시스템프로그래밍
assignment 2-2

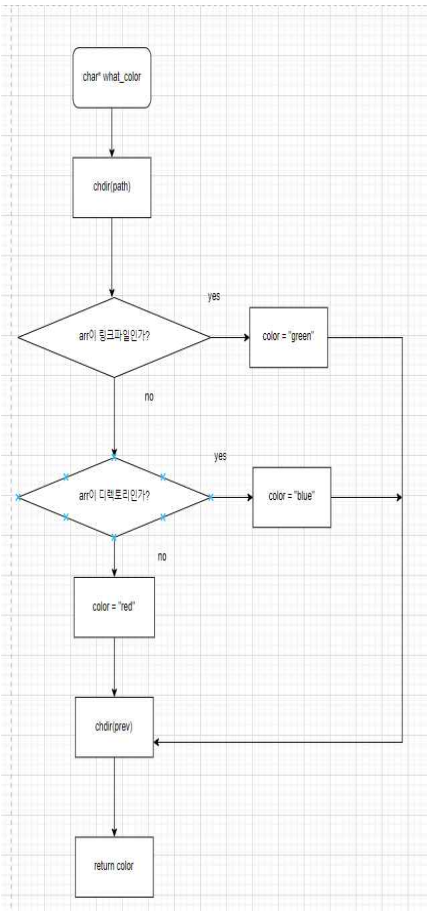
학번: 2021202045
이름: 김예은
담당 교수님: 최상호 교수님

Ⅰ. Introduction

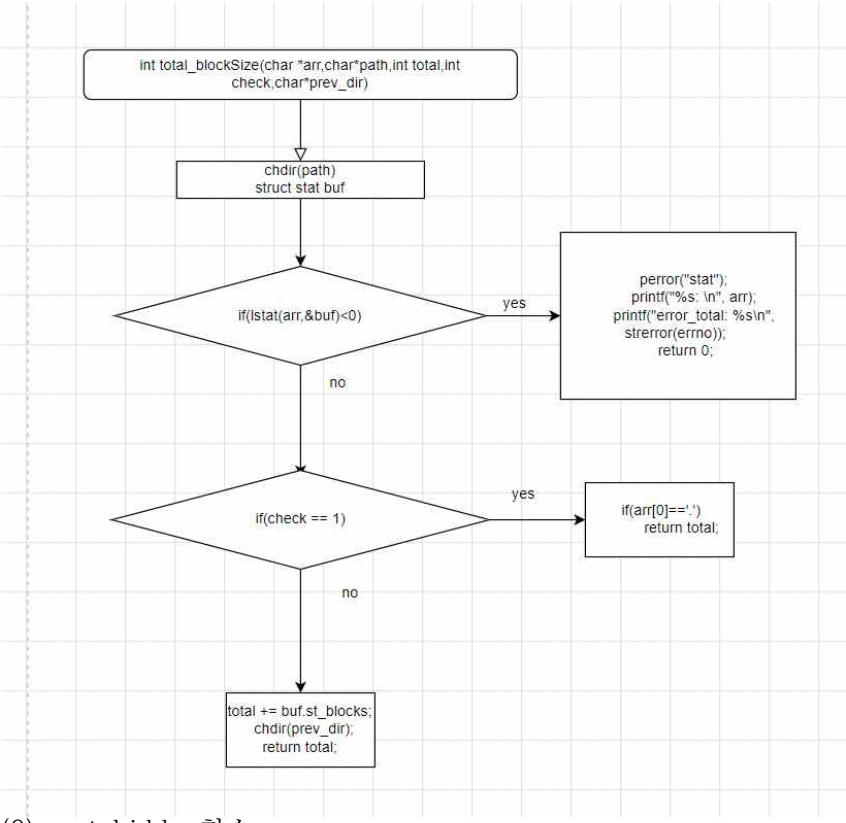
이번 과제는 root directory냐, 하위 directory냐에 따라 -l 옵션과 -al 옵션을 사용하여 result를 terminal대신 request를 요청한 client server에 display 해주는 과제이다. root directory에 접근할 때와 하위 directory에 접근할 시 다른 문구가 맨 위에 뜨게 한다. 이미지 파일, 디렉토리, 그 외 파일들을 분류하여 해당 entry를 누를 때 적절한 result(또는 download)가 표시되도록 한다.

