OS/NW 과제-6

단일 서버, 멀티 클라이언트 연결(fork) & 문자열 합치기

32184074 컴퓨터공학과 정민준 - 2022년 10월 31일

본문

<echo_client.c 코드>

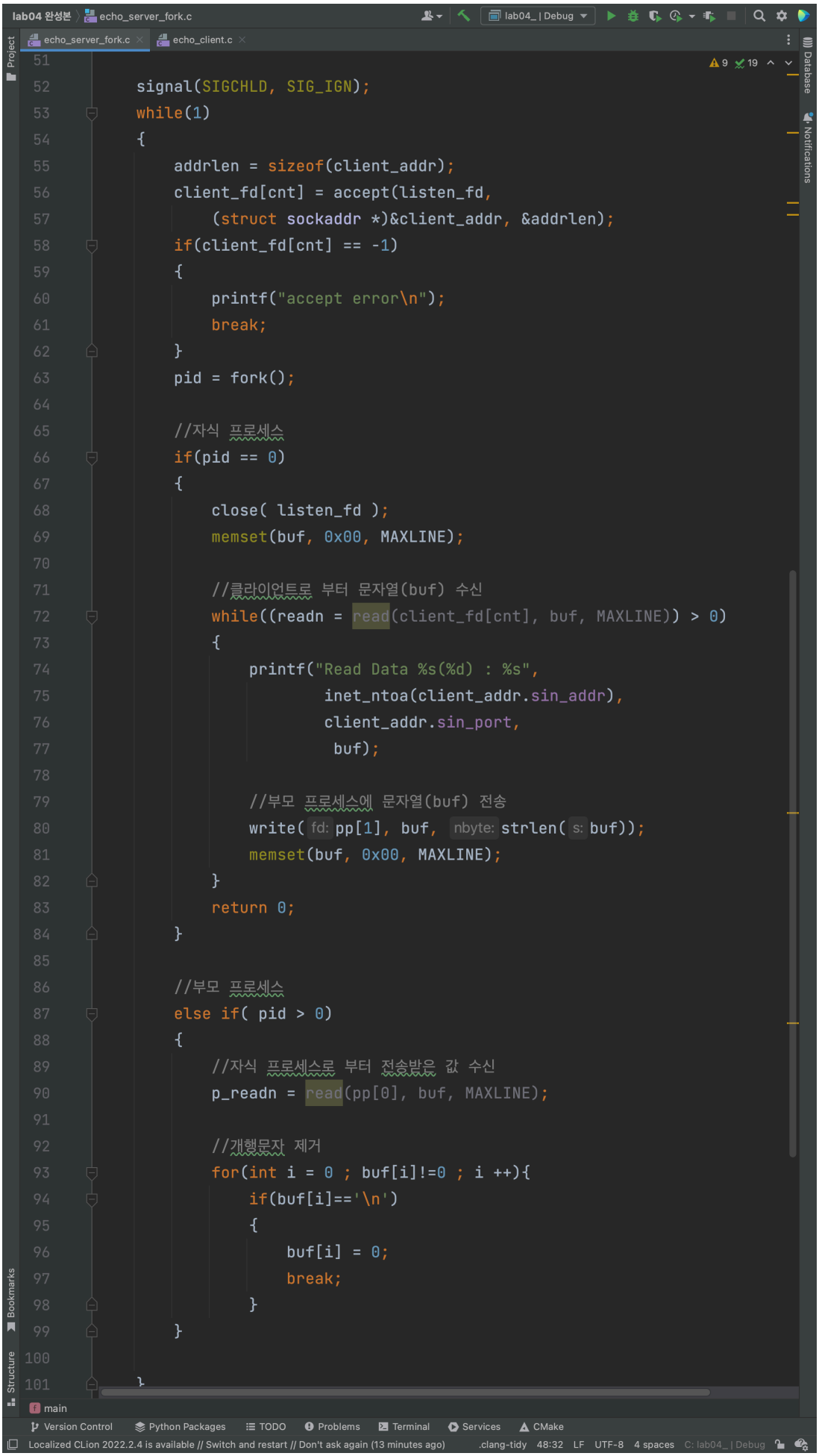
```c
int main(int argc, char **argv)
{
    struct sockaddr_in serveraddr;
    int server_sockfd;
    int client_len;
    char buf[MAXLINE];

    if ((server_sockfd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
    {
        perror("error :");
        return 1;
    }

    /* 연결요청할 서버의 주소와 포트번호 프로토콜등을 지정한다. */
    server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    serveraddr.sin_port = htons(3600);

    client_len = sizeof(serveraddr);

    /* 서버에 연결을 시도한다. */
    if (connect(server_sockfd, (struct sockaddr *)&serveraddr,
            client_len)  == -1)
    {
        perror("connect error :");
        return 1;
    }

    memset(buf, 0x00, MAXLINE);
    read(0, buf, MAXLINE);    /* 키보드 입력을 기다린다. */

    /* 입력 받은 데이터를 서버로 전송한다. */
    if (write( fd: server_sockfd, buf,  nbyte: MAXLINE) <= 0)
    {
        perror("write error : ");
        return 1;
    }
    memset(buf, 0x00, MAXLINE);
    /* 서버로 부터 데이터를 읽는다. */
    if (read(server_sockfd, buf, MAXLINE) <= 0)
    {
        perror("read error : ");
        return 1;
    }
    printf("read : %s", buf);
    printf("\n");
    close(server_sockfd);
    return 0;
}
```

