2025 년 1 학기 네트워크 프로그래밍 중간시험

☞ 주의 사항

- 기능에 대한 코드만 있고 동작이 안되면 해당 점수 없음
- 각 소스 파일에 학번, 이름(영문 가능) 주석이 없는 경우, 파일당 1 점 감점
- 화면 출력 시 쓰레기 값이 출력되거나, 라인번호가 출력되지 않으면 각 항목별 2 점씩 감점
- 문제에서 주어진 cmd 를 조건문으로 처리하지 않는 경우, 항목당 1 점 감점
- 주어진 조건(구조체, TCP 프로토콜)등을 사용하지 않는 경우, 0 점 처리함
- **검색을 위해 exec 관련 함수를 사용하여 grep 명령어를 직접 호출하는 경우 0 점 처리함**
- **AI 기능을 사용하거나 LMS 이외의 웹사이트에 접속하는 경우, F 학점 처리함**

1. TCP 프로토콜을 이용한 remote grep server/client 프로그램 (30 점)

제출파일: rgrep_server.c rgrep_client.c

서버에 저장되어 있는 파일 내용을 검색하여 클라이언트로 전송하는 remote grep 명령어를 구현하시오. (network.txt, udp.txt 파일 사용)

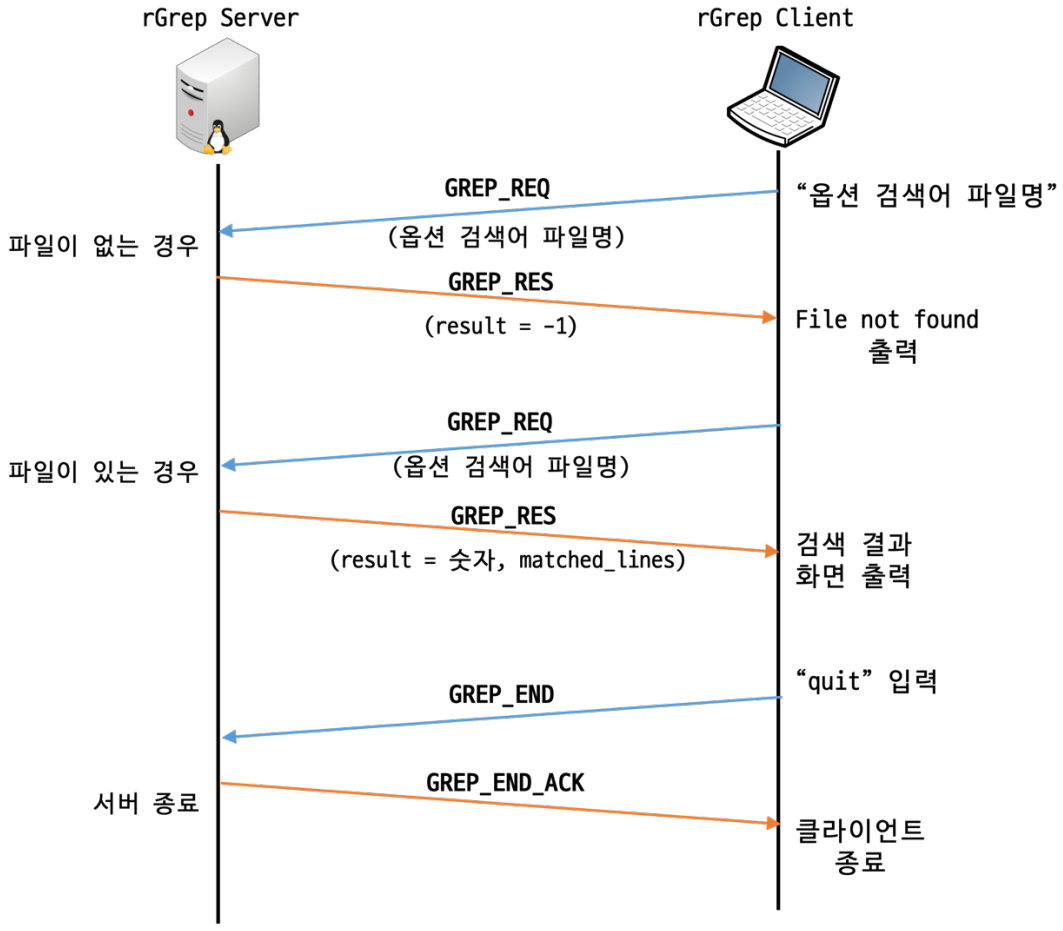
■ 데이터 전송 구조체

```c
#define GREP_REQ        1   // Client -> Server
#define GREP_RES        2   // Server -> Client
#define GREP_END        3   // Client -> Server
#define GREP_END_ACK    4   // Server -> Client

// Remote Grep Reqeust (Client -> Server)
typedef struct
{
    int cmd;           // GREP_REQ, GREP_END
    char options[100]; // "옵션 검색어 파일이름"
} REQ_PACKET;

// Remote Grep Response (Server -> Client)
typedef struct
{
    int cmd;                    // GREP_RES, GREP_END_ACK
    int result;                 // 검색된 라인 수
                                // -1: file not found
                                // -2: invalid option(-n, -v, -i 가 아닌 경우)
    char matched_lines[2048]; // 검색 결과(라인번호 및 검색된 파일 내용)
} RES_PACKET;
```

