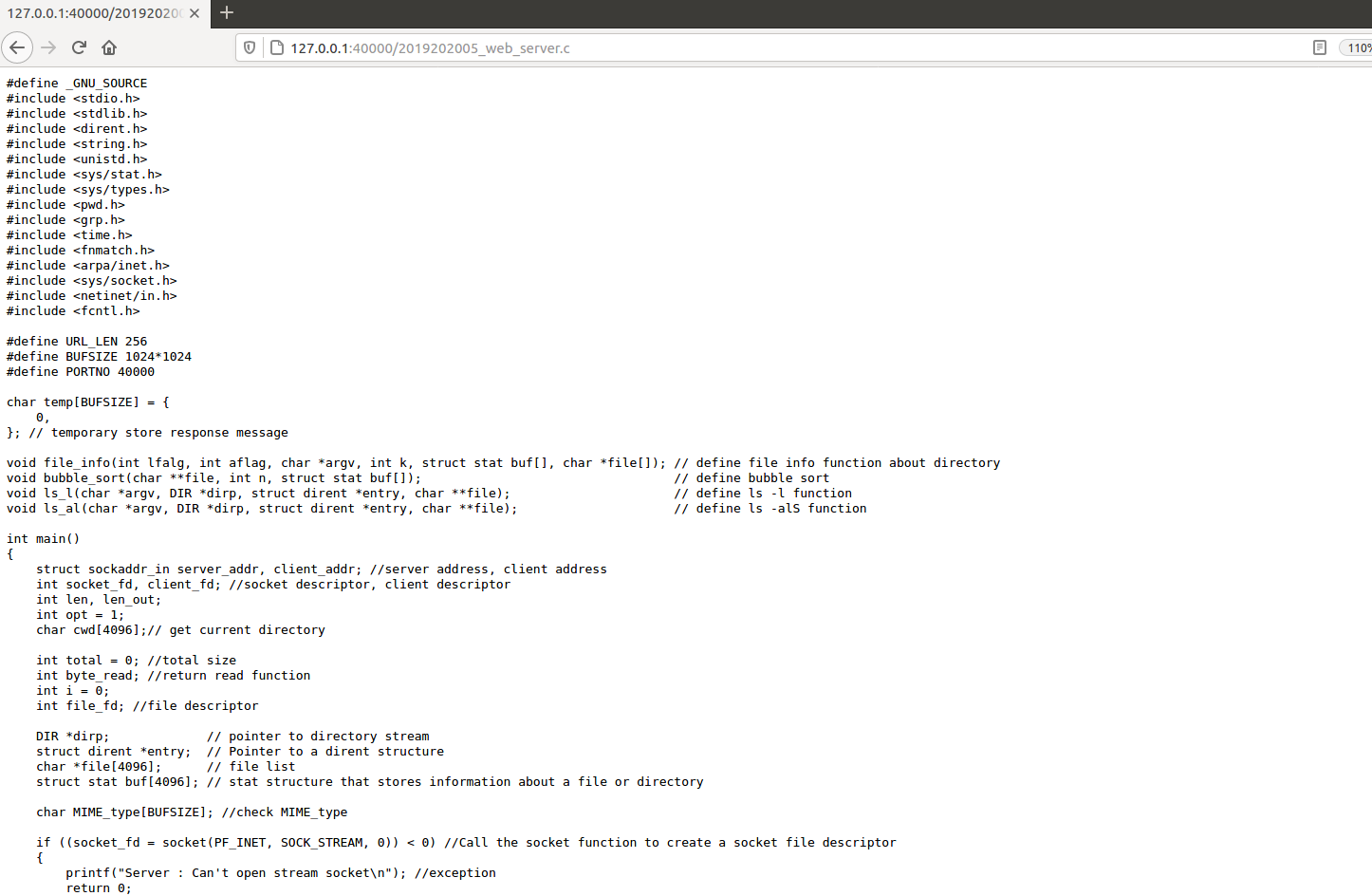
처음 실행 화면입니다. ./web\_server가 실행되는 root path에서는 -l옵션의 결과를 출력하고 상단에 welcome to system programming http를 출력한 화면입니다.

