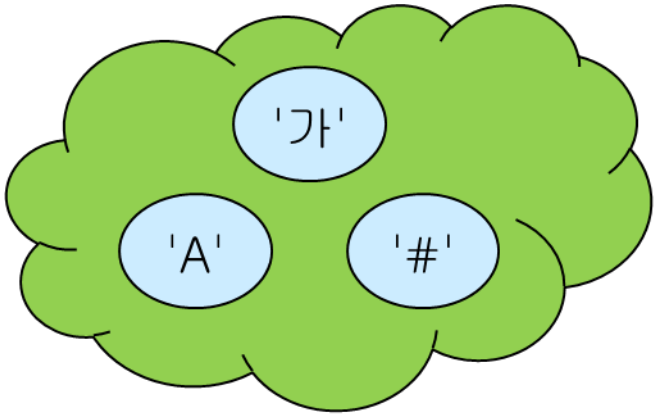
문자와 문자열 1

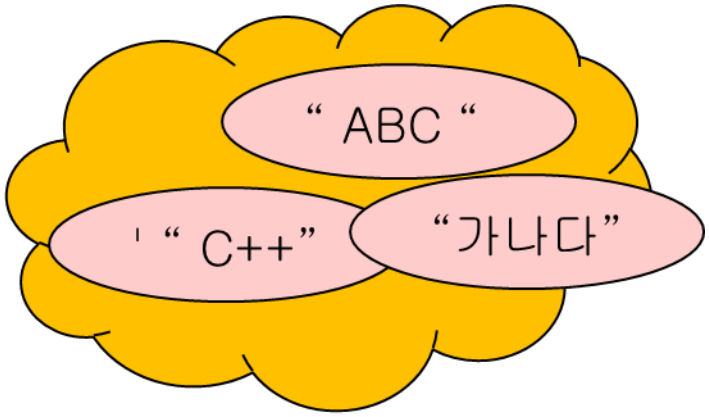
이번 장에서 학습할 내용

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

문자와 문자열



문자



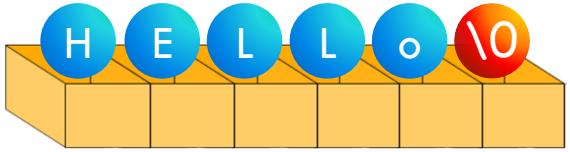
문자열

문자열 표현 방법

- *문자열(string):* 문자들이 여러 개 모인 것
  - "A "
  - "Hello World!"
  - "변수 score의 값은 %d입니다"

- *문자열 변수*
  - 변경 가능한 문자열을 저장할 수 있는 변수
  - 어디에 저장하면 좋은가?

하나의 문자는 char형 변수로 저장

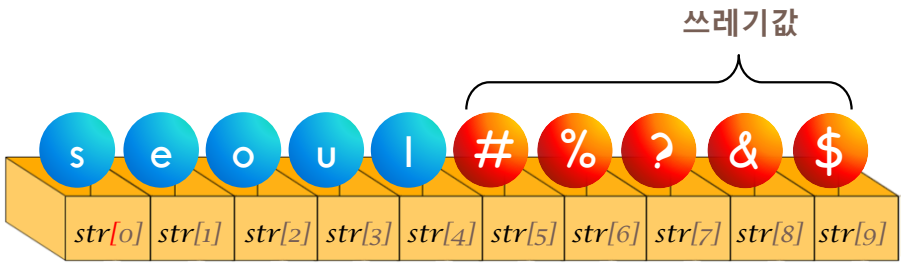


문자열은 char형 배열로 저장

NULL 문자

- NULL 문자: 문자열의 끝을 나타낸다.

- 문자열은 어디서 종료되는지 알 수가 없으므로 표시를 해주어야 한다.



예제 #1

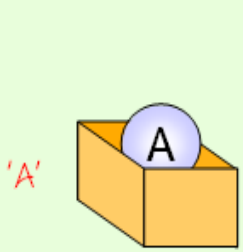
```c
#include <stdio.h>

int main(void)
{
  int i;
  char str[4];
  str[0] = 'a';
  str[1] = 'b';
  str[2] = 'c';
  str[3] = '\0';

  i = 0;
  while(str[i] != '\0')          {
          printf("%c", str[i]);
          i++;
  }
  return 0;
}
```

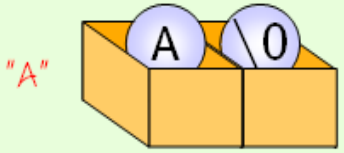
```c
int main(void)
{
    int i;
    char str[4]="abc";
    printf("%s", str);
    printf(str);
    return 0;
}
```

참고

참고 사항

여기서 잠깐 C 언어 퀴즈를 풀고 지나가자. C 언어 코드에서 A, 'A', "A"의 차이를 생각해보자.





