2008 Spring

Computer Engineering Programming 1

Lesson 11
- 제12장 문자와 문자열

Lecturer: JUNBEOM YOO
jbyoo@konkuk.ac.kr

※ 본 강의자료는 생능출판사의 “PPT 강의자료”를 기반으로 제작되었습니다.

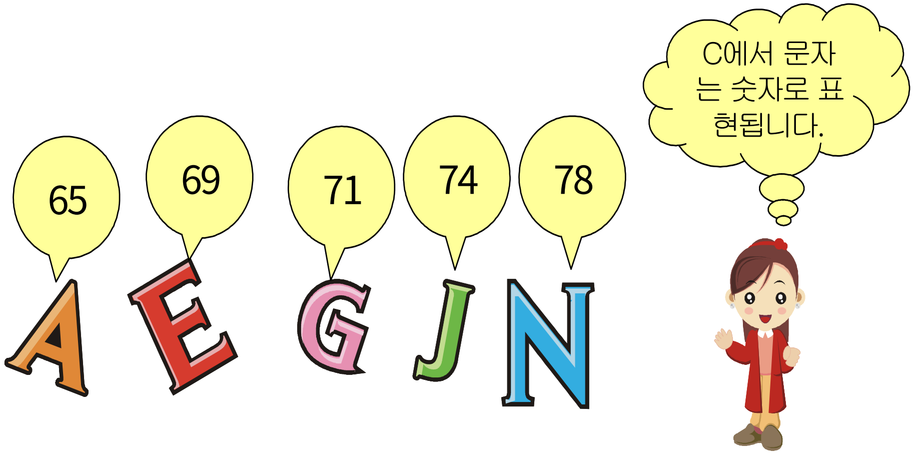
이번 장에서 학습할 내용

- 문자 표현 방법
- 문자열 표현 방법
- 문자열이란 무엇인가?
- 문자열의 입출력
- 문자처리 라이브러리 함수
- 표준입출력 라이브러리 함수

인간은 문자를 사용하여
정보를 표현하므로
문자열은 프로그램에서
중요한 위치를 차지하고
있다. 이번 장에서는
C에서의 문자열 처리
방법에 대하여 자세히
살펴볼 것입니다.

문자표현방법

- 컴퓨터에서는 각각의 문자에 숫자코드를 붙여서 표시한다.
- 아스키코드(ASCII code): 표준적인 8비트 문자코드
  - 0에서 127까지의 숫자를 이용하여 문자표현
- 유니코드(unicode): 표준적인 16비트 문자코드
  - 전세계의 모든 문자를 일관되게 표현하고 다룰 수 있도록 설계



문자 변수와 문자 상수

문자변수 문자상수

```
// 문자 상수
#include <stdio.h>

int main(void)
{
      char code1 = 'A';
      char code2 = 65;

      printf("code1=%c, code1=%d\n", code1,code1);
      printf("code2=%c, code2=%d\n", code2,code2);
      return 0;
}
```

```
code1=A, code1=65
code2=A, code2=65
```

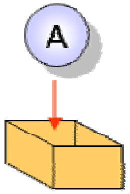
아스키 코드 출력

```c
// 아스키 코드 출력
#include <stdio.h>
int main(void)
{
	unsigned char code;

	for(code = 32; code < 128; code++)
	{
		printf("아스키 코드 %d은 %c입니다.\n", code, code);
	}
	return 0;
}
```

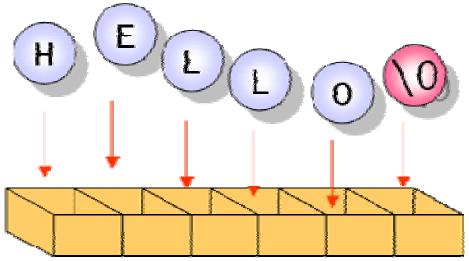
```
아스키 코드 32은  입니다.
아스키 코드 33은 !입니다.
...
아스키 코드 65은 A입니다.
아스키 코드 66은 B입니다.
...
아스키 코드 97은 a입니다.
아스키 코드 98은 b입니다.
...
아스키 코드 126은 ~입니다.
아스키 코드 127은 입니다.
```

문자열 표현 방법

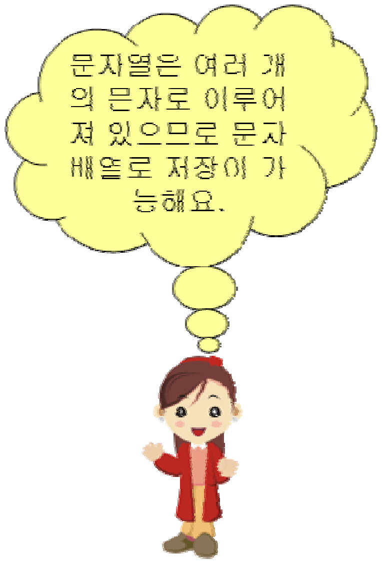
- *문자열(string):* 문자들이 여러 개 모인 것
  - "A"
  - "Hello World!"
  - "변수 score의 값은 %d입니다“
- *문자열 상수*
  - "Hello World"
  - "Hong"
  - "string!#$"
  - "guest123"
  - "ascii code = %d“
- *문자열 변수*
  - char형 배열



