시스템 프로그래밍 실습 1-2 과제

이름 : 이준휘

학번 : 2018202046

교수 : 최상호 교수님

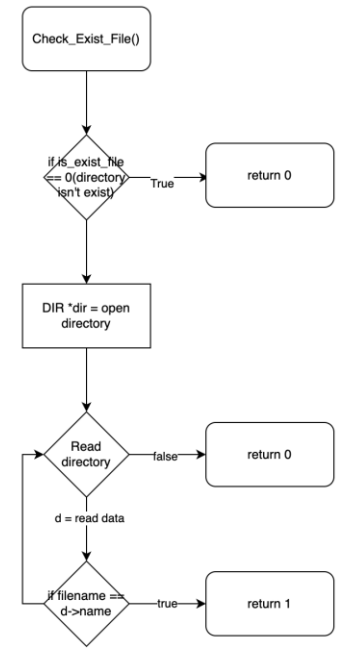
강의 시간 : 화

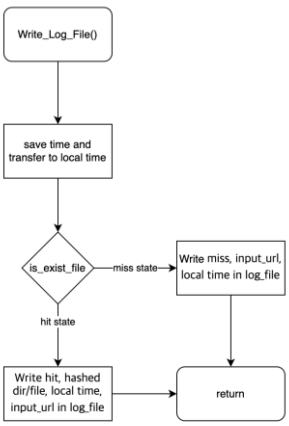
실습 분반 : 목 7, 8

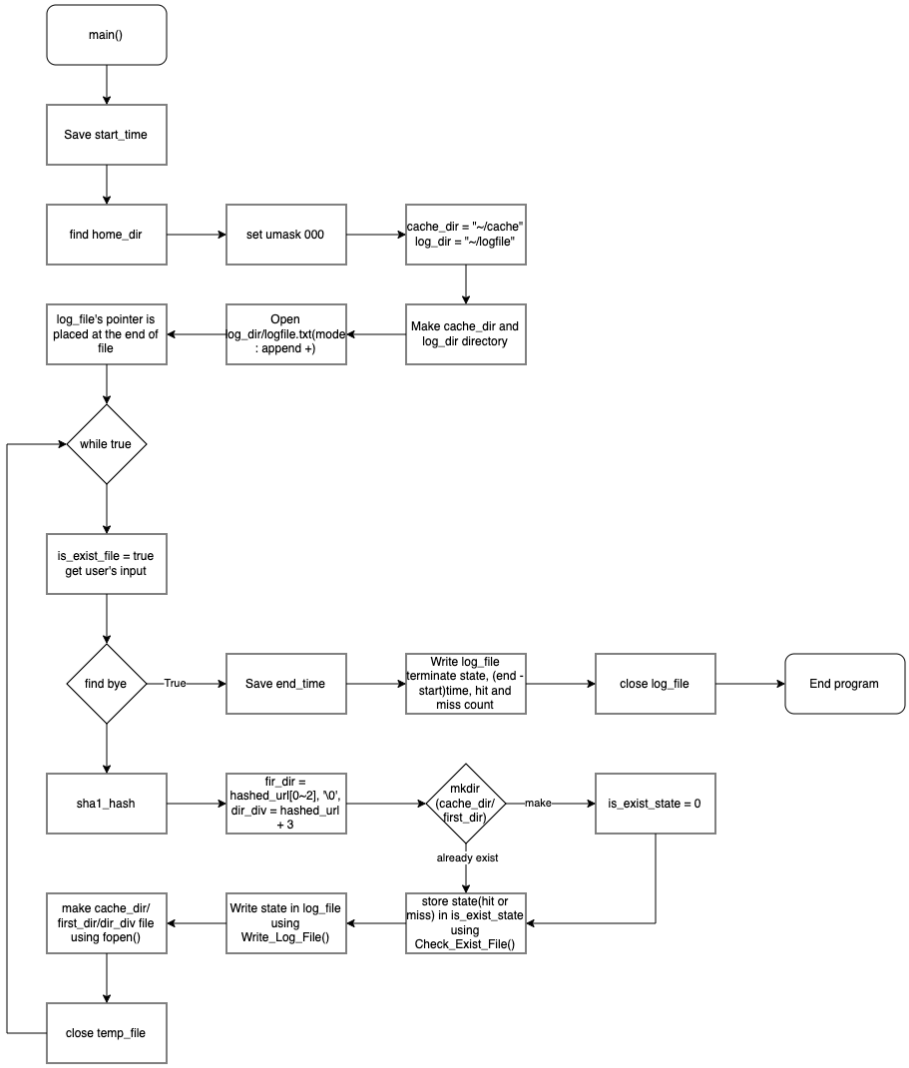
1. Introduction

해당 과제는 1-1 과제에 덧붙어서 진행된다. 프로그램을 실행하였을 때 /logfile/logfile.txt를 생성하여 이후의 작업에 대한 내용을 기록한다. 만약 기존에 cache에 존재하는 URL을 입력받은 경우 logfile.txt에 hit와 해당 hashed URL의 위치, 시간, 원본 URL을 기록한다. 만약 기존에 존재하지 않은 URL을 입력받은 경우 logfile.txt에 miss와 입력받은 URL, 시간을 입력한다. 이후에 작업은 1-1과 동일하다. 프로그램이 종료하는 경우에는 프로그램 작동 시간과 hit와 miss를 카운트한 수를 logfile.txt에 입력하는 과정을 추가하고 종료한다.

2. Flow Chart







3. Pseudo Code

```
Check_Exist_File(char *path, char *file_name, int is_exist_file){
    if(directory isn't exist)
        return 0;

    DIR *dir = Open path directory
    While(struct dirent *d = Read path directory){
        If(d->name == file name)
            Close directory and return 1;
```

```
    }
    Close directory and return 0;
}

Void Write_Log_File(File *log_file, char *input_url,
char *hashed_url_dir, char* hashed_url_file, int is_exist_file){
    time_t now;
    struct tm *ltp;

    ltp = current local time;
    if(miss state)
        Write miss, input_url, local time in log_file;
    Else