A : 컴파일러는 A를 변수의 이름으로 간주한다.

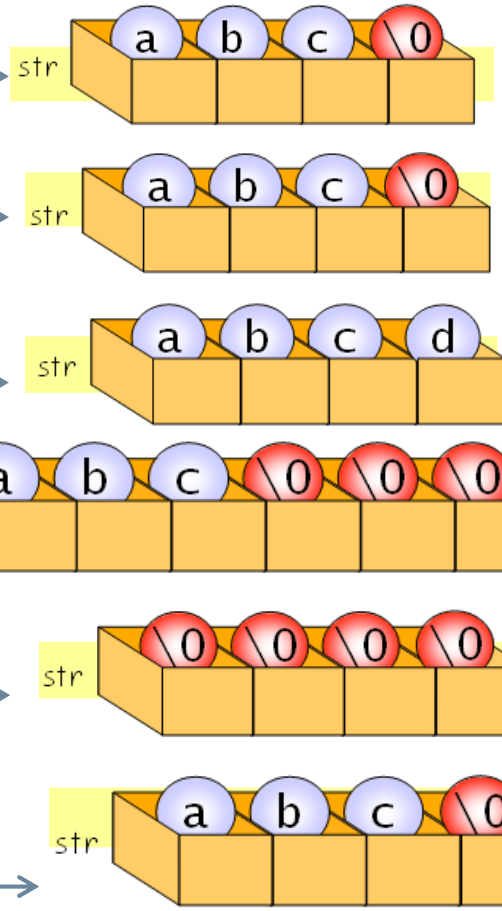
'A': 문자 A를 나타낸다.

"A": 문자 A만으로 이루어진 문자열을 나타낸다. 'A'와는 다르다.

여기서 주의해야 할 것은 'A'와 "A"의 차이점이다. 'A'는 하나의 문자를 나타내며 문자 A에 대한 아스키 코드와 같다. "A"는 문자열이며 A의 아스키 코드에 문자열 끝을 나타내는 NULL 문자가 추가된다.

문자 배열의 초기화

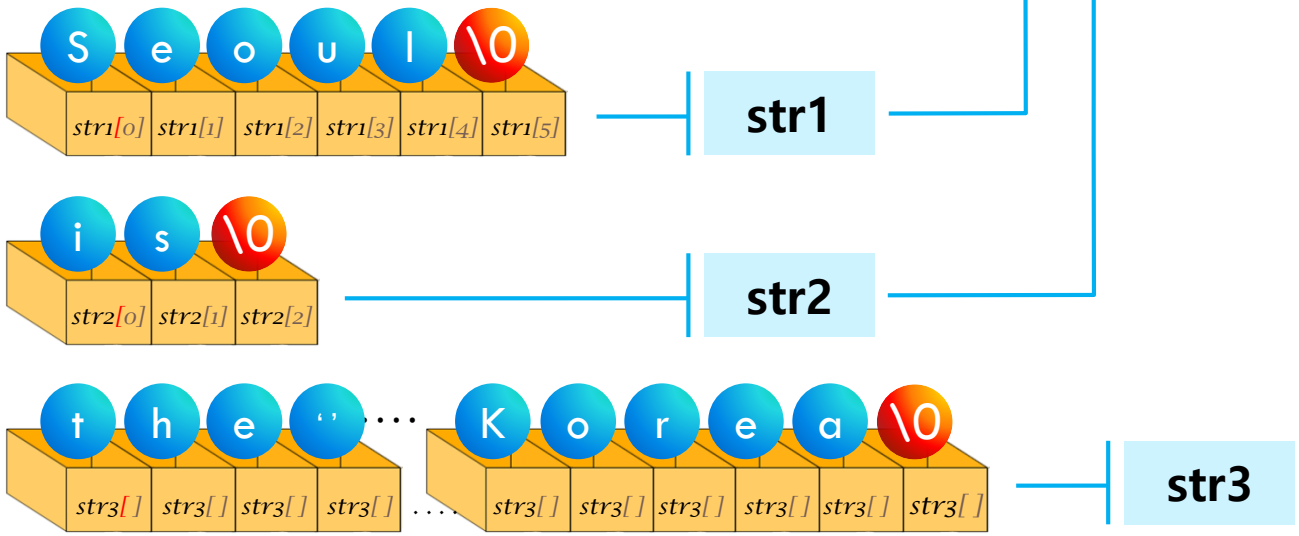
- char str[4] = { 'a', 'b', 'c', '\0' };
- char str[4] = "abc“;
- char str[4] = "abcdef";
- char str[6] = "abc";
- char str[4] = "";
- char str[] = ""abc;



