
# 문자열 비교 검색 라이브러리

한국기술교육대학교
온라인평생교육원

## 학습내용

- 라이브러리 함수 이해
- 라이브러리 함수 활용

## 학습목표

- 문자열 처리 관련 라이브러리의 종류를 설명할 수 있다.
- 문자열을 비교·검색하는 라이브러리 함수의 용도를 알고 구현할 수 있다.

# 라이브러리 함수 이해

## 1 종류

### 1 헤더 파일 : string.h

#### 1 비교함수

| | |
|---|---|
| memcmp | 메모리 블록을 비교 |
| strcmp | 문자열을 비교 |
| strncmp | 문자열 개수를 지정하여 비교 |

#### 2 검색함수

| | |
|---|---|
| memchr | 메모리 블록에서 문자열 검색 |
| strchr | 문자열에서 찾는 문자의 첫 번째 위치 |
| strrchr | 문자열의 마지막에서부터 문자 위치 검색 |
| strspn | 문자열에서 특정 문자로 구성된 문자열의 길이 |
| strcspn | 문자열에서 특정 문자로 구성에 포함되지 않는 문자열의 길이 |
| strstr | 부분 문자열 위치 |
| strtok | 토큰으로 문자열 분리 |

# 라이브러리 함수 활용

## 1 문자열 비교함수

### 1 memcmp

| 항목 | 내용 |
|---|---|
| 함수원형 | int memcmp ( const void * ptr1, const void * ptr2, size_t num ); |
| 헤더 | string.h |
| 기능 | 2개의 메모리 변수에 대해 내용을 비교 |
| 매개변수 | void *ptr1 ➡ 비교 대상 메모리 포인터<br>void *ptr2 ➡ 비교할 메모리 포인터<br>size_t num ➡ 비교할 바이트 크기 |
| 반환값 | 양의 정수 : s1이 s2보다 크다.<br>0 : s1과 s2가 같다.<br>음의 정수 : s1보다 s2가 크다. |

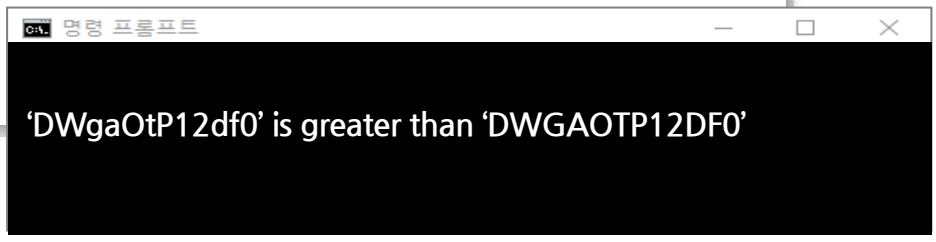
```
#include <stdio.h>
#include <string.h>

int main ()
{
  char buffer1[] = "DWgaOtP12df0";
  char buffer2[] = "DWGAOTP12DF0";
```

# 라이브러리 함수 활용

## 1 문자열 비교함수

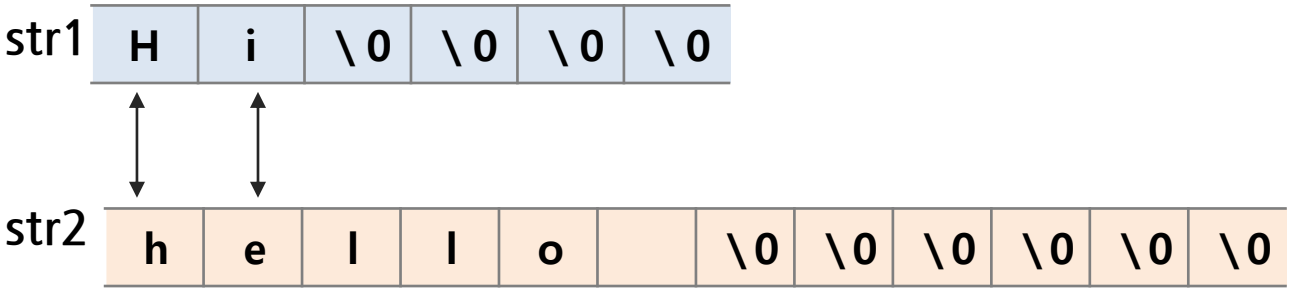
### 1 memcmp

```
int n;

n=memcmp ( buffer1, buffer2, sizeof(buffer1) );

if (n>0) printf ("'%s' is greater than '%s'.₩n",buffer1,buffer2);
else if (n<0) printf ("'%s' is less than '%s'.₩n",buffer1,buffer2);
else printf ("'%s' is the same as '%s'.₩n",buffer1,buffer2);

return 0;
}
```




