1-2

금3,4

과목명	시스템프로그래밍
담당교수	김태석 교수님
학과	컴퓨터정보공학부
학년	3학년
학번	2019202009
이름	서여지
제출일	21.04.06(화)

KWANGWOON UNIVERSITY

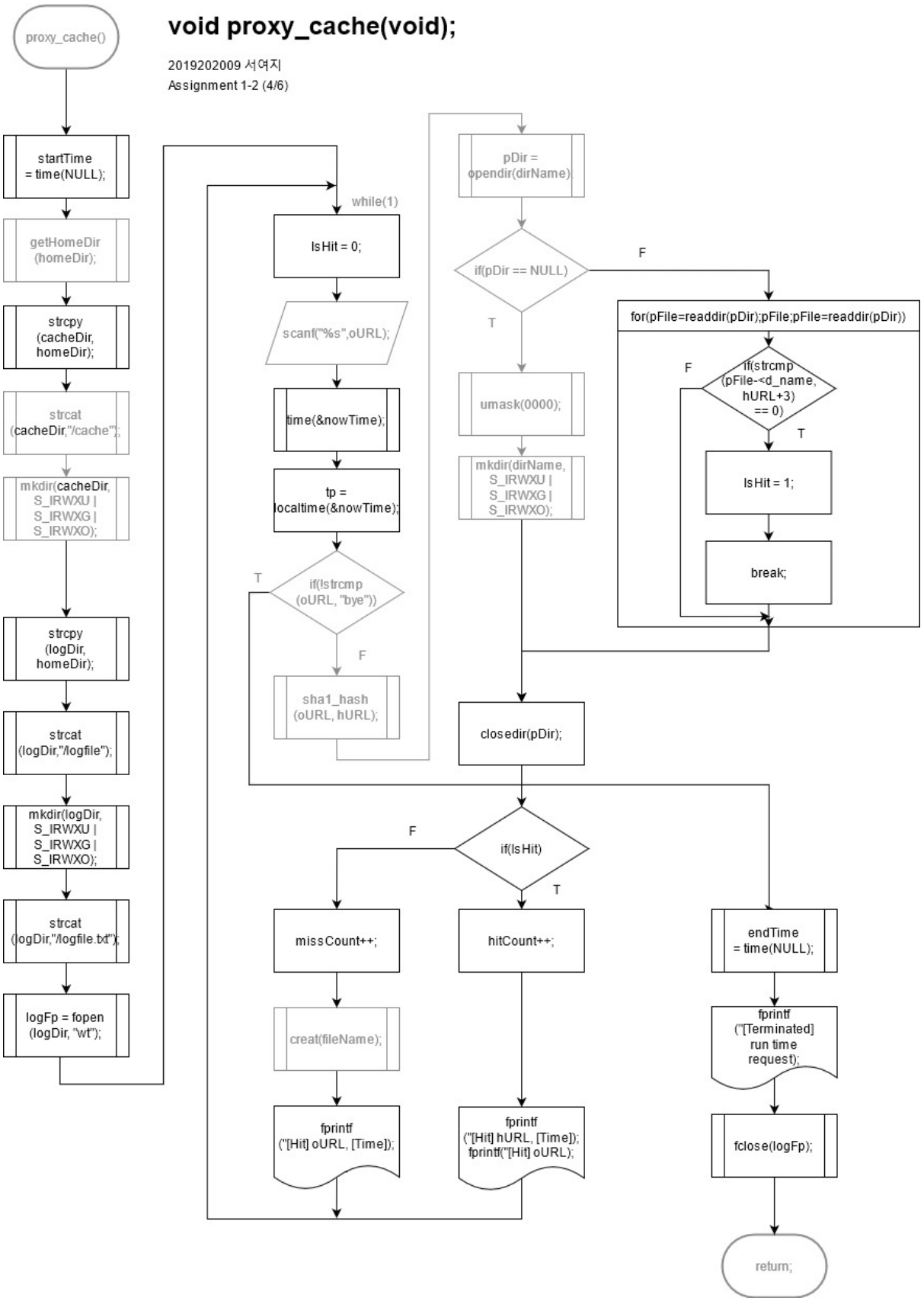
1. Introduction //과제 소개 – 5줄 내외(background 제외)

   저번 실습에서 구현한 Assignment 1-1에 logfile.txt를 생성하는 동작을 추가한다. logfile.txt는 cache의 HIT 혹은 MISS 여부와 입력한 URL, hash 결과, 요청한 시간에 대한 정보를 출력한다. 이때 HIT 혹은 MISS는 프로그램에서 hash결과 파일을 저장하는 cache directory를 읽어서 판별한다. 프로그램을 종료할 때 logfile의 마지막에 프로그램의 실행시간과 HIT, MISS의 수를 각각 기록한다.