예제 #2

```c
#include <stdio.h>

int main(void)
{
    char str1[6] = "Seoul";
    char str2[3] = { 'i', 's' , '\0' };
    char str3[] = "the capital city of Korea.";
    printf("%s %s %s\n", str1, str2, str3);
    return 0;
}
```

Seoul is the capital city of Korea.



```c
#include <stdio.h>
int main(void)
{
        char src[] = "The worst things to eat before you sleep";
        char dst[100];
        int i;
        printf("원본 문자열=%s\n", src);
        for(i=0 ; src[i] != '\0' ; i++)
                dst[i] = src[i];
        dst[i] = '\0';
        printf("복사된 문자열=%s\n", dst);
        return 0;
}
```

NULL과 '₩0'은 같다.

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;
}
```

NULL의 아스키코드값은 0
'₩0', NULL, 0 모두 가능

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

문자 배열을 strcpy()로 변경

1. strcpy()를 사용하여 문자열을 문자 배열에 복사

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


    printf("문자열₩"%s₩"₩n", str);
    strcpy(str, "World");
    printf("문자열₩"%s₩"₩n", str);

    return 0;
}
```

문자 입출력 라이브러리

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

getchar(), putchar() -> 버퍼사용

```
// getchar()의 사용
#include <stdio.h>
int main(void)
{
        int ch;             // 정수형에 주의
        while( (ch = getchar()) != EOF )
                putchar(ch);
        return 0;
}
```

End Of File을 나타내는 문자, EOF는 정수형이다.

_getch(), _putch() -> 버퍼사용 안함

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

int main(void)
{
        int ch;
        while( (ch = _getch()) != 'q' )
                _putch(ch);
        return 0;
}
```

_getche()와 비교하기

_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

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_s(char *s, int size)	한 줄의 문자열을 읽어서 문자 배열 s[]에 저장한다.
int puts(const char *s)	배열 s[]에 저장되어 있는 한 줄의 문자열을 출력한다.

예제

```c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    char name[100];
    char address[100];

    printf("이름을 입력하시오: ");
    scanf("%s", name);
    printf("주소를 입력하시오: ");
    scanf("%s", address);

    printf("이름: %s \n", name);
    printf("주소: %s \n", address);
    return 0;
}
```

scanf_s는 처리 불가능 scanf로 실습

gets_s(name, 100);

gets_s(address, 100);

puts(name)
puts(address)

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)
isspace(c)	c가 공백문자인가?(‘ ’, ‘\n', '\t', '\v', '\r')
toupper(c)	C를 대문자로 바꾼다
tolower(c)	C를 소문자로 바꾼다

예제

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

int main( void )
{
    int c;

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

```
abcdef
ABCDEF
^Z
```

EOF를 키보드에서 입력하려면 ^Z

Lab: 단어 세기

- 문자열 안에 들어 있는 단어의 개수를 세는 프로그램을 작성하여 보자. 문자열이 “the c book... ” 이라면 다음과 같은 출력이 나온다.

예제

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

int count_word(char* s);

int main(void)
{

    int wc = count_word("the c book...");
    printf("%d₩n", wc);
    return 0;
}

int count_word(char* s)
{
    int i, wc = 0, waiting = 1;
    for (i = 0; s[i] != NULL; ++i)
    {
        if (isalpha(s[i]))
        {
            if (waiting) {
                wc++;
                waiting = 0;
            }
        }
        else waiting = 1;
    }
    return wc;
}
```

