KWANGWOON UNIVERSITY

2024년 1학기 **시스템프로그래밍 & 시스템 프로그래밍 실습**

# Assignment3 - 1

**System Software Laboratory**
College of Software and Convergence
Kwangwoon Univ.

# Work Flow (1/3)

- **Client**
  - is executed with two parameter – **IP address and port number of Server.**
  - After successful connection, displays the message "****It is connected to Server****".
  - receives "**REJECTION**" or "**ACCEPTED**" from server
    - If it received "**REJECTION**" from server **(i.e. doesn't exist in the access.txt)**
      - than displays the message "****Connection refused****" and disconnects to server.
    - If it received "**ACCEPTED**" from server
      - than receives username and password.
  - receives user name and password from standard input & passes user name and password to server
    - If it received "**OK**" from server
      - than displays the message "**** User '…' logged in ****".
    - If it received "**FAIL**" from server
      - than displays the message "**** Log-in failed ****".
    - If it received "**DISCONNECTION**" from server **(i.e. failures during 3 times)**
      - than displays the message "**** Connection closed ****" and disconnects to server

# Work Flow (2/3)

- **Server**
  - is executed with one parameter – **Port number of server.**
  - After successful connection, displays the message "****Client is connected****", and displays client's IP, Port.
  - checks client's IP to confirm possibility of connection using "`access.txt`" file.
    - If the client's IP doesn't exist in the access.txt, send "**REJECTION**", and display the message "****It is NOT authenticated client****".
    - If the client's IP exist in the access.txt, send "**ACCEPTED**".
  - counts whenever client try to login, and displays the message "****User is trying to log-in (x/3)****".

# Work Flow (3/3)

- **Server (cont'd)**
  - receives the user name and password from client
    - searches user name and password in pre-define text file "`passwd`"
    - If it finds user name and password
      - then send "**OK**", and displays the message "****Success to log-in****".
    - If it can't find user name and password
      - If it is the first & second time, sends "**FAIL**", and displays the message "****Log-in failed****".
      - If it is the third time, don't send "FAIL", just display the message "****Log-in failed****".
      - If it is the third time, sends "**DISCONNECTION**", and display the message "****Fail to log-in****" ,and closes session.

# Skeleton Code (1/5)

- **Client**

```
/* 필요한 header file 선언 */

#define MAX_BUF 20
#define CONT_PORT       20001

int main(int argc, char *argv[])
{
    int sockfd, n, p_pid;
    struct sockaddr_in  servaddr;

    /* 코드 작성 */

    connect(sockfd,(struct sockaddr *)&servaddr, sizeof(servaddr));

    log_in(sockfd); close(sockfd);
    return 0;
}
```

# Skeleton Code (2/5)

- **Client (cont'd)**

```
void log_in(int sockfd){
    int n;
    char user[MAX_BUF], *passwd, buf[MAX_BUF];
    /* 코드 작성 (hint:  Check if the ip is acceptable ) */
    for(;;) {
        /* 코드 작성 (hint: pass username to server) */
        n = read(sockfd, buf, MAX_BUF);
        buf[n] = '\0';
        if(!strcmp(buf, "OK")){
            n = read(sockfd, buf, MAX_BUF);
            buf[n] = '\0';

            if(!strcmp(buf, "OK")) {
                /* 코드 작성 (hint: login success) */
            }
            else if(!strcmp(buf, "FAIL")){
                /* 코드 작성 (hint: login fail) */
            }
            else{ // buf is "DISCONNECTION"
                /* 코드 작성 (hint: three times fail) */
            }
        }
    }
}
```

# Skeleton Code (3/5)

- **Server**

```
/* 필요한 header file 선언 */

#define  MAX_BUF 20

int main(int argv, char *argv[]) {
    int listenfd, connfd;
    struct sockaddr_in servaddr, cliaddr;
    FILE *fp_checkIP;        // FILE stream to check client's IP

    /* 코드 작성 */

    listen(listenfd, 5);
    for(;;) {
        connfd = accept(listenfd, (struct sockaddr *) &cliaddr, &clilen) ;
        /* 코드 작성 (hint: Client의 IP가 접근 가능한지 확인) */
        if(log_auth(connfd) == 0) {    // if 3 times fail (ok : 1, fail : 0)
            printf("** Fail to log-in **\n");
            close(connfd);
            continue;
        }
        printf("** Success to log-in **\n");
        close(connfd);
        }
}
```