<echo_client.c>

<echo_server_fork.c 코드>

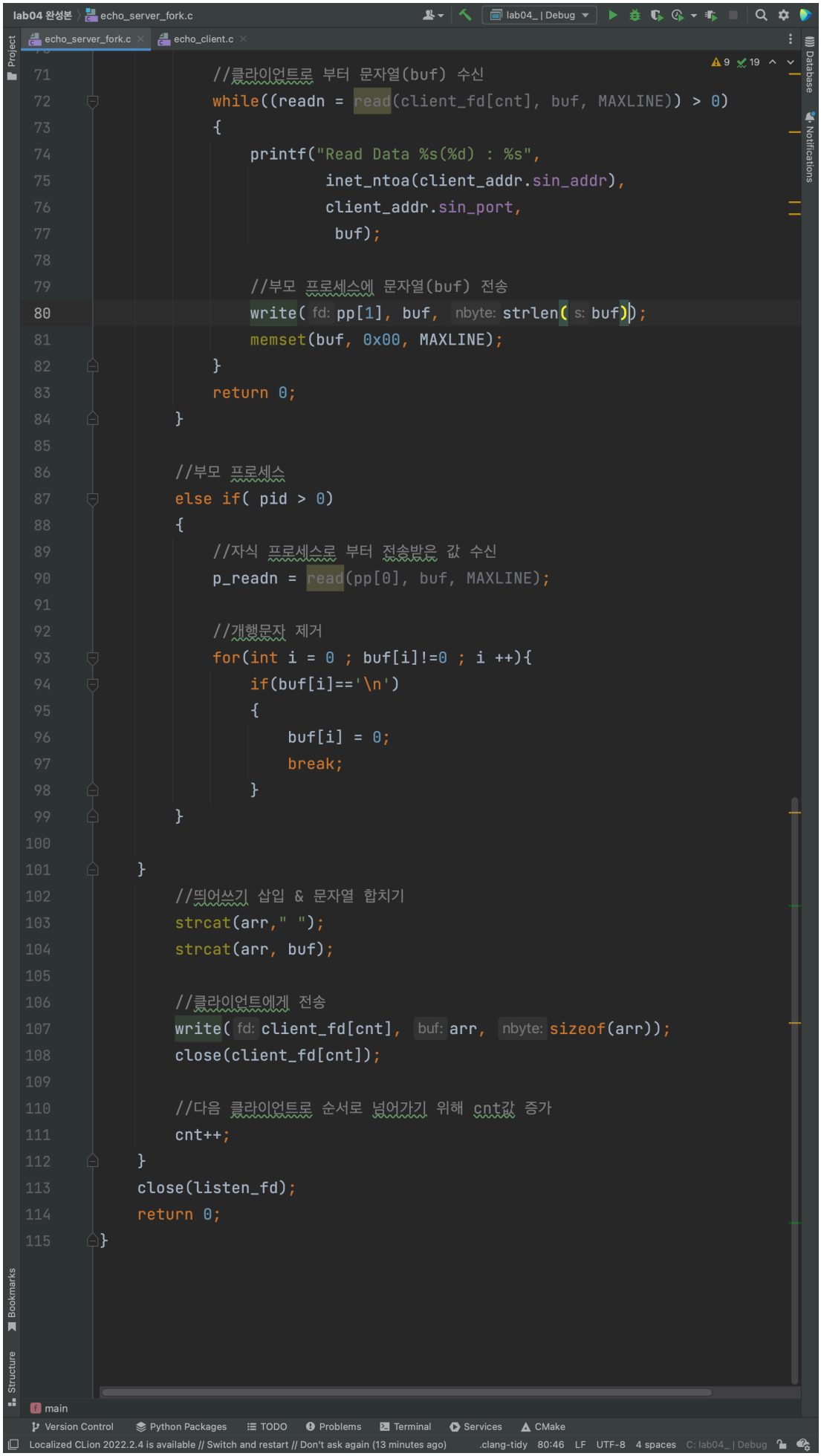
```c
#include ...

#define MAXLINE 1024
#define PORTNUM 3600

int main(int argc, char **argv)
{
    int listen_fd;
    int client_fd[3]; //클라이언트가 담길 배열 3개
    pid_t pid;
    socklen_t addrlen;
    int readn;
    int p_readn;
    char buf[MAXLINE] = {};
    struct sockaddr_in client_addr, server_addr;

    char arr[MAXLINE] = {}; //합쳐질 문자가 담길 문자열 초기화
    int cnt = 0; //클라이언트의 순서를 나타냄

    int pp[2]; //pipe 통신
    pipe(pp);


    if( (listen_fd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        return 1;
    }
    memset((void *)&server_addr, 0x00, sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    server_addr.sin_port = htons(PORTNUM);

    if(bind(listen_fd, (struct sockaddr *)&server_addr, sizeof(server_addr)
    {
        perror("bind error");
        return 1;
    }
    if(listen(listen_fd, 5) == -1)
    {
        perror("listen error");
        return 1;
    }

    signal(SIGCHLD, SIG_IGN);
    while(1)
    {
        addrlen = sizeof(client_addr);
        client_fd[cnt] = accept(listen_fd,
            (struct sockaddr *)&client_addr, &addrlen);
        if(client_fd[cnt] == -1)
```