Ⅱ.flowchart

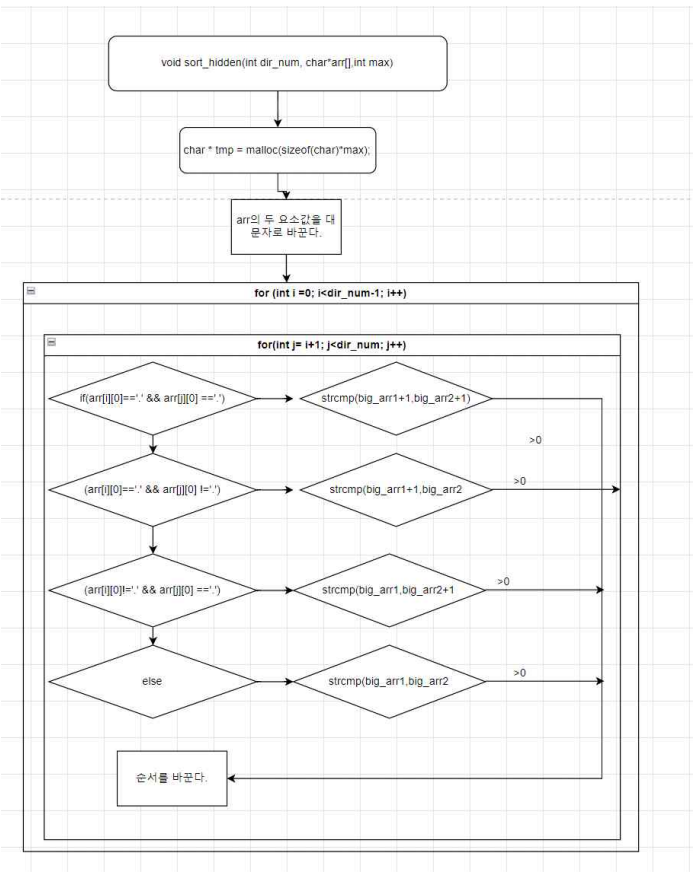
(1) what_color함수



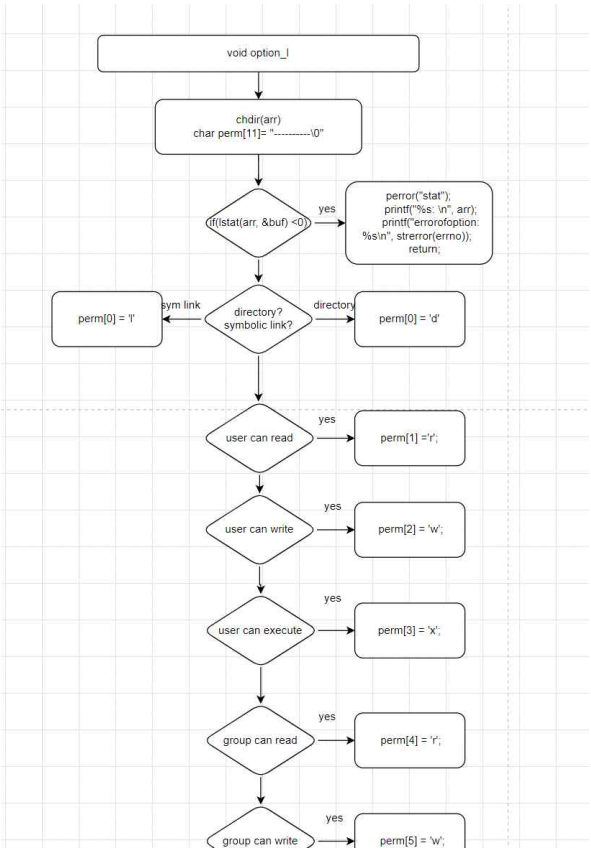
(2)total_blockSize함수

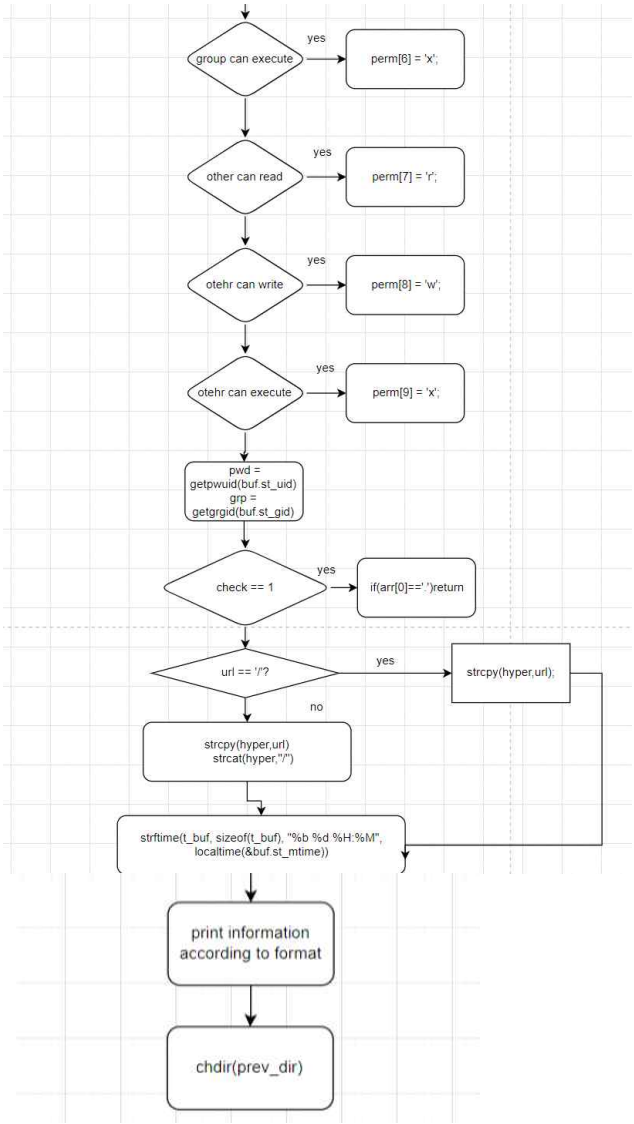


(3) sort_hidden함수

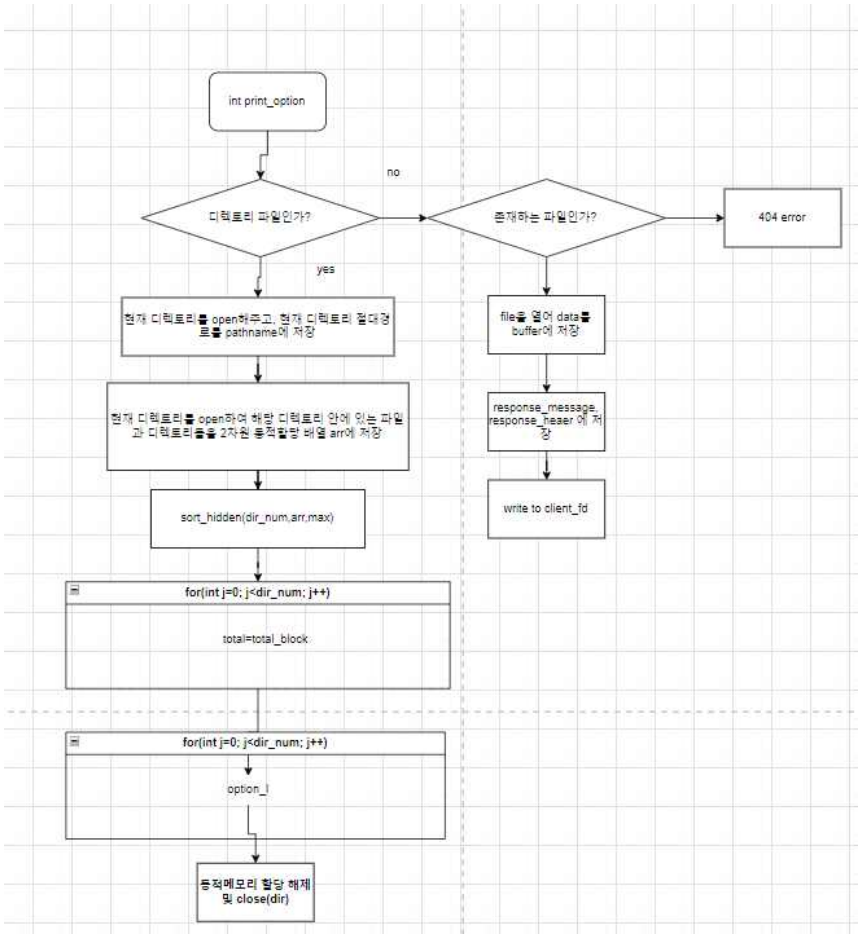


(4) option_l함수

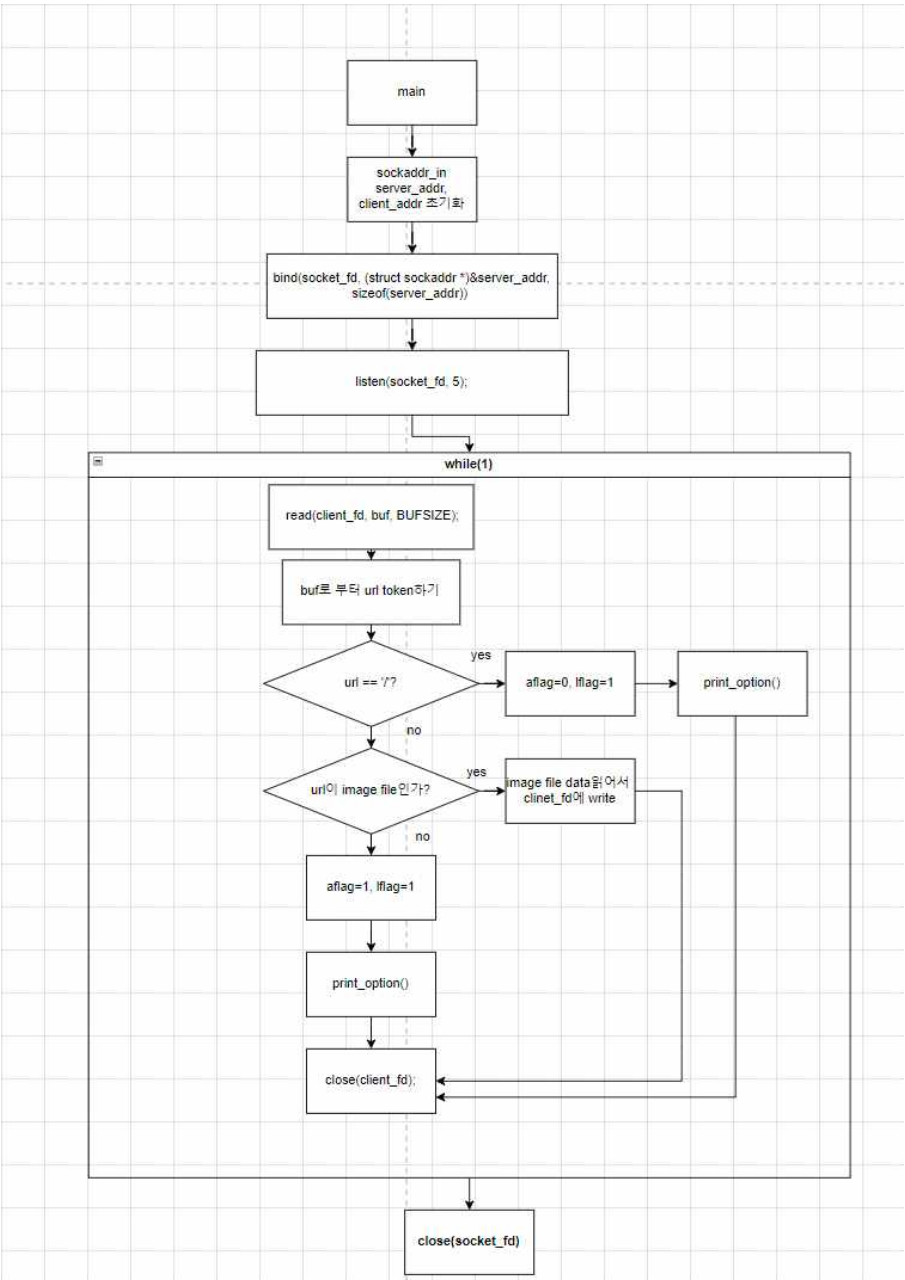

void option_l
chdir(arr)
char perm[11]= "----------\0"
if(lstat(arr, &buf) <0)
yes
perror("stat");
printf("%s: \n", arr);
printf("errorofoption:
%s\n", strerror(errno));
return;
directory?
symbolic link?
sym link
perm[0] = 'l'
directory
perm[0] = 'd'
user can read
yes
perm[1] ='r';
user can write
yes
perm[2] = 'w';
user can execute
yes
perm[3] = 'x';