        Write hit, hashed dir/file, local time, input_url in log_file;
}

main(void){

    char[100] input, home_dir, cache_dir, log_dir, temp_dir;
    char[60] hashed_url;
    char[4] first_dir;
    char *dir_div;
    int is_exist_file, hit = 0, miss = 0;
    FILE *temp_file, *log_file;
    time_t start_time, end_time;
```

```
Save start time;


home_dir = ~;
cache_dir = ~/cache;
log_dir = ~/logfile;
set umask 000;
make cache and log directory(permission = drwxrwxrwx);
temp_dir = ~/logfile/logfile/txt;
make log file(mode : a+);
log_file's pointer is placed in the end of file;


while(true){
   is_exist_file = 1;
   input = User's input;
   if(input is 'bye'){
      Save end time;
      Write end – start time, hit, miss count in logfile.txt;
      End program;
   }
   hashed_url = hashed input using sha1;
   first_dir = { hashed_url[0~3], '₩0' };
   dir_div = hashed_url + 3 address
   temp_dir = ~/cache/first_dir;
   if make temp_dir directory(permission = drwxrwxrwx)
      is_exist_file = 0;
   is_exist_file = hit or miss state;
```

Write state, url, dir/file name, time in logfile.txt;

temp_dir = ~/cache/first_dir/dir_div;

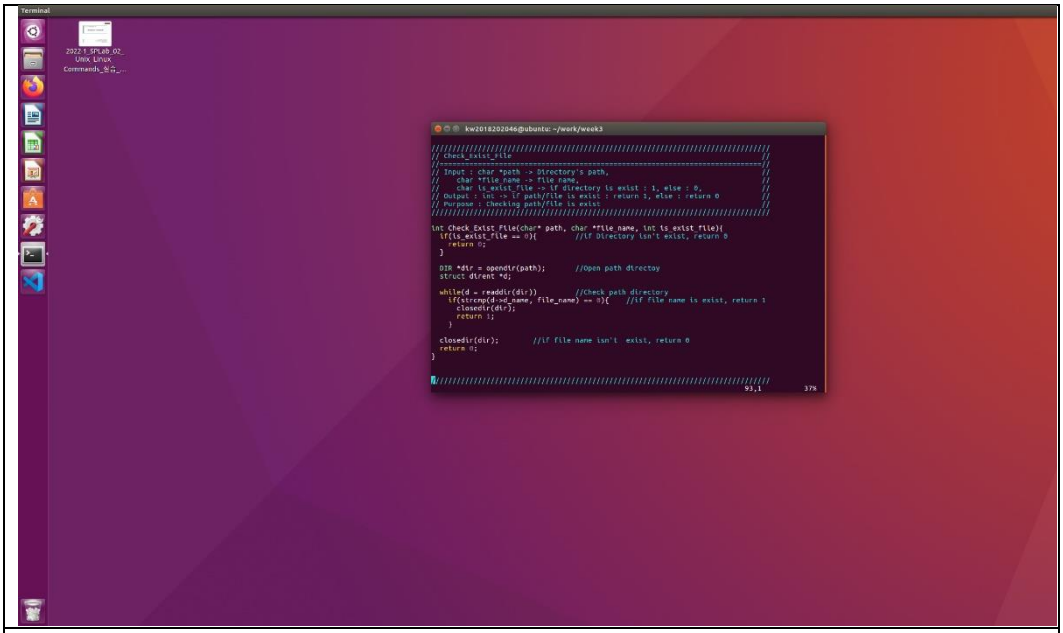
temp_file = make temp_dir file and open file;

