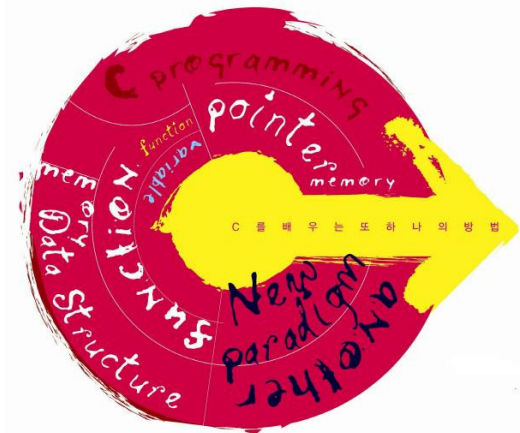


C Programming

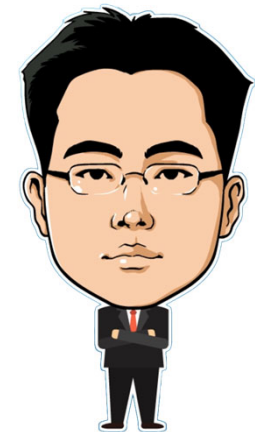
문자와 문자열 (Characters and Strings)



Seo, Doo-Ok

Clickseo.com

clickseo@gmail.com



목 차



백문이불여일타(百聞而不如一打)

- 문자 처리

- 문자열 처리



문자 처리



백문이불여일타(百聞而不如一打)

- 문자 처리

- 문자 분류

- 문자 변환

- 문자열 처리



문자 처리

- 매크로(Macro): 변수 및 상수, 매크로 함수

```
#include <ctype.h>
```

```
#define _UPPER          0x1      // upper case letter
```

```
#define _LOWER         0x2      // lower case letter
```

```
#define _DIGIT         0x4      // digit[0-9]
```

```
#define _SPACE         0x8      // tab, carriage return, newline,  
// vertical tab or form feed
```

```
#define _PUNCT         0x10     // punctuation character
```

```
#define _CONTROL       0x20     // control character
```

```
#define _BLANK         0x40     // space char
```

```
#define _HEX           0x80     // hexadecimal digit
```

```
#define _LEADBYTE      0x8000   // multibyte leadbyte
```

```
#define _ALPHA         (0x0100 | _UPPER | _LOWER) // alphabetic character
```



문자 처리

문자 분류



문자 분류 (1/4)

● 문자 분류(Character classification): 영문자

○ 영문 대소문자로 분류되는 문자인지 여부를 확인

- 변환 대상 문자는 **unsigned char** 로 표현 될 수 없다.
- EOF 와 같지 않은 경우의 동작은 정의되지 않는다.

```
#include <ctype.h>
```

```
int islower(int ch);           // 영문 소문자 여부 판단
```

```
int isupper(int ch);          // 영문 대문자 여부 판단
```

```
// 영문 대소문자 여부 판단
```

```
int isalpha(int ch);          // isupper(ch) 또는 islower(ch)가 만족되는 경우
```

판단 하려는 문자인 경우: 0 이외의 값을 반환
판단 하려는 문자가 아닌 경우: 0 값을 반환

Microsoft Visual Studio 디버그 콘솔

```
islower('A'): 0  
islower('a'): 2  
islower('0'): 0  
islower('*'): 0
```

C:\Users\click\OneDrive\문서\Click
이 창을 닫으려면 아무 키나 누르세요

Microsoft Visual Studio 디버그 콘솔

```
isupper('A'): 1  
isupper('a'): 0  
isupper('0'): 0  
isupper('*'): 0
```

C:\Users\click\OneDrive\문서\Click
이 창을 닫으려면 아무 키나 누르세요

Microsoft Visual Studio 디버그 콘솔

```
isalpha('A'): 1  
isalpha('a'): 2  
isalpha('0'): 0  
isalpha('*'): 0
```

C:\Users\click\OneDrive\문서\Clickseo\64
이 창을 닫으려면 아무 키나 누르세요...

문자 분류 (2/4)

● 문자 분류: 숫자

○ 숫자로 분류되는 문자인지 여부를 확인

- 변환 대상 문자는 **unsigned char** 로 표현 될 수 없다.
- EOF 와 같지 않은 경우의 동작은 정의되지 않는다.

```
#include <ctype.h>
```

```
int isdigit(int ch);           // 10진수 숫자 문자 여부 판단
```

```
int isxdigit(int ch);        // 16진수 숫자 문자 여부 판단
```

```
// 영문 대소문자와 숫자 여부 판단
```

```
int isalnum(int ch);         // isalpha(ch) 또는 isdigit(ch)가 만족되는 경우
```