group can read
yes
perm[4] = 'r';
group can write
yes
perm[5] = 'w';



(5)print_option함수


int print_option
no
디렉토리 파일인가?
존재하는 파일인가?
404 error
yes
현재 디렉토리를 open해주고, 현재 디렉토리 절대경로를 pathname에 저장
file을 열어 data를 buffer에 저장
현재 디렉토리를 open하여 해당 디렉토리 안에 있는 파일과 디렉토리들을 2차원 동적할당 배열 arr에 저장
response_message, response_heaer 에 저장
sort_hidden(dir_num,arr,max)
write to client_fd
for(int j=0; j<dir_num; j++)
total=total_block
for(int j=0; j<dir_num; j++)
option_l
동적메모리 할당 해제 및 close(dir)

(6)main함수


main
sockaddr_in server_addr, client_addr 초기화
bind(socket_fd, (struct sockaddr *)&server_addr, sizeof(server_addr))
listen(socket_fd, 5);
while(1)
read(client_fd, buf, BUFSIZE);
buf로 부터 url token하기
url == '/'?
yes
aflag=0, lflag=1
print_option()
no
url이 image file인가?
yes
image file data읽어서 clinet_fd에 write
no
aflag=1, lflag=1
print_option()
close(client_fd);
close(socket_fd)

Ⅲ. Pseudo code

int total_blockSize는 디렉토리나 파일의 총 블록 사이즈를 구하는 함수이다.
인자로 받은 path로 우선 이동한다. 그 다음, stat구조체 변수 buf를 선언해준다.
stat 인자로는 인자로 받은 디렉토리명과 buf를 넣어준다.
만약 lstat에 문제가 생길 시, perror와 errno를 통해 오류 메세지를 출력해준다.
함수 인자로 받은 check는 숨김 파일을 고려해야할지 안해야할지를 알려주는 signal로
만약 check == 1이라면,
    숨김파일일 때, 블록 사이즈를 더하지 않고 함수를 끝낸다.
아니라면,
    함수인자로 받은 기존 total를 업데이트 해준다.
이전 경로를 함수인자로 받았으므로 이전경로(prev_dir)로 다시 나가주고, total을 반환하면서 함수를 끝낸다.