temp_file close;

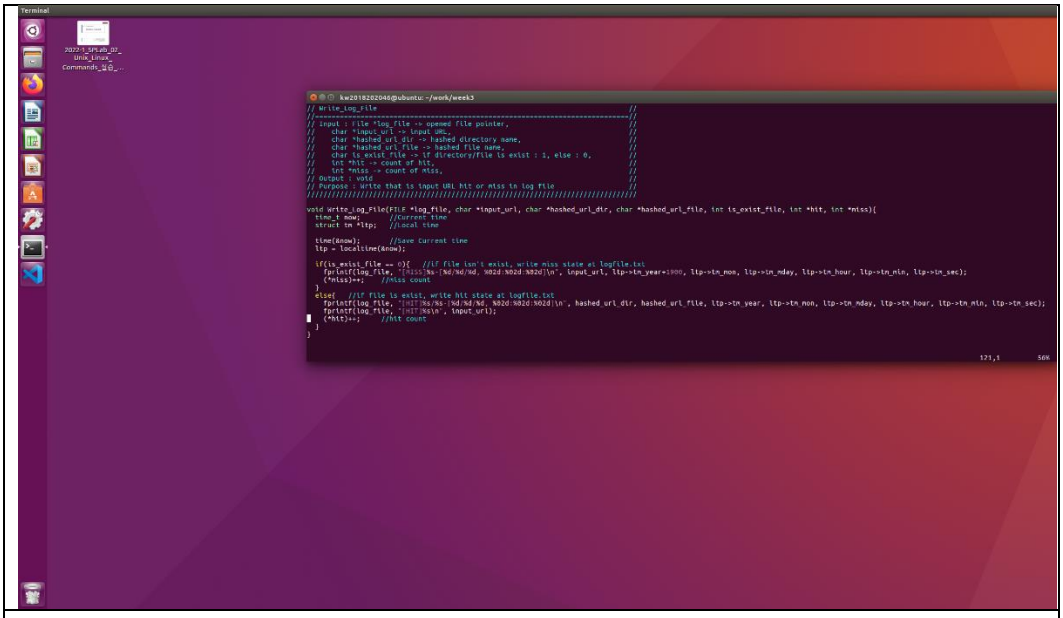
}

}

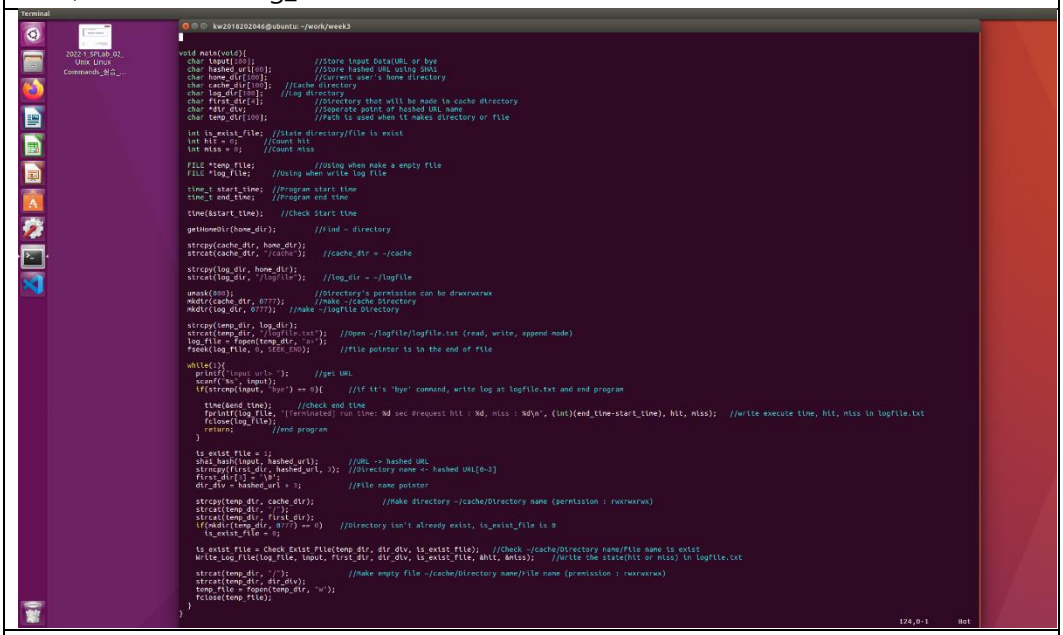
4. 결과 화면



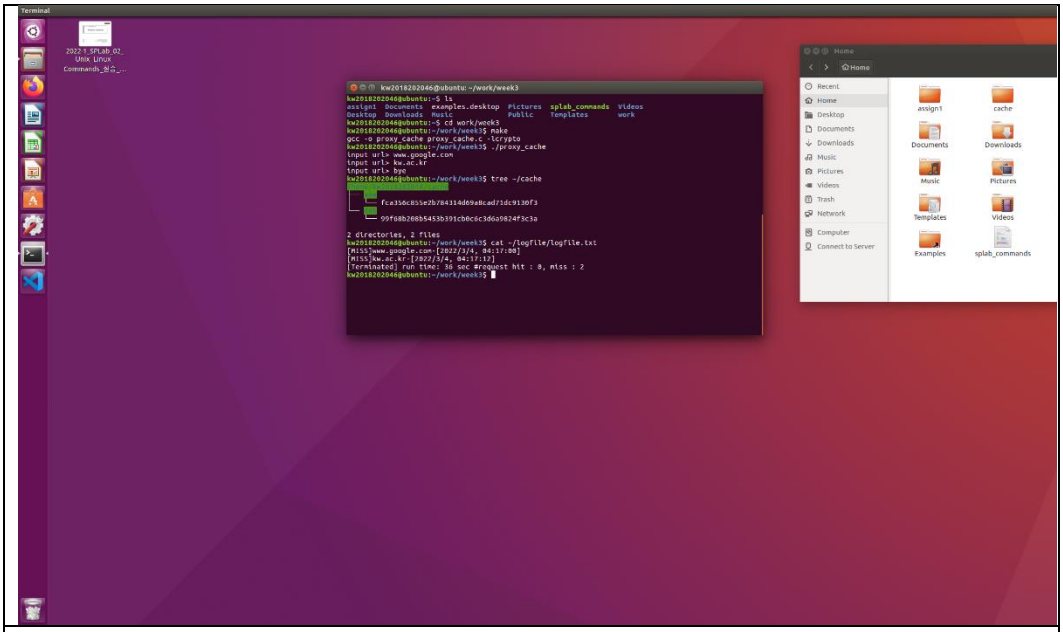
해당 함수는 1-2에서 추가된 Check_Exist_File() 함수다. 이 함수를 통해 miss(0)와 hit(1) 상태를 판별할 수 있다. 해당 함수는 만약 directory를 생성한 후 불러온 경우(is_exist_file == 1) miss를 반환한다. 이후에 opendir() 함수와 readdir()함수를 통해 directory를 탐색하며 같은 이름이 있을 경우 hit를 반환한다. 만약 찾지 못한 경우에는 miss를 반환한다.