### 2 strcmp

| 항목 | 내용 |
|---|---|
| 반환값 | 1 : str1 ! = str2<br>0 : str1 = str2<br>+ : 결과 값이면 str1 > str2<br>− : 결과 값이면 str1 < str2 |



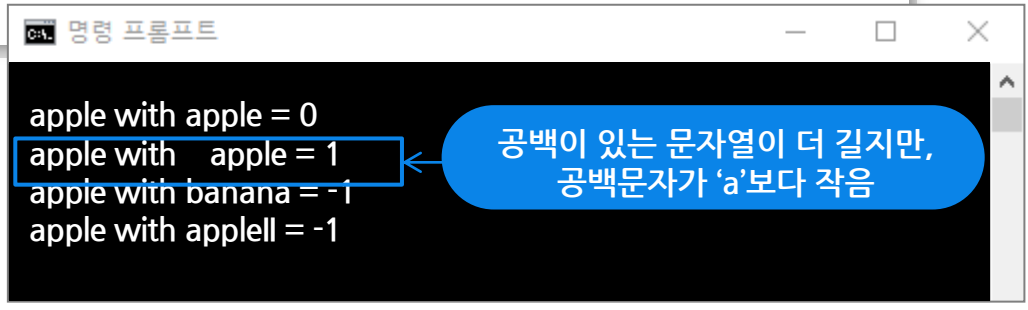
# 라이브러리 함수 활용

## 1 문자열 비교함수

### 2 strcmp

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

int main( void)
{
   char  str_apple[]  = “apple”;
   char  str_apple2[] = “   apple”;
   char  str_banana[] = “banana”;
   char  str_appleII[]= “appleII”;
```

```
printf( “%s with %s = %d₩n”, str_apple, str_apple  ,
strcmp( str_apple, str_apple  ) );
   printf( “%s with %s = %d₩n”, str_apple, str_apple2 ,
strcmp( str_apple, str_apple2 ) );
   printf( “%s with %s = %d₩n”, str_apple, str_banana ,
strcmp( str_apple, str_banana ) );
   printf( “%s with %s = %d₩n”, str_apple, str_appleII,
strcmp( str_apple, str_appleII) );

   return 0;
}
```



# 라이브러리 함수 활용

## 1 문자열 비교함수

### 3 strncmp

| 항목 | 내용 |
|---|---|
| 함수원형 | int strncmp ( const char * str1, const char * str2, size_t num ); |
| 헤더 | string.h |
| 기능 | 2개의 문자열을 지정한 문자 개수까지만 비교 |
| 매개변수 | char *str1 ➡ 비교할 대상 문자열<br>char *str2 ➡ 비교할 문자열<br>size_t n ➡ 비교할 문자의 개수 |
| 반환값 | 1 : str1 ! = str2<br>0 : str1 = str2<br>+ : 결과 값이면 str1 > str2<br>- : 결과 값이면 str1 < str2 |

```
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;
}
```

명령 프롬프트

```
Looking for R2 astromech droids...
found R2D2
found R2A6
```

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 1 memchr

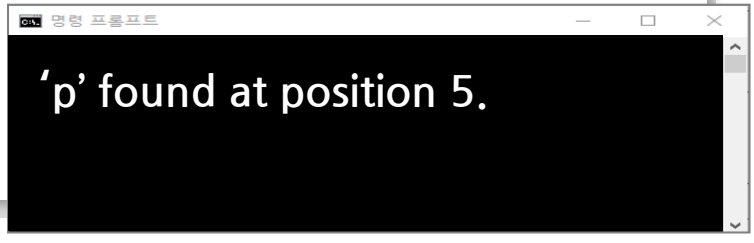
| 항목 | 내용 |
|---|---|
| 함수원형 | void * memchr ( const void * s, int c, size_t n); |
| 헤더 | string.h |
| 기능 | 메모리 영역에서 임의의 문자를 검색하고 있으면 그 위치의 포인터를 구함 |
| 매개변수 | void *s ➡ 검사할 메모리의 포인터<br>int c ➡ 검사할 문자 코드<br>size_t n ➡ 검사할 영역의 크기 |
| 반환값 | 처음 발견된 위치의 포인터, 발견하지 못하면 NULL |

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 1 memchr