void option_l함수는 인자로 받은 파일이나 디렉토리의 정보를 구한 후, 출력해준다.
permission에 대한 정보를 저장하기 위해 perm[11]을 선언 및 맨 처음엔 모두 '-'로 초기화 해준다.
구조체 stat 변수 buf를 정의한다.
user name, group name을 출력하기 위한 passwd, group 구조체 변수 pwd, grp도 각각 선언한다.
인자로 받은 path로 이동한다.
만약 lstat에 대한 에러가 발생할 때, 에러 메세지를 출력해준다.
buf에 저장된 파일이나 디렉토리의 타입을 분석하여 perm[0]에 표시한다.
user, group, other에 대한 접근 권한 총 9가지에 대해 if문을 통해 분석하고,
해당 접근권한이 있을 시 각 자리에 맞게 r,w,x를 써준다.
pwd를 통해 user id를 가져오고, grp를 이용해 group id를 가져온다.
t_buf에 strftime을 통해 time을 localtime으로 바꿔준후, format형으로 저장해준다.
만약 함수인자로 받은 check가 1이라면 파일명이 '.'으로 시작하는 것은 출력을 무시한다.
hyper link를 만들어준다.
만약 인자로 받은 url 마지막이 '/'로 끝난다면, '\0'을 넣는다.
hyper에 '/'를 넣는다.
hyper에 url을 추가한다.
hyper에 arr을 추가한다.
하이퍼링크를 걸어 response_message를 갱신한다.
접근권한,link수, user name, group name,size,수정시간,파일명을 '\t'로 구분하여 출력한 후,
prev_dir로 되돌아간다.

what_color함수
인자로 받은 path로 이동
인자로 받은 arr에 대해 lstat구조체에 넣는다.
만약 arr이 링크파일이라면

color는 green
만약 arr이 directory라면
color는 blue
만약 그외라면
color는 red
인자로 받은 prev로 이동
color를 return

print_option함수
open한 경로가 디렉토리가 아니라면{
 {존재하고 있는 파일이 아니라면
  response_header를 404 error를 출력하게끔 갱신한다.
  response_message를 에러 내용으로 갱신한다.}
 {존재하는 파일이라면
  해당 file을 open한다.
  response_header를 갱신한다.
  해당 file을 읽어 nread에 읽은 byte수를 저장하고 해당 내용을 response_message에 저장한다.
  response_message를 response_header에 붙여 쓴다.
  client_fd에 response_header(+response_message)를 쓴다.
}
}
open한 디렉토리에서 해당 디렉토리 안에 있는 파일과 디렉토리들을 2차원 동적할당 배열 arr에 저장한다.
디렉토리내의 entry 수들을 dir_num으로 정의하고, entry name중 가장 긴 이름을 max로 정의한다.
sort_hidden함수를 call한다.
그 다음 정렬된 entry들을 각각 total_blockSize()에 넣는다.
갱신된 total 변수를 response_message에 넣어 갱신한다.
반복문을 통해 모든 entry들을 option_l함수에 넣는다.

void sort_hidden함수는 '.'으로 시작되는 파일도 포함하여 파일명을 정렬해주는 함수이다.
bubble sorting을 써줄 것이기 때문에 임시 저장해줄 array tmp를 동적할당 해준다.
해당 array는
array[i]와 array[j]를 모두 대문자로 바꿔준후 각각 big_arr1, big_arr2에 저장한다.
big_arr1[i]와 big_arr2[j]를 비교할 때
숨김 파일과 숨김 파일일 때,
숨김 파일과 숨김 파일이 아닐때,
숨김파일이 아니고 숨김파일일때,
둘다 숨김 파일이 아닐때
총 4개의 경우의 수를 통해 먼저 '.'을 빼고 비교해준 뒤, 다시 '.'을 포함하여 정렬해준다.

만약 숨김 파일이라면 arr[i][0] == '.'이므로 arr[i]+1을 해주면 '.'이후의 문자열을 가리키는 것이다.
만약 arr[0]와 arr[1]의 위치가 바뀌어야 한다면, tmp에 arrp[0]값을 저장
arr[0]에 arr[1]값을 복사하고
arr[1]에는 tmp값을 복사하여 저장한다.

main함수
sockaddr_in 구조체 변수인 server_addr, client_addr를 초기화 해준다.
PF_INET,SOCK_STREAM type을 가진 socket을 만들어 해당 descriptor을 socket_fd에 저장한다.
server_addr과 해당 socket을 bind한다.
queue사이즈를 5로 하고, listen을 보낸다.
while(1)
{
socket_fd로부터 accept한 것을 client_fd에 저장한다.
client_fd로 부터 읽은 내용을 buf에 저장한다.
buf에 저장된 내용을 method와 url부분으로 token한다.
aflag =0
lflag=0
으로 초기화한다.

