오픈소스 소프트웨어 실습

Open-Source Software Lab

#9

실습 담당 조교 연락처

◆실습 조교 : 김민곤

◆연구실 : 학연산클러스터 601호

◆이메일 : phenix235@hanyang.ac.kr

메일양식: [오픈소스]학번_수업일(20190000)_이름

소켓 만들기

int socket(int domain, int type, int protocol)

소켓을 생성하고 소켓을 위한 파일 디스크립터를 리턴, 실패하면 -1을 리턴

- 인터넷 소켓
  fd = socket(AF_INET, SOCK_STREAM, DEFAULT _PROTOCOL);
- 유닉스 소켓
  fd = socket(AF_UNIX, SOCK_STREAM, DEFAULT _PROTOCOL);

```
listenfd = socket(AF_UNIX, SOCK_STREAM, DEFAULT_PROTOCOL);
```

```
clientfd = socket(AF_UNIX, SOCK_STREAM, DEFAULT_PROTOCOL);
```

소켓에 이름(주소) 주기

int bind(int fd, struct sockaddr* address, int addressLen)

소켓에 대한 이름 바인딩이 성공하면 0, 실패하면 -1을 리턴

```
bind(listenfd, (struct sockaddr *) &serverAddr, sizeof(serverAddr));
```

소켓 큐 생성

int listen(int fd, int queueLength)

소켓 fd에 대한 연결 요청을 기다린다. 성공하면 0, 실패하면 -1을 리턴

```
listen(listenfd, 5);
```

소켓에 연결 요청

int connet(int fd, struct sockaddr* address, int addressLen)

성공하면 0, 실패하면 -1을 리턴

```
result = connect(clientfd, (struct sockaddr *) &serverAddr, sizeof(serverAddr));
```

연결 요청 수락

int accpet(int fd, struct sockaddr* address, int* addressLen)

성공하면 새로 만들어진 복사본 소켓의 파일 디스크립터, 실패하면 -1을 리턴

```
connfd = accept(listenfd, (struct sockaddr *) &clientAddr, &clientlen);
```

대문자 변환 서버

- 프로그램
  - ✓ 입력받은 문자열을 소문자를 대문자로 변환한다.
  - ✓ 서버와 클라이언트로 구성된다.
- 서버
  - ✓ 소켓을 통해 클라이언트로부터 받은 문자열을 소문자를 대문자로 변환하여 소켓을 통해 클라이언트에 다시 보낸다.
- 클라이언트
  - ✓ 표준입력으로부터 문자열을 입력받는다.
  - ✓ 이를 소켓을 통해 서버에 보낸다.
  - ✓ 대문자로 변환된 문자열을 다시 받아 표준출력에 출력한다.

cserver.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>
#define DEFAULT_PROTOCOL 0
#define MAXLINE 100

/* 소문자를 대문자로 변환하는 서버 프로그램 */
int main ()
{
    int listenfd, connfd, clientlen;
    char inmsg[MAXLINE], outmsg[MAXLINE];
    struct sockaddr_un serverAddr, clientAddr;

    signal(SIGCHLD, SIG_IGN);
    clientlen = sizeof(clientAddr);

    listenfd = socket(AF_UNIX, SOCK_STREAM, DEFAULT_PROTOCOL);
    serverAddr.sun_family = AF_UNIX;
    strcpy(serverAddr.sun_path, "convert");
    unlink("convert");
    bind(listenfd, (struct sockaddr *) &serverAddr, sizeof(serverAddr));
```

cserver.c

```
    unlink("convert");
    bind(listenfd, (struct sockaddr *) &serverAddr, sizeof(serverAddr));

    listen(listenfd, 5);

    while (1) {  /* 소켓 연결 요청 수락 */
        connfd = accept(listenfd, (struct sockaddr *) &clientAddr, &clientlen);
        if (fork ( ) == 0) {
            /* 소켓으로부터 한 줄을 읽어 대문자로 변환하여 보냄 */
            readLine(connfd, inmsg);
            toUpper(inmsg, outmsg);
            write(connfd, outmsg, strlen(outmsg)+1);
            close(connfd);
            exit (0);
        } else close(connfd);
    }
}
```

cserver.c