```
int main ()
{
  char * pch;
  char str[] = “Example string”;
  pch = (char*) memchr (str, ‘p’, strlen(str));
  if (pch!=NULL)
    printf (“ ‘p’ found at position %d.₩n”, pch-str+1);
  else
    printf (“ ‘p’ not found.₩n”);
  return 0;
}
```



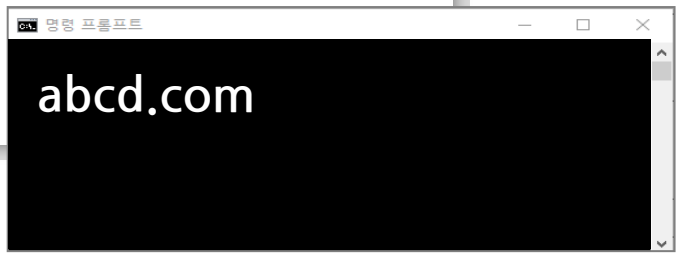

| 항목 | 내용 |
|---|---|
| 함수원형 | char *strchr( const char *str, int chr) |
| 헤더 | string.h |
| 기능 | 문자열에서 임의의 문자가 처음으로 발견된 위치 |
| 매개변수 | char *str ➧ 검색 대상 문자열<br>int chr ➧ 찾는 문자 |
| 반환값 | 찾고자 하는 문자가 발견된 첫 번째의 포인터를 반환, 찾지 못하면 NULL을 반환 |

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 2 strchr

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

int main( void)
{
   char   *str = "www.abcd.com";

   printf( "%s₩n", strchr( str, 'a'));

   return 0;
}
```




### 3 strrchr

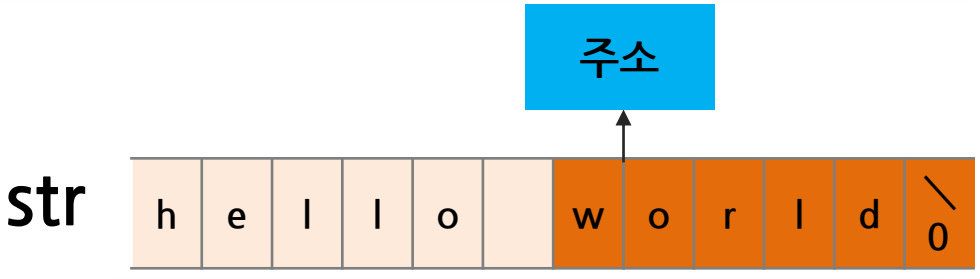
| 항목 | 내용 |
|---|---|
| 함수원형 | char *strrchr( const char *str, int chr); |
| 헤더 | string.h |
| 기능 | 문자열에서 임의의 문자가 마지막으로 발견된 위치를 포인터로 반환(몇 번째 X) |
| 매개변수 | char *str ➡ 검색 대상 문자열<br>int chr ➡ 찾는 문자 |
| 반환값 | 찾고자 하는 문자가 발견된 마지막 위치의 포인터를 반환, 찾지 못하면 NULL을 반환 |

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 3 strrchr

```
strrchr("hello world", 'o');
```



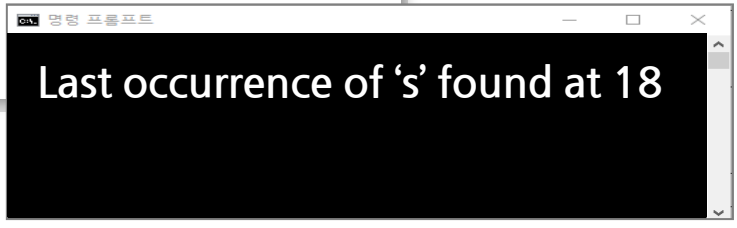

```
int main( void)
{
   char   *str = "www.world.com";

   printf( "%s₩n", strrchr( str, 'w'));

   return 0;
}
```