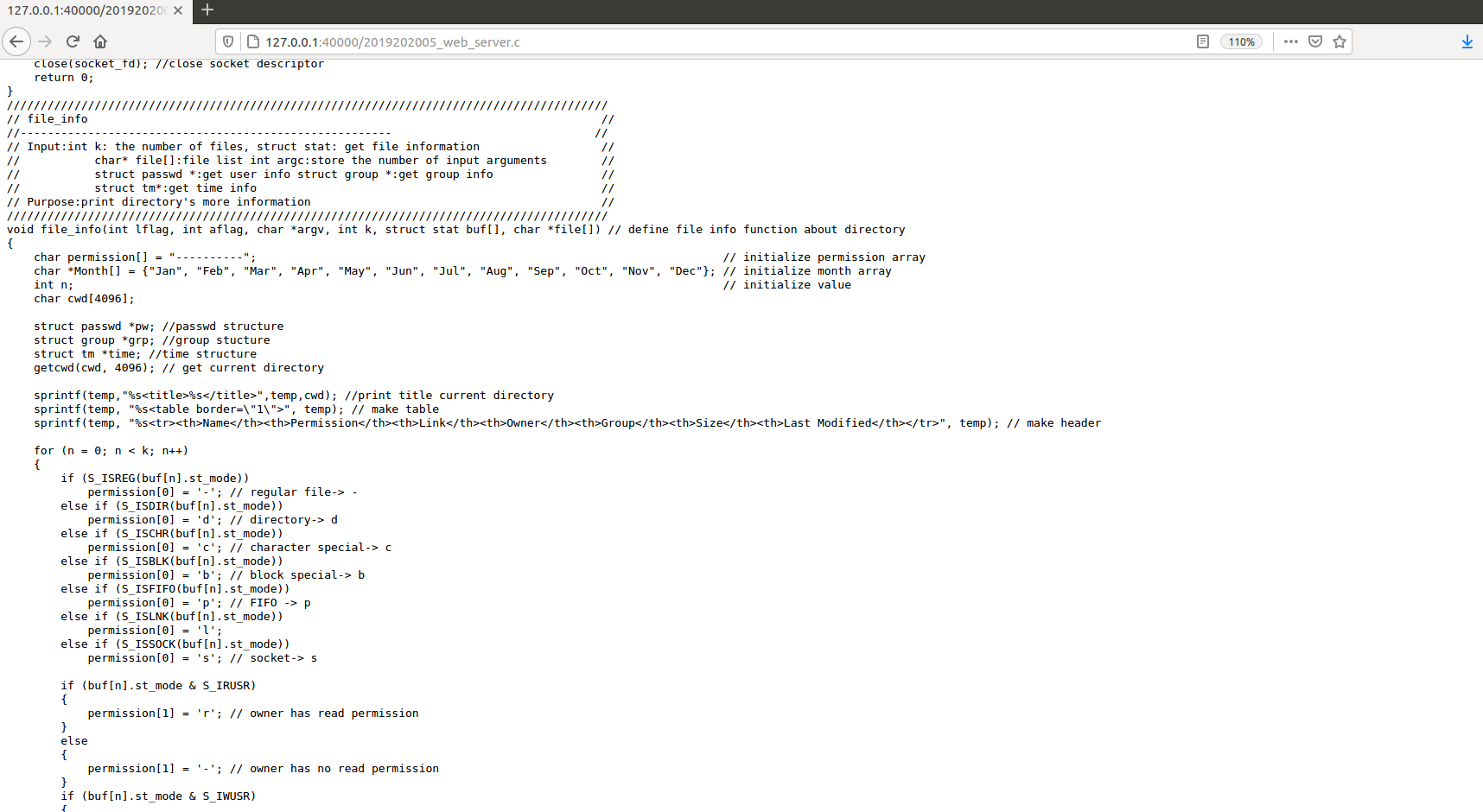
Directory path에는 현재 working directory의 위치를 출력하고 total에는 총 크기를 출력했습니다. 또한 title태그를 현재 디렉토리의 위치를 명시해주었습니다.

이 화면에서 디렉토리는 파란색, 파일은 빨간색, 링크 파일은 초록색입니다. 각각의 이름들은 모두 하이퍼링크로 연결 되어있으며 디렉토리 클릭 시 하위 디랙토리가 출력되고 파일 클릭 시 text파일 및 소스 코드일 경우에 파일의 내용을 출력해주어야 합니다. 만약 이미지 파일 클릭 시에는 이미지의 open 및 download창이 뜨며 이미지를 출력해주어야 합니다.

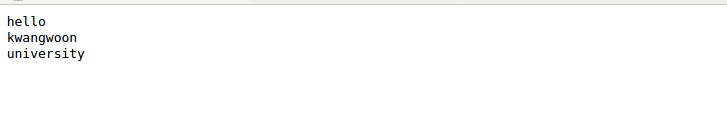


다음은 root directory의 하위 directory인 kwangwoon 디렉토리를 출력한 결과입니다. 하위 디렉토리를 출력하는 경우에는 system programming http를 상단에 출력해주어야 하며 directory path와 url주소 뒤에 현재 디렉토리의 이름이 붙어서 출력되는 모습을 확인할 수 있습니다. 그리고 이때는 ls -al을 이용하여 파일들을 출력해주어야 하며 .과 ..의 히든 파일을 출력했습니다. 여기서 .을 클릭하면 현재 화면이 계속 출력되며 ..클릭 시에는 상위 디렉토리로 이동하게 됩니다.