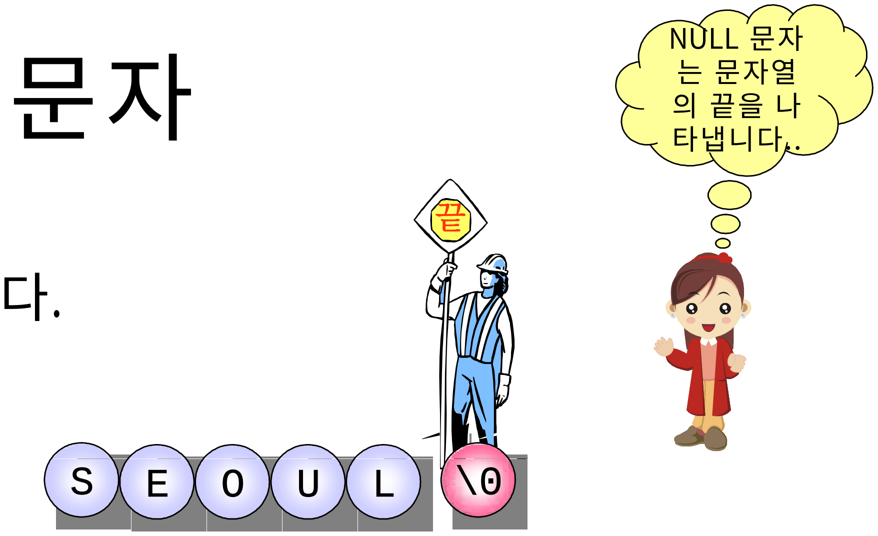
하나의 문자는 char형 변수로 저장



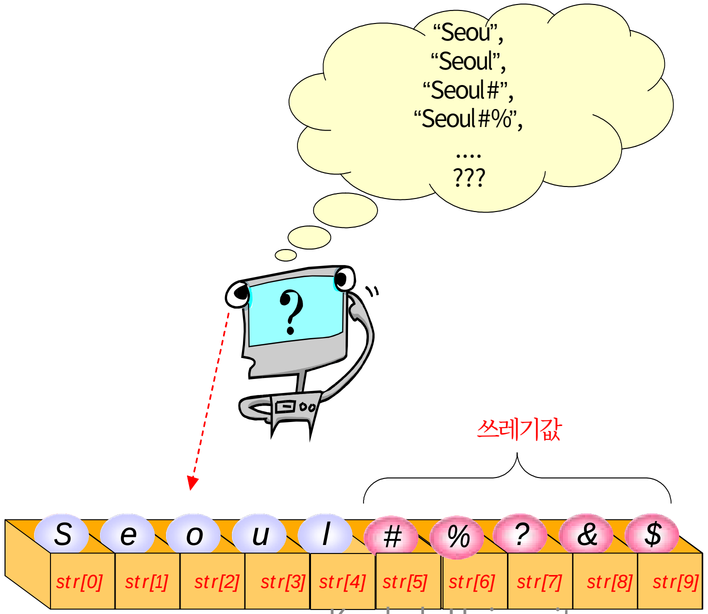
문자열은 char형 배열로 저장


문자열은 여러 개의 문자로 이루어져 있으므로 문자 배열로 저장이 가능해요.

NULL 문자



- NULL 문자: 문자열의 끝을 나타낸다.
- 문자열은 어디서 종료되는지 알 수가 없으므로 표시를 해주어야 한다.



문자 배열의 초기화

1. 문자 배열 원소들을 중괄호 안에 넣어주는 방법

   – char str[6] = { 'H', 'e', 'l', 'l', 'o', '\ 0' };

2. 문자열 상수를 사용하여 초기화하는 방법

   – char str[6] = "Hello";

3. 만약 배열을 크기를 지정하지 않으면 컴파일러가 자동으로 배열의 크기를 초기화값에 맞추어 설정

   – char str[] = "C Bible"; // 배열의 크기는 7이 된다.

문자 배열에 문자를 저장

1. 각각의 문자 배열 원소에 원하는 문자를 개별적으로 대입하는 방법이다.
   - str[0] = 'W';
   - str[1] = 'o';
   - str[2] = 'r';
   - str[3] = 'l';
   - str[4] = 'd';
   - str[5] = '\0';

2. strcpy()를 사용하여 문자열을 문자 배열에 복사
   - strcpy(str, "World");

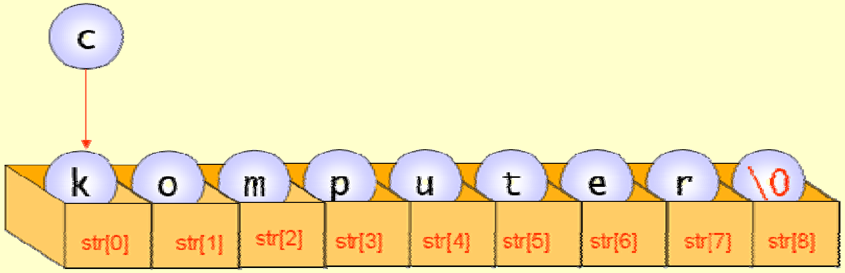
예제 #1

```c
#include <stdio.h>

int main(void)
{
      char str1[6] = "Seoul"
      char str2[3] = { 'i', 's' };
      char str3[] = "the capital city of Korea."

      printf("%s %s %s\n", str1, str2, str3);
}
```

```
Seoul is the capital city of Korea.
```