2. Flow Chart //코드 작성 순서도

void proxy_cache(void);

2019202009 서여지
Assignment 1-2 (4/6)



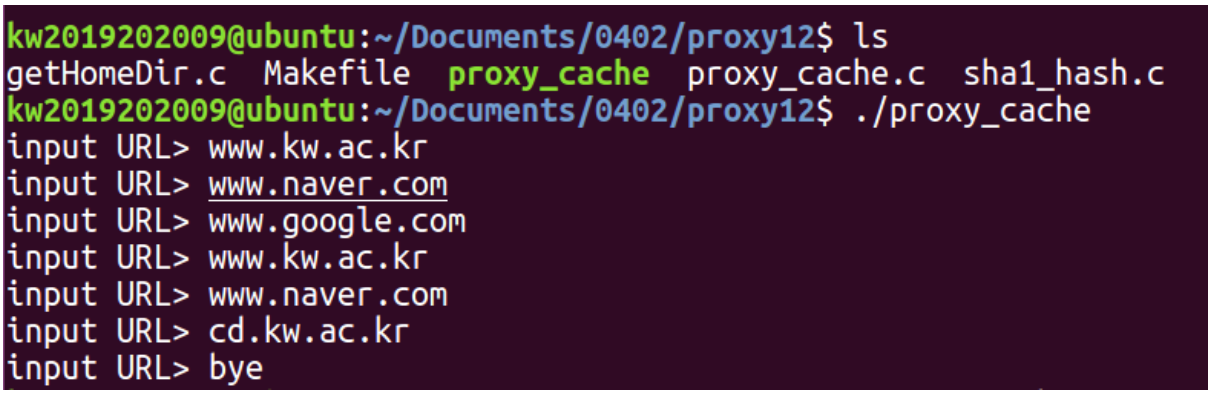
3. Pseudo code //알고리즘

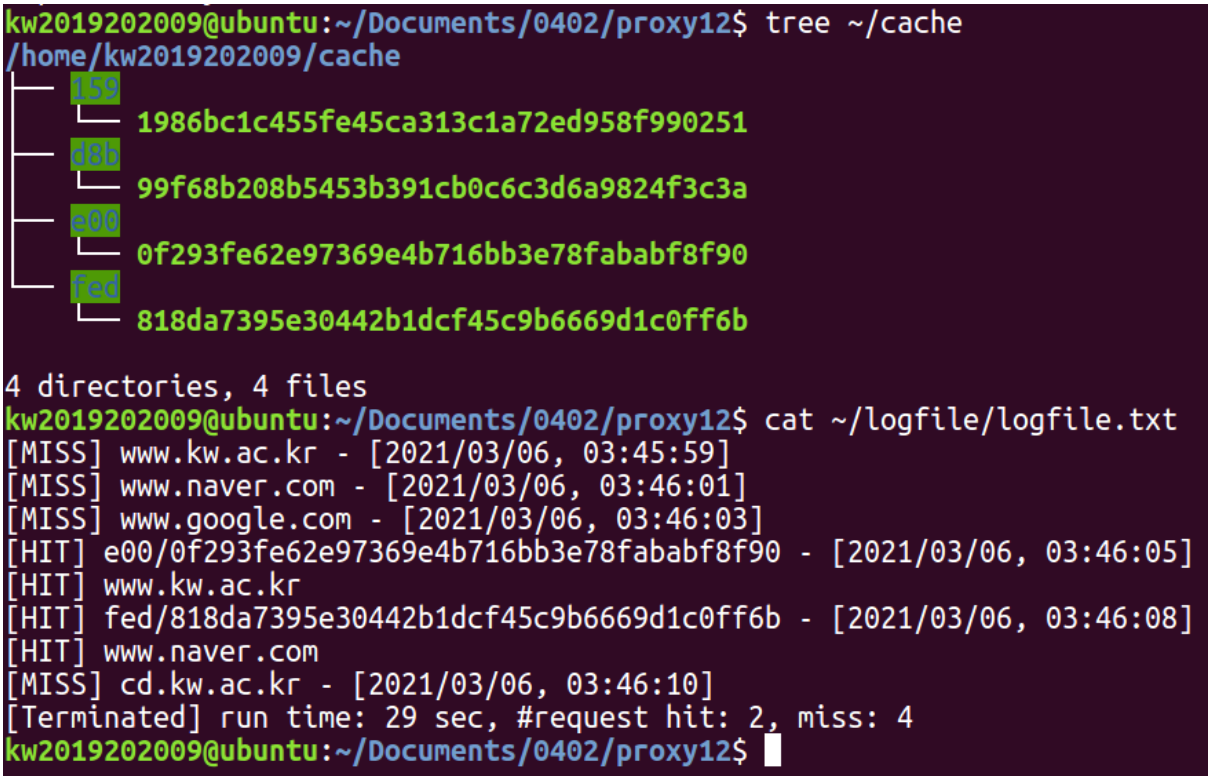
```
void proxy_cache(void){
    char oURL[40], hURL[41], dirName[45], fileName[81];
    char homeDir[40], cacheDir[40], logDir[40];
    DIR *pDir;
    struch dirent *pFile;
    int IsHit = 0, hitCount = 0, miscount = 0;
    FILE *logFp;
    time_t startTime, endTime, nowTimel
    struct tm *tp

    startTime = time(NULL);      //get start time
    getHomeDir(homeDir);      //get home dir

    cacheDir = homeDir + "/cache";
    mkdir(cacheDir, 0777);      //make cache directory
    cacheDir += "/";

    logDir = homeDir + "/logfile");
    mkdir(logDir,0777);      //make logfile directory
    logDir += "/logfile.txt";
    logFp = fopen(logDir, "wt");      //make logfile.txt

    while(1) {
        IsHit = 0;
        printf("input URL> ");     //get URL
        scanf("%s", oURL);

        time(&nowTime);      //get time
        tp = localtime(&nowTime);

        if (oURL == "bye") break;

        sha1_hash(oURL, hURL);
        dirName = cacheDir + hURL[0:2];      //get cache dir name
        filename = dirName + "/" + hURL[2:];      //get cache file name

        pDir = opendir(dirName);
        if (pDir == NULL) {