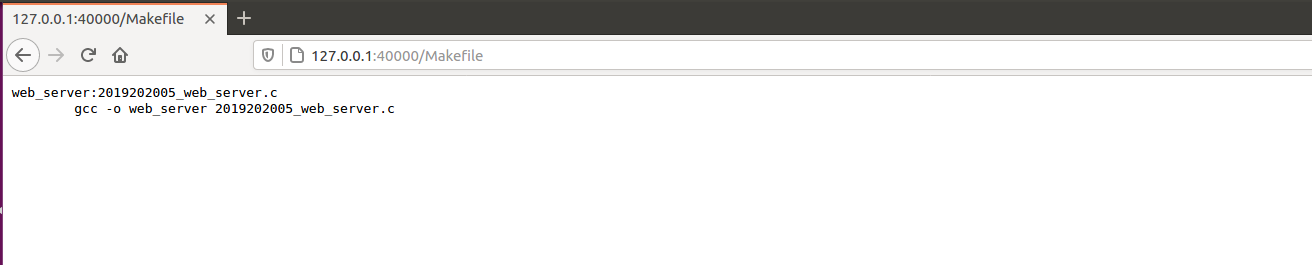




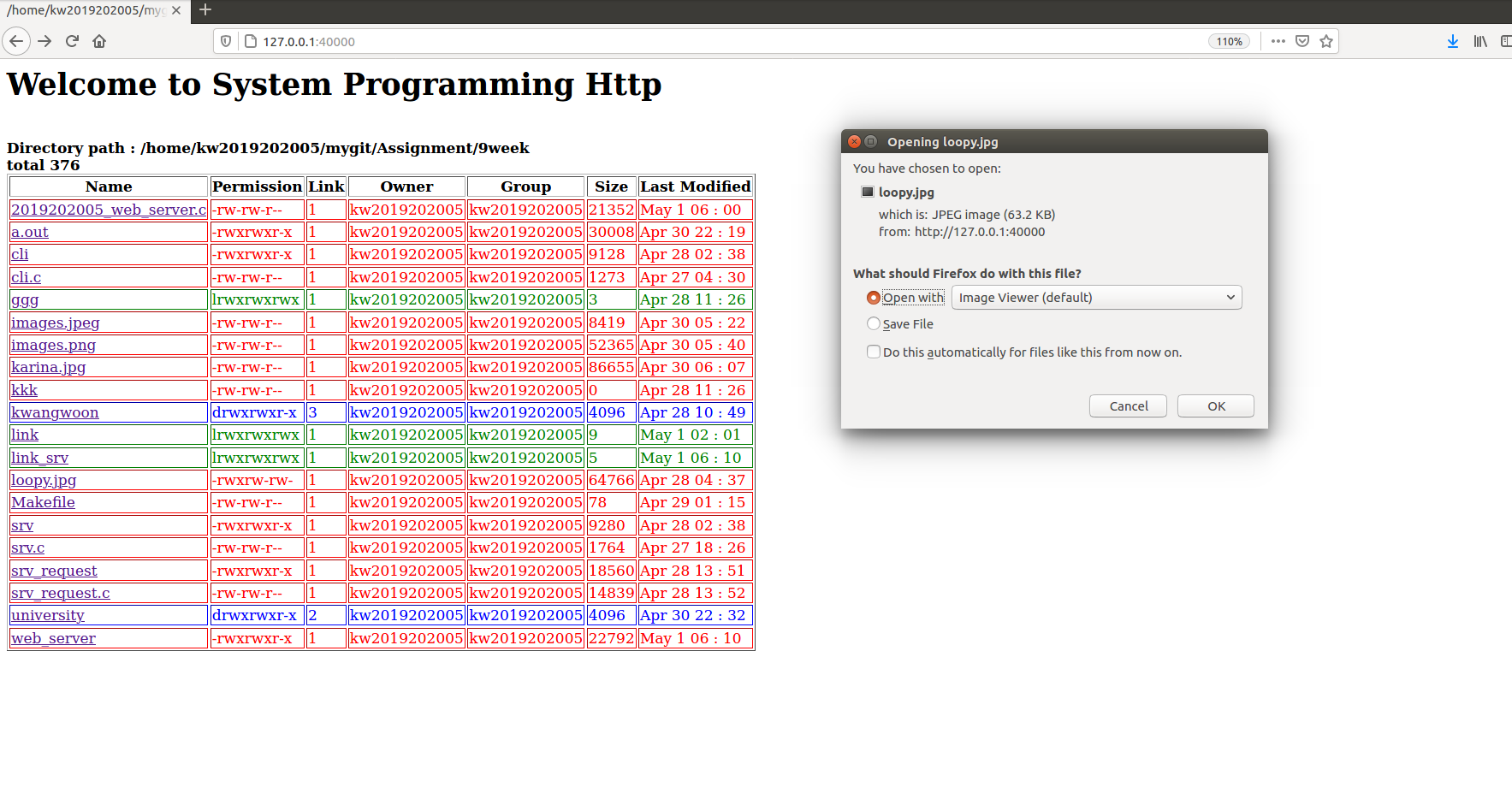
다음은 소스 코드파일을 클릭한 화면입니다. 소스 코드파일 클릭 시에는 파일의 내용이 출력해주어야 하며 2019202005\_web\_server.c의 파일의 내용이 출력된 모습입니다. 파일의 내용이 길어 초반 부분의 결과창을 캡처하여 첨부했습니다.



