1-3

금3,4

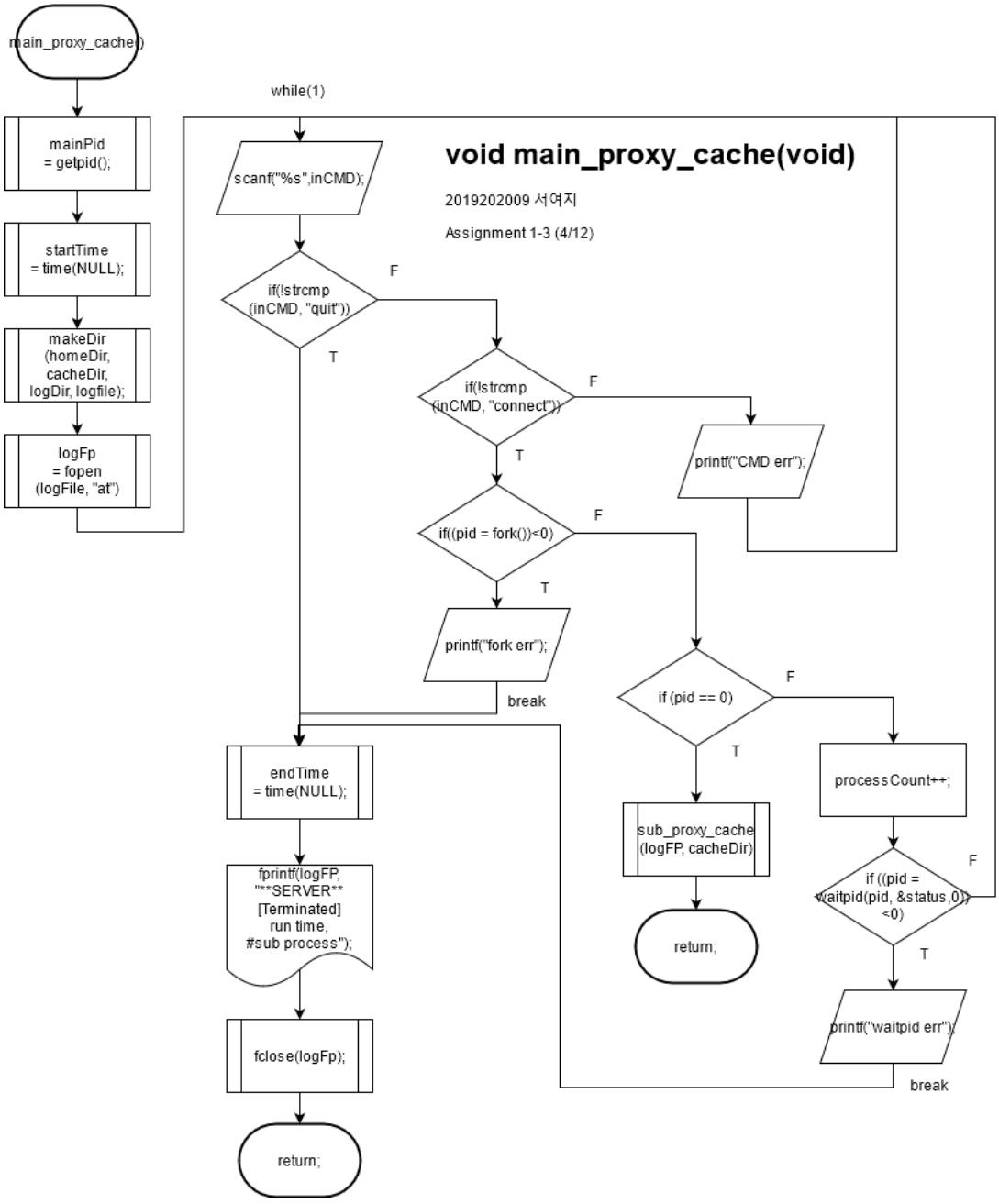
과목명	시스템프로그래밍
담당교수	김태석 교수님
학과	컴퓨터정보공학부
학년	3학년
학번	2019202009
이름	서여지
제출일	21.04.13(화)