```c
/* 소문자를 대문자로 변환 */
toUpper(char* in, char* out)
{
    int i;
    for (i = 0; i < strlen(in); i++)
        if (islower(in[i]))
            out[i] = toupper(in[i]);
        else out[i] = in[i];
    out[i] = '\0';
}

/* 한 줄 읽기 */
readLine(int fd, char* str)
{
    int n;
    do {
        n = read(fd, str, 1);
    } while(n > 0 && *str++ != '\0');
    return(n > 0);
}
```

cclient.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>
#define DEFAULT_PROTOCOL 0
#define MAXLINE 100

/* 소문자-대문자 변환: 클라이언트 프로그램 */
int main ( )
{
    int clientfd, result;
    char inmsg[MAXLINE], outmsg[MAXLINE];
    struct sockaddr_un serverAddr;

    clientfd = socket(AF_UNIX, SOCK_STREAM, DEFAULT_PROTOCOL);
    serverAddr.sun_family = AF_UNIX;
    strcpy(serverAddr.sun_path, "convert");

    do {          /* 연결 요청 */
        result = connect(clientfd, (struct sockaddr *) &serverAddr, sizeof(serverAddr));
        if (result == -1) sleep(1);
    } while (result == -1);
```

cclient.c

```
    } while (result == -1);

    printf("변환할 문자열 입력:\n");
    fgets(inmsg, MAXLINE, stdin);
    write(clientfd,inmsg,strlen(inmsg)+1);  // 변환할 문자열 보내기

    /* 소켓으로부터 변환된 문자열을 한 줄 읽어서 프린트 */
    readLine(clientfd,outmsg);
    printf("%s --> \n%s", inmsg, outmsg);
    close(clientfd);
    exit(0);
}

readLine(int fd, char* str)
{
    int n;
    do {
        n = read(fd, str, 1);
    } while (n > 0 && *str++ != '\0');
    return(n > 0);
}
```

대문자 변환 서버

1. 서버 실행

```
[1111222333@node1 class9]$ ./server
```

2. 클라이언트 실행

```
[1111222333@node1 class9]$ ./c
변 환 할  문 자 열  입 력 :
hello world!
hello world!
  -->
HELLO WORLD!
```

DNS 관련 함수

struct hostent *gethostbyaddr(const char* addr, int len, int type);

길이가 len이고 주소 타입 type인 호스트 주소 addr에 해당하는 hostent 구조체를 리턴

struct hostent *gethostbyname(char* name);

도메인 이름에 대응하는 hostent 구조체에 대한 포인터를 리턴

DNS 관련 함수

char* inet_ntoa(struct in_addr address);

IP 주소 address에 대응하는 A. B. C. D 포맷의 스트링을 리턴

unsigned long inet addr(char* string);

A. B. C. D 포맷의 IP 주소를 네트워크 바이트 순서로 된 이진 데이터로 변환하여 리턴

파일 서버 클라이언트

- 서버
  - ✓ 파일 이름을 받아 해당 파일을 찾아 그 내용을 보내주는 서비스
  - ✓ 명령줄 인수로 포트 번호를 받아 해당 소켓을 만든다.
  - ✓ 이 소켓을 통해 클라이언트로부터 파일 이름을 받아 해당 파일을 열고 그 내용을 이 소켓을 통해 클라이언트에게 보낸다.
- 클라이언트
  - ✓ 명령줄 인수로 연결할 서버의 이름과 포트 번호를 받아 해당 서버에 소켓 연결을 한다.
  - ✓ 이 연결을 통해 서버에 원하는 파일 이름을 보낸다.
  - ✓ 서버로부터 해당 파일 내용을 받아 사용자에게 출력한다.

fserver.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#define DEFAULT_PROTOCOL 0
#define MAXLINE 100