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 3 strrchr

```
#include <stdio.h>
#include <string.h>
int main ()
{
  char str[] = “This is a sample string”;
  char * pch;
  pch=strrchr(str,‘s’);
  printf (“Last occurence of ‘s’ found at %d
₩n”,pch-str+1);
  return 0;
}
```




| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| T | h | i | s | | i | s | | a | |

| 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| s | a | m | p | l | e | | s | t | r | i | n | g | \0 |

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 4 strspn

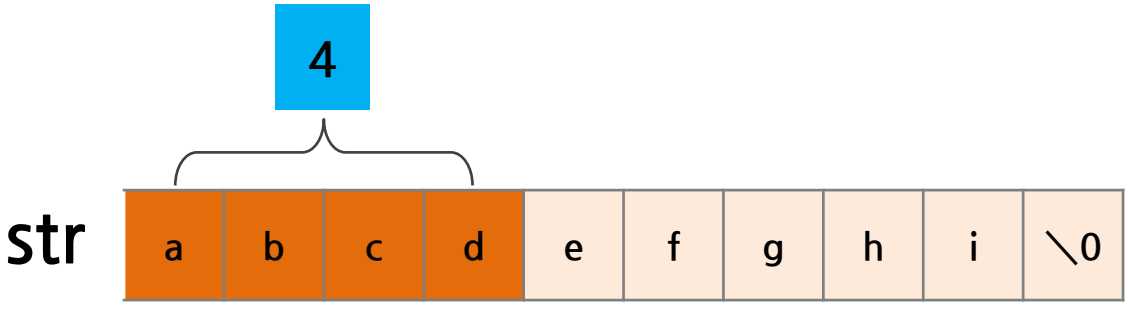
| 항목 | 내용 |
|---|---|
| 함수원형 | size_t strspn(const char *str1, const char *str2); |
| 헤더 | string.h |
| 기능 | ▪ 문자열에서 지정된 문자들로 구성된 초기 문자열의 길이를 검색<br>▪ 검색 대상의 문자열에 대해 첫 번째 바이트부터 차례대로 검색하면서 두 번째 인수의 문자에 포함되는 문자인지를 확인<br>▪ 두 번째 인수에 **포함되지 않는 문자**를 만나면 그 전까지의 문자열 길이를 반환 |
| 매개변수 | char *str1 ➡ 검색 대상 문자열<br>char *str2 ➡ 검색에 사용되는 문자들의 모임 |
| 반환값 | 문자열의 길이를 반환 |

# 라이브러리 함수 활용

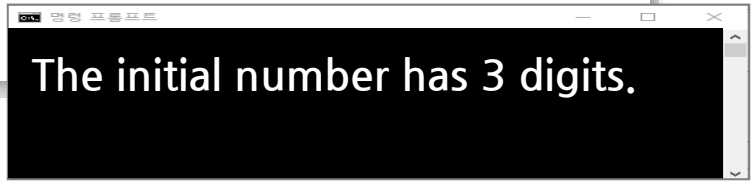
## 2 문자열 검색함수

### 4 strspn

```
strspn(str, “bdca”);
```




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

int main ()
{
  int i;
  char strtext[] = “129th”;
  char cset[] = “1234567890”;