# Skeleton Code (4/5)

- **Server (cont'd)**

```
int log_auth(int connfd)
{
    char user[MAX_BUF], passwd[MAX_BUF];
    int n, count=1;

    while(1) {
        /* 코드 작성 (hint: username과 password를 client로부터 받는다) */
        write(connfd, "OK", MAX_BUF);

        if((n = user_match(user, passwd)) == 1){
            /*  코드 작성 (hint: 인증 OK) */
        }

        else if(n == 0){
            if(count >= 3) {
                /* 코드 작성 (hint: 3 times fail) */
            }
            write(connfd, "FAIL", MAX_BUF);
            count++;
            continue;
        }
    }
    return 1;
}
```

# Skeleton Code (5/5)

- **Server (cont'd)**

```
int user_match(char *user, char *passwd)
{
    FILE *fp;
    struct passwd *pw;

    fp = fopen("passwd", "r");

    /* 코드 작성 (hint: 인증 성공 시 return 1, 인증 실패 시 return 0  */

}
```

# Execution Example (1/3)

- 접속이 불가한 IP를 가진 Client가 접속할 경우

**access.txt**

```
```

**passwd**

```
test1:12:0:0:SPLab1:/home/1:sh1
test2:34:1:0:SPLab2:/home/2:sh2
test3:56:2:0:SPLab3:/home/3:sh3
```

**Server Process**

```
$ ./srv 12345
** Client is trying to connect **
 - IP:   127.0.0.1
 - Port: 35261
** It is NOT authenticated client **
```

**Client Process**

```
$ ./cli 127.0.0.1 12345
** Connection refused **
$
```

사용자 입력
출력된 메시지

# Execution Example (2/3)

- 성공적으로 로그인을 마친 경우

**access.txt**

```
*.*.*.*
```

**passwd**

```
test1:12:0:0:SPLab1:/home/1:sh1
test2:34:1:0:SPLab2:/home/2:sh2
test3:56:2:0:SPLab3:/home/3:sh3
```

**Server Process**

```
$ ./srv 12345
** Client is trying to connect **
 - IP:   127.0.0.1
 - Port: 35262
** Client is connected **
** User is trying to log-in (1/3) **
** Success to log-in **
```

**Client Process**

```
$ ./cli 127.0.0.1 12345
** It is connected to Server **
Input ID : test1
Input Password : 12
** User 'test1' logged in **
$
```

사용자 입력
출력된 메시지

# Execution Example (3/3)

- 로그인을 세 번 시도했지만 모두 실패한 경우

**access.txt**

```
*.*.*.*
```

**passwd**

```
test1:12:0:0:SPLab1:/home/1:sh1
test2:34:1:0:SPLab2:/home/2:sh2
test3:56:2:0:SPLab3:/home/3:sh3
```

**Server Process**

```
$ ./srv 12345
** Client is trying to connect **
 - IP:   127.0.0.1
 - Port: 35263
** Client is connected **
** User is trying to log-in (1/3) **
** Log-in failed **
** User is trying to log-in (2/3) **
** Log-in failed **
** User is trying to log-in (3/3) **
** Log-in failed **
** Fail to log-in **
```

**Client Process**

```
$ ./cli 127.0.0.1 12345
** It is connected to Server **
Input ID : test
Input Password : 1234
** Log-in failed **
Input ID : test1
Input Password : 34
** Log-in failed **
Input ID : test1
Input Password : 45
** Connection closed **
$
```

사용자 입력
출력된 메시지

# 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 필히 명시
      - 과제 이름 ➔ Assignment3-1
  - **과제 내용**
    - Introduction
      - 과제 소개 - 4줄 이상(background 제외) 작성
    - Flow chart(4주차 강의자료 appendlx 참고)
    - Pesudo code(4주차 강의자료 appendlx 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

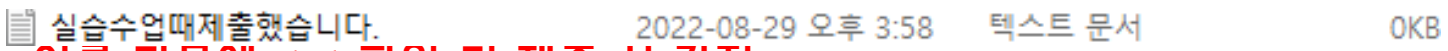
- **Softcopy Upload**
  - 제출 파일
    - 보고서 + 소스파일 하나의 압축 파일로 압축하여 제출(tar.gz)
    - **보고서(.pdf. 파일 변환)**
    - **소스코드**
      - **cli.c, srv.c**
      - **Makefile**
      - **실행파일명: cli, srv**
      - **소스 코드, 실행파일명 다르게 작성 시 감점**
  - Tar 압축 및 해제 방법
    - 압축 시 ➔ tar –zcvf [압축 파일명].tar.gz[폴더 명]
    - 해제 시 ➔ tar –zxvf 파일명.tar.gz
  - 보고서 및 압축 파일 명 양식
  - **Assignment3_1_수강분류코드_학번 으로 작성**

| 수강요일 | 이론1<br>월5수6 | 이론2<br>목4 | 실습1<br>금12 | 실습2<br>금56 | 실습3<br>금78 |
|---|---|---|---|---|---|
| 수강분류<br>코드 | A | B | C | D | E |