판단 하려는 문자인 경우: 0 이외의 값을 반환
판단 하려는 문자가 아닌 경우: 0 값을 반환

Microsoft Visual Studio 디버그 콘솔

```
isdigit('A'): 0  
isdigit('a'): 0  
isdigit('0'): 4  
isdigit('*'): 0
```

C:\Users\click\OneDrive\문서\cli
이 창을 닫으려면 아무 키나 누르세요

Microsoft Visual Studio 디버그 콘솔

```
isxdigit('A'): 128  
isxdigit('a'): 128  
isxdigit('0'): 128  
isxdigit('*'): 0
```

C:\Users\click\OneDrive\문서\cli
이 창을 닫으려면 아무 키나 누르세요

Microsoft Visual Studio 디버그 콘솔

```
isalnum('A'): 1  
isalnum('a'): 2  
isalnum('0'): 4  
isalnum('*'): 0
```

C:\Users\click\OneDrive\문서\clickseo\64
이 창을 닫으려면 아무 키나 누르세요...

문자 분류 (3/4)

- 문자 분류: 다양한 문자

- 공백과 제어 문자 그리고 그래픽 문자 등 다양한 문자에 대하여 분류되는 문자인지 여부를 확인하는 함수

- 변환 대상 문자는 **unsigned char** 로 표현 될 수 없다.
- EOF 와 같지 않은 경우의 동작은 정의되지 않는다.

```
#include <ctype.h>
```

```
int isblank(int ch);
```

```
// ' ', '\t'
```

```
int isspace(int ch);
```

```
// ' ', '\t', '\n', '\r', '\f', '\v'
```

```
int iscntrl(int ch);
```

```
// 제어 문자 여부 판단
```

```
int isgraph(int ch);
```

```
// 그래픽 문자 여부 판단
```

```
int isprint(int ch);
```

```
// 주어진 문자가 인쇄될 수 있는지 판단
```

```
int ispunct(int ch);
```

```
// 구두 문자(punctuation character) 판단
```

판단 하려는 문자인 경우: 0 이외의 값을 반환

판단 하려는 문자가 아닌 경우: 0 값을 반환

문자 분류 (4/4)

ASCII values (hex)	characters	isctrl iswctrl	isprint iswprint	isspace iswspace	isblank iswblank	isgraph iswgraph	ispunct iswpunct	isalnum iswalnum	isalpha iswalpha	isupper iswupper	islower iswlower	isdigit iswdigit	isxdigit iswxdigit
0 - 8 0x00 - 0x08	control codes (NUL, etc.)	≠0	0	0	0	0	0	0	0	0	0	0	0
9 0x09	tab (\t)	≠0	0	≠0	≠0	0	0	0	0	0	0	0	0
10 - 13 0x0A - 0x0D	whitespaces (\n.\v.\f.\r)	≠0	0	≠0	0	0	0	0	0	0	0	0	0
14 - 31 0x0E - 0x1F	control codes	≠0	0	0	0	0	0	0	0	0	0	0	0
32 0x20	space	0	≠0	≠0	≠0	0	0	0	0	0	0	0	0
33 - 47 0x21 - 0x2F	!"#\$%&'()*+,-./	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
48 - 57 0x30 - 0x39	0123456789	0	≠0	0	0	≠0	0	≠0	0	0	0	≠0	≠0
58 - 64 0x3a - 0x40	::<=>?@	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
65 - 70 0x41 - 0x46	ABCDEF	0	≠0	0	0	≠0	0	≠0	≠0	≠0	0	0	≠0
71 - 90 0x47 - 0x5A	GHIJKLMNOPQRSTUVWXYZ	0	≠0	0	0	≠0	0	≠0	≠0	≠0	0	0	0
91 - 96 0x5B - 0x60	[\]^_`	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
97 - 102 0x61 - 0x66	abcdef	0	≠0	0	0	≠0	0	≠0	≠0	0	≠0	0	≠0
103 - 122 0x67 - 0x7A	ghijklmnopqrstuvwxyz	0	≠0	0	0	≠0	0	≠0	≠0	0	≠0	0	0
123 - 126 0x7B - 0x7E	{ } ~	0	≠0	0	0	≠0	≠0	0	0	0	0	0	0
127 0x7F	backspace character (DEL)	≠0	0	0	0	0	0	0	0	0	0	0	0



문자 처리

문자 변환



문자 변환 (1/2)

● 문자 조작: 영문 대소문자

○ 영문 대문자를 소문자로 또는 영문 소문자를 대문자로 변환

- 변환 대상 문자는 **unsigned char** 로 표현 될 수 없다.
- EOF 와 같지 않은 경우의 동작은 정의되지 않는다.

```
#include <ctype.h>
```

대소문자 변환 성공: 변환 된 문자 ch 값(ASCII code) 반환
대소문자 변환 실패: 기존 문자 ch 값(ASCII code) 반환

```
int tolower(int ch);           // 주어진 문자 ch 를 소문자로 변환  
int toupper(int ch);         // 주어진 문자 ch 를 대문자로 변환
```

Microsoft Visual Studio 디버그 콘솔

```
tolower('A'): a  
tolower('a'): a  
tolower('0'): 0  
tolower('*'): *
```

C:\Users\click\OneDrive\문서\clickseo\64
이 창을 닫으려면 아무 키나 누르세요...