<echo_server_fork.c - 1>



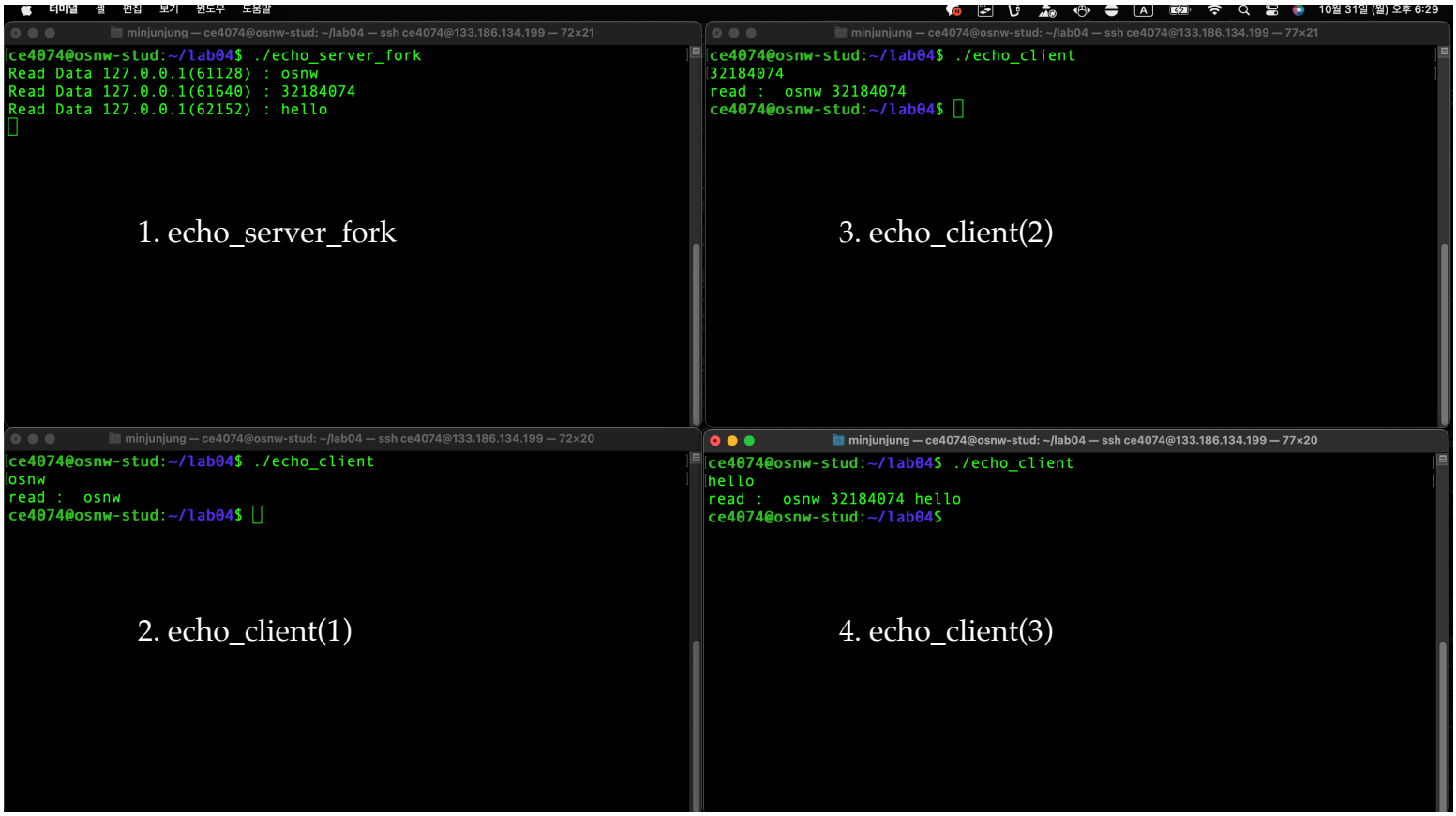

```
51
52      signal(SIGCHLD, SIG_IGN);
53      while(1)
54      {
55          addrlen = sizeof(client_addr);
56          client_fd[cnt] = accept(listen_fd,
57              (struct sockaddr *)&client_addr, &addrlen);
58          if(client_fd[cnt] == -1)
59          {
60              printf("accept error\n");
61              break;
62          }
63          pid = fork();
64
65          //자식 프로세스
66          if(pid == 0)
67          {
68              close( listen_fd );
69              memset(buf, 0x00, MAXLINE);
70
71              //클라이언트로 부터 문자열(buf) 수신
72              while((readn = read(client_fd[cnt], buf, MAXLINE)) > 0)
73              {
74                  printf("Read Data %s(%d) : %s",
75                         inet_ntoa(client_addr.sin_addr),
76                         client_addr.sin_port,
77                          buf);
78
79                  //부모 프로세스에 문자열(buf) 전송
