

## 모르는거 리스트

strncpy / strncmp / sprintf / jump(longjump,setjump)  
basic\_server.c / basic\_clnt.c

### 1.strncpy

char \*strncpy(char \*destination, const char \*source, size\_t num);

문자열에서 일부 문자들만 복사한다.

num 보다 source 의 문자 수가 더 적으면 모자란 부분은 0 으로 생각되서 dest 에 복사가된다.

즉 source 가 “abc”인데 num 이 5 라면 “abc\0\0” 이렇게 된다.

strncpy 함수는 복사시 dest 끝에 반드시 NULL 문자를 붙이는 것이 아니다.

destination - 문자열이 복사될 곳.

Source - 복사 될 c 문자열

num - source 에서 복사 할 문자의 개수

예제))

```
#include <stdio.h>
```

```
#include <string.h>
```

```
int main ()
```

```
{
```

```
char str1[]= "To be or not to be";
```

```
char str2[6];
```

```
strncpy (str2,str1,5);
```

```
str2[5]='\0';
```

```
puts (str2);
```

```
return 0;
```

```
}
```

### 2.strncmp

int strncmp(const char \*str1, const char \*str2,size\_t num);

두 문자열의 일부 문자들을 비교한다.

C 형식 문자열인 str1 의 처음 num 개의 문자를 다른 C 형식 문자열인 str2 의 처음 num 개의 문자와 비교.  
다른문자나 NULL 문자가 나올때까지.

예제))

```
#include <stdio.h>
#include <string.h>

int main ()
{
    char str[][5] = { "R2D2" , "C3PO" , "R2A6" };
    int n;
    puts ("Looking for R2 astromech droids...");
    for (n=0 ; n<3 ; n++)
        if (strncmp (str[n],"R2xx",2) == 0)
        {
            printf ("found %s\n",str[n]);
        }
    return 0;
}
```

### 3.sprintf

```
int sprintf(char *str, const char * format, ...);
```

str 에 데이터를 형식에 맞추어 쓴다.

문자열에 숫자를 넣어야 할때가 있음. 이때 sprintf 를 씀.

예제))

```
char buf[256] ;
```

```
int a = 9 ;
```

```
sprintf( buf , " test%d", a ) ;
```

→ buf 에는 test9 라는 문자열이 저장.

응용하면

```
char *string = "world" ;  
sprintf( buf, "Hello %s %d", string, a ) ;
```

는

buf 에 Hello world 9 을 저장.

더 응용하면

```
char buf[256] ;  
  
for( int i = 0 ; i < 3 ; ++i )  
{  
    sprintf( buf, "test %d", i ) ;  
    printf( "%s\n", buf ) ;  
}
```

이렇게 쓰면,

```
test0  
test1  
test2
```

이런식으로 출력.

## 4.setjump() 와 longjmp()

```
int setjmp( jmp_buf env )
```

함수가 호출되는 순간 스택값들은 env 에 저장.

- setjmp()호출, longjmp 할 곳을 지정.

```
void longjmp( jmp_buf env, int val )
```

- longjmp 를 호출하면 setjmp()를 한 곳으로 돌아감.

예제))

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <setjmp.h>
```

```
#include <signal.h>
```

```
void p1();
```

```
void intHandler();
```

```
jmp_buf env;
```

```
int main()
```

```
{
```

```
    signal( SIGINT, intHandler );
```

```
    if( setjmp( env ) != 0 ){
```

```
        printf( "오류로 인해 복귀\n" );
```

```
        exit( 0 );
```

```
    }
```

```
    else
```

```
        printf( "처음 통과\n" );
```

```
    p1();
```

```
}
```

```
void p1()
```

```

{
    while( 1 ){
        printf( "루프\n" );
        sleep( 1 );
    }
}

```

```

void intHandler()
{
    printf( "인터럽트\n" );
    longjmp( env, 1 );
}

```

## 5.basic\_server.c

```

typedef struct sockaddr_in si;
typedef struct sockaddr *sap;

void err_handler(char *msg)
{
    fputs(msg,stderr);
    fputc('\n',stderr);
    exit(1);
}

int main(int argc, char **argv)
{

```

```
int serv_sock;
int clnt_sock;
si serv_addr;
si clnt_addr;
socklen_t clnt_addr_size;

char msg[] = "Hello Network Progrmming";
```

```
if(argc != 2)//이건 왜하는걸까?
{
    printf("use:%s<port>\n", argv[0]);
    exit(1);
}
```

```
/* 서버 소켓 생성*/
```

```
serv_sock = socket(PF_INET,SOCK_STREAM,0);
```

```
if(serv_sock == -1)
    err_handler("socket() error");
```

```
memset(&serv_addr,0,sizeof(serv_addr));
```

```
serv_addr.sin_addr.s_addr
    =htonl(INADDR_ANY);
serv_addr.sin_port = htons(atoi(argv[1]));
```

```
/* 소켓에 주소할당 */
```

```
if(bind(serv_sock, (sap)&serv_addr,sizeof(serv_addr)) == -1)
    err_handler("bind() error");
```

```
/* 연결 요청 대기상태로 진입 */
```

```
if(listen(serv_sock,5) == -1)
    err_handler("listen() error");
```

```
/* 연결 요청 수락 */
```

```
clnt_addr_size = sizeof(clnt_addr);
```

```

    clnt_sock = accept(serv_sock,(struct sockaddr
*)&clnt_addr,&clnt_addr_size);

if(clnt_sock == -1)
    err_handler("accept() error");
    /* 데이터 전송 */
    write(clnt_sock,msg,sizeof(msg));
    /* 연결 종료 */
    close(clnt_sock);
    close(serv_sock);
    return 0;
}

```

## 6.basic\_clnt.c

```

typedef struct sockaddr_in si;
typedef struct sockaddr *sap;

void err_handler(char *msg)
{
    fputs(msg,stderr);
    fputc('\n',stderr);
    exit(1);
}

int main(int argc, char **argv)
{
    int sock;
    int str_len = 0;
    si serv_addr;
    char msg[32] = {0};
    int idx =0, read_len =0;

    if(argc != 3)

```

```
{  
    printf("use:%s<IP><port>\n",argv[0]);  
    exit(1);  
}
```

```
sock=socket(PF_INET,SOCK_STREAM,0);
```

```
if(sock == -1)  
    err_handler("socket() error");
```

```
memset(&serv_addr,0,sizeof(serv_addr));  
serv_addr.sin_family = AF_INET;  
serv_addr.sin_addr.s_addr = inet_addr(argv[1]);
```

```
serv_addr.sin_port = htons(atoi(argv[2]));
```

```
if(connect(sock,(sap)&serv_addr,sizeof(serv_addr)) == -1)  
    err_handler("connect() error!");
```

```
while(read_len = read(sock,&msg[idx++],1))  
{  
    if(read_len == -1)  
    {  
        err_handler("read() error");  
        idx--;  
    }  
    str_len += read_len;  
}
```

```
printf("msg form serv:%s\n",msg);  
printf("read count: %d\n",str_len);  
close(sock);  
return 0;
```