

2025년 1학기 시스템프로그래밍실습 4주차

Introduction of Proxy Server

System Software Laboratory
College of Software and Convergence
Kwangwoon Univ.

과제 세부 일정

■ 1st Assignments

- Overview & Create Cache Directory
- Get local time & log file
- Create server process

■ 2nd Assignments

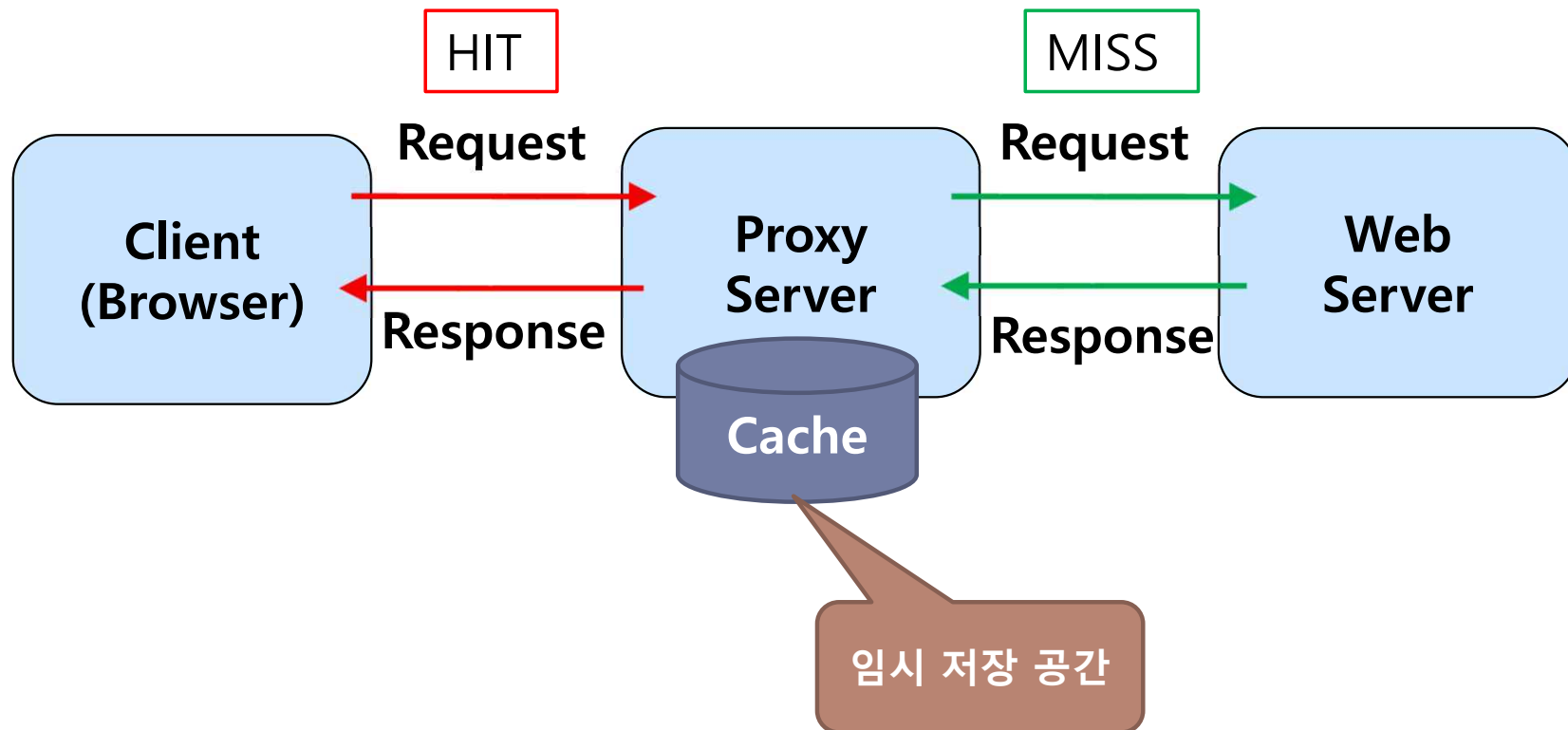
- Implement server/client (socket programming)
- Forward HTTP request to Web server & print the HTTP response
- Integrate server side and client side into proxy server

■ 3rd Assignments

- Add cache(assignment_1) to proxy server(assignment_2)
- Synchronize the shared resource

Introduction of Proxy Server (1/3)

- Two cases in general
 - HIT
 - MISS



Introduction of Proxy Server (2/3)

- **Proxy Server**

- 인터넷의 캐시 프로그램으로 사용자가 계정을 가진 호스트 컴퓨터에서 최근에 가져온 자료들을 보관하는 저장 장소를 의미
- 같은 자료를 다시 요구할 경우 Proxy 내의 cache에 있는 자료를 제공함으로써 자료가 빠르게 전송
- 사용자는 Proxy를 사용해서 자료를 받아 오겠다는 사실을 호스트에 알려 주어야 Proxy를 통해서 자료를 받는 것이 가능
- 전송 속도가 빠를 수도 있지만 Proxy에 미리 저장되지 않은 자료를 찾는 경우는 더 느려 질 수도 있음

Introduction of Proxy Server (3/3)