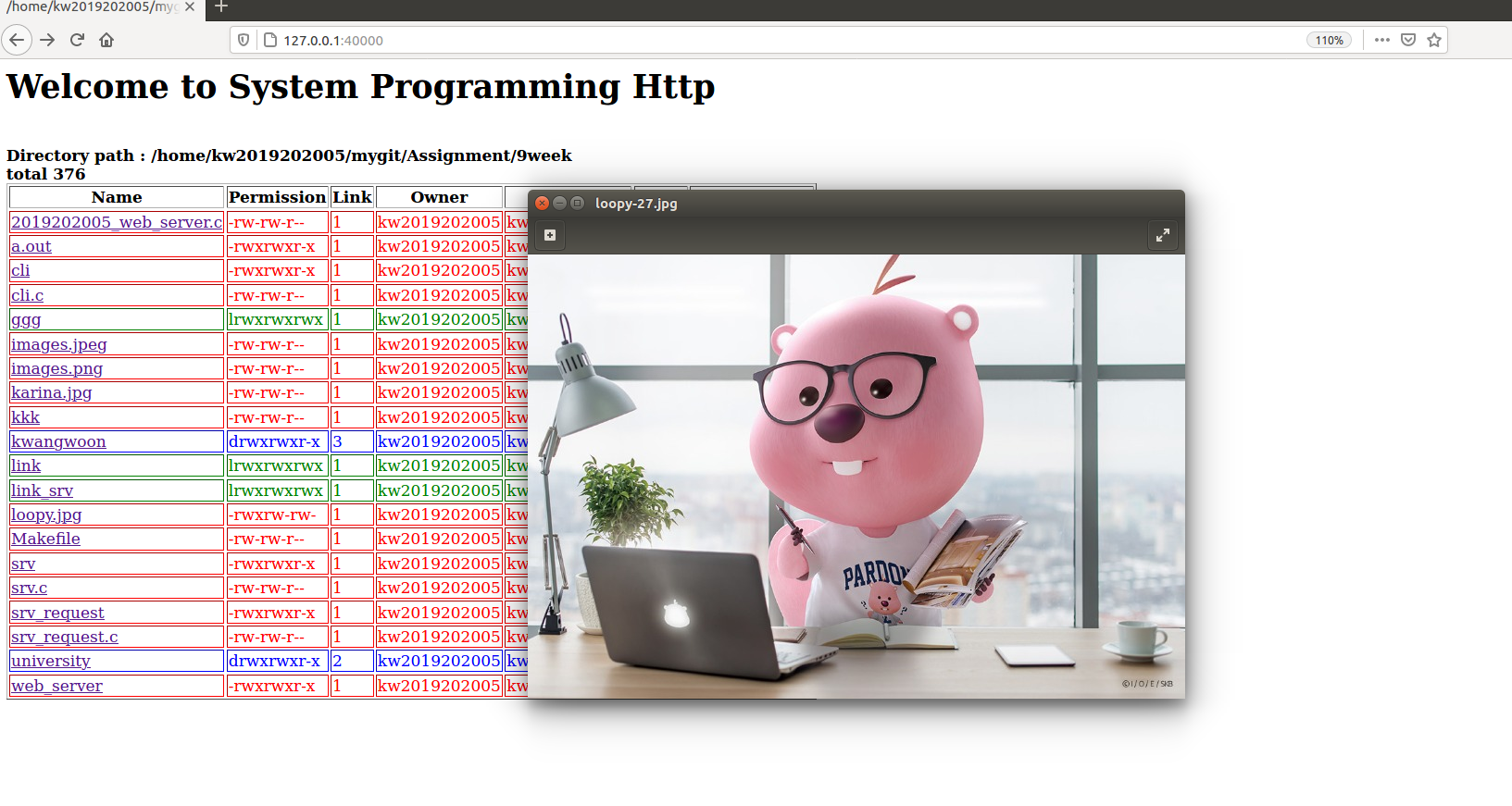
다음은 링크 파일 클릭 시 출력되는 화면입니다. 원본파일의 내용이 출력되는 점을 확인할 수 있습니다.