  i = strspn (strtext,cset);
  printf (“The initial number has %d digits.₩n”,i);
  return 0;
}
```

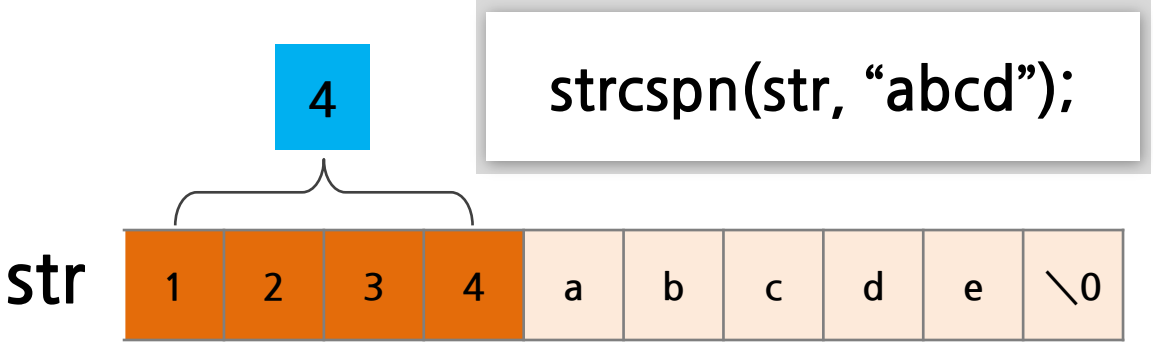


# 라이브러리 함수 활용

## 2 문자열 검색함수

### 5 strcspn

| 항목 | 내용 |
|---|---|
| 함수원형 | size_t strcspn(const char *str1, const char *str2); |
| 헤더 | string.h |
| 기능 | ▪ 문자열에서 지정된 문자들로 구성된 초기 문자열의 위치를 검색<br>▪ 검색 대상의 문자열에 대해 첫 번째 바이트부터 차례대로 검색하면서 두 번째 인수의 문자열에 포함되는 문자인지를 확인<br>▪ 2번째 인수에 포함되는 문자를 만나면 그전까지의 길이를 반환 |
| 매개변수 | char *str1 ➡ 검색 대상 문자열<br>char *str2 ➡ 검색에 사용되는 문자들의 모임 |
| 반환값 | 검색된 문자의 위치를 반환 |

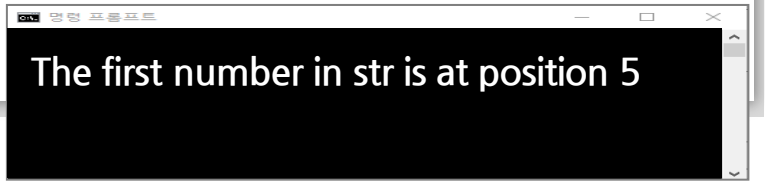


# 라이브러리 함수 활용

## 2 문자열 검색함수

### 5 strcspn

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

int main ()
{
  char str[] = “fcba73”;
  char keys[] = “1234567890”;
  int i;
  i = strcspn (str,keys);
  printf (“The first number in str is at position %d.₩n”,i+1);
  return 0;
}
```




### 6 strstr

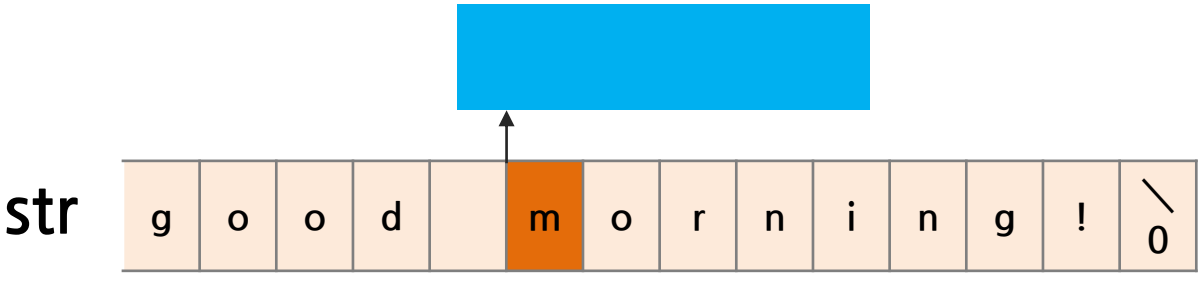
| 항목 | 내용 |
|---|---|
| 함수원형 | char *strstr( const char *str1, const char *str2) |
| 헤더 | string.h |
| 기능 | 문자열에서 임의의 문자열이 시작하는 위치 |
| 매개변수 | char *str1 ➡ 검색 대상 문자열<br>char *str2 ➡ 찾는 문자열 |
| 반환값 | 찾고자 하는 문자열이 발견된 첫 번째 위치의 포인터를 반환, 찾지 못하면 NULL을 반환 |

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 6 strstr

```
strstr(str, “morning”);
```