- **Caching Proxy Server**

- 자신을 통과하는 모든 페이지를 저장
- 사용자가 페이지를 요청했을 때, 요청한 페이지를 가지고 있는지 검사
 - 가지고 있으면, 페이지가 아직 유효한 페이지인지 검사
 - 아직 유효하다면, 사용자에게 페이지 전달
 - 그렇지 않으면, 웹 서버로부터 새로운 페이지를 로드

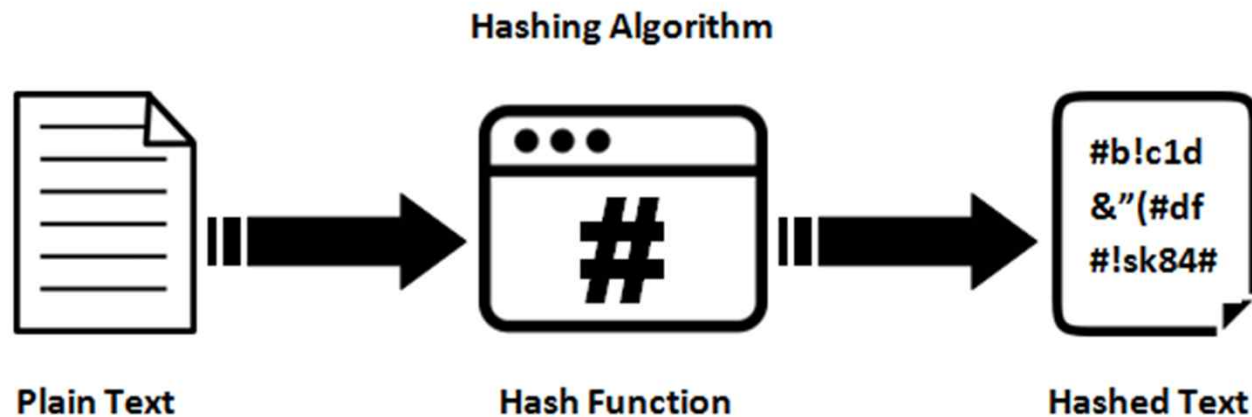
2022년 1학기 시스템프로그래밍실습 4주차

Implementation of the Proxy Cache

System Software Laboratory
College of Software and Convergence
Kwangwoon Univ.

Hash function

- 해시 함수(hash function)
 - 임의의 길이의 데이터를 고정된 길이의 데이터로 매핑하는 함수
 - 해시 함수에 의해 얻어지는 값은 해시 값, 해시 코드, 해시 체크섬 또는 간단하게 해시라고 함
- 해시 함수의 종류로는 MD5, SHA, CRC32등이 있음



SHA-1

- **SHA (Secure Hash Algorithm)**

- A family of cryptographic hash functions
- Features
 - 미국 국립표준기술연구소(NIST)에서 발표하는 미국 연방정부 정보처리 표준(FIPS).
 - 최초의 함수는 SHA이며, 이후의 변형들을 SHA-1, SHA-2, SHA-3 로 묶어서 부름.

- **SHA-1**

- 1993년에 미국 국가 안보국(NSA)에 의해 전자 서명 알고리즘의 일부분으로 개발됨
- SHA-1에 대한 공격 발견되었음.
- 그러나, 아직 여러 보안 프로토콜이나 프로그램에서 많이 사용됨.
 - e.g. SSH(Secure Shell), SSL(Secure Socket Layer), ...
- 입력 한 text data(최대 2^{64} bits)를 160 bits의 hashed data로 반환함.

- **In this assignment,**

- Cache에 저장하기 전에, URL을 hash 하기 위해 사용

Create Directory (1/3)

- **System call**

- mkdir()
 - Make directory

```
#include <sys/types.h>
#include <sys/stat.h>

int mkdir (const char *pathname, mode_t mode);

//Returns : 0 if OK, -1 on error
```

- Parameter
 - pathname : 생성하려는 디렉터리 이름 또는 path
 - mode : 생성시 설정할 디렉터리에 대한 접근 권한 설정 값

- mode_t
 - S_IRUSR, S_IWUSR, S_IXUSR, S_IRWXU
 - S_IRGRP, S_IWGRP, S_IXGRP, S_IRWXG
 - S_IROTH, S_IWOTH, S_IXOTH, S_IRWXO
 - ex) S_IRUSR | S_IWUSR | S_IROTH → 0604
- 숫자로 쓴다면 8진수 기입 → 0xxx
 - ex) 0777, 0755, 0544 (o)
 - ex) 777, 755, 544 (x)

Create Directory (2/3)

- **The nine file access permission bits <sys/stat.h>**

- S_IRUS : user-read : 0400
- S_IWUSR : user-write : 0200
- S_IXUSR : user-execute : 0100

- S_IRGRP : group-read : 0040
- S_IWGR : group-write : 0020
- S_IXGRP : group-execute : 0010

- S_IROTH : other-read : 0004
- S_IWOTH : other-write : 0002
- S_IXOTH : other-execute : 0001

- Grouping into threes

- e.g. S_IRWXU equals S_IRUSR | S_IWUSR | S_IXUSR

(실습 1) Create Directory (3/3)