■ 문자열 검색 함수

```
#include <string.h>
char* strstr(const char* str, const char* substr);
    - str: 원본 문자열, substr: 검색 문자열 (대소문자 구분)
    - str이 substr을 포함하지 않으면 NULL 리턴 (문자열 일치가 아닌 포함)

char* strcasetstr(const char* str, const char* substr);
    - strstr()함수와 동일하며 대소문자 구분 없이 substr 검색
    - substr을 포함하지 않으면 NULL 리턴
```

■ 메시지 프로토콜



■ 문자열 검색 기능 (26점)

✓ 검색어 처리 및 파일 확인 기능 (6점)

- 클라이언트는 화면에서 "옵션 검색어 파일명"을 입력 받고 REQ_PACKET의 options[] 배열에 저장 후 서버로 전송(cmd=GREP_REQ) (2점)
- 서버는 전송된 GREP_REQ 패킷의 options[] 배열의 내용을 공백을 기준으로 분리(옵션, 검색어, 파일명) (2점)
- 전송된 파일이 서버의 디렉토리에 존재하는지 검사하여 파일이 없으면

RES_PACKET의 result변수에 -1을 저장하여 전송(cmd=GREP_RES) (1점)

- 클라이언트가 잘못된 옵션을 전송한 경우, 서버는 GREP_RES 패킷의 result 변수에 -2를 저장하여 전송하고 클라이언트는 **"Invalid option"**을 화면에 출력 (1점)

✓ **"-n" 옵션 기능 구현 (검색어 대소문자 구분)** (5점)

- 서버는 파일에서 한 라인씩 읽고, strstr() 함수를 이용하여 검색어를 포함하는 라인에 대해서 해당 라인 번호 및 파일 내용을 화면에 출력

✓ **"-v" 옵션 기능 구현 (대소문자 구분)** (5점)

- 서버는 strstr() 함수를 이용하여 검색어를 포함하지 않는 라인에 대해 라인 번호 및 파일 내용을 화면에 출력

✓ **"-i" 옵션 기능 구현 (검색어 대소문자 구분 없음)** (5점)

- 서버는 strcasestr() 함수를 이용하여 대소문자 구분 없이 검색어를 포함하는 라인에 대해 라인 번호 및 파일 내용을 화면에 출력

✓ **검색 결과 전송 및 클라이언트 출력 기능** (5점)

- 서버는 검색 결과를 RES_PACKET의 matched_lines[] 배열에 저장하여 클라이언트로 전송함 (검색된 결과는 matched_lines[2048]에 저장 후 한 번에 전송함) (3점)
  - 검색 결과가 없는 경우 result=0 저장,
  - 검색 결과가 있는 경우, result=검색된 라인수 저장
- 클라이언트는 GREP_RES 메시지 내용을 화면에 출력 (2점)

■ GREP_END, GREP_END_ACK 메시지 처리 기능 (4점)

✓ 클라이언트가 화면에서 "quit" 메시지를 입력하면 REQ_PACKET에 cmd=GREP_END를 저장하여 서버로 전송 후 GREP_END_ACK를 수신하면 클라이언트 종료 (2점)

✓ GREP_END를 수신한 서버는 RES_PACKET cmd=GREP_END_ACK를 저장하여 전송 후 서버는 종료함 (2점)