Microsoft Visual Studio 디버그 콘솔

```
toupper('A'): A  
toupper('a'): A  
toupper('0'): 0  
toupper('*'): *
```

C:\Users\click\OneDrive\문서\clickseo\64
이 창을 닫으려면 아무 키나 누르세요...

문자 변환 (2/2)

예제 8-1: 문자 분류 및 변환 -- islower, isupper, tolower, toupper

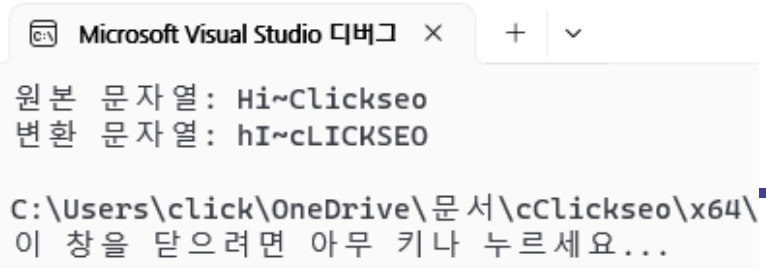
```
#include <stdio.h>
#include <ctype.h>      // islower, isupper, tolower, toupper

int main(void)
{
    char    str[] = "Hi~Clickseo";
    char    *pStr = str;

    fputs("원본 문자열: ", stdout);
    puts(str);

    printf("변환 문자열: ");
    for (; *pStr != '\0'; pStr++) {
        if (islower(*pStr))    putchar(toupper(*pStr));
        else if (isupper(*pStr))    putchar(tolower(*pStr));
        else                    putchar(*pStr);
    }
    *pStr = '\0';
    putchar('\n');

    return 0;
}
```



문자열 처리



백문이불여일타(百聞而不如一打)

- 문자 처리

- 문자열 처리

- 문자열 조작

- 문자열 검사

- 문자열과 숫자 변환



문자열 처리 (1/2)

- 문자열 처리(String Handling): string.h
 - C 표준 라이브러리: **<string.h>**
 - 널 문자를 제외한 문자 개수를 검증하는 과정

```
size_t my_strlen(const char *pStr) {  
    if (pStr == NULL) {           // 잘못된 문자열  
        return 0;  
    }  
  
    int count = 0;  
    while (*pStr++)  
        count++;  
    return count;  
}
```

```
#include <string.h>  
size_t  strlen (const char *pStr);
```

문자열 처리 (2/2)

- 문자열 처리: 데이터 유형과 매크로 상수

- 데이터 유형

```
#include <string.h>

typedef unsigned int      size_t;
typedef unsigned short   wchar_t;
```

- 매크로 상수

```
#include <string.h>

#define NULL      ((void*) 0)
```



문자열 처리

문자열 조작

`strcpy, strncpy / strcpy_s, strncpy_s`

`strcat, strncat / strcat_s, strncat_s`



문자열 조작 (1/4)

- 문자열 복사: **strcpy, strncpy / strcpy_s, strncpy_s**

- 널 문자를 포함한 원본 문자열을 대상 문자열로 복사

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

```
char *strcpy(char *dest, const char *src); // until C99
```

```
char *strcpy(char *restrict dest, const char *restrict src); // since C99
```

```
errno_t strcpy_s(char *restrict dest, rsize_t destsz, const char *restrict src); // since C11
```

```
char *strncpy(char *dest, const char *src, size_t count); // until C99
```

```
char *strncpy(char *restrict dest, const char *restrict src, size_t count); // since C99
```

```
errno_t strncpy_s(char *restrict dest, rsize_t destsz, const char *restrict src, rsize_t count);  
// since C11
```

문자열 조작 (2/4)

예제 8-2: 문자열 복사 -- strcpy, strncpy / strcpy_s, strncpy_s

```
#include <stdio.h>
#include <string.h>      // strcpy, strncpy / strcpy_s, strncpy_s
int main(void)
{
    const char    *src = "Hi~ Clickseo";
    char          dest[1024] = {'\0'};

    printf("원본 문자열(src) : %s \n", src );
    printf("대상 문자열(dest): %s \n", dest );

    // strcpy(dest, src);
    strcpy_s(dest, sizeof(dest), src);
    printf("대상 문자열(dest): %s \n\n", dest );

    char    temp[] = "1234567890";

    // strncpy(dest, temp, 4);
    strncpy_s(dest, sizeof(dest), temp, 4);
    printf("원본 문자열(temp): %s \n", temp );
    printf("대상 문자열(dest): %s \n", dest );

    return 0;
}
```

```
Microsoft Visual Studio 디버그 x + v
원본 문자열(src) : Hi~ Clickseo
대상 문자열(dest):
대상 문자열(dest): Hi~ Clickseo

원본 문자열(temp): 1234567890
대상 문자열(dest): 1234

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...|
```

