**lect11 HW**

**시스템프로그래밍 1분반 12180626 성시열**

4) Implement simple ftp server and client. If the client doesn't follow the protocol, the server should stop the communication. This is not chatting program. Do not type "hello", "what file do you want?", etc. The server and client should automatically send or receive the ftp protocol messages. The user will only provide the file name to download. Do not use the code from Problem 3, which will make the coding very hard. Modify the code from Problem 1.

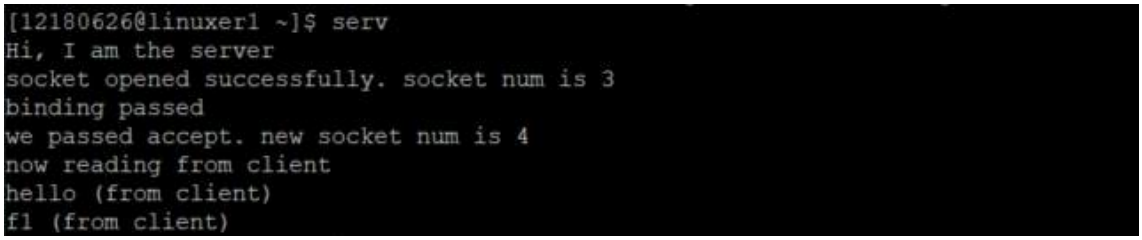
Simple ftp protocol

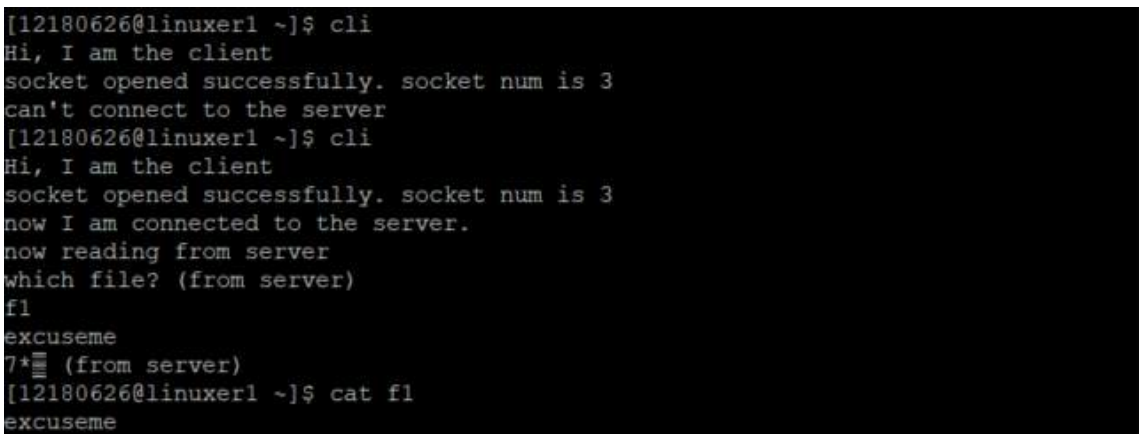
client => server: hello
server => client: what file do you want?
client => server: file name
server => client: file contents

```
    // read msg from client
    // hello
    printf("now reading from client\n");
    y=read(s2, buf, 50);
    buf[y]=0;
    printf("%s (from client)\n", buf);

    //which file?
    write(s2, "which file?", 11);

    //file name
    y = read(s2, buf, 50);
    buf[y] = 0;
    printf("%s (from client)\n", buf);
    file = open(buf, O_RDONLY, 00777);
    if (file < 0){
       printf("file not found\n");
       close(s2);
       close(s1);
       exit(1);
    }

    //content of file
    y = read(file, buf2, 50);
    write(s2, buf2, 50);

    close(s2);    // disconnect the connection
    close(s1);    // close the original socket
}
```

```
    // send msg to the server
    // hello
    printf("now I am connected to the server.\n");
    write(x, "hello", 5);

    // read from server
    printf("now reading from server\n");
    // which file?
    y=read(x, buf, 50);
    buf[y]=0;
    printf("%s (from server)\n", buf);

    // send file name to the server
    // file name
    gets(fname);
    write(x, fname, 50);
    // content of file
    y = read(x, buf, 50);
    buf[y] = 0;
    printf("%s (from server)\n", buf);

    close(x);  // disconnect the communication
}
```