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
void main(int argc, char *argv[])
{
    if(argc < 2){
        printf("error\n");
        return;
    }
    mkdir(argv[1], S_IRWXU | S_IRWXG | S_IRWXO);
}
```

- 본 코드를 통해 생성된 directory의 권한이 mkdir의 인자로 준 것과 일치하지 않음
- 이러한 일이 발생한 이유를 조사하고 해결할 것.
 - 즉, 과제에서 생성하는 디렉토리의 권한을 제대로 부여하여야 함
 - Hint : **umask**

Reading a Directory (1/5)

- **dirent structure - directory entry**
 - filename (14 bytes) , i-node number(2 bytes)
 - i-node : 파일의 특성이 저장됨
 - 소유권 , 디스크 내 물리적 주소, 링크 수, 형태, 크기
 - 생성/사용/수정시간, i-node 최근 수정시간

```
#include <dirent.h>

struct dirent
{
    ino_t d_ino;        // i-node number
    char  d_name[];     // name of filename
};
```

Reading a Directory (2/5)

- DIR 구조체
 - 디렉토리 스트림(DIR *)
 - 개방된 디렉토리를 접근하는데 필요한 구조체
 - 표준 라이브러리 I/O의 FILE 구조체와 유사한 역할
 - Header file → **<dirent.h>**

Reading a Directory (3/5)

- Opens a directory

```
#include <sys/types.h>
#include <dirent.h>

DIR *opendir(const char *pathname);
```

- opens a directory stream corresponding to the directory **name**
- returns a pointer to the directory stream (on error, NULL is returned)
- The stream is positioned at the first entry in the directory

Reading a Directory (4/5)

- Reads a directory

```
#include <sys/types.h>
#include <dirent.h>

struct dirent *readdir(DIR *dp);
```

- DIR ***dp**

- returns a pointer to a dirent structure representing the next directory entry in the directory stream pointed to by **dp**
- It returns NULL on reaching the end of the directory stream or if an error occurred.

- Closes a directory

```
int closedir(DIR *dp);
```

- DIR ***dp**

- closes the directory stream associated with **dp**
- returns 0 on success. On error, -1 is returned

(실습 2) Reading a Directory (5/5)

```
#include <sys/types.h>
```

```
#include <dirent.h>
```

```
#include <stdio.h>
```

```
int main(int argc, char *argv[])
```

```
{
```

```
    struct dirent *pFile;
```

```
    DIR *pDir;
```

```
    if( argc < 2 ) return;
```

```
    pDir=opendir(argv[1]);
```

```
    if( pDir == NULL ) {
```

```
        printf("Dir read error\n");
```

```
        return;
```

```
    }
```

```
    for(pFile=readdir(pDir); pFile; pFile=readdir(pDir))
```

```
        printf( "%s\n", pFile->d_name );
```

```
    closedir(pDir);
```

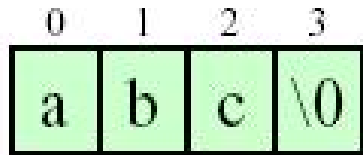
```
    return 1;
```

```
}
```

```
sslslab@ubuntu:~/sslslab/test$ ls
a.out  read_dir.c
sslslab@ubuntu:~/sslslab/test$ mkdir a
sslslab@ubuntu:~/sslslab/test$ ls -al
total 28
drwxrwxr-x 3 sslslab sslslab 4096 Mar 17 21:40 .
drwxrwxr-x 3 sslslab sslslab 4096 Mar 17 20:34 ..
drwxrwxr-x 2 sslslab sslslab 4096 Mar 17 21:40 a
-rwxrwxr-x 1 sslslab sslslab 8768 Mar 17 21:38 a.out
-rw-rw-r-- 1 sslslab sslslab  374 Mar 17 21:04 read_dir.c
sslslab@ubuntu:~/sslslab/test$ touch a/a.txt
sslslab@ubuntu:~/sslslab/test$ mkdir a/b
sslslab@ubuntu:~/sslslab/test$ ls -al a
total 12
drwxrwxr-x 3 sslslab sslslab 4096 Mar 17 21:40 .
drwxrwxr-x 3 sslslab sslslab 4096 Mar 17 21:40 ..
-rw-rw-r-- 1 sslslab sslslab    0 Mar 17 21:40 a.txt
drwxrwxr-x 2 sslslab sslslab 4096 Mar 17 21:40 b
sslslab@ubuntu:~/sslslab/test$ ./a.out a
b
a.txt
..
.
sslslab@ubuntu:~/sslslab/test$ ./a.out a/b
..
.
```


String Manipulation (1/7)

- **String**
 - null-terminated single/multi-byte characters
 - "abc"



- **String manipulation routines**
 - included in `<string.h>`
 - most routines operate on null-terminated strings

String Manipulation (2/7)

- **strlen**

- Get the length of a string
 - excluding the terminal NULL
- **size_t strlen (const char **string*);**

- **Example**