예제 #2

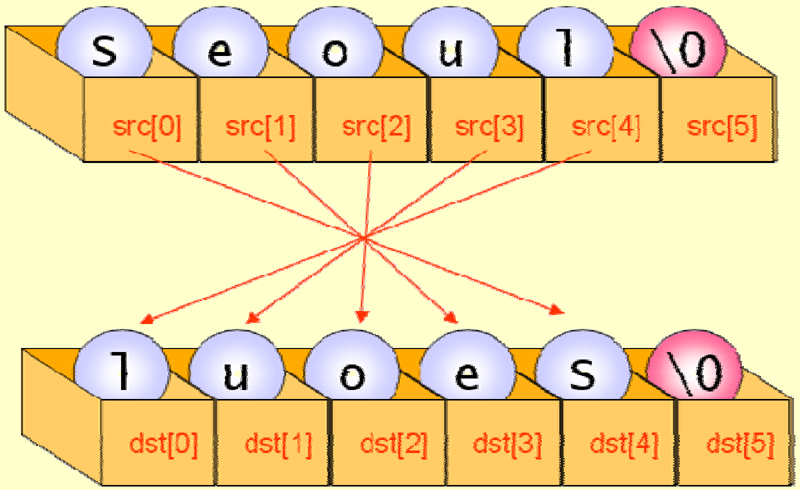
```c
#include <stdio.h>

int main(void)
{
    char str[] = "komputer";
    int i;

    for(i=0;i<8;i++)
        printf("%c ", str[i]);

    str[0] = 'c';
    printf("\n");

    for(i=0;i<8;i++)
        printf("%c ", str[i]);
    return 0
```



```
k o m p u t e r
c o m p u t e r
```

문자열 역순 예제

```
#include <stdio.h>

int main(void)
{
      char src[] = "Seoul";
      char dst[6];
      int i;

      printf("원래 문자열=%s\n", src);

      i = 0;
      while(src[i] != '\0')
      {
                dst[i] = src[4 - i];
                i++;
      }
      dst[i] = '\0';

      printf("역순 문자열=%s\n", dst);
      return 0;
}
```



```
원래 문자열=Seoul
역순 문자열=luoeS
```

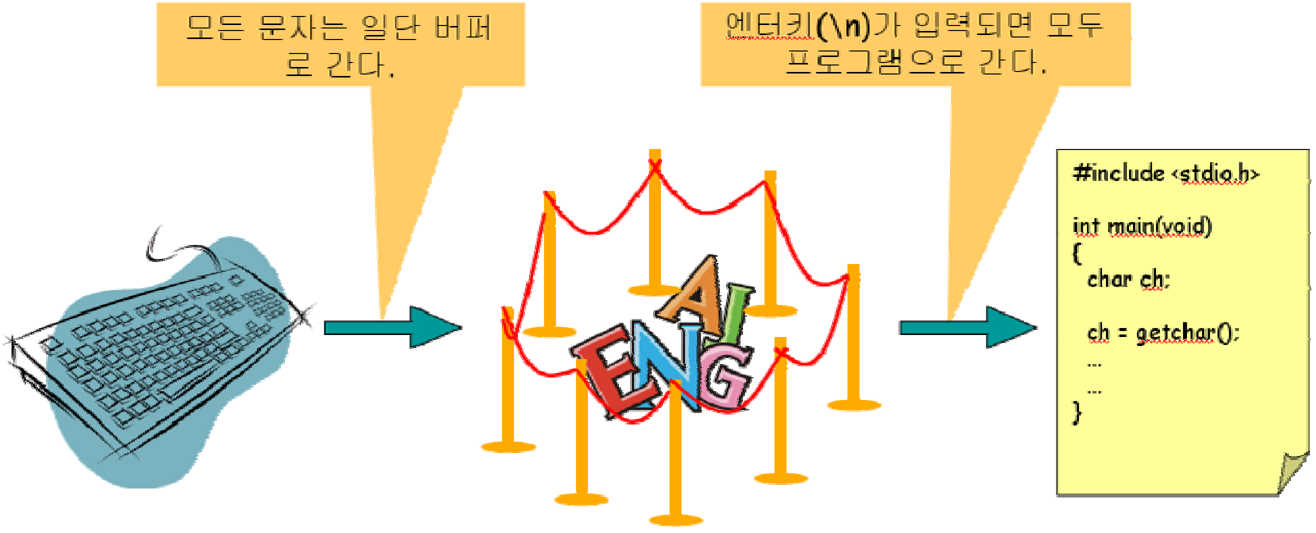
문자열 길이 계산 예제

```c
// 문자열의 길이를 구하는 프로그램
#include <stdio.h>

int main(void)
{
	char str[30] = "C language is easy";
	int i = 0;

	while(str[i] != 0)
		i++;

	printf("문자열\"%s\"의 길이는 %d입니다.\n", str, i);

	return 0;
}
```

```
문자열 "C language is easy"의 길이는 18입니다.
```

문자 입출력 라이브러리

입출력 함수	설명
int getchar(void)	하나의 문자를 읽어서 반환한다.
void putchar(int c)	변수 c에 저장된 문자를 출력한다.
int getch(void)	하나의 문자를 읽어서 반환한다(버퍼를 사용하지 않음).
void putch(int c)	변수 c에 저장된 문자를 출력한다(버퍼를 사용하지 않음).
scanf("%c", &c)	하나의 문자를 읽어서 변수 c에 저장한다.
printf("%c", c);	변수 c에 저장된 문자를 출력한다.