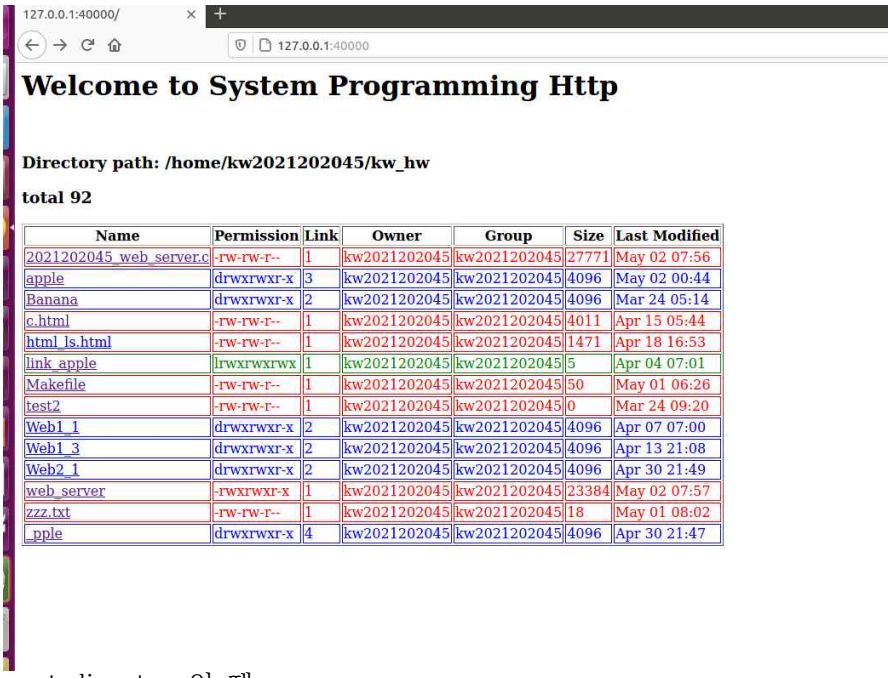
만약, url이 '/'이라면,
lflag=1로 update한다.
현재 디렉토리를 link_path에 저장한다.
response_header(type : text/html)와 response_message를 update한다.
print_option함수를 call한다.

만약 url이 이미지파일(확장자가 .jpg, .png, .jpeg)라면,
response_header는 type: image/*로 update한 후, client_fd에 send한다.
url을 file open한다.
파일 끝까지 읽을 때까지,
사진 데이터를 읽어 buffer에 저장한다.
이 값을 client_fd에 send한다.
파일을 닫는다.

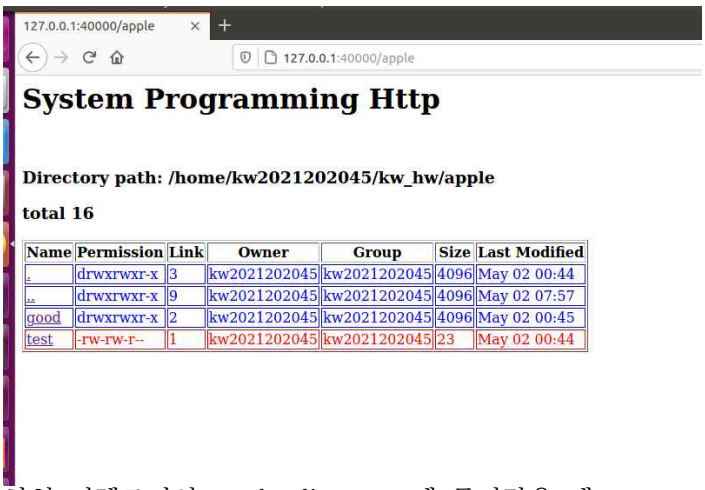
그외라면,
aflag=1, lflag=1로 update한다.
현재 디렉토리를 link_path, prev에 copy한다.
link_path에 url 부분을 추가한다.
response_header는 type : text/html로 갱신한다.
response_message를 갱신한다.

print_option함수를 call한다.
client_fd를 닫는다.
}
socket_fd를 닫는다.