```
#include <string.h>
#include <stdio.h>
void main( void )
{
    char buffer[61] = "How long am I?";
    int len;
    len = strlen( buffer );
    printf( "'%s' is %d characters long\n", buffer, len );
}
```

- **Result**

'How long am I?' is 14 characters long

String Manipulation (3/7)

- **strcmp**

- Compare strings.
- `int strcmp(const char *str1, const char *str2);`
- **Return value**
 - indicates the lexicographic relation of `str1` to `str2`

| Value | Relationship of <code>str1</code> to <code>str2</code> |
|-------|--------------------------------------------------------|
| < 0 | str1 is less than <code>str2</code> |
| = 0 | str1 is identical to <code>str2</code> |
| > 0 | str1 is greater than <code>str2</code> |

- **strncmp**

- `int strncmp(const char *str1, const char *str2, size_t count);`

String Manipulation (4/7)

- **strcpy**
 - Copy a string
 - including the terminating NULL
 - `char* strcpy(char *strDest, const char *strSrc);`
 - **Return value**
 - the pointer of a destination string
 - No return value is reserved to indicate an **error**.
 - **Parameters**
 - *strDest* → Destination string
 - *strSrc* → Null-terminated source string
- **strncpy**
 - a null character is not appended automatically.

String Manipulation (5/7)

- **strcat**

- Append a string.
 - The initial character of strSrc overwrites the NULL of strDest.
- `char* strcat(char *strDest, const char *strSrc);`
- **Return value**
 - Return value
 - No return value is reserved to indicate an **error**.
- Parameters
 - *strSrc/ Dest*. Null-terminated source/destination string

- **strncat**

- terminated with a null character

String Manipulation (6/7)

- **strtok**

- String을 정해진 character에 따라서 자름
- `char *strtok(char *s, const char *delim);`

- **Example**

ex) string : "Car Airplane bicycle"

1) strcpy (buf1, strtok(string, " "));

→ string의 첫 번째 문자열을 " "(공백)으로 구분하여 buf1에 저장

→ buf1 = "Car"

2) strcpy(buf2, strtok(NULL, " \n"));

→ 앞에서 사용한 string의 다음 위치부터 두 번째 문자열을 " "(공백) 혹은 "/n"(new line)으로 구분하여 buf2에 저장

→ buf2 = "Airplane"

String Manipulation (7/7)

- Example

```
/* This program uses strcpy and strcat to build a phrase */  
#include <string.h>  
#include <stdio.h>  
void main( void )  
{  
    char string[80];  
    strcpy( string, "Hello world from " );  
    strcat( string, "strcpy " );  
    strcat( string, "and " );  
    strcat( string, "strcat!" );  
    printf( "String = %s\n", string );  
}
```

- Result

String = Hello world from strcpy and strcat!

2025년 1학기 시스템프로그래밍 & 시스템 프로그래밍 실습

Proxy #1-1

System Software Laboratory
College of Software and Convergence
Kwangwoon Univ.

Assignment 1-1 Contents

- 1. mkdir()
- 2. proxy #1-1
- Report Requirements
- Appendix

1. mkdir()

- mkdir() 이용하여 디렉터리 생성 시 발생하는 권한 문제
 - 아래 코드를 사용 시 directory의 권한이 mkdir의 인자로 준 것과 일치 하지 않음
 - **발생한 이유와 문제를 해결한 소스코드 캡처하여 보고서에 작성**
 - Hint : umask

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
void main(int argc, char *argv[])
{
    if(argc < 2){
        printf("error\n");
        return;
    }
    mkdir(argv[1], S_IRWXU | S_IRWXG | S_IRWXO);
}
```

2. proxy #1-1 Requirements

- Input url은 웹사이트 명으로 입력
 - e.x) www.kw.ac.kr
- Bye command를 입력으로 받았을 때만 프로그램 종료
- 제공된 sha1_hash 함수를 사용하여 입력 받은 URL을 Hashed URL로 변환
- 제공된 getHomedir 함수를 사용하여 Home directory path를 얻고, Home directory에 cache 디렉토리를 생성
- 디렉토리 이름은 hashed_url의 앞 세 글자로함
 - (e.g. Hahsed_url : e000f293fe62e97369e4b716bb3e78fababf8f90
 - 디렉토리 이름 : e00
 - 파일이름 : 0f293fe62e97369e4b716bb3e78fababf8f90
- 파일 생성시 다양한 함수 사용가능

2) proxy #1-1 -Create Cache Directory and File

■ To do list

- url을 SHA-1으로 hashing
 - SHA-1를 이용한 hashing 함수를 사용
 - SHA-1 library를 설치해 주어야 함.
 - **\$ sudo apt-get install libssl-dev**
 - SHA-1 library를 사용한 코드 컴파일 시, 옵션이 필요함.
 - **-lcrypto**
 - e.g. gcc proxy_cache.c **-lcrypto**
- Hashed된 URL에 해당하는 directory와 file 생성
 - 루트 디렉토리는 "~/cache/" 로.
 - Hashed URL에 해당하는 위치에 file이 있는지 검사하고, 없을 경우에 file 생성

2) proxy #1-1 -Create Cache Directory and File

■ Hashing Function using SHA-1

- 아래의 빈 칸을 채워서 사용

```
#include <stdio.h>           // sprintf()
#include <string.h>          // strcpy()
#include <openssl/sha.h>     // SHA1()

char *sha1_hash(char *input_url, char *hashed_url) {
    unsigned char hashed_160bits[20];
    char hashed_hex[41];
    int i;

    SHA1(  );

    for(i=0; i<sizeof(hashed_160bits); i++)
        sprintf(hashed_hex + i*2, "%02x", hashed_160bits[i]);

    strcpy(  );

    return hashed_url;
}
```