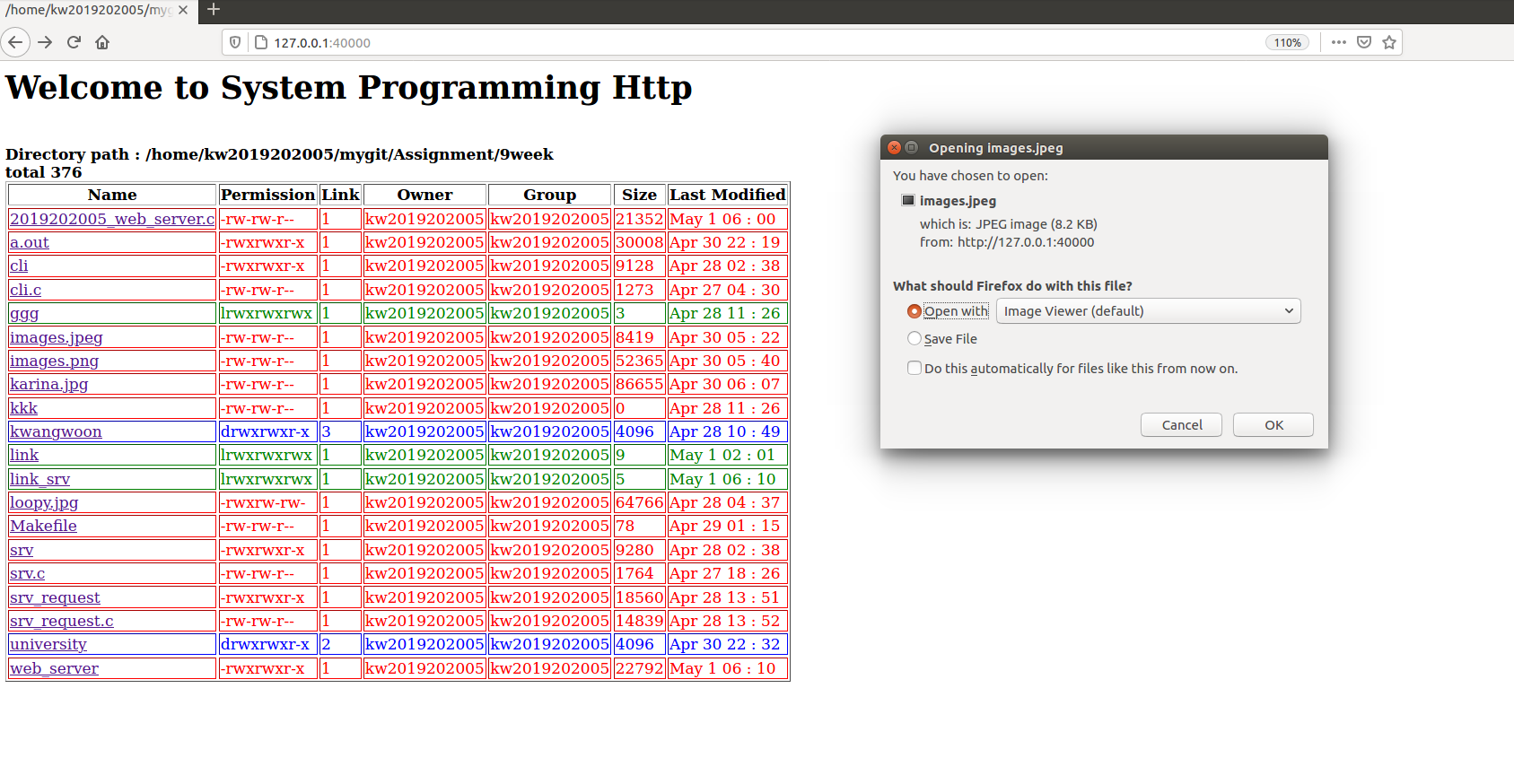
다음은 Makefile 클릭 시 출력되는 화면입니다. Makefile의 내용이 잘 출력되는 모습을 확인할 수 있습니다.