문자열 조작 (3/4)

- 문자열 추가: **strcat, strncat / strcat_s, strncat_s**

- 대상 문자열에 원본 문자열을 추가(연결)

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

```
char *strcat(char *dest, const char *src); // until C99
```

```
char *strcat(char *restrict dest, const char *restrict src); // since C99
```

```
errno_t strcat_s(char *restrict dest, rsize_t destsz, // since C11  
    const char *restrict src);
```

```
char *strncat(char *dest, const char *src, size_t count); // until C99
```

```
char *strncat(char *restrict dest, const char *restrict src, size_t count); // since C99
```

```
errno_t strncat_s(char *restrict dest, rsize_t destsz, // since C11  
    const char *restrict src, rsize_t count);
```

문자열 조작 (4/4)

예제 8-3: 문자열 추가(연결) -- strcat, strncat / strcat_s, strncat_s

```
#include <stdio.h>
#include <string.h> // strcat, strncat / strcat_s, strncat_s
int main(void)
{
    const char *src = "Clickseo";
    char dest[1024] = "Hi~ ";

    printf("원본 문자열(src) : %s \n", src );
    printf("대상 문자열(dest): %s \n", dest );

    // strcat(dest, src);
    strcat_s(dest, sizeof(dest), src);
    printf("대상 문자열(dest): %s \n\n", dest );

    char temp[] = "1234567890";

    // strncat(dest, temp, 4);
    strncat_s(dest, sizeof(dest), temp, 4);
    printf("대상 문자열(dest): %s \n", dest );

    return 0;
}
```

```
Microsoft Visual Studio 디버그 x + v
원본 문자열(src) : Clickseo
대상 문자열(dest): Hi~
대상 문자열(dest): Hi~ Clickseo

대상 문자열(dest): Hi~ Clickseo1234

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```



문자열 처리

문자열 검사

`strlen`, `strcmp`, `strchr`, `strstr`



문자열 검사 (1/13)

- 문자열 길이: **strlen** / **strlen_s**

- 주어진 문자열의 길이(문자 개수)를 검사한다.

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

```
size_t  strlen(const char *str);
```

```
size_t  strlen_s(const char *str, size_t strsz);           // since C11
```

```
char str[] = "Hi~ Clickseo";
```

```
printf("문자열 길이: %d \n", strlen(str) );
```

```
printf("배열 크기  : %d \n\n", sizeof(str) );
```

```
char    temp[1024];
```

```
printf("문자열 길이: %d \n", strlen(temp) );
```

```
printf("문자열 길이: %d \n", strlen_s(temp, sizeof(temp)) );
```

문자열 검사 (2/13)

- **문자열 비교: strcmp, strncmp**

- 주어진 두 개의 문자열이 동일한지 여부를 비교한다.

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

```
int strcmp(const char *lhs, const char *rhs);
```

```
int strncmp(const char *lhs, const char *rhs, size_t count);
```

- 두 문자열이 동일한 경우: 0 값을 반환
- 두 문자열이 동일하지 않은 경우
 - 사전순으로 왼쪽 문자가 더 큰 ASCII 값인 경우: 1
 - 사전순으로 오른쪽 문자가 더 큰 ASCII 값인 경우: -1

문자열 검사 (3/13)

예제 8-4: 문자열 비교 -- strcmp

```
#include <stdio.h>
#include <string.h> // strcmp, strncmp
int main(void)
{
    const char *lStr = "Hi~ Clickseo";
    const char *rStr = "Hi~ Clickseo";

    printf("왼쪽 문자열 : %s \n", lStr);
    printf("오른쪽 문자열 : %s \n\n", rStr);

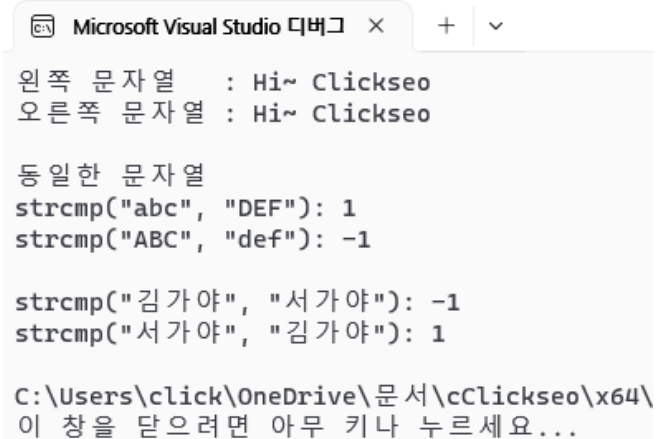
    int res = strcmp(lStr, rStr);