■ Function

- unsigned char ***SHA1** (const unsigned char ***d**, unsigned long **n**, unsigned char ***md**);
 - **d** : Hashing 할 데이터
 - **n** : Hashing 할 데이터의 길이
 - **md** : 해시 된 데이터를 저장할 배열

2) proxy #1-1 -Create Cache Directory and File

■ Hashing Function using SHA-1

- 아래의 빈 칸을 채워서 사용

```
#include <stdio.h>           // sprintf()
#include <string.h>          // strcpy()
#include <openssl/sha.h>     // SHA1()

char *sha1_hash(char *input_url, char *hashed_url) {
    unsigned char hashed_160bits[20];
    char hashed_hex[41];
    int i;

    SHA1(   );

    for(i=0; i<sizeof(hashed_160bits); i++)
        sprintf(hashed_hex + i*2, "%02x", hashed_160bits[i]);

    strcpy(   );

    return hashed_url;
}
```

- Variables
 - **input_url** : 원본 URL
 - **hashed_url** : 16진수로 변환된 hashed URL
 - **hashed_160bits** : 해시 된 160bits의 값. (160bits = 5 bits * 32 blocks)
 - **hashed_hex** : hashed_160bits를 1byte씩 16진수로 표현하고, 이를 문자열로 변환한 것

2) proxy #1-1 -Get Path of Home Directory

- Home directory path가 필요한 경우, 아래의 함수를 정의하여 사용

```
#include <sys/types.h>
#include <unistd.h>
#include <pwd.h>
#include <string.h>

char *getHomeDir(char *home) {
    struct passwd *usr_info = getpwuid(getuid());
    strcpy(home, usr_info->pw_dir);

    return home;
}
```

- mkdir(), opendir() 함수 등에서 인자로 "~/"와 같이 ~를 입력하여 home directory에 접근하는 것이 불가능 함.
- 이러한 경우, 절대 경로(e.g. /home/sslabs/)를 사용하지 말고,
위의 함수를 이용하여 home directory path를 얻을 것.
- 본 함수 구현에 대한 자세한 내용은 Proxy 1-3 강의자료에서 다룰 예정임

2) proxy #1-1 Requirements - Input

- Read URL from the STDIN (as a shell)
- Use 'bye' command to exit a program
- e.g.

```
ssl@ubuntu:~/ssl$ ls
Makefile proxy_cache.c
ssl@ubuntu:~/ssl$ make
gcc -o proxy_cache proxy_cache.c -lcrypto
ssl@ubuntu:~/ssl$ ls
Makefile proxy_cache proxy_cache.c
ssl@ubuntu:~/ssl$ ./proxy_cache
input url> www.kw.ac.kr
input url> www.google.com
input url> bye
ssl@ubuntu:~/ssl$
```


2) proxy #1-1 Requirements - Output

- Create cache directory

- HIT인 경우. (Assignment#1-2에서 구현)
 - 아무 것도 하지 않음.

- MISS인 경우

- 입력 받은 URL을 SHA-1 모듈을 사용하여 Hashed URL로 변환
- Hashed URL에 따라 directory와 cached file을 구성함.

- Cache directories

- Using hash function results
- The root directory is "~/cache/"
- 생성한 모든 Directory는 모든 권한을 갖도록 구현함.
 - ex) 8진수로 777 할당

* tree 명령어는 추가 package 설치 후 사용 가능
\$ sudo apt-get install tree

```
sslab@ubuntu:~/sslab$ ./proxy_cache
input url> www.kw.ac.kr
input url> www.google.com
input url> www.naver.com
input url> klas.kw.ac.kr
input url> bye
sslab@ubuntu:~/sslab$ tree ~/cache/
/home/sslab/cache/
├── kcf
│   └── 9fd210fb8e00c8114ff978d282258ed8a48ea
├── g0b
│   └── 99f68b208b5453b391cb0c6c3d6a9824f3c3a
├── 000
│   └── 0f293fe62e97369e4b716bb3e78fababf8f90
└── fcd
    └── 818da7395e30442b1dcf45c9b6669d1c0ff6b

4 directories, 4 files
```

2) proxy #1-1 Requirements

■ Create Cache Directory and File

- Hashed된 URL에 해당하는 directory와 file 생성
 - e.g.

```
sslslab@ubuntu:~/sslslab$ ./proxy_cache
input url> www.kw.ac.kr
input url> www.google.com
input url> www.naver.com
input url> klas.kw.ac.kr
input url> bye
```

```
/home/sslslab/cache/
├── 3ef
│   └── 9fd210fb8e00c8114ff978d282258ed8a48ea
├── d8b
│   └── 99f68b208b5453b391cb0c6c3d6a9824f3c3a
├── e00
│   └── 0f293fe62e97369e4b716bb3e78fababf8f90
└── fed
    └── 818da7395e30442b1dcf45c9b6669d1c0ff6b
```

| URL | | Hashed URL |
|----------------|---|-------------------------------------------|
| www.kw.ac.kr | | e00/0f293fe62e97369e4b716bb3e78fababf8f90 |
| www.google.com | → | d8b/99f68b208b5453b391cb0c6c3d6a9824f3c3a |
| www.naver.com | | fed/818da7395e30442b1dcf45c9b6669d1c0ff6b |
| klas.kw.ac.kr | | 3ef/9fd210fb8e00c8114ff978d282258ed8a48ea |