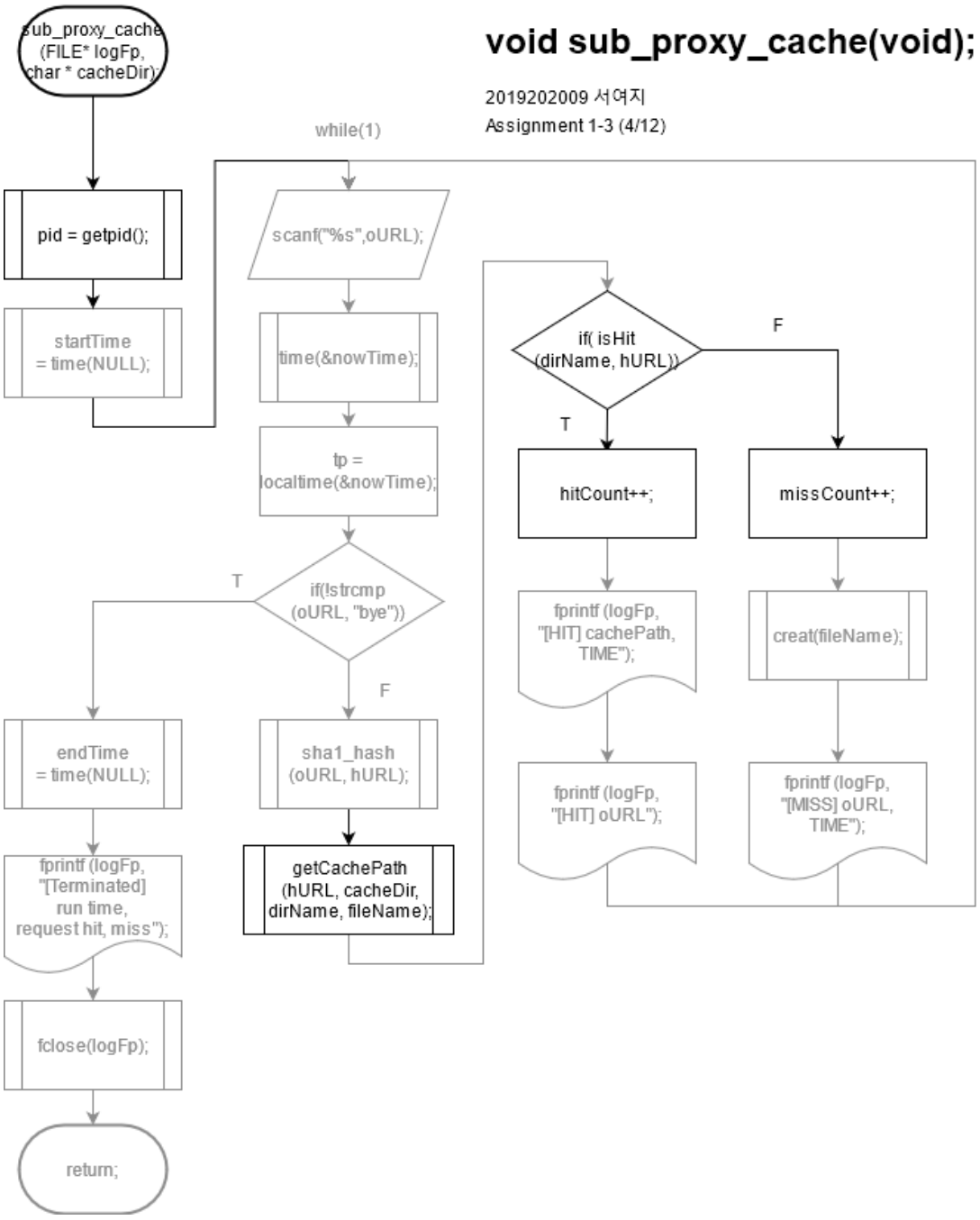
KWANGWOON UNIVERSITY

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

지난 실습에서 구현한 Assignment 1-2에 fork함수를 적용하여 새로운 프로세스를 생성한다. 생성된 sub process에서는 지금까지 구현한 동작을 실행하고, main process는 sub process의 종료까지 대기한다. 프로그램은 이전 sub process가 생성한 cache file을 유지하고, logfile을 이어서 작성한다. main process는 quit명령어를 입력 받았을 때 logfile에 정보를 출력하고 종료된다.