모든 문자는 일단 버퍼로 간다.
엔터키(\n)가 입력되면 모두 프로그램으로 간다.
AI
ENG

```
#include <stdio.h>

int main(void)
{
  char ch;

  ch = getchar();
  ...
  ...
}
```

키보드

버퍼

프로그램

getchar(), putchar()

```
// getchar()의 사용
#include <stdio.h>

int main(void)
{
    int ch;                         // 정수형에 주의

    while(1)
    {
        ch = getchar();             // 문자를 입력 받는다.
        if( ch == 'q' ) break;
        putchar(ch);
    }
    return 0;
}
```

```
A
A
B
B
q
```

getch(), putch()

```
// getch()의 사용
#include <conio.h>

int main(void)
{
      int ch;               // 정수형에 주의

      while(1)
      {
            ch = getch();         // 문자를 입력받는다.
            if( ch == 'q' ) break;
            putch(ch);
      }
      return 0;
}
```

버퍼를 사용하지 않는다

```
ABCDEFGH
```

getch(), getche(), getchar()

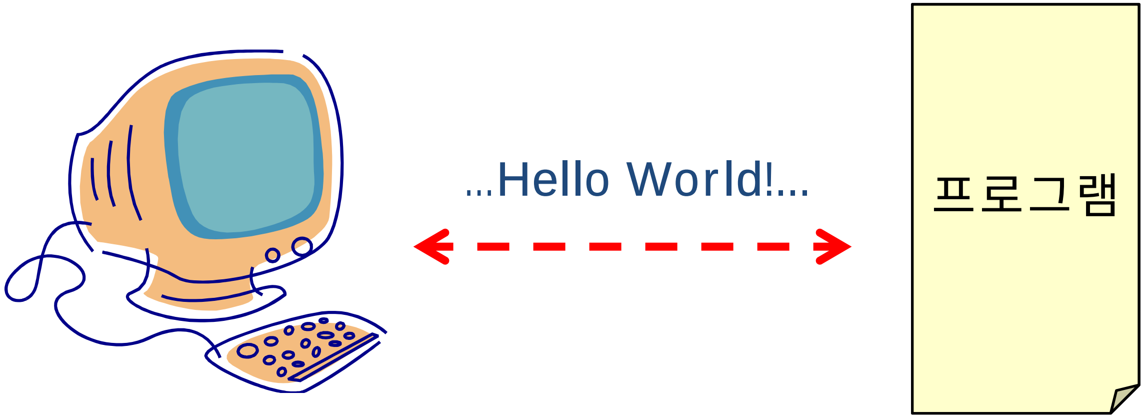
	헤더파일	버퍼사용여부	에코여부	응답성	문자수정여부
getchar()	<stdio.h>	사용함 (엔터키를 눌러입력됨)	에코	줄단위	가능
getch()	<conio.h>	사용하지 않음	에코하지 않음	문자단위	불가능
getche()	<conio.h>	사용하지 않음	에코	문자단위	불가능

용도에 맞는 것을 골라 사용하세요!

버퍼가 없이 바로 받으려면 getch()를 사용합니다.

문자열 입출력 라이브러리 함수

입출력 함수	설명
int scanf("%s", s)	문자열을 읽어서 문자배열 s[]에 저장
int printf("%s", s)	배열 s[]에 저장되어 있는 문자열을 출력한다.
char *gets(char *s)	한 줄의 문자열을 읽어서 문자 배열 s[]에 저장한다.
int puts(const char *s)	배열 s[]에 저장되어 있는 한 줄의 문자열을 출력한다.



scanf(), printf() 문자열 입출력

- scanf()의 사용법
  - char str[10];
  - scanf("%s", str);

- scanf()는 한 번에 두개 이상의 문자열도 받아들일 수 있다.
  - char s1[10];
  - char s2[10];
  - char s3[10];
  - scanf("%s%s%s", s1,s2,s3);
  - // 사용자가 one two three와 같이 입력하면 s1에는 one이, s2에는 two가, s3에는 three가 할당된다.

gets()와 puts() 문자열 입출력

- gets()
  - 표준 입력으로부터 엔터키가 나올 때까지 한 줄의 라인을 입력
  - 문자열에 줄바꿈 문자('\n')는 포함되지 않으며 대신에 자동으로 NULL 문자('\0')를 추가한다.
  - 입력 받은 문자열은 buffer가 가리키는 주소에 저장된다.

```
char *gets(char *buffer);
int puts(const char *str);
```

- puts()
  - str이 가리키는 문자열을 받아서 화면에 출력
  - NULL 문자('\0')는 줄바꿈 문자('\n')로 변경

```
        char *menu = "파일열기: open, 파일닫기: close";
        puts("메뉴에서 하나를 선택하시오.");
        puts(str);
```

예제

```
#include <stdio.h>

int main( void )
{
      char buffer[21]; // 20개의 문자와 '\0'을 저장할 수 있다.

      printf("문자열을 입력하시오.\n");
      gets( buffer );

      printf("입력된 라인은 다음과 같습니다.\n");
      puts(buffer);
      return 0;
}
```

```
문자열을 입력하시오.
Hello!
입력된 라인은 다음과 같습니다.
Hello!
```

문자 처리 라이브러리 함수

- 문자를 검사하거나 문자를 변환한다.

함수	설명
isalpha(c)	c가 영문자인가?(a-z, A-Z)
isupper(c)	c가 대문자인가?(A-Z)
islower(c)	c가 소문자인가?(a-z)
isdigit(c)	c가 숫자인가?(0-9)
isalnum(c)	c가 영문자이나 숫자인가?(a-z, A-Z, 0-9)
isxdigit(c)	c가 16진수의 숫자인가?(0-9, A-F, a-f)
isspace(c)	c가 공백문자인가?(‘ ’, ‘\n', '\t', '\v', '\r')
ispunct(c)	c가 구두점 문자인가?
isprint(c)	C가 출력가능한 문자인가?
iscntrl(c)	c가 제어 문자인가?
isascii(c)	c가 아스키 코드인가?
toupper(c)	c를 대문자로 바꾼다.
tolower(c)	c를 소문자로 바꾼다.
toascii(c)	c를 아스키 코드로 바꾼다.