■ 기능 검증 방법

✓ grep -n 검색어 파일명

✓ grep -nv 검색어 파일명

✓ grep -ni 검색어 파일명

■ **실행 결과: 서버**

```
$ ./server 9190
[Rx] GREP_REQ(1), options: -n connect test.txt
File not found: test.txt
-----------------------------------
[Tx] GREP_RES(2), result: -1        <--- test.txt 파일 없음
-----------------------------------
```

```
[Rx] GREP_REQ(1), options: -x connect network.txt
Invalid option: -x
-----------------------------------
[Tx] GREP_RES(2), result: -2          <--- "-x" 옵션이 없기 때문에 -2 리턴
-----------------------------------
[Rx] GREP_REQ(1), options: -n program network.txt
-----------------------------------
[Tx] GREP_RES(2), result: 3
-----------------------------------
  1: Computer network programming
  3: Computer network programming involves writing computer programs
  9: by the communication protocol, and not by application programming interface(API).
-----------------------------------
[Rx] GREP_REQ(1), options: -v connect network.txt
-----------------------------------
[Tx] GREP_RES(2), result: 20
-----------------------------------
  1: Computer network programming
  2:
  3: Computer network programming involves writing computer programs
  4: that enable processes to communicate with each other across a computer network.
  5:
  9: by the communication protocol, and not by application programming interface(API).
 12:
 13: Clients and servers
 16: this party is usually referred to as "server".
 18: this party is usually referred to as "client".
 19:
 20: Transmission Control Protocol (TCP)
 21: The Transmission Control Protocol (TCP) is one of the Internet protocol suite.
 22: It originated in the initial network implementation in which it complemented
 23: the Internet Protocol (IP).
 24: Therefore, the entire suite is commonly referred to as TCP/IP.
 25:
 26: User Datagram Protocol (UDP)
 29: UDP provides checksums for data integrity, and port numbers for addressing different
 30: functions at the source and destination of the datagram.
-----------------------------------
[Rx] GREP_REQ(1), options: -i computer network.txt
-----------------------------------
[Tx] GREP_RES(2), result: 3
-----------------------------------
  1: Computer network programming
  3: Computer network programming involves writing computer programs
  4: that enable processes to communicate with each other across a computer network.
-----------------------------------
[Rx] GREP_REQ(1), options: -i protocol udp.txt
-----------------------------------
[Tx] GREP_RES(2), result: 8
-----------------------------------
  1:                          User Datagram Protocol
  6: This User Datagram  Protocol  (UDP)  is  defined  to  make  available  a
  9: protocol  assumes  that the Internet  Protocol  (IP)  [1] is used as the
```

```
  10: underlying protocol.
  12: This protocol  provides  a procedure  for application  programs  to send
  13: messages  to other programs  with a minimum  of protocol mechanism.  The
  14: protocol  is transaction oriented, and delivery and duplicate protection
  16: streams of data should use the Transmission Control Protocol (TCP) [2].
-----------------------------------
[Rx] GREP_REQ(1), options: -n aaa udp.txt     <-- 검색어 "aaa"는 udp.txt 에 없음
-----------------------------------
[Tx] GREP_RES(2), result: 0
-----------------------------------
-----------------------------------
[Rx] GREP_END(3)
[Tx] GREP_END_ACK(3)

Exit rGrep Server
$
```

■ 실행 결과: 클라이언트

```
$ ./client 127.0.0.1 9190
Type [option] [keyword] [filename]: -n connect test.txt
[Tx] GREP_REQ(1) options: -n connect test.txt
-----------------------------------
[Rx] GREP_RES(2), result: -1
-----------------------------------
File not found!
Type [option] [keyword] [filename]: -x connect network.txt
[Tx] GREP_REQ(1) options: -x connect network.txt
-----------------------------------
[Rx] GREP_RES(2), result: -2
-----------------------------------
Invalid option
Type [option] [keyword] [filename]: -n program network.txt
[Tx] GREP_REQ(1) options: -n program network.txt
-----------------------------------
[Rx] GREP_RES(2), result: 3
-----------------------------------
  1: Computer network programming
  3: Computer network programming involves writing computer programs
  9: by the communication protocol, and not by application programming interface(API).
-----------------------------------
Type [option] [keyword] [filename]: -v connect network.txt
[Tx] GREP_REQ(1) options: -v connect network.txt
-----------------------------------
[Rx] GREP_RES(2), result: 20
-----------------------------------
  1: Computer network programming
  2:
  3: Computer network programming involves writing computer programs
  4: that enable processes to communicate with each other across a computer network.
  5:
  9: by the communication protocol, and not by application programming interface(API).
```

```
 12:
 13: Clients and servers
 16: this party is usually referred to as "server".
 18: this party is usually referred to as "client".
 19:
 20: Transmission Control Protocol (TCP)
 21: The Transmission Control Protocol (TCP) is one of the Internet protocol suite.
 22: It originated in the initial network implementation in which it complemented
 23: the Internet Protocol (IP).
 24: Therefore, the entire suite is commonly referred to as TCP/IP.
 25:
 26: User Datagram Protocol (UDP)
 29: UDP provides checksums for data integrity, and port numbers for addressing different
 30: functions at the source and destination of the datagram.
-----------------------------------
Type [option] [keyword] [filename]: -i computer network.txt
[Tx] GREP_REQ(1) options: -i computer network.txt
-----------------------------------
[Rx] GREP_RES(2), result: 3
-----------------------------------
  1: Computer network programming
  3: Computer network programming involves writing computer programs
  4: that enable processes to communicate with each other across a computer network.
-----------------------------------
Type [option] [keyword] [filename]: -i protocol udp.txt
[Tx] GREP_REQ(1) options: -i protocol udp.txt
-----------------------------------
[Rx] GREP_RES(2), result: 8
-----------------------------------
  1:                          User Datagram Protocol
  6: This User Datagram  Protocol  (UDP)  is  defined  to  make  available  a
  9: protocol  assumes  that the Internet  Protocol  (IP)  [1] is used as the
 10: underlying protocol.
 12: This protocol  provides  a procedure  for application  programs  to send
 13: messages  to other programs  with a minimum  of protocol mechanism.  The
 14: protocol  is transaction oriented, and delivery and duplicate protection
 16: streams of data should use the Transmission Control Protocol (TCP) [2].
-----------------------------------
Type [option] [keyword] [filename]: -n aaa udp.txt
[Tx] GREP_REQ(1) options: -n aaa udp.txt
-----------------------------------
[Rx] GREP_RES(2), result: 0
-----------------------------------
-----------------------------------
Type [option] [keyword] [filename]: quit
[Tx] GREP_END(3)
[Rx] GREP_END_ACK(4)

Exit rGrep Client
$
```