    - **예시–이론 월5 수6 수강하는 학생인 경우**
      - **보고서 Assignment3_1_A_2024123456.pdf**
      - **압축 파일 명: Assignment3_1_A_2024123456.tar.gz**

# Report Requirements

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

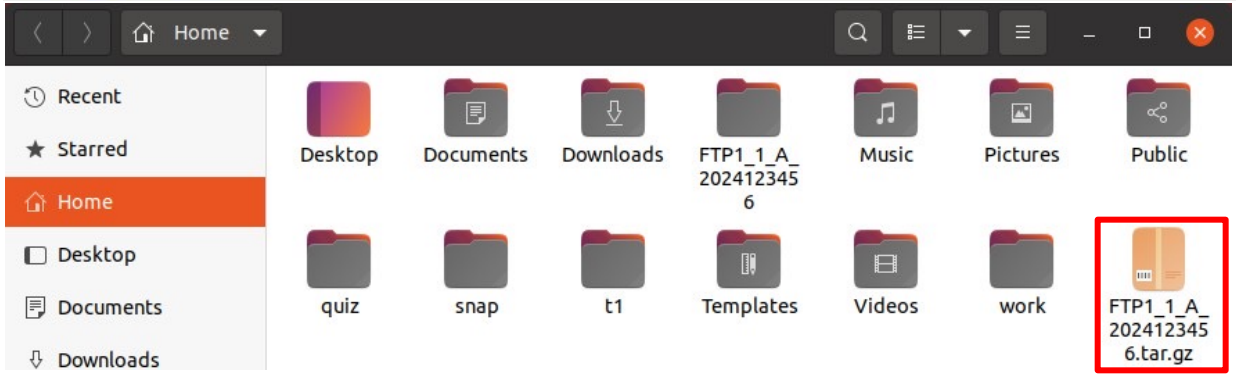



    - **이론 과목에 .txt 파일 미 제출 시 감점**
    - **.tar.gz 파일로 제출 하지 않을 시 감점**

- **과제 제출**
  - **KLAS – 강의 과제 제출**
  - **2024년 5월 23일 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

# Appendix A. tar.gz compression

```
sslab@ubuntu: ~
sslab@ubuntu:~$ ls
Desktop    Downloads            Music     Public  snap            t1         Videos
Documents  FTP1_1_A_2024123456  Pictures  quiz    splab_commands  Templates  work
sslab@ubuntu:~$ cd FTP1_1_A_2024123456/
sslab@ubuntu:~/FTP1_1_A_2024123456$ ls
FTP1_1_A_2024123456.pdf  Makefile  othercom.c
sslab@ubuntu:~/FTP1_1_A_2024123456$ cd ..
sslab@ubuntu:~$ tar -zcvf FTP1_1_A_2024123456.tar.gz FTP1_1_A_2024123456/
FTP1_1_A_2024123456/
FTP1_1_A_2024123456/othercom.c
FTP1_1_A_2024123456/FTP1_1_A_2024123456.pdf
FTP1_1_A_2024123456/Makefile
sslab@ubuntu:~$ ls
Desktop    FTP1_1_A_2024123456          Pictures  snap            Templates
Documents  FTP1_1_A_2024123456.tar.gz   Public    splab_commands  Videos
Downloads  Music                        quiz      t1              work
sslab@ubuntu:~$
```



# Appendix B. Comment 작성 요령 (1/3)

- **File Head Comment**

```
/////////////////////////////////////////////////////////////////////////////
// File Name      : Main.c                                                 //
// Date           : 2024/03/01                                             //
// OS             : Ubuntu 20.04.6 LTS 64bits
          //
// Author         : Hong Gil Dong                                          //
// Student ID     : 2024123456                                             //
// ------------------------------------------------------------------------------------------------------------------------ //
// Title : System Programming Assignment #1-1  ( ftp server )              //
// Description : ...                                                       //
/////////////////////////////////////////////////////////////////////////////
```

