COSE341 2018320182 김상언

```c
int main(void) {
    char receive_msg[BUFFER_SIZE], send_msg[BUFFER_SIZE];
    int receive_fd, send_fd;
    /*---------------------------------------*/
    /* TODO 1 : init receive_fd and send_fd  */

    if ((receive_fd = open(NP_RECEIVE, O_RDWR)) == -1)
        return -1;

    if ((send_fd = open(NP_SEND, O_WRONLY)) == -1)
        return -1;

    /* TODO 1 : END                          */
    /*---------------------------------------*/

    for (int i=12; i<16; i++) {
        printf("client : send %d\n", i);
        sprintf(send_msg, "%d", i);
        /*---------------------------------------*/
        /* TODO 2 : send msg and receive msg     */

        if (write(send_fd, send_msg, sizeof(send_msg)) == -1)
            return -1;

        if (read(receive_fd, receive_msg, sizeof(receive_msg)) == -1)
            return -1;

        /* TODO 2 : END                          */
        /*---------------------------------------*/
```

Named PIPE를 이용하는 경우이다. Read와 write를 하기 위해서 둘 다 open하여 동기화 되도록 할 것인데 만약 open이 비정상적으로 작동할 경우 종료하도록 한다. TODO2 에서는 메시지를 send하고 receive하는 부분인데 마찬가지로 전송할 메시지를 작성하는 과정이 비정상적이라면 종료하고 수신 메시지를 읽는데 비정상적으로 작동하면 종료하도록 한다.

```c
/*---------------------------------------*/
/* TODO 3 : make pipes and init fds      */
/* (1) make NP_RECEIVE pipe              */
/* (2) make NP_SEND pipe                 */
/* (3) init receive_fd and send_fd       */

if(access(NP_RECEIVE, F_OK) == 0)
    unlink(NP_RECEIVE);

if(mkfifo(NP_SEND, 0666) == -1)
    return -1;

if((receive_fd = open(NP_RECEIVE, O_RDWR)) == -1)
    return -1;

if(access(NP_SEND, F_OK) == 0)
    unlink(NP_SEND);

if(mkfifo(NP_SEND, 0666) == -1)
    return -1;

if((send_fd = open(NP_SEND, O_RDWR)) == -1)
    return -1;

/* TODO 3 : END                          */
/*---------------------------------------*/

while (1) {
    /*---------------------------------------*/
    /* TODO 4 : read msg                     */

    if(read(receive_fd, receive_msg, sizeof(receive_msg)) == -1)
        return -1;

    /* TODO 4 : END                          */
    /*---------------------------------------*/

    printf("server : receive %s\n", receive_msg);

    value = atoi(receive_msg);

    sprintf(send_msg, "%d", value*value);
    printf("server : send %s\n", send_msg);

    /*---------------------------------------*/
    /* TODO 5 : write msg                    */

    if(write(send_fd, send_msg, sizeof(send_msg)) == -1)
        return -1;

    /* TODO 5 : END                          */
    /*---------------------------------------*/
```

TODO3에서 우선 receive와 send의 경우로 나누어서 작성하였다. 우선 pipe가 존재하는지를 access 함수를 이용하여 체크하여 존재한다면 unlink하여 제거한다. 없다면 mkfifo를 이용하여 생성을 해준다. 그러고 난 뒤 open을 하여 둔다. Send도 receive와 마찬가지로 진행한다.

이후 TODO4에서 메시지를 read하는데 만약 read가 비정상적으로 종료되는 경우 종료하도록 한다.

TODO5에서는 메시지를 write한느 부분으로 마찬가지로 write가 비정상적으로 종료되는 경우 종료되도록 한다.