Report Requirements

- **Ubuntu 20.04.6 Desktop 64bits 환경에서 채점**
- **Copy 발견 시 0점 처리**
- **보고서 구성**
 - **보고서 표지**
 - 수업 명, 과제 이름, 담당 교수님, 학번, 이름, 강의 시간 필히 명시
 - 과제 이름 → Proxy 1-1
 - 아래의 내용은 보고서에 필히 포함
 - Introduction
 - 과제 소개 – 4줄 이상(background 제외) 작성
 - Flow Chart
 - 코드 작성 순서도
 - sha1_hash 함수, getHomedir 함수 필수 작성 아님
 - 강의자료 appendix 내용 참고
 - Pseudo code
 - 알고리즘
 - Sha1_hash 함수, getHomedir 함수 필수 작성 아님
 - 강의자료 appendix 내용 참고
 - 결과화면
 - 수행한 내용을 캡처 및 설명
 - 고찰
 - 과제를 수행하면서 느낀점 작성
 - Reference
 - 과제를 수행하면서 참고한 내용을 구체적으로 기록
 - 강의자료만 이용한 경우 생략 가능

Report Requirements

- Softcopy Upload

- 제출 파일

- 보고서 + 소스파일 하나의 압축 파일로 압축하여 제출(tar.xz)
 - 1)보고서:
 - 보고서를 **pdf로 변환**하여 제출
 - 보고서 이름은 *Proxy1-1_수강분류코드_학번_이름* 으로 작성
 - 2)C 파일 명:
 - **proxy_cache.c**
 - **Comment 작성(Appendix 내용 참고)**
 - 3)Makefile:
 - 실행파일명: **proxy_cache**
 - C 파일명, 실행파일명 지정한 이름 외 다른 명으로 작성 시 감점


- tar.xz 압축 방법

- **(Appendix 내용 참고)**

- 컴파일은 무조건 Makefile(makefile)을 이용한 make로 함.
 - Makefile(makefile) 없거나 실행 불가시 0점
 - 파일 압축 오류 시, 0점 처리

Report Requirements

- 실습 수업을 수강하는 학생인 경우
 - 실습 과목에 과제를 제출(.tar.xz)
 - 이론 과목에 간단한 .txt 파일로 제출

 실습수업때 제출했습니다.

2022-08-29 오후 3:58

텍스트 문서

0KB

- 이론 과목에 .txt 파일 미 제출 시 감점
- .tar.xz 파일로 제출 하지 않을 시 감점

- 예시-이론 월5 수6 수강하는 학생인 경우
 - 보고서: Proxy1-1_A_2025123456_홍길동.pdf
 - 압축 파일 명: Proxy1-1_A_2025123456_홍길동.tar.xz

| | | | |
|------------|-------------|-----------|------------|
| 수강요일 | 이론1 월5수6 | 이론2 목4 | 실습1 목12 |
| 수강분류 코드 | A | B | C |

- 과제 제출
 - KLAS – 강의 과제 제출**
 - 2025년 4월 3일 목요일 23:59까지 제출**
 - 딜레이 받지 않음
 - 제출 마감 시간 내 미제출시 해당 과제 **0점 처리**
 - 교내 서버 문제 발생 시, 메일로 과제 제출 허용

Report Requirements

- 실습 수업을 수강하는 학생인 경우

- 실습 과목에 과제를 제출
- 이론 과목에 간단한 .txt 파일로 제출

실습수업때 제출했습니다.

2022-08-29 오후 3:58

텍스트 문서

0KB

- 이론 과목에 .txt 파일 미 제출 시 감점
- .tar.xz 파일로 제출 하지 않을 시 감점

- 과제 제출

- KLAS – 강의 과제 제출
- 2025년 4월 3일 목요일 23:59까지 제출
- 딜레이 받지 않음
 - 제출 마감 시간 내 미제출시 해당 과제 0점 처리
 - 교내 서버 문제 발생 시, 메일로 과제 제출 허용(제출 기한 내)