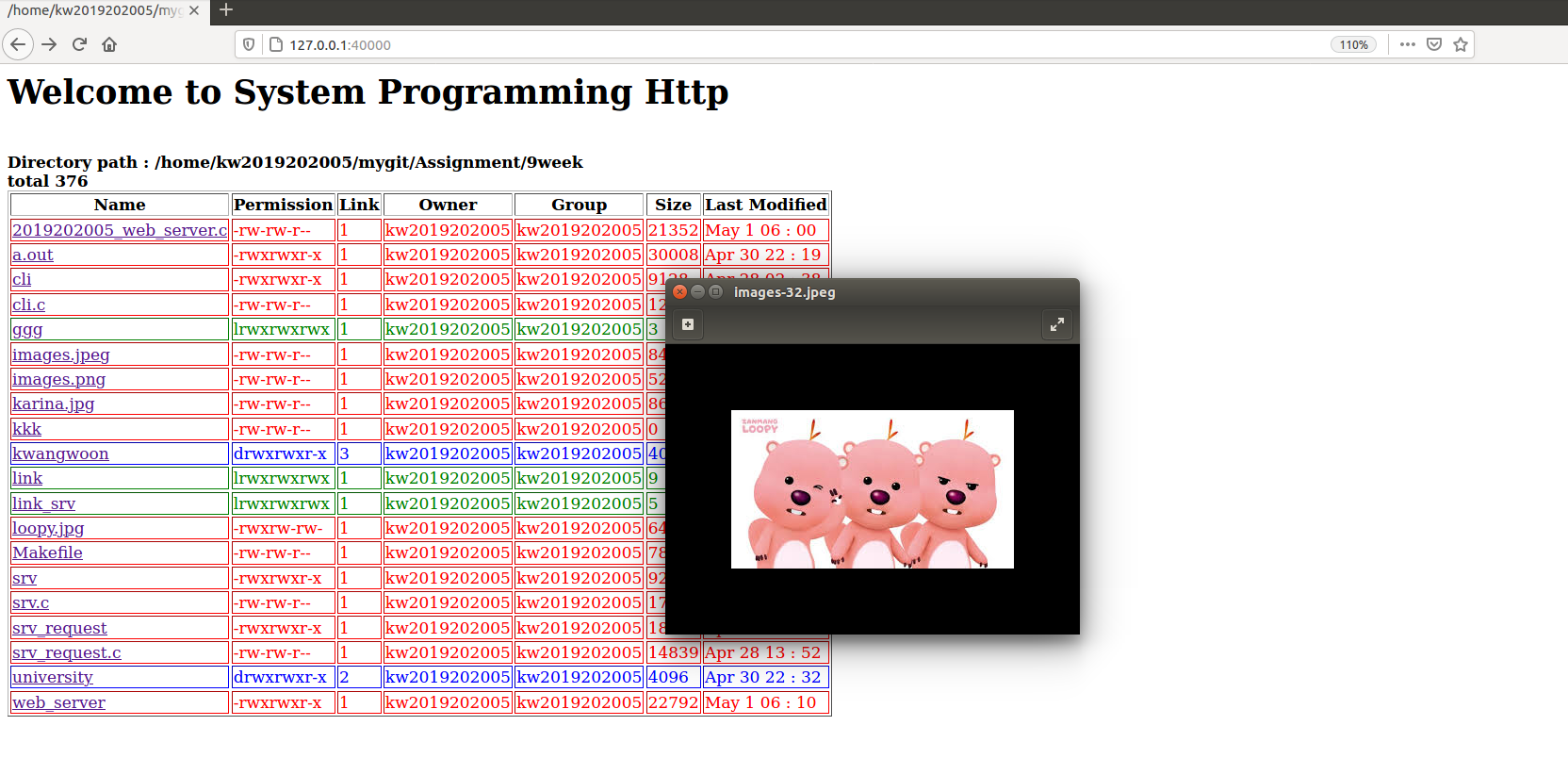


다음은 .jpg파일을 클릭 시 출력되는 화면입니다. 먼저 이미지 파일을 클릭하면 파일을 open할 것인지 download할 것인지 물어보는 창이 출력된 다음 ok를 누르면 이미지 파일이 출력됩니다.

