|  |
| --- |
| **퓨터네트워크** |
| **과제 #05 문제 및 보고서** |

|  |  |
| --- | --- |
| **이름** | 이풍헌 |
| **학번** | 201924548 |
| **소속**  **학과/대학** | 정보컴퓨터공학과 |
| **분반** | 061 |

## <주의사항>

* 각각의 문제 바로 아래에 답을 작성 후 제출해 주세요.
* PLATO 제출 데드라인: 11/07(화)~11/20(월) 23:55까지
* 데드라인을 지나서 제출하면 0점
* 주말/휴일/학교행사 등으로 인한 데드라인 연장 없음
* 부정행위 적발 시, 원본(보여준 사람)과 복사본(베낀 사람) 모두 0점 처리함
* PLATO에 아래의 파일을 제출해 주세요
* 보고서
  + 보고서 파일명에 이름과 학번을 입력 해 주세요.
  + 보고서 첫 페이지(직전 페이지)에 이름, 학번, 소속을 작성해 주세요.
* 이번 과제를 수행하면서 소스코드, 스크립트, Makefile 등을 작성한 경우, 모든 파일을 압축하여 제출하세요(미제출 시 감점)

## <개요>

이번 과제는 소켓 프로그래밍을 통한 통신 프로그램을 구현하는 내용으로 구성되어 있습니다.

소켓 프로그래밍 참고자료는 아래 링크를 방문하세요

|  |
| --- |
| AF\_UNIT 도메인 소켓 프로그래밍 예시:  - 서버 : https://www.ibm.com/docs/en/i/7.2?topic=uauaf-example-server-application-that-uses-af-unix-address-family  - 클라이언트 : https://www.ibm.com/docs/en/i/7.2?topic=uauaf-example-client-application-that-uses-af-unix-address-family |
| AF\_INET 도메인 소켓 프로그래밍 예시  - 서버 & 클라이언트: https://www.geeksforgeeks.org/socket-programming-cc/ |

\*\* 주의: 전체 소스코드를 압축하여 첨부파일로 제출하세요.

## <실습 과제>

|  |
| --- |
| **[Q 1] AF\_UNIX 도메인 STREAM 소켓 프로그래밍 [배점: 30]**  AF\_UNIX/SOCK\_STREAM 타입의 소켓을 사용하는 서버-클라이언트 프로그램을 작성하는 문제입니다. 소켓 파일 경로는 ./sock\_addr 를 사용하세요.  간단한 단방향 메시지 전송 프로그램을 작성하는 구현하는 문제입니다. 서버-클라이언트간 연결이 설정되면(=클라이언트의 connect 요청에 서버가 accept로 반응하면)…   * 클라이언트 프로그램은 사용자 터미널로부터 전달받은 문자열을 서버로 전송하고 * 서버 프로그램은 클라이언트로부터 받은 메시지를 터미널에 출력하는   과정을 반복합니다. 클라이언트에서 \quit 이라는 메지시를 입력하면 클라이언트와 서버 프로그램 모두 종료합니다. 주의: 서버 프로그램을 먼저 실행하고, 다음으로 클라이언트 프로그램을 실행하세요. 클라이언트에서 사용자가 \quit 메시지를 입력하면, 서버로 해당 메시지를 전달하고 난 다음에 클라이언트를 종료하세요.   * (문제 1) 서버와 클라이언트 프로그램을 서로 다른 터미널에서 구동하고, 그 상태에서 세 번째 터미널을 실행하세요. 서버 및 클라이언트 프로그램이 저장된 디렉토리로 이동한 뒤 $ls 명령을 입력하세요. 터미널 결과를 캡처하여 아래에 첨부하세요. 주의: $ls 명령의 결과에 sock\_addr 이라는 파일이 나타나야 합니다. * (문제 2) 서버-클라이언트 프로그램을 구동하고, 클라이언트는 1) hello world 메시지를 보내고, 다음으로 2) nice to meet you 메시지를 보내고, 마지막으로 3) \quit 메시지를 전송합니다. 서버측 터미널 출력 결과를 캡처하여 아래에 첨부하세요. |

답변 1) 텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

답변 2)

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

|  |
| --- |
| **[Q 2] AF\_INET 도메인 STREAM 소켓 프로그래밍 : 동기형 1대1 채팅 [배점: 30]**  AF\_INET/SOCK\_STREAM 타입의 서버-클라이언트 소켓 프로그램을 작성하는 문제입니다. 이번에는 서버와 클라이언트가 1:1로 순서에 따라 채팅하는 프로그램을 작성하세요.  동기화된 양방향 메시지 전송 프로그램을 작성하는 구현하는 문제입니다. 서버-클라이언트간 연결이 설정되면(=클라이언트의 accept 요청에 서버가 accept로 반응하면) 아래의 동작을 순서대로 반복합니다.   1. [클라이언트] 사용자로부터 전달받은 문자열을 서버로 전송 (사용자는 터미널에서 문자열 입력) 2. [서버] 클라이언트로부터 받은 메시지를 터미널에 출력 3. [서버] 사용자 터미널로부터 전달받은 문자열을 클라이언트로 전송 4. [클라이언트] 프로그램은 서버로부터 받은 메시지를 터미널에 출력 5. 위의 1~4 과정을 반복 (서버와 클라이언트는 순서에 맞게 채팅 메시지를 입력함)   클라이언트 또는 서버에서 \quit 이라는 메지시를 입력하면 클라이언트와 서버 프로그램 모두 종료합니다. 상대방으로부터 전달받은 메시지를 터미널에 출력할 때, [You] 라는 문자열을 먼저 출력하고, 다음으로 상대방의 메시지를 출력하세요. 예를 들어, 클라이언트가 hello 라는 메시지를 전송하면, 서버는 [You] hello 라고 출력해야 합니다.  주의: 서버 프로그램을 먼저 실행하고, 다음으로 클라이언트 프로그램을 실행하세요. 클라이언트 >> 서버 >> 클라이언트 >> 서버 … 순으로 메시지를 전송해야 합니다 (= 동기형 1:1 채팅).  문제) 클라이언트와 서버 각각 3번씩 메시지를 입력하도록 하고, 서버와 클라이언트의 터미널을 모두 캡처해서 아래에 첨부하세요. |