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

int main ()
{
  char str[] =“This is a simple string”;
  char * pch;
  pch = strstr (str,“simple”);
  strncpy (pch,“sample”,6);
  puts (str);
  return 0;
}
```


참고
strrstr()은 문자열 우측에서분터 검사

# 라이브러리 함수 활용

## 2 문자열 검색함수

### 6 strtok

| 항목 | 내용 |
|---|---|
| 함수원형 | char * strtok ( char * str, const char * delimiters ); |
| 헤더 | string.h |
| 기능 | 문자열을 문자로 자르는 함수 |
| 매개변수 | char *str ➡ 자르기 대상 문자열<br>char *delimiters ➡ 잘라내기 위한 문자 모임 |
| 반환값 | 잘라내기 한 문자열의 첫 번째 포인터를 반환,<br>문자열이 없다면 NULL을 반환 |

```
strtok(str , del);
```

```
strtok(NULL, del);
```

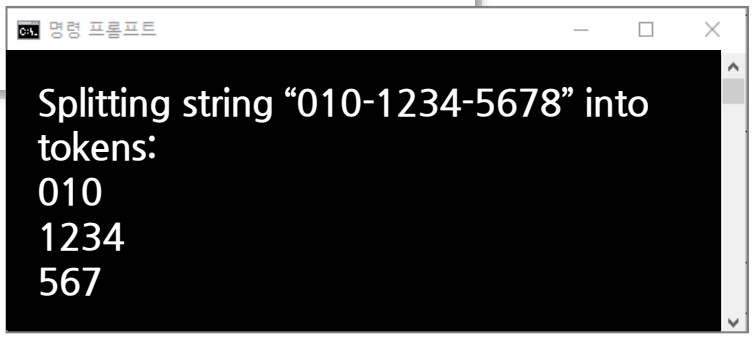
```
#include <stdio.h>
#include <string.h>

int main ()
{
  char str[] ="010-1234-5678";
  char * pch;
  printf ("Splitting string ₩"%s₩" into
tokens:₩n",str);
  pch = strtok (str,"-");
```

# 라이브러리 함수 활용

## 2 문자열 검색함수

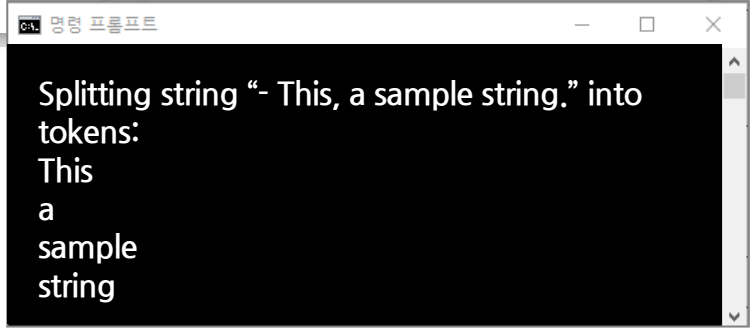
### 7 strtok

```
while (pch != NULL)
  {
    printf ("%s₩n",pch);
    pch = strtok (NULL, " ,.-");
  }
  return 0;
}
```




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

int main ()
{
  char str[] ="This, a sample string.";
  char * pch;
  printf ("Splitting string ₩"%s₩"
into tokens:₩n",str);
  pch = strtok (str," ,.-");
```

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

int main ()
{
  char str[] ="This, a sample string.";
  char * pch;
  printf ("Splitting string ₩"%s₩"
into tokens:₩n",str);
  pch = strtok (str," ,.-");
```



## 학습정리

### 1. 라이브러리 함수 이해

- 문자열 처리 관련 함수는 string.h를 include 함
- 문자열 비교함수 : memcmp, strcmp, strncmp
- 문자열 검색함수 : memchr, strchr, strrchr, strspn,strcspn,strstr
- 문자열을 검색하여 분리하는 함수 : strtok

### 2. 라이브러리 함수 활용

- 문자열 비교함수는 첫 번째 매개변수가 더 큰 값이면 양수, 같으면 0, 두 번째 매개변수가 크면 음수를 반환
- 문자열에서 특정 문자의 위치를 검색하는 함수 : strchr, strrchr
- 문자열에서 특정 문자열의 위치를 검색하는 함수 : strst
- strtok()함수는 문자열을 토큰으로 분리