80                  write( fd: pp[1], buf,  nbyte: strlen( s: buf));
81                  memset(buf, 0x00, MAXLINE);
82              }
83              return 0;
84          }
85
86          //부모 프로세스
87          else if( pid > 0)
88          {
89              //자식 프로세스로 부터 전송받은 값 수신
90              p_readn = read(pp[0], buf, MAXLINE);
91
92              //개행문자 제거
93              for(int i = 0 ; buf[i]!=0 ; i ++){
94                  if(buf[i]=='\n')
95                  {
96                      buf[i] = 0;
97                      break;
98                  }
99          }
100
101     }
```

<echo_server_fork.c - 2>



<echo_server_fork.c - 3>

<결과 화면>



<결과 출력>

입력 받은 문자열값을 순서대로 연결, 각 클라이언트에 순서대로 반환 해주고 있습니다.

1. 3명의 클라이언트를 받아야 하기 때문에 정수형 client_fd 배열을 크기 3으로 선언합니다.
2. fork()를 사용해서 자식 프로세스를 생성해서 클라이언트로 부터 입력을 받습니다.
3. 클라이언트로 부터 입력받은 값을 자식 프로세스는 pipe 통신으로 부모 프로세스로 전달합니다.
4. 부모 프로세스는 입력받은 값을 개행문자를 제거하고 strcat을 통해서 빈 문자열 arr을 이어 붙이면서 새로운 문자열을 만듭니다.
5. 클라이언트에 새롭게 만들어진 문자열 arr을 보내줍니다.