    if(res == 0) printf("동일한 문자열 \n");
    else printf("서로 다른 문자열 \n");

    printf("strcmp(\"abc\", \"DEF\") : %d\n", strcmp("abc", "DEF") );
    printf("strcmp(\"ABC\", \"def\") : %d\n\n", strcmp("ABC", "def") );

    printf("strcmp(\"김가야\", \"서가야\") : %d\n", strcmp("김가야", "서가야") );
    printf("strcmp(\"서가야\", \"김가야\") : %d\n", strcmp("서가야", "김가야") );

    return 0;
}
```



```
Microsoft Visual Studio 디버그 x + v
왼쪽 문자열 : Hi~ Clickseo
오른쪽 문자열 : Hi~ Clickseo

동일한 문자열
strcmp("abc", "DEF"): 1
strcmp("ABC", "def"): -1

strcmp("김가야", "서가야"): -1
strcmp("서가야", "김가야"): 1

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```


문자열 검사 (4/13)

예제 8-5: 문자열 비교 -- strcmp

```
#include <stdio.h>
#include <string.h> // strcmp, strncmp
int main(void)
{
    const char *lStr = "Hi~ Clickseo";
    const char *rStr = "Hi~ Click";

    printf("왼쪽 문자열 : %s \n", lStr );
    printf("오른쪽 문자열 : %s \n\n", rStr );

    int res = strcmp(lStr, rStr, 3);
    printf("strcmp(lStr, rStr, 3) : %d \n", res );

    res = strcmp(lStr, rStr, 10);
    printf("strcmp(lStr, rStr, 10): %d \n", res );

    res = strcmp(lStr, rStr, 50);
    printf("strcmp(lStr, rStr, 50): %d \n", res );

    return 0;
}
```

```
Microsoft Visual Studio 디버그 x + v
왼쪽 문자열 : Hi~ Clickseo
오른쪽 문자열 : Hi~ Click

strcmp(lStr, rStr, 3) : 0
strcmp(lStr, rStr, 10): 1
strcmp(lStr, rStr, 50): 1

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```

문자열 검사 (5/13)

- 문자 또는 문자열 검색: **strchr, strrchr, strstr**

- 문자열에서 지정된 문자 또는 문자열이 존재하는지 검색한다.

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

```
// 문자열에 지정된 문자가 존재하는지 검색하는 함수
```

```
char *strchr(const char *str, int ch);
```

```
char *strrchr(const char *str, int ch);
```

```
// 문자열에서 부분 문자열이 존재하는지 검색하는 함수
```

```
char *strstr(const char *str, const char *substr);
```

- **strchr** : 문자열의 처음 부터 처음으로 일치하는 문자를 검색
- **strrchr** : 문자열의 끝에서 부터 일치하는 문자를 검색
- **문자열에서 문자(또는 문자열)가 존재하는지 검색한다.**
 - 지정된 문자(또는 문자열)이 존재: 검색된 문자(또는 문자열)의 메모리 주소 반환
 - 지정된 문자(또는 문자열) 존재하지 않음: NULL 값 반환

문자열 검사 (6/13)

예제 8-6: 문자 또는 문자열 검색 -- strchr

```
#include <stdio.h>
#include <string.h>    // strchr
int main(void)
{
    const char *str = "Hi~ Clickseo";

    printf("원본 문자열 :%s \n\n", str );
    printf("검색 할 문자 :");
    char ch = getchar();

    char *pChar = strchr(str, ch);

    if (pChar == NULL)    printf("존재하지 않는 문자!!! \n");
    else                  printf("검색 문자: %c \n", *pChar );

    return 0;
}
```

문자열 검사 (7/13)

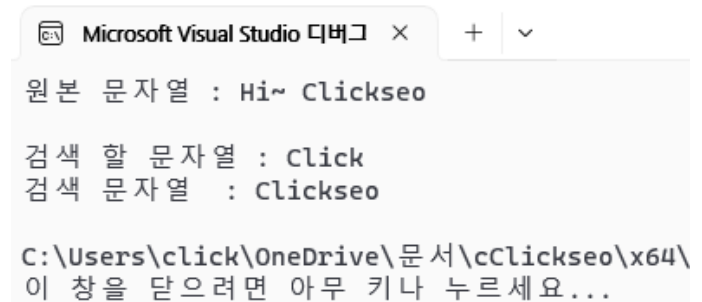
예제 8-7: 문자 또는 문자열 검색 -- strstr