# Appendix B. Comment 작성 요령 (2/3)

- **Function Head Comment**

```
////////////////////////////////////////////////////////////////////////
// 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                                            //
////////////////////////////////////////////////////////////////////////
```

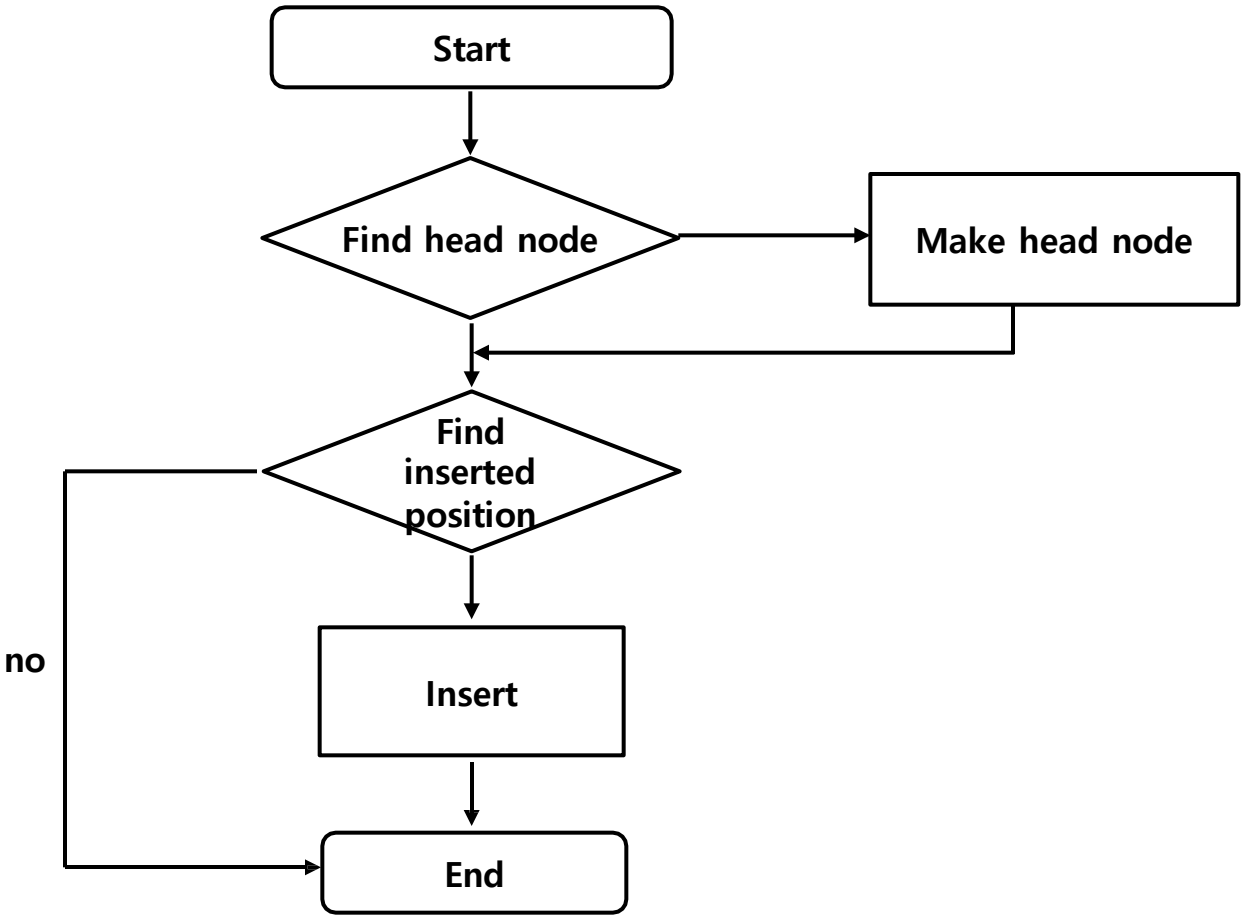
# Appendix B. Comment 작성 요령 (3/3)

- **In-line Comment**

```
////////////////////// Row insert ////////////////////
   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;
////////////////////// End of row insert ///////////////
```

# Appendix C. 보고서 작성 요령 (1/2)

- **Algorithm – Flow Chart (Each function)**
  - E.g.



# Appendix C. 보고서 작성 요령 (2/2)

- **Algorithm – Pseudo Code**

```
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 lagerChild's key to vacant;
                        vacant = largerChild;
                }
                else exit loop;
        }
}
```