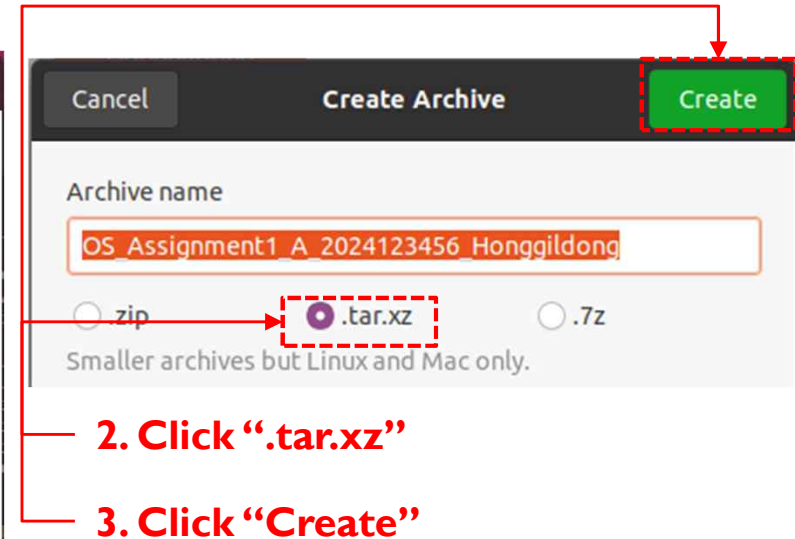
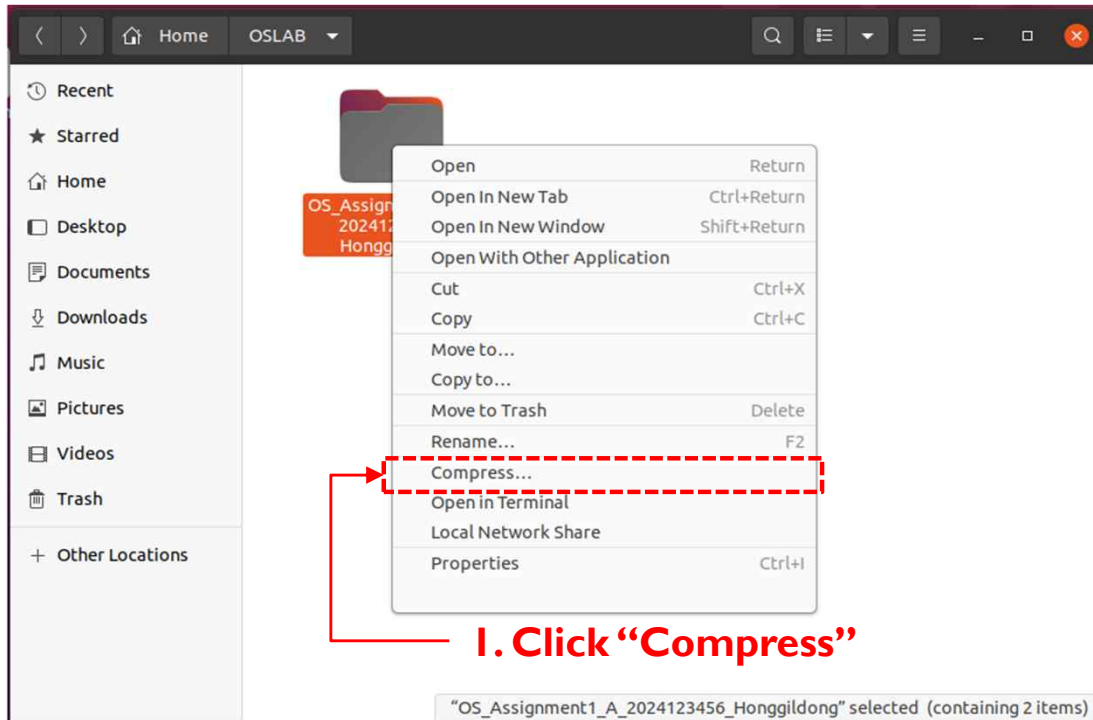
2025년 1학기 시스템프로그래밍 실습

Appendix

System Software Laboratory
College of Software and Convergence
Kwangwoon Univ.

tar.xz compression

■ Solution 01. Compress using GUI



tar.xz compression(Cont'd)