예제

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

int main( void )
{
        int c;

        while((c = getchar()) != EOF)
        {
                if( islower(c) )
                        c = toupper(c);
                putchar(c);
        }
        return 0;
}
```

소문자인지 검사

대문자로 변환

```
abcdef
ABCDEF
^Z
```

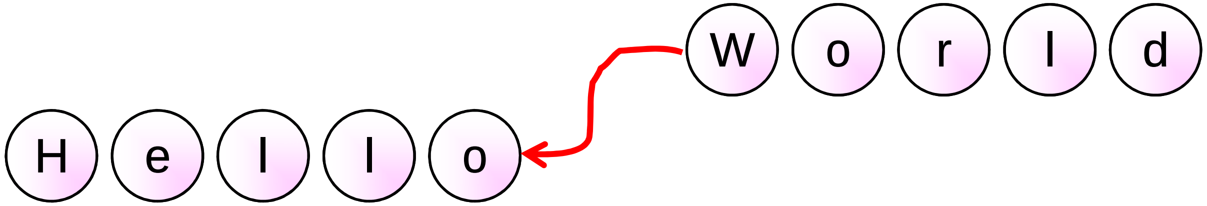
예제

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

int main( void )
{
     int c;

     while((c = getch()) != 'z')
     {
               printf("------------------------\n");
               printf("isdigit(%c) = %d\n", c, isdigit(c));
               printf("isalpha(%c) = %d\n", c, isalpha(c));
               printf("islower(%c) = %d\n", c, islower(c));
               printf("ispunct(%c) = %d\n", c, ispunct(c));
               printf("isxdigit(%c) = %d\n", c, isxdigit(c));
               printf("isprint(%c) = %d\n", c, isprint(c));
               printf("------------------------\n\n");
     }
     return 0;
}
```

```
------------------------
isdigit(') = 0
isalpha(') = 0
islower(') = 0
ispunct(') = 16
isxdigit(') = 0
isprint(') = 16
------------------------
…
```

문자열 처리 라이브러리

함수	설명
strlen(s)	문자열 s의 길이를 구한다.
strcpy(s1, s2)	s2를 s1에 복사한다.
strcat(s1, s2)	s2를 s1의 끝에 붙여넣는다.
strcmp(s1, s2)	s1과 s2를 비교한다.
strncpy(s1, s2, n)	s2의 최대 n개의 문자를 s1에 복사한다.
strncat(s1, s2, n)	s2의 최대 n개의 문자를 s1의 끝에 붙여넣는다.
strncmp(s1, s2, n)	최대 n개의 문자까지 s1과 s2를 비교한다.
strchr(s, c)	문자열 s안에서 문자 c를 찾는다.
strstr(s1, s2)	문자열 s1에서 문자열 s2를 찾는다.