```
#include <stdio.h>
#include <string.h> // strstr
int main(void)
{
    const char *str = "Hi~ Clickseo";
    char search[24];

    printf("원본 문자열 : %s \n\n", str );
    printf("검색 할 문자열 : ");
    gets_s(search, sizeof(search));

    char *pStr = strstr(str, search);
    if (pStr == NULL) printf("존재하지 않는 문자열!!! \n");
    else printf("검색 문자열 : %s \n", pStr );

    return 0;
}
```



```
Microsoft Visual Studio 디버그 x + v
원본 문자열 : Hi~ Clickseo
검색 할 문자열 : Click
검색 문자열 : Clickseo
C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```



문자열 처리

문자열 검사

`strspn`, `strpbrk`, `strtok`



문자열 검사 (8/13)

● 특정 문자가 연속해서 존재하는지 여부

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

```
size_t strstr(const char *dest, const char *src);
```

```
size_t strcspn(const char *dest, const char *src);
```

○ 문자열에서 특정 문자가 첫 번째 문자부터 연속해서 존재하는 길이를 검색한다.

- 특정 문자가 연속해서 존재: 검색된 부분의 길이를 반환
- 특정 문자가 연속해서 존재하지 않음: 0 값을 반환
- **strstr** 함수
 - 대상 문자열을 구성하는 문자가 원본 문자열에 연속적으로 존재하는지 길이를 구하는 함수(단, 순서는 중요하지 않다)
- **strcspn** 함수
 - 대상 문자열을 구성하는 문자가 원본 문자열에 연속적으로 존재하지 않는 길이를 구하는 함수(단, 순서는 중요하지 않다)

문자열 검사 (9/13)

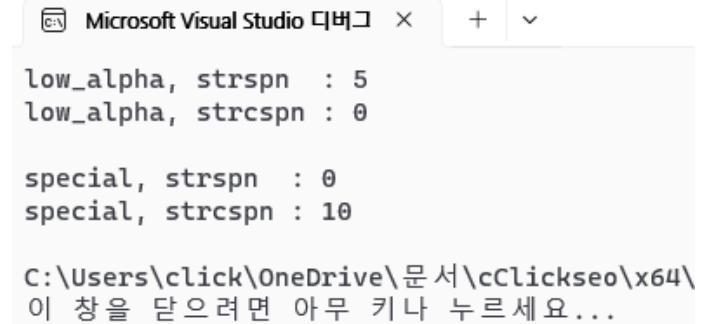
예제 8-8: 특정 문자가 연속해서 존재하는지 여부 -- `strspn`, `strcspn`

```
#include <stdio.h>
#include <string.h> // strspn, strcspn
int main(void)
{
    const char *str = "abcde12345$&*";

    // 영문 소문자
    const char *low_alpha = "abcdefghijklmnopqrstuvwxy";
    printf("low_alpha, strspn : %zd \n", strspn(str, low_alpha) );
    printf("low_alpha, strcspn : %zd \n\n", strcspn(str, low_alpha) );

    // 특수 문자
    const char *special = "~!@#$%^&*()";
    printf("special, strspn : %zd \n", strspn(str, special) );
    printf("special, strcspn : %zd \n", strcspn(str, special) );

    return 0;
}
```



```
Microsoft Visual Studio 디버그 x + v
low_alpha, strspn : 5
low_alpha, strcspn : 0

special, strspn : 0
special, strcspn : 10

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```

문자열 검사 (10/13)

● 특정 문자가 존재하는지 여부

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

```
char *strpbrk(const char *dest, const char *breakset);
```

○ 문자열에서 여러 개의 문자 중 하나라도 존재하는지 여부

- 특정 문자가 존재: 검색된 문자의 메모리 주소를 반환
- 특정 문자가 존재하지 않음: NULL 반환
- **strpbrk 함수**
 - 특정 문자열에서 여러 개의 문자 중 하나라도 있는지를 검색하는 함수
 - 대상 문자열의 문자가 하나라도 존재하는 위치에서 검색을 중단한다.
 - » 검색된 문자의 메모리 주소를 반환한다.

문자열 검사 (11/13)

예제 8-9: 특정 문자가 존재하는지 여부 -- strpbrk

```
#include <stdio.h>
#include <string.h>      // strpbrk, strspan
int main(void)
{
    const char *str = "Hello World, Hi~ Clickseo ^..^";
    const char *separator = ",!";

    printf("원본 문자열: %s \n", str);
    printf("구분자: %s \n", separator);

    unsigned int count = 0;
    do {
        str = strpbrk(str, separator);    // 구분자 탐색
        if(str)
            str += strspn(str, separator); // 구분자 건너 뛴
            count++;                       // 단어 수
    } while( str && *str );
    printf("단어 수: %d \n", count );
    return 0;
}
```