/* 파일 서버 프로그램 */
int main (int argc, char* argv[])
{
    int sfd, cfd, port, clientlen;
    FILE *fp;
    char inmsg[MAXLINE], outmsg[MAXLINE];
    struct sockaddr_in serveraddr, clientaddr;
    struct hostent *hp;
    char *haddrp;

    signal(SIGCHLD, SIG_IGN);

    if (argc != 2) {
        fprintf(stderr, "사용법: %s <port>\n", argv[0]);
        exit(0);
    }
```

fserver.c

```
    port = atoi(argv[1]);
    sfd = socket(AF_INET, SOCK_STREAM, DEFAULT_PROTOCOL);

    bzero((char *) &serveraddr, sizeof(serveraddr));
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);
    serveraddr.sin_port = htons((unsigned short)port);
    bind(sfd, (struct sockaddr *) &serveraddr, sizeof(serveraddr));
    listen(sfd, 5);

    while (1) {
        clientlen = sizeof(clientaddr);
        cfd = accept(sfd, (struct sockaddr *) &clientaddr, &clientlen); // 연결 요청 수락
        haddrp = inet_ntoa(clientaddr.sin_addr);
        printf("서버: %s (%d)에 연결됨\n", haddrp, clientaddr.sin_port);

        if (fork ( ) == 0) {
            readLine(cfd, inmsg); /* 소켓에서 파일 이름을 읽는다 */
            fp = fopen(inmsg, "r");
        if (fp == NULL) {
               write(cfd, "해당 파일 없음", 10);
            } else {    /* 파일에서 한 줄씩 읽어 소켓을 통해 보낸다 */
                while(fgets(outmsg, MAXLINE, fp) != NULL)
                    write(cfd, outmsg, strlen(outmsg)+1);
            }
            close(cfd);
            exit (0);
        } else close(cfd);
    }
}
```

fserver.c

```
readLine(int fd, char* str)
{
    int n;
    do {
        n = read(fd, str, 1);
    } while (n > 0 && *str++ != '\0');
    return(n > 0);
}
```

fclient.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#define DEFAULT_PROTOCOL 0
#define MAXLINE 100

/* 파일 클라이언트 프로그램 */
int main (int argc, char* argv[])
{
    int sfd, port, result;
    char *host, inmsg[MAXLINE], outmsg[MAXLINE];
    struct sockaddr_in serverAddr;
    struct hostent *hp;

    if (argc != 3) {
        fprintf(stderr, "사용법 : %s <host> <port>\n", argv[0]);
        exit(0);
    }
```

fclient.c

```
host = /**/;
port = atoi(/**/);

sfd = socket(/**/);

/* 서버의 IP 주소와 포트 번호를 채운다. */
if ((hp = gethostbyname(host)) == NULL)
    perror("gethostbyname error"); // 호스트 찾기 오류
bzero((char *) &serverAddr, sizeof(serverAddr));
serverAddr.sin_family = AF_INET;
bcopy((char *)hp->h_addr_list[0],
      (char *)&serverAddr.sin_addr.s_addr, hp->h_length);
serverAddr.sin_port = htons(port);

do { /* 연결 요청 */
    result = connect(/**/);
    if (result == -1) sleep(1);
} while (result == -1);

printf("다운로드할 파일 이름 입력:");
scanf("%s", inmsg);
write(sfd,inmsg,strlen(inmsg)+1);
```

fclient.c

```
    /* 소켓으로부터 파일 내용 읽어서 프린트 */
    while (/**/)
        /**/
    close(sfd);
    exit(0);
}

readLine(int fd, char* str)
{
    int n;
    do {
        n = read(fd, str, 1);
    } while (n > 0 && *str++ != '\0');
    return(n > 0);
}
```

실습 과제

✓ /**/ 채워 fclient.c 완성

✓ 실행화면

```
[1111222333@node1 class9]$ ./fserver 9999
서 버 : 127.0.0.1 (41618)에  연 결 됨
```

```
[1111222333@node1 class9]$ ./fclient 127.0.0.1 9999
다 운 로 드 할  파 일  이 름  입 력 :hello.txt
hello world!
```

✓ fclient.c 작성 내용과 실행화면 캡쳐(총2개)해서 양식 맞춰 메일

끝