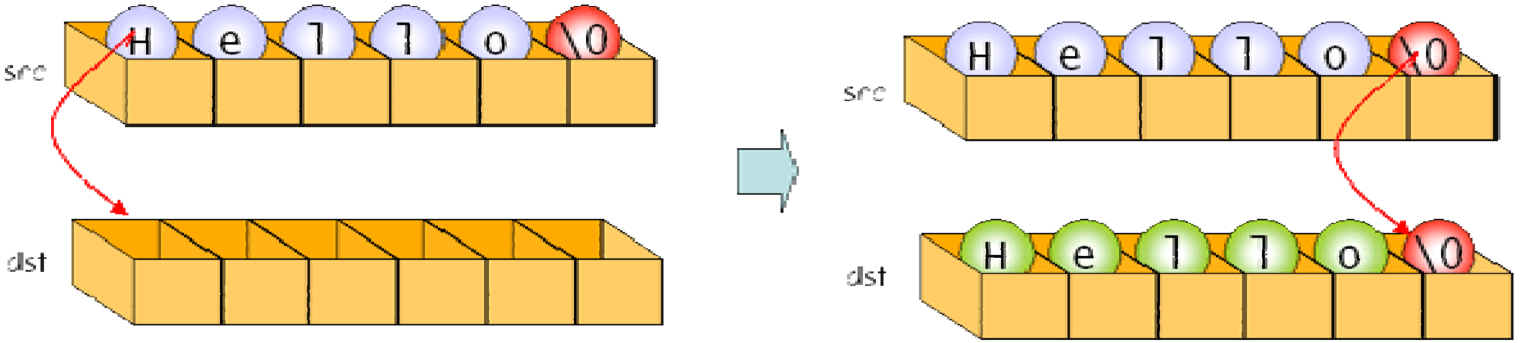


문자열 길이, 복사

- 문자열의 길이
  - strlen(“Hello”)는 5를 반환

- 문자열 복사

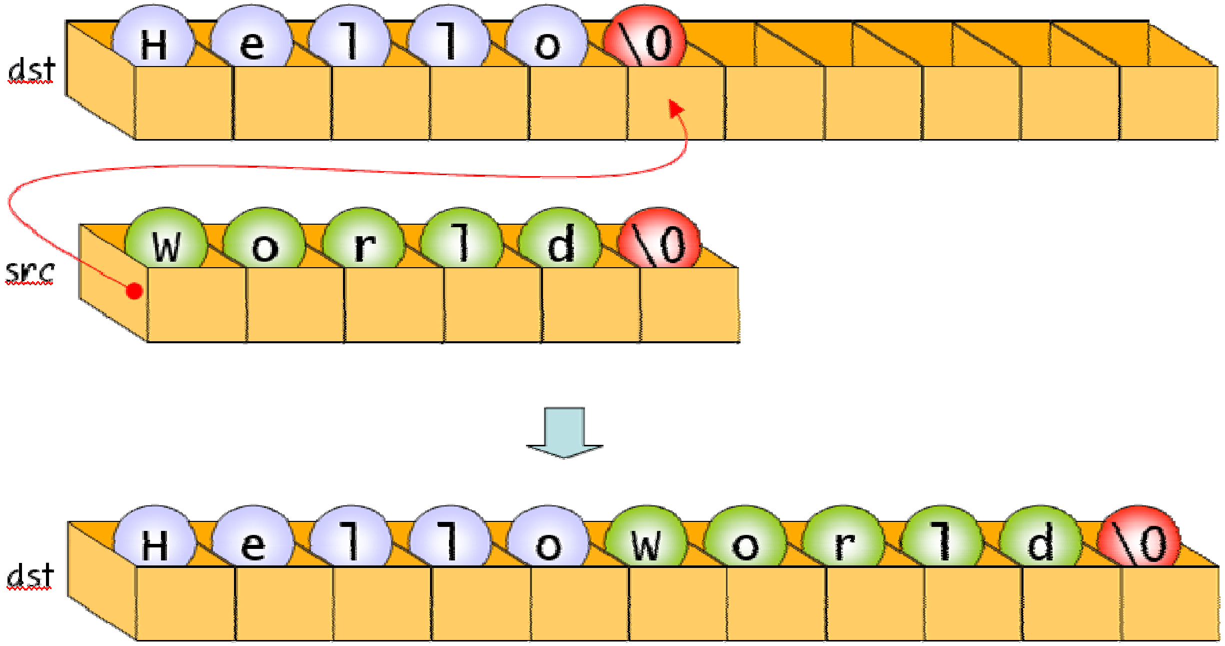
```
char dst[6];
char src[6] = “Hello";
strcpy(dst, src);
```



문자열 연결

- 문자열 연결

```
char dst[12] = "Hello";
char src[6] = "World";
strcat(dst, src);
```



예제

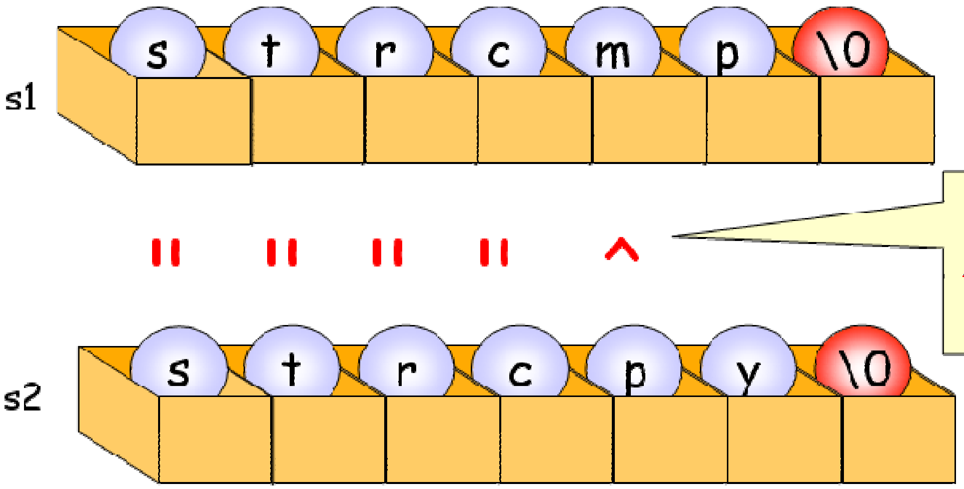
```c
// strcpy와 strcat
#include <string.h>
#include <stdio.h>

int main( void )
{
      char string[80];

      strcpy( string, "Hello world from " );
      strcat( string, "strcpy " );
      strcat( string, "and " );
      strcat( string, "strcat!" );
      printf( "string = %s\n", string );
      return 0;
}
```

```
string = Hello world from strcpy and strcat!
```

문자열 비교

```
int strcmp( const char *s1,  const char *s2 );
```

반환값	s1과 s2의 관계
<0	s1이 s2보다 작다
0	s1이 s2와 같다.
>0	s1이 s2보다 크다.



m이 p보다 아스키 코드값이 작으므로 음수가 반환된다.

예제

```c
// strcmp() 함수
#include <string.h>
#include <stdio.h>

int main( void )
{
	char s1[80];		// 첫번째 단어를 저장할 문자배열
	char s2[80];		// 두번째 단어를 저장할 문자배열
	int result;

	printf("첫번째 단어를 입력하시오:");
	scanf("%s", s1);
	printf("두번째 단어를 입력하시오:");
	scanf("%s", s2);

	result = strcmp(s1, s2);
	if( result < 0 )
		printf("%s가 %s보다 앞에 있습니다.\n", s1, s2);
	else if( result == 0 )
		printf("%s가 %s와 같습니다.\n", s1, s2);
	else
		printf("%s가 %s보다 뒤에 있습니다.\n", s1, s2);
	return 0;
}
```

```
첫번째 단어를 입력하시오:Hello
두번째 단어를 입력하시오:World
Hello가 World보다 앞에 있습니다.
```

문자 검색, 문자열 검색

- 문자열에서 문자 검색

```
char s[] = "language";      // 문자열
char c = 'g';               // 찾고자 하는 문자
char *p;                    // 문자 포인터

p = strchr(s, c);           // str에서 c를 찾는다.
```

- 문자열에서 문자열 검색

```
char s[] = "A joy that's shared is a joy made double"; // 입력 문자열
char sub[] = "joy";                                   // 찾으려고 하는 문자열
char *p;                                              // 문자 검색 위치 저장 포인터

p = strstr(s, sub);                                   // s에서 sub를 찾는다.
```