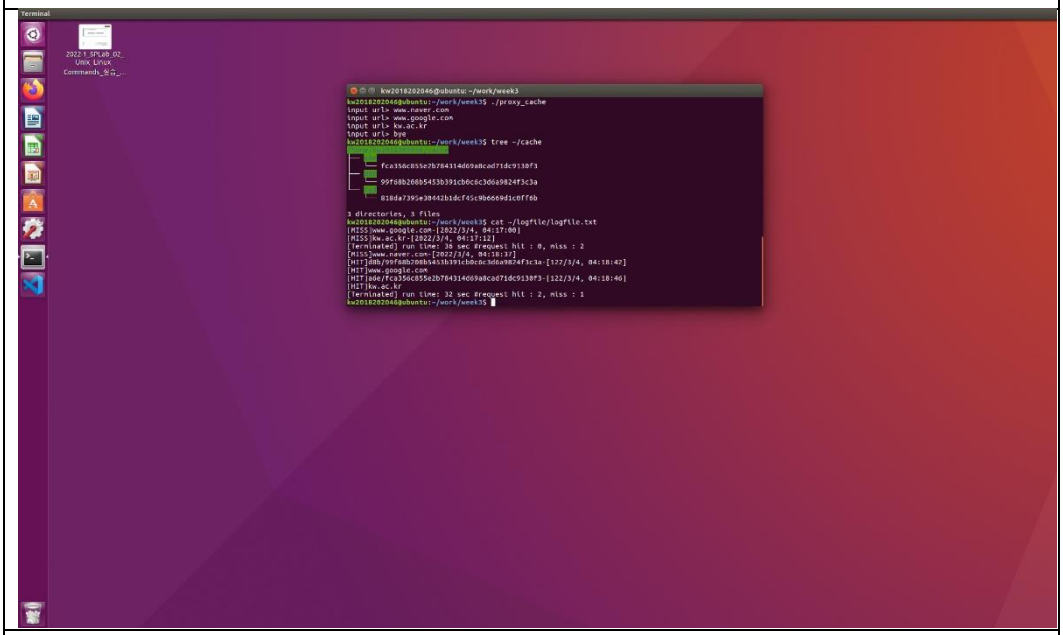
해당 함수 또한 1-2에서 추가된 함수로 logfile에 지정된 출력을 입력하는 역할을 수행한다. 우선 현재 local 시간을 기록하여 저장한다. 그 후 is_exist_file이 miss(0)인 경우 miss 상태와 함께 URL과 시간을 log_file에 출력하고 miss 변수를 1 증가시킨다. 만약 hit(1) 상태일 경우에는 hit 상태와 함께 hashed된 디렉토리와 파일명, 시간, 원본 URL를 log_file에 출력하고 hit 변수를 1 증가시킨다.



위의 사진은 메인 함수를 나타낸 것이다. 이전 1-1과 점을 설명하겠다. 우선 ~/logfile/logfile.txt를 생성하여 로그파일을 작성하는 점이다. 그리고 다음으로는 프로그램 시작 시 시간을 기록하고 종료 시 시간을 기록하여 프로그램이 얼마나 동작했는지, 그리고 hit와 miss가 몇 번 일어났는지 logfile.txt에 기록한다는 점이다. 그리고 마지막으로 다른 점은 hashed directory 생성 시 만들어지지 않은 경우 0으로 상태를 기록하고 Check_Exist_File() 함수를 통해 현재 hit인지 miss인지를 판별하여 이를 Write_Log_File() 함수를 통해 기록하는 점이다.



해당 그림은 기존에 cache와 logfile directory가 없는 상태에서 실행하여 2개의 인자를 입력한 모습이다. 해당 그림에서 볼 수 있다시피 기존처럼 cache폴더와 파일은 정상적으로 만들어지고, logfile/logfile.txt 파일이 정상적으로 만들어졌으며, 해당 내용 또한 miss 2개로 잘 기록된 모습을 볼 수 있다. 시간의 출력 포맷도 조건에 맞게 출력되고 있다.



해당 그림은 한번 더 프로그램을 동작시키는 모습이다. 해당 그림에서 볼 수 있다시피 기존에 존재하는 URL을 2개 입력하고 새로운 URL을 1개 입력하였다. 이 때 hit는 2개, miss는 1개로 정상적으로 logfile.txt에 작성되었고, cache 폴더와 파일 또한 정상적으로 작동한 것을 알 수 있다.

5. 고찰

해당 과제를 통해서 opendir() 함수를 활용하여 directory을 열고 readdir() 함수를 통해 정보를 읽어와 dirent 구조체에 정보를 저장하는 부분에 대해 공부할 수 있었다. 또한 이를 통해 proxy server가 hit와 miss를 판단할 수 있도록 하는 메커니즘을 이해할 수 있었다. 그리고 기존에 단순하게 난수의 시드 설정에만 사용하였던 time_t를 localtime()함수와 tm 구조체를 통해 현지 시간대로 변환하여 사용하면서 시간을 활용할 수 있는 다양한 방법을 익힐 수 있었다.

6. Reference

강의 자료만을 참고