serv.c와 cli.c를 한 단계씩 비교하며 코드를 작성하였다.
먼저 서버와 클라이언트가 연결된 뒤,
 클라이언트에서 "hello"라는 문자열을 x에 저장하여 서버에 전달한다.
 서버는 "hello"라는 문자열을 s2에서 받고 이를 buf에 저장한 뒤 출력한다.
서버에서 "which file?"이라는 문자열을 s2에 저장하여 클라이언트에 전달한다.
클라이언트는 "which file?"라는 문자열을 x에서 받고 이를 buf에 저장한 뒤 출력한다.
 클라이언트에서 내용을 알고싶은 파일이름을 gets를 통해 입력받고 x에 저장한다.
 서버는 s2에 넘어온 파일이름을 buf에 저장하고, open으로 파일번호를 file에 저장한다.
 file이 0보다 작을 때의 오류상황을 조건문을 통해 처리한다.
서버에서 클라이언트가 넘겨준 파일의 번호가 file이고, 이 내용을 buf2에 저장한 뒤, s2에 저장하여 클라이언트에게 넘겨준다.
클라이언트는 파일내용을 buf에 받고 이를 출력한다.




컴파일 후 실행한 결과, 값이 전달됨을 확인할 수 있었고, 클라이언트 측에서 7*()라는 문자들이 추가로 출력되었는데 원인을 찾을 수 없었다.

==> 추후에 확인해본 결과 파일에 저장된 값을 최대 50byte를 읽고 그 값을 그대로 printf()하여 뒤 쪽의 쓰레기값까지 출력된 것으로 예상된다. 6번에서 write를 활용하여 오류를 해결하였다.

5) Modify your ftp server such that it can handle multiple clients at the same time.

```
for(i=0; i<20; i++){
   // read msg from client
   // hello
   s2 = accept(s1, (struct sockaddr *)&cli_addr, &xx);
   ch = fork();
   if(ch==0){
      printf("now reading from client\n");
      y=read(s2, buf, 50);
      buf[y]=0;
      printf("%s (from client)\n", buf);

      //which file?
      write(s2, "which file?", 11);

      //file name
      y = read(s2, buf, 50);
      buf[y] = 0;
      printf("%s (from client)\n", buf);
      file = open(buf, O_RDONLY, 00777);
      if (file < 0){
         printf("file not found\n");
         close(s2);
         close(s1);
         exit(1);
      }

      //content of file
      y = read(file, buf2, 50);
      write(s2, buf2, 50);

      close(s2);    // disconnect the connection
      close(s1);    // close the original socket
      exit(0);
   }
}
```

serv.c의 코드를 수정하였다. 상단에 정수형 변수 i, ch를 선언하였다 이후 handling multiple clients code templit에 따라 코드를 작성하였다. 반복문에서 서버는 클라이언트와 연결된 뒤 다음 클라이언트를 기다리는 과정(accept)을 20번 진행한다. 새로운 client가 들어오면 그 client를 처리할 child를 fork()를 통해 만들어준다. 이후 child일 때 이전에 실행했던 코드를 실행시켜준다.

만약 성공적으로 파일 전달을 마쳤다면 exit(0)을 통해 프로그램을 빠져나가 fork()가 무한히 반복되는 것을 막는다.

```
[12180626@linuxer1 ~]$ serv
Hi, I am the server
socket opened successfully. socket num is 3
binding passed
we passed accept. new socket num is -2097141952
now reading from client
hello (from client)
f1 (from client)
now reading from client
hello (from client)
f2 (from client)
now reading from client
hello (from client)
f3 (from client)
```

```
[12180626@linuxer1 ~]$ cli
Hi, I am the client
socket opened successfully. socket num is 3
now I am connected to the server.
now reading from server
which file? (from server)
f1
excuseme
% (from server)
[12180626@linuxer1 ~]$ cli
Hi, I am the client
socket opened successfully. socket num is 3
now I am connected to the server.
now reading from server
which file? (from server)
f2
12180626
% (from server)
[12180626@linuxer1 ~]$ cli
Hi, I am the client
socket opened successfully. socket num is 3
now I am connected to the server.
now reading from server
which file? (from server)
f3
12180626
% (from server)
```

컴파일 후 실행결과 첫 cli에서 하나의 파일을 전달받고 서버는 다음 client가 오길 기다린다. 이후 반복적으로 cli를 실행하면 계속해서 여러 파일을 전달 받을 수 있다.