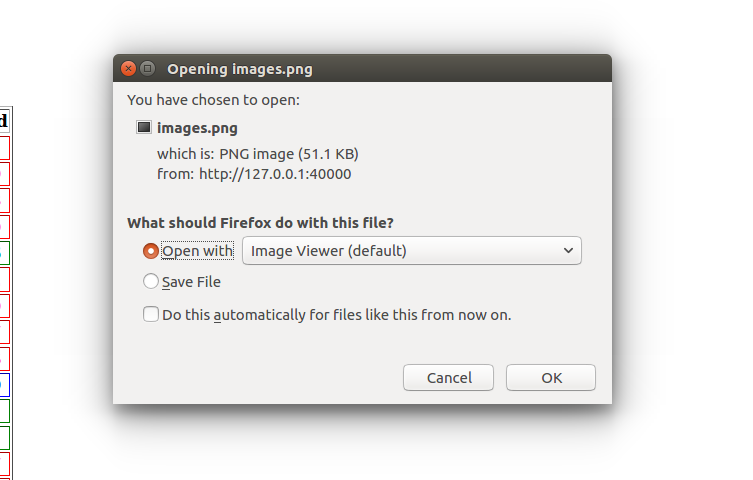


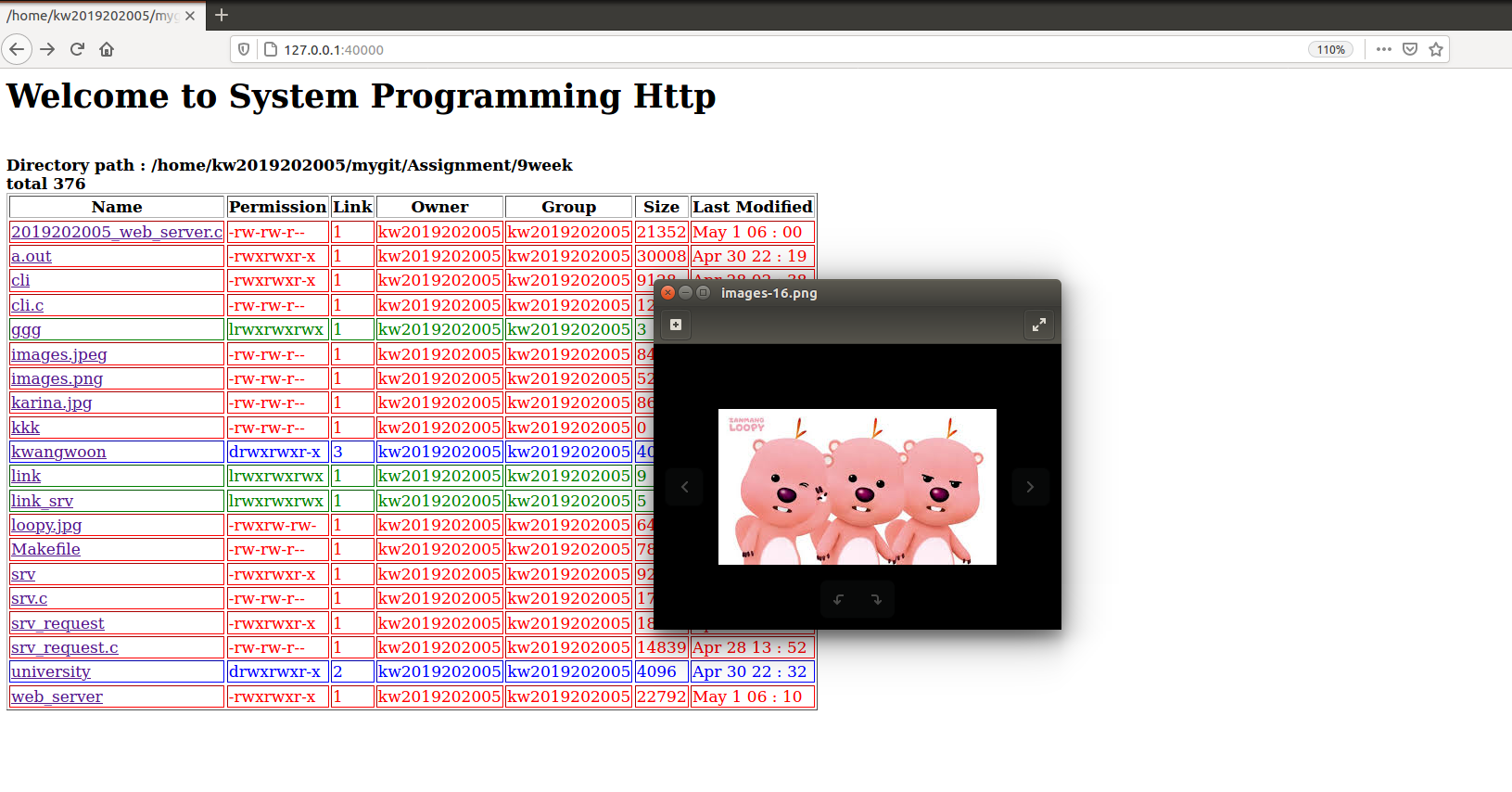
다음은 이미지 .jpg파일이 출력되는 화면입니다.



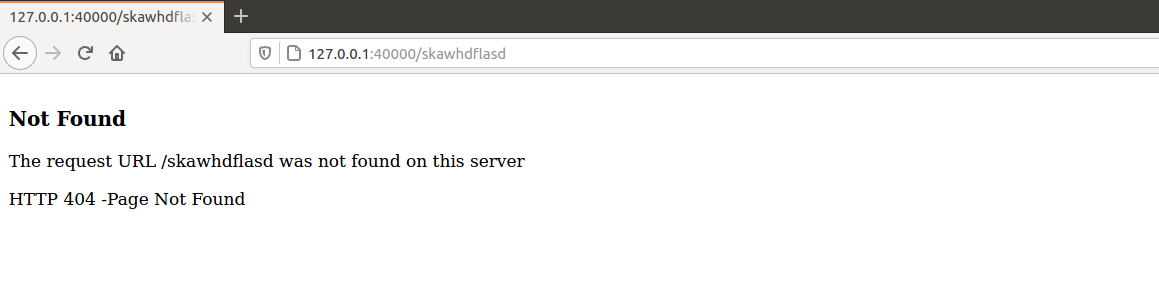


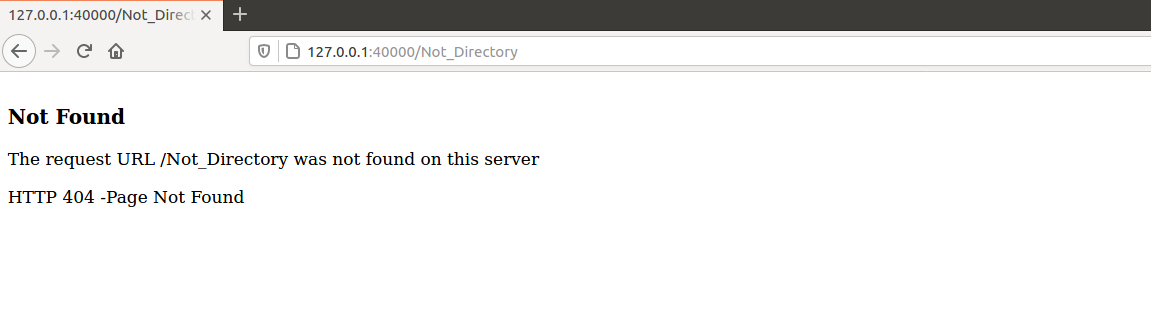
다음은 .jpeg파일이 클릭 시 출력되는 화면입니다.