- Solution 02. Compress using CLI

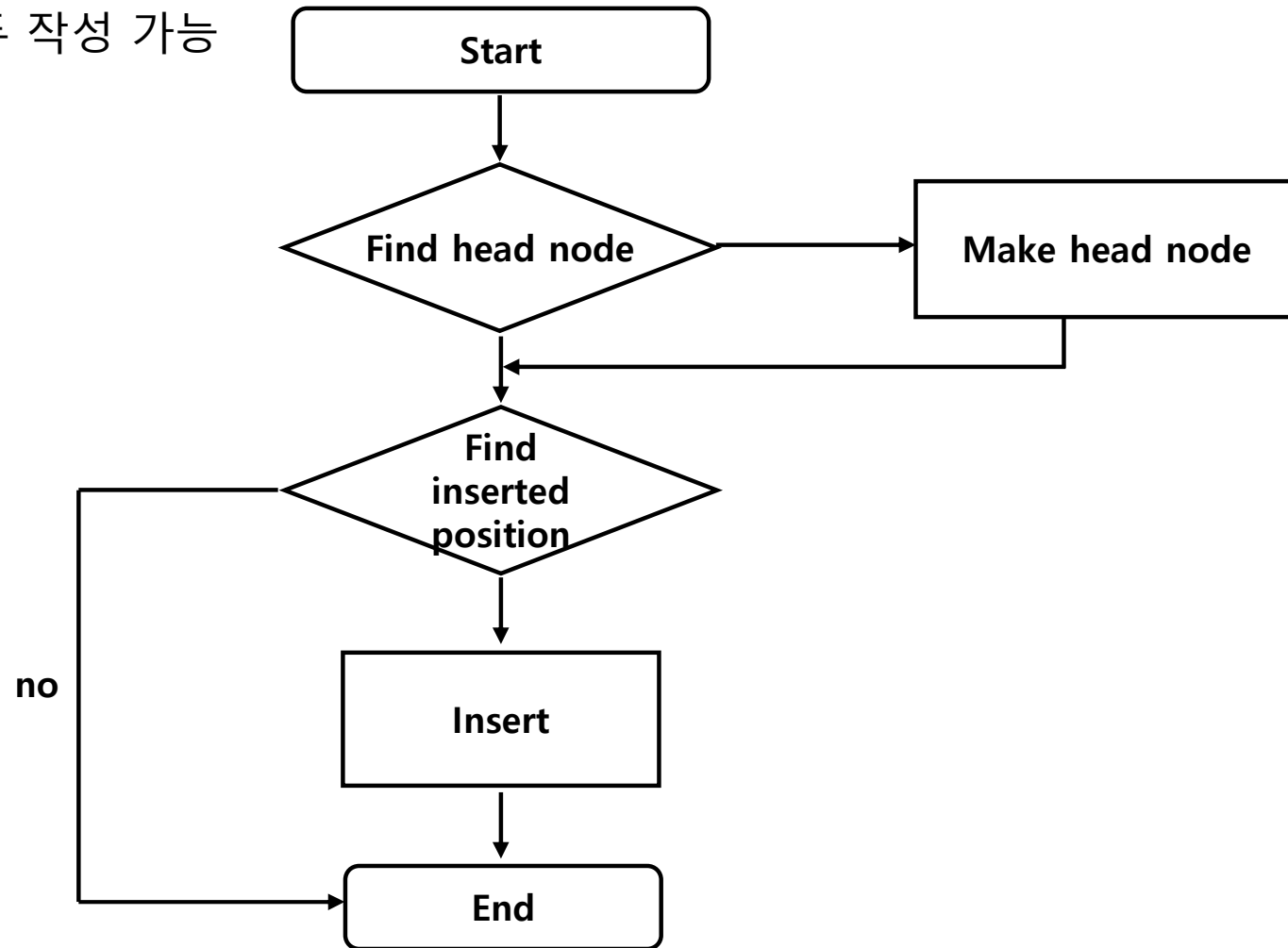
```
test12@ubuntu:~/OSLAB$ ls
OS_Assignment1_A_2024123456_Honggildong
test12@ubuntu:~/OSLAB$ tar Jcvf OS_Assignment1_A_2024123456_Honggildong.tar.xz OS_Assignment1_A_2024123456_Honggildong
OS_Assignment1_A_2024123456_Honggildong/
OS_Assignment1_A_2024123456_Honggildong/code.c
OS_Assignment1_A_2024123456_Honggildong/Report.pdf
```

- 압축방법(Compress)
 - \$tar -Jcvf [File_name.tar.xz] [Target]
 - \$tar -Jcvf [압축파일이름.tar.xz] [압축대상]
- 압축해제(Extract)
 - \$tar -Jxvf [File_name.tar.xz]
 - \$tar -Jxvf [압축된 파일 이름.tar.xz]

보고서 작성 요령 (1/2)

■ Algorithm – Flow Chart (Each function)

- E.g.
- 국문, 영문 모두 작성 가능



보고서 작성 요령 (2/2)

■ Algorithm – Pseudo Code

- E.g.
- 국문, 영문 모두 작성 가능

```
FixHeap(Node *root, Key k)
{
    Node vacant, largerChild;
    vacant = root;
    while( vacant is not leaf ) {
        largerChild = the child of vacant with the larger key;
        if( k < largerChild's Key ) {
            copy largerChild's key to vacant;
            vacant = largerChild;
        }
        else exit loop;
    }
}
```

Set grade counter to one

While grade counter is less than or equal to ten

 Input the next grade

 Add the grade into the total

Set the class average to the total divided by ten

Print the class average

Comment 작성 요령 (1/2)

■ File Head Comment

- E.g.
- 국문, 영문 모두 작성 가능

```
////////////////////////////////////  
// File Name      : Main.c                      //  
// Date           : 2022/03/01                  //  
// Os             : Ubuntu 16.04 LTS 64bits      //  
// Author         : Hong Gil Dong               //  
// Student ID     : 2022123456                  //  
// ----- //  
// Title : System Programming Assignment #1-1 (proxy server) //  
// Description : ...                            //  
////////////////////////////////////
```

Comment 작성 요령 (2/2)

■ Function Head Comment

- E.g.
- 국문, 영문 모두 작성 가능

```
////////////////////////////////////  
// InsertNode                                                    //  
// =====                                                    //  
// Input: Node* -> Insert Node,                                  //  
//           Node* -> Column node before insert node           //  
//           Node* -> Row node before insert node               //  
//           (Input parameter Description)                        //  
// Output: int  - 1 success                                       //  
//           0 fail                                              //  
//           (Out parameter Description)                          //  
// Purpose: Inserting node                                       //  
////////////////////////////////////
```

Comment 작성 요령 (3/3)

■ In-line Comment

- e.g.
- 국문, 영문 모두 작성 가능

```
if( pRowPos->pNextRow != pRowPos ) {  
    pTemp->pNextRow = pRowPos->pNextRow;           // pTemp set next row  
    if( !( pRowPos->pNextRow->bHead ) ){  
        pRowPos->pNextRow->NodeItem.pPrevRow = pTemp;  
    } // end of if  
} // end of if  
else {  
    pTemp->pNextRow = pRowPos;                       // pTemp set next row  
} // end of else  
pTemp->NodeItem.pPrevRow = pRowPos;                 // pTemp set previous row  
pRowPos->pNextRow = pTemp;
```