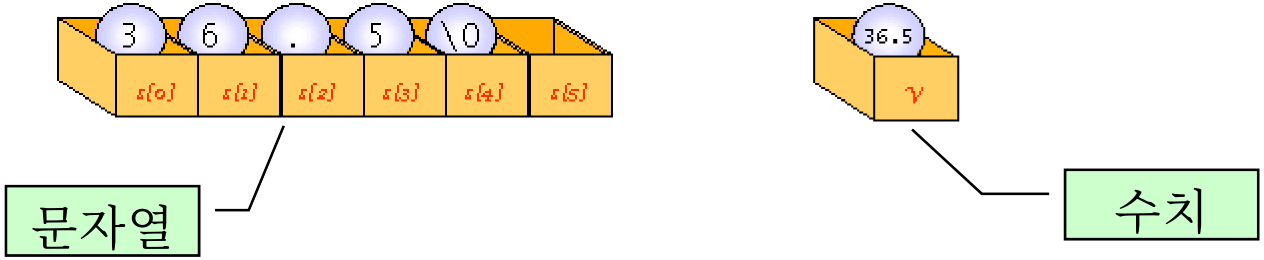
문자열 토큰 분리

```
// strtok 함수의사용예
#include <string.h>
#include <stdio.h>

char s[] = "Man is immortal, because he has a soul";
char seps[]   = " ,\t\n";
char *token;

int main( void )
{
      // 문자열을 전달하고 다음 토큰을 얻는다.
      token = strtok( s, seps );
      while( token != NULL )
      {
                // 문자열 s에 토큰이 있는 동안 반복한다.
           printf( "토큰: %s\n", token );
                // 다음 토큰을 얻는다.
           token = strtok( NULL, seps ); //
      }
}
```

```
토큰: Man
토큰: is
토큰: immortal
토큰: because
토큰: he
토큰: has
토큰: a
토큰: soul
```

문자열 수치 변환

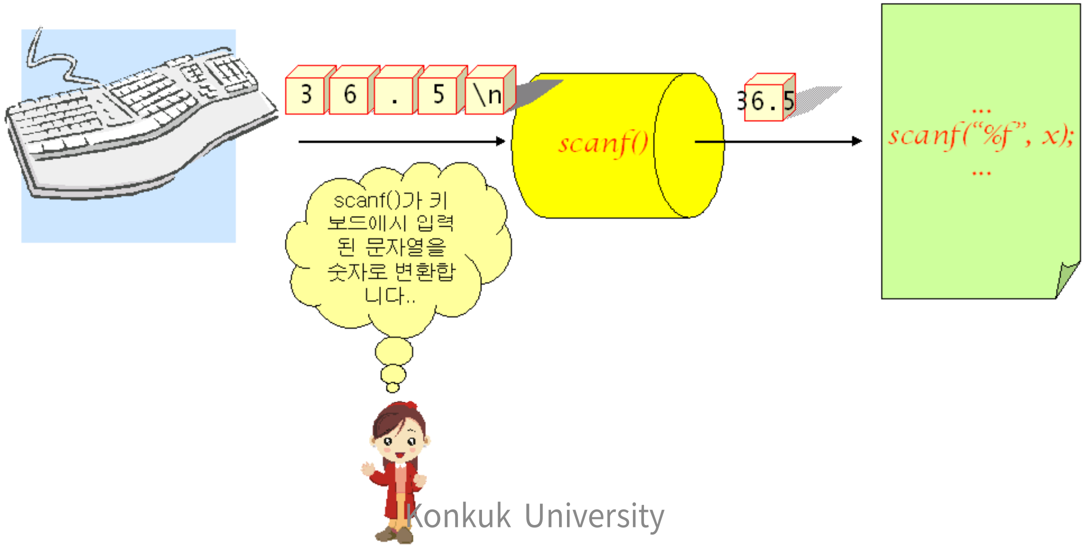
- 문자열과 수치



문자열

수치

- scanf() 함수는 문자열을 수치로 변환한다.



scanf()가 키보드에서 입력된 문자열을 숫자로 변환합니다..

문자열을 수치로 변환하는 전용함수

- 전용 함수는 scanf()보다 크기가 작다.
- stdlib.h에 원형 정의- 반드시 포함

함수	설명
int atoi(const char *str);	str을 int형으로 변환한다.
long atoi(const char *str);	str을 long형으로 변환한다.
double atof(const char *str);	str을 double형으로 변환한다.

문자열 토큰 분리

```c
// atoi() 함수
#include <stdio.h>
#include <stdlib.h>

int main( void )
{
	char s[30];
	char t[] = "36.5";
	int i;
	double v;

	printf("정수를 입력하시오:");
	gets(s);
	i = atoi(s);
	printf("입력된 정수: %d \n", i);

	v = atof(t);
	printf("변환된 실수: %f", v);

	return 0;
}
```

```
정수를 입력하시오:89
입력된 정수: 89
변환된 실수: 36.500000
```

sscanf(), sprintf()

함수	설명
sscanf(s,...)	문자열 s로부터 지정된 형식으로 수치를 읽어서 변수에 저장한다.
sprintf(s,...)	변수의 값을 형식 지정자에 따라 문자열 형태로 문자 배열 s에 저장한다.

```
int main( void )
{
        char s1[] = "100";
        char s2[] = "12.93";
        char buffer[100];

        int i;
        double d;
        double result;

        sscanf(s1, "%d", &i);
        sscanf(s2, "%lf", &d);

        result = i + d;

        sprintf(buffer, "%f", result);
        printf("연산 결과는 %s입니다.\n", buffer);

        return 0;
}
```