다음은 .png파일 클릭 시 출력되는 화면입니다.





다음은 존재하지 않는 디렉토리 URL입력 시 출력되는 화면입니다.

**고찰**

이번 과제에서 제가 느끼기에는 이전 과제들과 진행되는 방식이 많이 다르다고 느껴서 처음에 접근하기에 조금 어려움이 있었습니다. 물론 시스템 프로그래밍 수업과 컴퓨터 네트워크 수업에서 배운 내용이지만 socket programming을 직접 해보는 것은 처음이어서 낯설었던 것 같습니다. 이전 과제를 진행하기 위해서는 직전 과제의 코드에서 덧붙이는 형식으로 진행했다면 이번 과제는 처음부터 다시 코드를 구현했습니다. 다만 클라이언트와 서버를 연결하기 위해 각각이 사용하는 함수를 이용하기 위해 실습 강의자료의 srv\_request.c파일의 내용을 참고하여 시작했습니다.

먼저 포트 번호는 40000을 사용했으며 ip 주소는 127.0.0.1을 사용했습니다.

실습 강의를 듣기 전에 과제를 진행하면서 서버 프로그램을 종료할 때 ctrl z를 사용했는데 이때마다 계속 bind함수에서 오류가 났는데 원인을 찾지 못해 과제를 진행할 수 없었습니다. 하지만 실습 강의를 수강 후 ctrl z를 사용해 프로그램 중지가 아닌 ctrl c를 사용해 프로그램을 강제종료해야 오류가 나지 않는다고 하셔서 그제서야 과제를 진행할 수 있었습니다.

또 오류가 있었는데 디렉토리 클릭 시 하위 디렉토리로 잘 넘어갔는데 파일과 이미지 파일 클릭 시 파일의 내용과 이미지가 출력되지 않았습니다. 아무것도 출력되지 않아서 계속 생각해보니 일단 파일을 열지도 않고 열지 않았으니 파일의 내용을 읽지 못해서 계속 없는 파일을 write함수를 호출하는 상태였습니다. 그래서 일단 fopen함수를 통해 파일을 open한 후 read함수를 통해 파일의 내용을 읽었습니다. 하지만 이래도 출력되지 않아 확인해보니 파일의 크기를 전달해주는 과정에서 오류가 있었습니다. 그래서 파일의 정보를 받아와서 파일의 크기를 계산해 이를 response\_header의 content-length로 넘겨줬습니다. 이를 통해 파일의 내용을 출력 문제는 해결하였지만 이미지 파일이 계속 출력되지 않았습니다. 이 문제에서 가장 시간을 많이 소요했던 것 같습니다.

제가 response\_message를 전달하는 과정에서 temp를 전역변수로 선언 후 temp에다가 모든 내용을 받아와서 main함수에서 write하기 직전에 strcpy함수를 통해 temp에 있는 내용을 response\_message에 전달 후 write함수를 진행했습니다. 이때 확인해보니 이미지 파일에 있어서는 파일의 내용을 복사하는 과정에서 제대로 복사되지 않는 점을 확인할 수 있었습니다. 그래서 이미지 파일인 경우와 이미지 파일이 아닌 경우 디렉토리인 경우 이 세가지의 경우를 나누어 이미지 파일인 경우에는 temp에 담아 복사하는 과정 없이 바로 response\_message에 담아 이를 출력해주었습니다. 이런 과정을 통해 이미지 파일이 오류없이 전달되어 잘 출력할 수 있었습니다.

그리고 server application실행 후 root directory에서 아무것도 하지 않았는데 client의 접속 요청을 sever가 수락하여 segment fault가 발생하는 상황이 있었습니다. 자세히 확인해보니 하위 파일 및 디렉토리를 클릭하기도 전에 하이퍼링크에 마우스 커서를 올려 두기만 해도 접속 요청이 수락되었습니다. 하지만 사실은 client쪽에서 아무 일도 하지 않았으니 계속 오류가 발생했습니다.

자세히 확인해보니 이는 firefox의 파일 및 디렉토리 하이퍼링크를 클릭하기 전에 미리보기 기능 때문이었습니다. read함수에서 반환 값은 읽어 들인 버퍼의 바이트 수인데 이때는 read함수가 0을 반환하여 계속 오류가 났었습니다. 그래서 continue를 이용해 read함수가 0을 반환하는 이 미리보기 기능이 활성화 되어있을 때 segment fault가 발생해 server application이 멈추는 현상을 예외처리 해주었습니다.

**Reference**

2023년 1학기 시스템프로그래밍 & 시스템 프로그래밍 실습 강의자료

Assignment 2-2

2023년 1학기 시스템 프로그래밍 1학기 강의자료 3. Files and directories

2023년 1학기 시스템 프로그래밍 1학기 강의자료 6. sockets

2023년 1학기 시스템 프로그래밍 실습 9주차 강의자료 Basic web server