6) Write a client in your PC as follows and let it talk to the server program in the lab server. To compile the client program:

PC에서 실행하기 전 어디서 어디로 전송하는 것인지 명확히 하기 위해 코드를 수정하였다. 컴파일 후 실행결과 결과값을 통해 파일의 흐름을 좀 더 명확히 파악할 수 있었다.

```
   for(i=0; i<20; i++){
      // read msg from client
      // below
      s2 = accept(s1, (struct sockaddr *)&cli_addr, &xx);
      ch = fork();
      if(ch==0){
         printf("now reading from client\n");
         y=read(s2, buf, 50);
         buf[y]=0;
         printf("client => server : %s\n", buf);

         //which file?
         write(s2, "which file?", 11);
         printf("server => client : which file?\n", buf);

         //file name
         y = read(s2, buf, 50);
         buf[y] = 0;
         printf("client => server : %s\n", buf);

         file = open(buf, O_RDONLY, 00777);
         if (file < 0){
            printf("file not found\n");
            close(s2);
            close(s1);
            exit(0);
         }

         //content of file
         printf("server => client : contents of file\n", buf);
         for(;;){
            y = read(file, buf, 50);
            if (y==0) break;
            write(s2, buf, y);
            write(1, buf, y);
         }
         printf("\n");
         close(file);
         close(s2);    // disconnect the connection
         close(s1);    // close the original socket
         exit(0);
      }
      close(s2);
   }
   close(s1);
}
```

```
   // send msg to the server
   // hello
   printf("now I am connected to the server.\n");
   write(x, "hello", 5);
   printf("client => server : hello\n");
   // read from server
   // which file?
   y=read(x, buf, 50);
   buf[y]=0;
   printf("server => client : %s\n", buf);

   // send file name to the server
   // file name
   printf("client => server : ");
   gets(fname);
   write(x, fname, 50);
   // content of file
   printf("server => client : file contents\n");
   for(;;){
      y = read(x, buf, 50);
      if(y==0) break;
      write(1, buf, y);
   }
   close(x);  // disconnect the communication
```

```
[12180626@linuxer1 ~]$ serv
Hi, I am the server
socket opened successfully. socket num is 3
binding passed
we passed accept. new socket num is 1483598608
now reading from client
client => server : hello
server => client : which file?
client => server : f1
server => client : contents of file
excuseme

now reading from client
client => server : hello
server => client : which file?
client => server : f2
server => client : contents of file
12180626

now reading from client
client => server : hello
server => client : which file?
client => server : f3
server => client : contents of file
12180626
```

```
[12180626@linuxer1 ~]$ cli
Hi, I am the client
socket opened successfully. socket num is 3
now I am connected to the server.
client => server : hello
server => client : which file?
client => server : f1
server => client : file contents
excuseme
[12180626@linuxer1 ~]$ cli
Hi, I am the client
socket opened successfully. socket num is 3
now I am connected to the server.
client => server : hello
server => client : which file?
client => server : f2
server => client : file contents
12180626
[12180626@linuxer1 ~]$ cli f3
Hi, I am the client
socket opened successfully. socket num is 3
now I am connected to the server.
client => server : hello
server => client : which file?
client => server : f3
server => client : file contents
12180626
```

이후 Visual Studio에서
프로젝트 > 속성 > 매니페스트 도구 > 입력 및 출력 > 매니페스트 포함 > 아니요
프로젝트 > 속성 > 링커 > 입력 > 추가 종속성 > 드롭박스 > 편집 > ws2_32.lib 추가
를 통해 설정을 하고 강의노트의 c++ 코드를 복사 붙여넣기 하였다. 이 때 port 번호를 62608로, IP 주소를 165.246.38.151로 맞춰준다.
이후 F7를 통해 솔루션 빌드를 진행하였다.

```
[12180626@linuxer1 ~]$ serv
Hi, I am the server
socket opened successfully. socket num is 3
binding passed
we passed accept. new socket num is -1049434000
now reading from client
client => server : Hi I'm Visual Studio.
server => client : which file?
client => server :
file not found
```

```
선택 Microsoft Visual Studio 디버그 콘솔
enter a string to send to server
Hi I'm Visual Studio.
Bytes Sent: 21
received: which file?

C:₩Users₩User₩Desktop₩시스템프로그래밍₩lect11-network2₩syspr
드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

serv를 미리 실행시킨 상태로 VisualStudio에서 디버깅하였다. server로 보낼 문장을 보낸 결과 서버에서 받고 기존에 보내려던 which file?이라는 문자열이 전달된 것을 알 수 있다.

7) Write an ftp client in your PC and let it talk to the ftp server you made in problem 5). Use this client to download a file from the lab server.