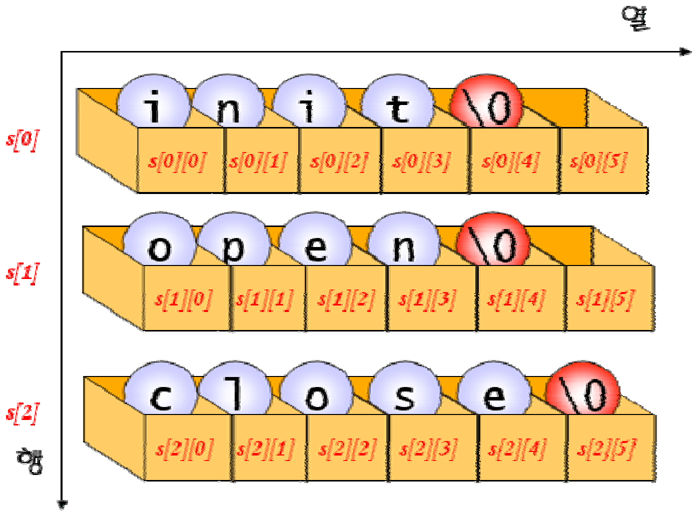
```
연산 결과는 112.930000입니다.
```

문자열의 배열

- (Q)문자열이 여러 개 있는 경우에는 어떤 구조를 사용하여 저장하면 제일 좋을까?
- (A) 여러 개의 문자 배열을 각각 만들어도 되지만 문자열의 배열을 만드는 것이 여러모로 간편하다.

- 문자열이 문자 배열에 저장되므로 문자열의 배열은 배열의 배열, 즉 2차원 문자 배열이 된다.

```
char s[3][6] = {
      "init",
      "open",
      "close"
};
```


열
s[0]
i n i t \0
s[0][0] s[0][1] s[0][2] s[0][3] s[0][4] s[0][5]
s[1]
o p e n \0
s[1][0] s[1][1] s[1][2] s[1][3] s[1][4] s[1][5]
s[2]
c l o s e \0
s[2][0] s[2][1] s[2][2] s[2][3] s[2][4] s[2][5]
행

메뉴 디스플레이

```c
#include <stdio.h>

int main( void )
{
    int i;
    char menu[5][10] = {
            "init",
            "open",
            "close",
            "read",
            "write"
    };

    for(i = 0; i < 5; i++)
            printf("%d 번째 메뉴: %s \n", i, menu[i]);

    return 0;
}
```

```
0 번째 메뉴: init
1 번째 메뉴: open
2 번째 메뉴: close
3 번째 메뉴: read
4 번째 메뉴: write
```

메뉴 선택

```c
#include <stdio.h>

int main( void )
{
    int i;
    char buffer[10];
    char menu[5][10] = {
            "init",
            "open",
            "close",
            "read",
            "write"
    };

    printf("메뉴를 입력하시오:");
    scanf("%s", buffer);

    for(i = 0; i < 5; i++)
            if( strcmp(buffer, menu[i]) == 0 )
                        printf("%d번째 메뉴를 입력하였습니다.\n", i);

    return 0;
}
```

```
메뉴를 입력하시오:open
1번째 메뉴를 입력하였습니다.
```

단어 카운팅

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

int count_word(const char *s);

int main( void )
{
    printf("%d\n", count_word("the c book..."));

    return 0;
}

int count_word (  const char * s )
{
    int i, wc = 0, waiting = 1;

    for( i = 0; s[i] != NULL; ++i)     // s의 각 글자 조사
           if( isalpha(s[i]) )                    // s의 글자가 알파벳이면
           {
                      if( waiting )               // 워드를 기다리고 있으면
                      {
                                 wc++;                       // 카운터를 증가
                                 waiting = 0;  // 워드를 처리하는 중
                      }
           }
           else                                              // 알파벳이 아니면
                      waiting = 1;                // 워드를 기다린다.

    return wc;
}
```

문자열 비교

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

int str_ncmp(const char *s1, const char *s2, int count);

int main( void )
{
	printf("%d\n", str_ncmp("language C++", "language C", 5));

	return 0;
}

//	returns <0 if s1 < s2
//	returns  0 if s1 == s2
//	returns >0 if s1 > s2
int str_ncmp (  const char * s1, const char * s2, int count )
{
	if (!count)
		return(0);

	while (--count && *s1 && *s1 == *s2)
	{
		s1++;
		s2++;
	}

	return( *s1 - *s2 );
}
```

한영 사전 구현

```
#define ENTRIES 5

int main( void )
{
      int i, index;
      char dic[ENTRIES][2][30] = {
               {"book", "책"},
               {"boy", "소년"},
               {"computer", "컴퓨터"},
               {"lanuguage", "언어"},
               {"rain", "비"},
      };
      char word[30];

      printf("단어를 입력하시오:");
      scanf("%s", word);

      index = 0;
      for(i = 0; i < ENTRIES; i++)
      {
               if( strcmp(dic[index][0], word) == 0 )
               {
                              printf("%s: %s\n", word, dic[index][1]);
                              return 0;
               }
               index++;
      }
      printf("사전에서 발견되지 않았습니다.\n");
}
```

문자열->정수

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

int stoi( const char *s );

int main(void)
{
	printf("%d\n", stoi("-123"));
}

int stoi( const char *s )
{
	int c;		// 현재의 글자
	int total =0;	// 현재의 합계
	int sign;

	c = *s++;
	sign = c;		// 부호를 저장한다.
	if (c == '-' || c == '+')
		c = *s++;	// 부호를 제거한다.

	while (isdigit(c)) {
		total = 10 * total + (c - '0');	// 누적시킨다.
		c = *s++;	// 다음 글자를 얻는다.
	}
	if (sign == '-')
		return -total;
	else
		return total;	// 필요하면 음수로 만든다.
}
```

Q & A