답변 (서버의 터미널 화면 캡처)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

답변 (클라이언트의 터미널 화면 캡처)

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

|  |  |  |
| --- | --- | --- |
| **[Q 3] AF\_INET 도메인 STREAM 소켓 : 멀티 서비스 [배점: 40]**  이번에는 서버 프로그램과 클라이언트 프로그램을 서로 다른 디렉토리에 저장해야 합니다. 디렉토리를 아래와 같이 구성하세요.   * MultiService/server * MultiService/client   ‘server’ 디렉토리 아래에 server.c 소스코드를 생성하여 코딩하고, 첨부된 Book.txt, Linux.png 파일을 서버 쪽 디렉토리에 저장하세요. ‘client’ 디렉토리 아래에 client.c 소스코드를 생성하여 코딩하세요.  클라이언트가 서버에 접속하면, 다음과 같은 메시지를 서버로부터 전달받습니다.   |  | | --- | | [Service List]  1. Get Current Time  2. Download File  3. Echo Server  Enter: |   클라이언트는 메시지를 터미널에 출력하고, 사용자 입력을 기다립니다.   * 사용자가 1을 입력하면   + \service 1 이라는 메시지가 서버로 전달됩니다.   + 서버는 \service 1 이라는 메시지를 받으면, 현재 시간을 문자열 형태로 클라이언트에게 전달합니다 (현재 시간을 문자열 형태로 얻는 코드는 첨부된 get\_localtime.c 파일을 참고하세요). 클라이언트는 전달받은 메시지를 터미널에 출력합니다.   + 서버는 다음으로 [Service List] 메뉴 메시지를 클라이언트에게 전달합니다. 클라이언트는 메뉴 메시지를 터미널에 출력하고 사용자 입력을 기다립니다. * 사용자가 2를 입력하면   + \service 2 라는 메시지가 서버로 전달됩니다.   + 서버는 \service 2 라는 메시지를 받으면, 아래의 메시지를 클라이언트에게 전달합니다.  |  | | --- | | [Available File List]  1. Book.txt  2. Linux.png  3. Go back  Enter: |  * + 클라이언트는 메시지를 터미널에 출력하고 사용자 입력을 기다립니다.   + 사용자가     - 1 또는 2를 입력하여 서버에 전달하면, 클라이언트는 해당 파일을 서버로부터 다운 받고 서버에 저장된 파일의 이름과 동일한 이름으로 파일을 현재 디렉토리에 저장합니다. 파일 전송이 완료되면 서버는 [Service List] 메뉴 메시지를 클라이언트에게 전달합니다. 클라이언트는 메뉴 메시지를 터미널에 출력하고 사용자 입력을 기다립니다.     - 3을 입력하여 서버에 전달하면, 서버는 [Service List] 메뉴 메시지를 클라이언트에게 전달합니다. 클라이언트는 메뉴 메시지를 터미널에 출력하고 사용자 입력을 기다립니다. * 사용자가 3을 입력하면   + \service 3 이라는 메시지가 서버로 전달되고, 서버는 ECHO SERVER로 동작합니다. 즉, 클라이언트가 메시지를 입력하면, 해당 메시지는 서버로 전달되고, 동일한 메시지가 다시 클라이언트로 전달됩니다. 클라이언트는 서버로부터 수신한 메시지는 [You] … 이런 식으로 터미널에 출력합니다.     - 사용자가 \quit 이라고 입력하면 ECHO SERVER는 중지되고, 서버는 [Service List] 메뉴 메시지를 클라이언트에게 전달합니다. 클라이언트는 메뉴 메시지를 터미널에 출력하고 사용자 입력을 기다립니다. * (문제 1) 클라이언트가 1번 서비스를 사용하게 하고, 터미널 출력 화면을 캡처하여 아래에 첨부하세요. * (문제 2) 클라이언트가 2번 서비스를 사용하게 하고, 텍스트 파일과 이미지 파일을 모두 다운 받은 후, 클라이언트 터미널 출력 화면을 캡처하여 아래에 첨부하세요. 클라이언트 터미널에서, 프로그램 구동 전 $ls 결과, 그리고 구동 후 $ls 결과를 캡처하여 아래에 첨부하세요. 프로그램 구동 후에는 $ls 명령 입력 시, 다운받은 파일이 조회되어야 합니다. * (문제 3) 클라이언트가 3번 서비스를 사용하게 하고, 클라이언트가 세 번 메시지를 전송하도록 하세요. 터미널 출력 화면을 캡처하여 아래에 첨부하세요.   [참고] 터미널 화면 캡처 시, 출력된 문자열이 너무 많다면, 최근의 출력문만 캡처해도 괜찮습니다. |

답변 1)

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

답변 2) 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

답변 3)

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

**끝! 수고하셨습니다 ☺**