2. Flow Chart //코드 작성 순서도





3. Pseudo code //알고리즘

```
void main_proxy_cache(void) {
      mainPid = getpid();       //main pid
      startTime = time(NULL)       //start time

      makeDir(homeDir, cacheDir, logDir, logFile);       //make path string
      logFp = fopen(logFile, “at”);       //open logfile.txt
```

```
    while(1) {
        printf("[mainPid]input CMD>");
        scanf("%s", inCMD);
        if(inCMD == "quit") break;
        else if (inCMD == "connect"){
            if ((pid=fork())<0) break;      //fork error
            else if (pid==0) {
                sub_proxy_cache(logFp, cacheDir);
                return;      //sub process return
            }

            processCount++;
            if ((pid = waitpid(pid, &status, 0))<0) break;      //waitpid err
        }
    }
    //quit
    endTime = time(NULL);
    fprintf(logFp, "**SERVER** [Terminated] run time, #sub process count");
    fclose(logFp);
    return;      //main process return
}
```

```
void sub_proxy_cache(FILE* logFp, char* cacheDir){
    pid = getpid();      //sub process pid
    startTime = time(NULL);

    while(1){
        printf("[pid]input URL>");
        scanf("%s", oURL);
        time(&nowTime);
        tp = localtime(&nowTime);

        if(oURL=="bye") break;
        sha1_hash(oURL, hURL);      //get hash
        getCachePath(hURL, cacheDir, dirName, filename); //get path string

        if( isHit(dirName, hURL)){      // [HIT]
            hitCount++;
            fprintf(logFp, "[HIT] cachePath, [TIME]");
            fprintf(logFp, "[HIT] oURL");
        }
        else {      //[MISS]
            missCount++;
            fprintf(logFp, "[MISS] oURL, [TIME]");
        }
    }
    //bye
    endTime = time(NULL);
    fprintf(logFp, "[Terminated] run time, #request hit count, miss count");
    fclose(logFp);
    return;
}
```

4. 결과화면 //수행한 내용을 캡쳐 및 설명

강의자료에 제시된 예시를 이용하여 결과를 확인하였다. 프로그램을 실행했을 때 pid가 정상적으로 출력되었고, logfile.txt와 cache 모두 기대한 결과와 같이 나타났다.

```
kw2019202009@ubuntu:~/Documents/0409/proxy13$ ./proxy_cache
[2608]input CMD> connect
[2609]input URL> www.kw.ac.kr
[2609]input URL> www.google.com
[2609]input URL> bye
[2608]input CMD> connect
[2612]input URL> www.kw.ac.kr
[2612]input URL> www.naver.com
[2612]input URL> bye
[2608]input CMD> quit
kw2019202009@ubuntu:~/Documents/0409/proxy13$ cat ~/logfile/logfile.txt
[MISS] www.kw.ac.kr - [2021/03/12, 08:45:57]
[MISS] www.google.com - [2021/03/12, 08:46:01]
[Terminated] run time: 9 sec, #request hit: 0, miss: 2
[HIT] e00/0f293fe62e97369e4b716bb3e78fababf8f90 - [2021/03/12, 08:46:07]
[HIT] www.kw.ac.kr
[MISS] www.naver.com - [2021/03/12, 08:46:10]
[Terminated] run time: 7 sec, #request hit: 1, miss: 1
**SERVER** [Terminated] run time: 21 sec, #sub process: 2
kw2019202009@ubuntu:~/Documents/0409/proxy13$ tree ~/cache
/home/kw2019202009/cache
├── d8b
│   └── 99f68b208b5453b391cb0c6c3d6a9824f3c3a
├── e00
│   └── 0f293fe62e97369e4b716bb3e78fababf8f90
└── fed
    └── 818da7395e30442b1dcf45c9b6669d1c0ff6b

3 directories, 3 files
kw2019202009@ubuntu:~/Documents/0409/proxy13$
```

5. 결론 및 고찰

fork함수와 waitpid함수를 이용하여 sub process를 생성하고 관리할 수 있었다. 여러 개로 작성했던 .c file을 하나로 작성하고, 크기가 커졌던 proxy_cache 함수를 기능별로 분리하였다. cache의 중복을 확인하기 위하여 사용했던 IsHit flag를 삭제하고, isHit함수에서 cache file을 확인하여 1또는 0의 값을 반환하도록 작성하였다. 그 밖에도 cache와 logfile의 path를 구하기 위하여 사용한 부분을 함수로 분리하였다.

1-3 과제를 구현하는데 어려움은 없었지만 zombie process에 대해 이해하는데 어려움을 겪었다. zombie process와 orphan process의 차이가 잘 이해되지 않았는데, 이것은 시스템프로그래밍 강의시간에 sub process가 종료되었을 때 자원을 회수하기 전의 상태가 zombie process이고, orphan process는 main process가 먼저 종료된 것이며, 이때 init process가 자원을 회수한다는 설명을 듣고 문제를 해결할 수 있었다.

6. reference //과제를 수행하면서 참고한 내용을 구체적으로 기록

강의자료 이용