```
Microsoft Visual Studio 디버그 x + v
원본 문자열: Hello World, Hi~ Clickseo ^..^
구분자: ,!
단어 수: 5
C:\Users\click\OneDrive\문서\cClickseo\x64\Debug\
이 창을 닫으려면 아무 키나 누르세요...
```

문자열 검사 (12/13)

- 문자열 분할 함수: **strtok**

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

```
char *strtok(char *str, const char* token); // until C99  
char *strtok(char *restrict str, const char *restrict token); // since C99  
char *strtok_s(char *restrict str, rsize_t *restrict strMax, // since C11  
const char *restrict token, char **restrict ptr);
```

- 대상 문자열을 지정된 특정 구분자로 문자열을 분할한다.
 - 특정 구분자가 존재: 검색된 token 의 위치(메모리 주소)를 반환
 - 특정 구분자가 존재하지 않음: NULL 반환
- **strtok** 함수
 - 원하는 구분자를 이용하여 문자열을 쪼갤 수 있도록 하는 함수
 - 문자열을 분할할 때의 구분자는 두 번째 인자로 지정한다.

문자열 검사 (13/13)

예제 8-10: 문자열 분할 -- strtok / strtok_s

```
#include <stdio.h>
#include <string.h> // strtok
int main(void)
{
    char str[] = "서두옥 70 80 90 240 80.0";
    const char *token = " "; // 구분자
    char *nextToken; // 다음 작업이 진행될 위치의 주소

    printf("대상 문자열(str) : %p %s \n", str, str);
    printf("구분자 : %s \n\n", " ");

    // char *pToken = strtok(str, token);
    char *pToken = strtok_s(str, token, &nextToken);
    while (pToken != NULL) {
        printf("%p : %s \n", pToken, pToken );
        printf("%p : %s \n\n", nextToken, nextToken );
        // pToken = strtok(NULL, token);
        pToken = strtok_s(NULL, token, &nextToken);
    }
    return 0;
}
```

```
Microsoft Visual Studio 디버그
대상 문자열(str) : 000000322C8FF9C8 서두옥 70 80 90 240 80.0
구분자 :

000000322C8FF9C8 : 서두옥
000000322C8FF9CF : 70 80 90 240 80.0

000000322C8FF9CF : 70
000000322C8FF9D2 : 80 90 240 80.0

000000322C8FF9D2 : 80
000000322C8FF9D5 : 90 240 80.0

000000322C8FF9D5 : 90
000000322C8FF9D8 : 240 80.0

000000322C8FF9D8 : 240
000000322C8FF9DC : 80.0

000000322C8FF9DC : 80.0
000000322C8FF9E0 :

C:\Users\click\OneDrive\문서\cClickseo\x64\Debug\cClickseo.exe
이 창을 닫으려면 아무 키나 누르세요...
```



문자열 처리

문자열과 숫자 변환

atoi, atof / itoa, ltoa



문자열과 숫자 변환 (1/4)

- 문자열을 숫자로 변환: **atoi, atof**
 - 문자열을 숫자(정수형 또는 실수형)로 변환

```
#include <stdlib.h>

// 문자열을 정수형(int, long, long long)으로 변환하는 함수
int      atoi(const char *str);
long     atol(const char *str);
long long atoll(const char *str);           // since C99

// 문자열을 실수형(double)으로 변환하는 함수
double   atof(const char *str);
```

문자열과 숫자 변환 (2/4)

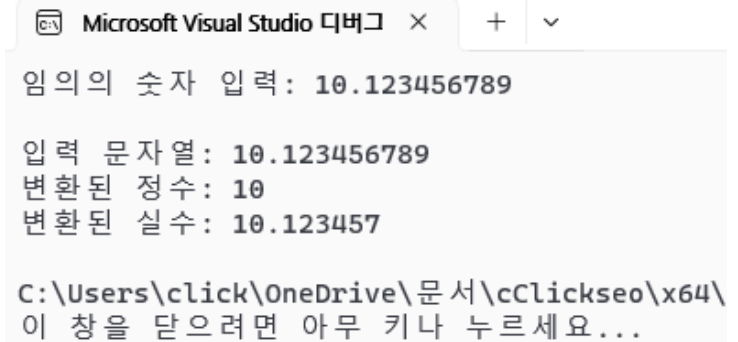
예제 8-11: 문자열을 숫자로 변환 -- atoi, atof

```
#include <stdio.h>
#include <stdlib.h> // atoi, atof
int main(void)
{
    char    str[24];

    printf("임의의 숫자 입력: ");
    gets_s(str, sizeof(str)); // gets(str);

    printf("\n입력 문자열: %s \n", str );
    printf("변환된 정수: %d \n", atoi(str) );
    printf("변환된 실수: %lf \n", atof(str) );

    return 0;
}
```



Microsoft Visual Studio 디버그

임의의 숫자 입력: 10.123456789

입력 문자열: 10.123456789
변환된 정수: 10
변환된 실수: 10.123457

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...

문자열과 숫자 변환 (3/4)

- 숫자를 문자열로 변환하: **itoa, Itoa**

- 정수형 숫자를 2진수, 8진수, 10진수 또는 16진수의 문자열로 변환