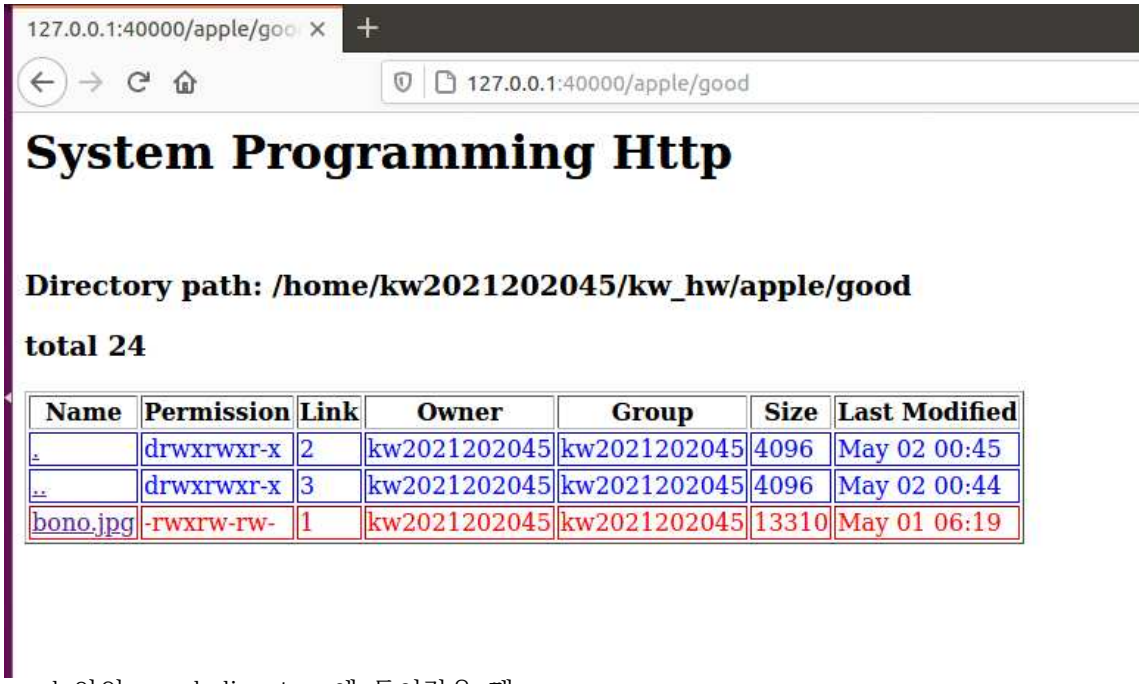
Ⅳ.결과화면



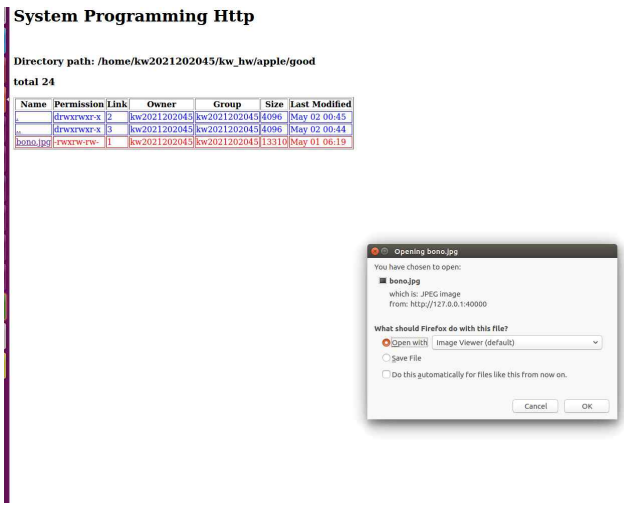
root directory일 때



하위 디렉토리인 apple directory에 들어갔을 때

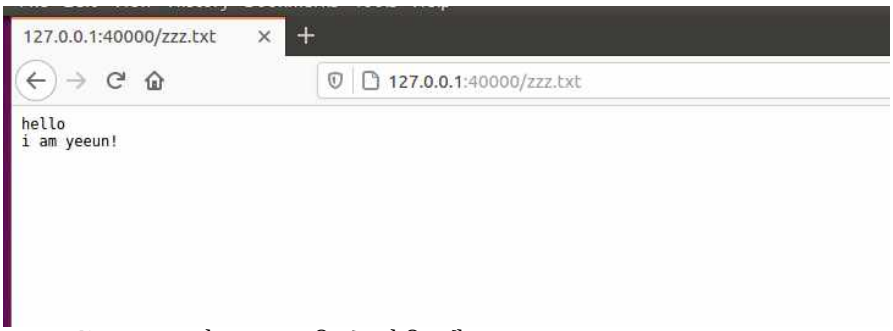


apple안의 good directory에 들어갔을 때

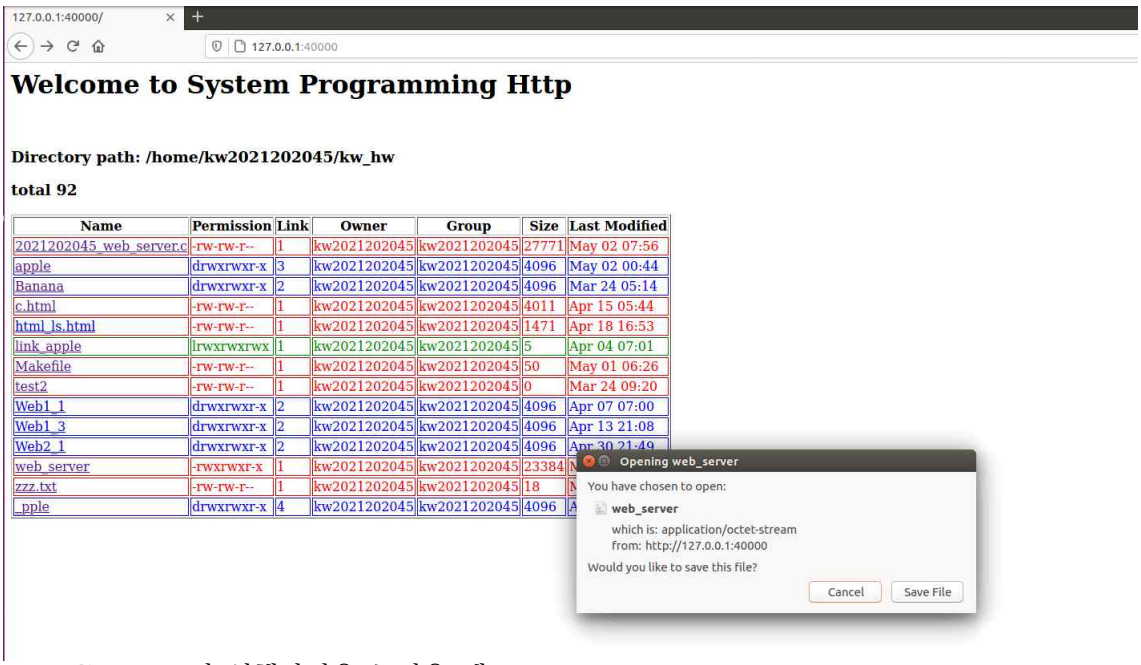


bono.jpg를 눌렀을 때

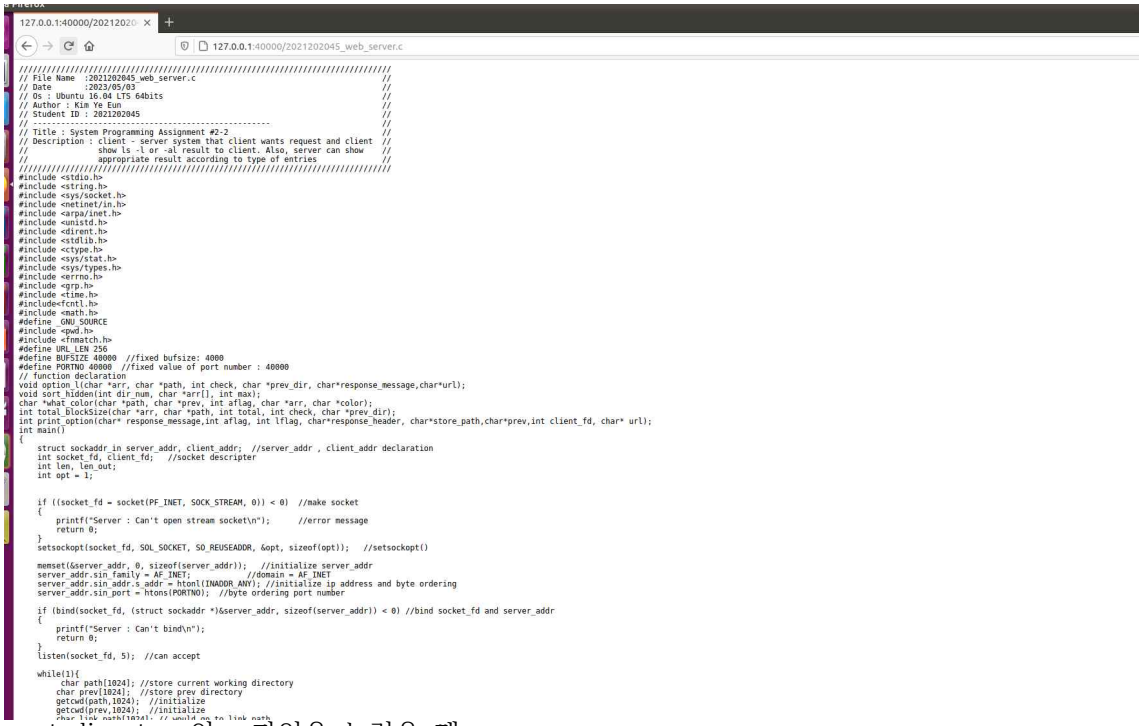
root directory의 c.html을 눌렀을 때



root directory의 zzz.txt을 눌렀을 때



root directory의 실행파일을 눌렀을 때



root directory의 .c파일을 눌렀을 때

없는 파일에 대해 접근할 때

Ⅵ.고찰

terminal로 request를 보내는 것이 아니라 web page를 열어서 server에 요청을 하고, 적절한 result를 직접 client web에 내보내는 것이 생소한 개념이라 이해하느라 시간이 걸렸다. 특히 response_header의 중요성을 깨닫지 못하고, html 태그가 문자열로 표시되어서 고치느라 시간이 걸렸고, 하위 디렉토리로 계속 들어가기 위해 url을 추가해주는 부분에서 '/'가 계속 추가 되어 이 부분을 고치느라 시간이 걸렸다. char point를 하나 선언하여 이 포인터가 url을 가리키게 하고 맨처음이나 마지막이 '/'가 있으면 무시할 수 있도록 pointer를 이동해주거나, 해당 index를 '\0'으로 바꾸게 하여 해결했다. response_message는 sprintf로 추가해줬는데 추가가 아니라 덮어씌우기를 해버려서 결국 response_tmp를 만들어 여기에 잠시 저장했다가 기존 response_message에 response_tmp를 strcat을 하는 방식으로 추가해주었다. 시간은 오래 걸렸지만 basic server를 구현하면서 실제로 web page작동 원리를 얕게나마 구현해볼 수 있어서 신기했다.