```
59      //---- SEND bytes ------------------------------------------
60      printf("enter a string to send to server₩n");
61      gets_s(buf, 99);
62      bytesSent = send(s, buf, strlen(buf), 0); // use "send" in windows
63      printf("Bytes Sent: %ld ₩n", bytesSent);
64
65      // now receive
66      int n;
67      n = recv(s, buf, 50, 0); // read max 50 bytes
68      buf[n] = 0; // make a string
69      printf("received: %s₩n", buf);
70
71      //-------------------------------------------------------
72      closesocket(s);
73      WSACleanup();
74
75      return 0;
76  }
```

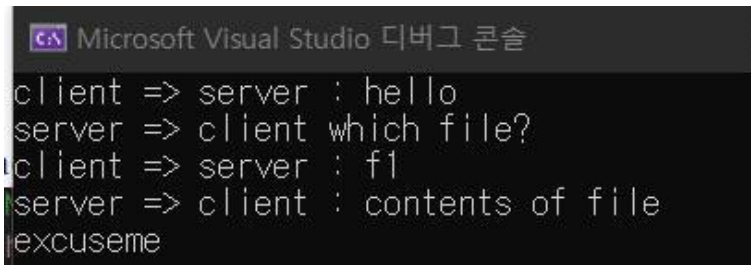
serv.c는 그대로 활용하였고, client는 SEND bytes 이후로 수정하였다.

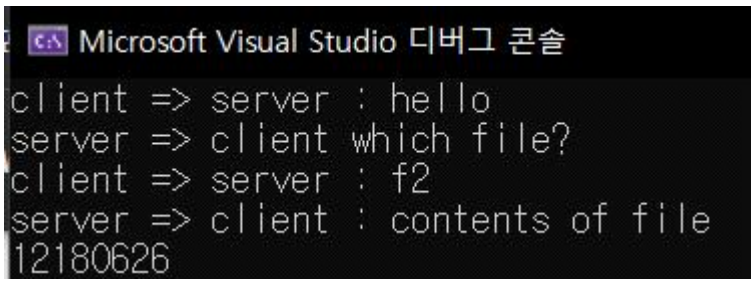
```
59      //---- SEND bytes ------------------------------------------
60      send(s, "hello", 5, 0);
61      printf("client => server : hello₩n");
62
63      //now receive
64      int n;
65      n = recv(s, buf, 50, 0);
66      buf[n] = 0;
67      printf("server => client %s₩n", buf);
68
69      //-------------------------------------------------------
70      //send file name
71      printf("client => server : ");
72      gets_s(buf, 99);
73      send(s, buf, strlen(buf), 0);
74
75      //receive file contents
76      printf("server => client : contents of file₩n");
77      for (;;) {
78          n = recv(s, buf, 50, 0);
79          buf[n] = 0;
80          if (n == 0) break;
81          printf("%s", buf);
82      }
83      closesocket(s);
84      WSACleanup();
85
86      return 0;
87  }
```

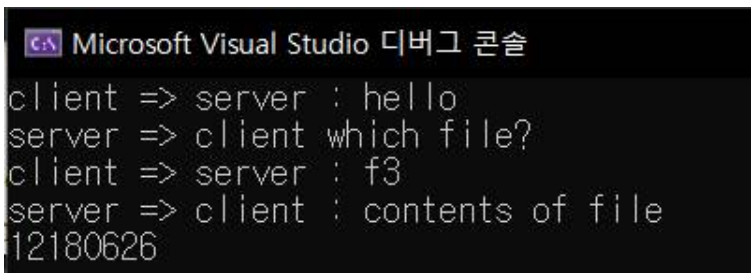
cli.c를 참고하여 코드를 수정하였다.

수행동작은 같으나 window에서 실행해야하므로 window에 맞게 send, closesocket 등으로 변경하여 사용하였다. read(s, buf, 50)를 recv(s, buf, 50, 0)으로, write(s, buf, strlen(buf))를 send(s, buf, strlen(buf), 0)으로 교체하였고, 0은 소켓 모드의 기본값을 의미한다.

```
[12180626@linuxer1 ~]$ serv
Hi, I am the server
socket opened successfully. socket num is 3
binding passed
we passed accept. new socket num is -553797696
now reading from client
client => server : hello
server => client : which file?
client => server : f1
server => client : contents of file
excuseme

now reading from client
client => server : hello
server => client : which file?
client => server : f2
server => client : contents of file
12180626

now reading from client
client => server : hello
server => client : which file?
client => server : f3
server => client : contents of file
12180626
```




컴파일 후 실행한 결과 Visual Studio에서 입력한 값들이 serv에 잘 출력됨을 알 수 있다.