            umask(0000);
            mkdir(dirName, 0777);      //make cache dir
        }
        else {
            for (pFile=readdir(pDir); pDir; end of pDir) {
                if(pFile->d_name == hURL+3) {
                    IsHit = 1;
                    break;
                }
            }
        }
        closedir(pDir);
```

```
        if (IsHit) {      //HIT
            hitCount++;
            fprintf(logFp, "[HIT] hURL - [TIME]₩n");
            fprintf(logFp, "[HIT] oURL₩n);
        }
        else {      //MISS
            missCount++
            create(fileName);      //make cache file
            fprintf(logFp, "[MISS] oURL - [TIME]₩n");
        }
    }     //while end

    endTime = time(NULL);
    fprintf(logFp, "[Terminated] run time, request count ₩n";
    fclose(logFp);

    return;
}
```

4. 결과화면 //수행한 내용을 캡쳐 및 설명

강의자료에 제시된 예시를 이용하여 결과를 확인하였다. 입력된 4종류의 cache가 각각의 cache file을 생성하였고, logfile이 정상적으로 출력된 것을 확인할 수 있다.





5. 결론 및 고찰

시간 정보를 얻는 다양한 함수와 time_t, tm을 사용하여 시간정보를 출력할 수 있었다. logfile을 생성하기 위해 directory를 생성하는 부분과 필요한 정보를 얻는 부분이 추가되었다. 또한 조건문을 이용하여 cache 파일을 항상 생성하던 구조에서 cache의 중복을 확인하고, IsHit flag를 이용하여 miss인 경우에만 cache 파일을 생성하는 구조로 변경하였다. 1-2 과제를 구현하는데 큰 어려움은 없었지만, proxy_cache 함수의 길이가 과도하게 길어진 것 같다. 다음 과제를 진행하며 함수를 분리하는 것이 좋을 것 같다.

6. reference //과제를 수행하면서 참고한 내용을 구체적으로 기록

강의자료 이용