```
#include <stdlib.h>

// 정수형(int) 숫자를 문자열 str 로 변환
char      *itoa(int value, char *buffer, int radix);           // POSIX
errno_t   _itoa_s(int value, char *buffer, size_t size, int radix); // ISO

// 정수형(long) 숫자를 문자열 str 로 변환
char      *ltoa(long value, char *buffer, int radix);         // POSIX
errno_t   _ltoa_s(long value, char *buffer, size_t size, int radix); // ISO
```

// itoa, Itoa 함수 사용 시 Visual Studio 2019부터 오류(error) 메시지

Error C4996:

The **POSIX** name for this item is deprecated.
instead, use the ISO C and C++ conformant name: **_itoa**.

See online help for details.

문자열과 숫자 변환 (4/4)

예제 8-12: 숫자를 문자열로 변환 -- itoa / _itoa_s

```
#include <stdio.h>
#include <stdlib.h>      // itoa / _itoa_s
int main(void)
{
    int    num;
    char   buffer[1024];

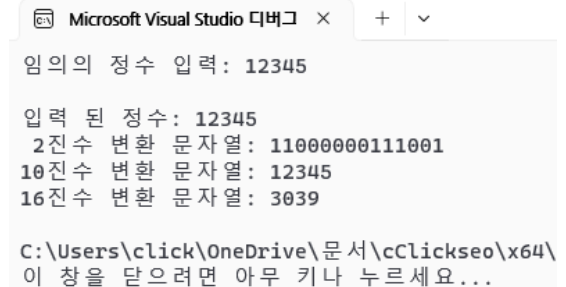
    printf("임의의 정수 입력: ");
    scanf_s("%d", &num);          // scanf ("%d", &num);

    printf("\n입력 된 정수: %d \n", num );
    // printf(" 2진수 변환 문자열: %s \n", itoa(num, buffer, 2) );
    // printf("10진수 변환 문자열: %s \n", itoa(num, buffer, 10) );
    // printf("16진수 변환 문자열: %s \n", itoa(num, buffer, 16) );
    _itoa_s(num, buffer, sizeof(buffer), 2);
    printf(" 2진수 변환 문자열: %s \n", buffer );

    _itoa_s(num, buffer, sizeof(buffer), 10);
    printf("10진수 변환 문자열: %s \n", buffer );

    _itoa_s(num, buffer, sizeof(buffer), 16);
    printf("16진수 변환 문자열: %s \n", buffer );

    return 0;
}
```



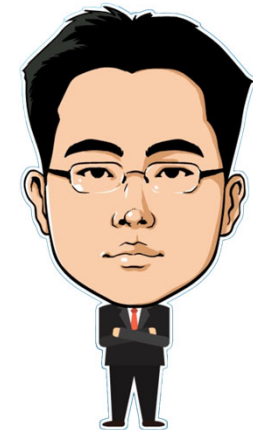
```
Microsoft Visual Studio 디버그 x + v
임의의 정수 입력: 12345

입력 된 정수: 12345
2진수 변환 문자열: 11000000111001
10진수 변환 문자열: 12345
16진수 변환 문자열: 3039

C:\Users\click\OneDrive\문서\cClickseo\x64\
이 창을 닫으려면 아무 키나 누르세요...
```


참고문헌

- [1] 서현우, "혼자 공부하는 C 언어 : 1:1 과외 하듯 배우는 프로그래밍 자습서", 한빛미디어, 2023.
- [2] Paul Deitel, Harvey Deitel, "C How to Program", Global Edition, 8/E, Pearson, 2016.
- [3] Kamran Amini, 박지윤 번역, "전문가를 위한 C : 동시성, OOP부터 최신 C, 고급 기능까지!", 한빛미디어, 2022.
- [4] 서두옥, "(열혈강의) 또 하나의 C : 프로그래밍은 셀프입니다", 프리렉, 2012.
- [5] Behrouz A. Forouzan, Richard F. Gilberg, 김진 외 7인 공역, "구조적 프로그래밍 기법을 위한 C", 도서출판 인터비전, 2004.
- [6] Brian W. Kernighan, Dennis M. Ritchie, 김석환 외 2인 공역, "The C Programming Language", 2/E, 대영사, 2004.
- [7] "C reference", cppreference.com, 2023 of viewing the site, <https://en.cppreference.com/w/c>.



이 강의자료는 저작권법에 따라 보호받는 저작물이므로 무단 전제와 무단 복제를 금지하며, 내용의 전부 또는 일부를 이용하려면 반드시 저작권자의 서면 동의를 받아야 합니다.

Copyright © Clickseo.com. All rights reserved.