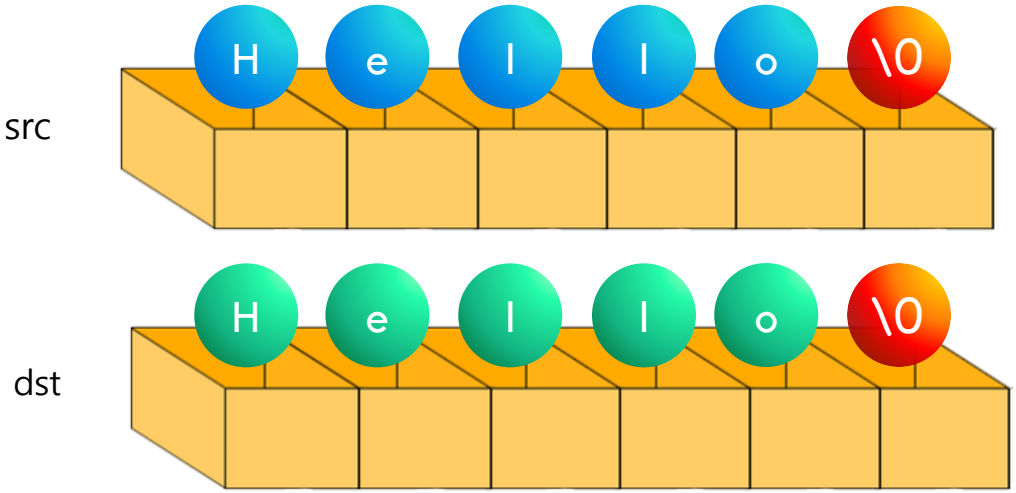
문자열 처리 라이브러리

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 복사

- 문자열 복사

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

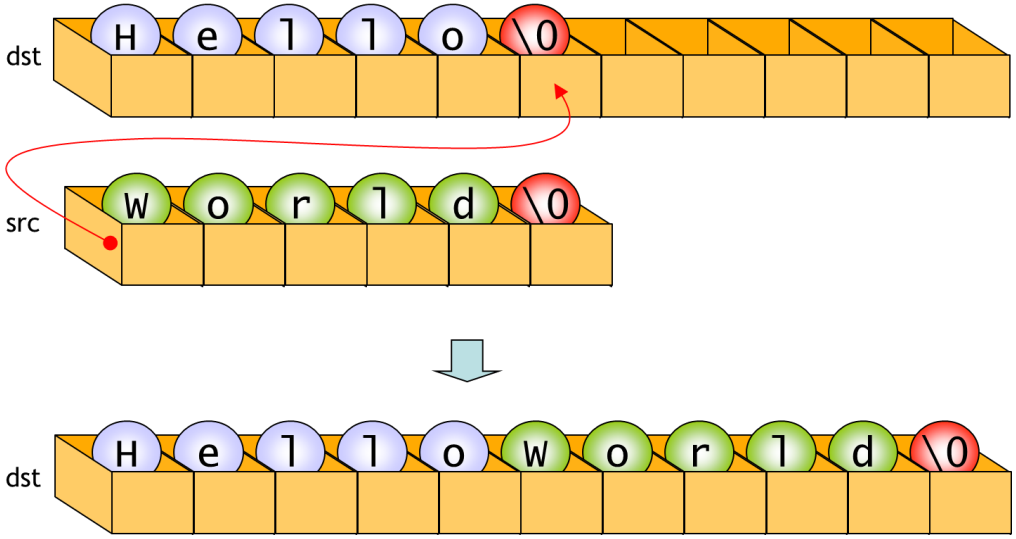


문자열 연결

Syntax: strcat()

예

```
char dst[12]= "Hello";
char src[6] = "World";
strcat(dst, src);        // dst가 "HelloWorld"가 된다.
```



예제

```c
#include<stdio.h>

int main()
{
    char s[80];
    strcpy(s, "Hello world from ");
    printf("s = %s₩n", s);
    strcat(s, "strcpy ");
    printf("s = %s₩n", s);
    strcat(s, "and ");
    printf("s = %s₩n", s);
    strcat(s, "strcat!");
    printf("s = %s₩n", s);
    return 0;
}
```

문자열 비교

Syntax: strcmp()

예

```
int result = strcmp("dog", "dog");      // 0이 반환된다.
```

strcmp()는 문자열 s1과 s2를 비교하여 사전적인(lexicographic) 순서에서 s1이 앞에 있으면 음수가 반환되고, 같으면 0이, 뒤에 있으면 양수가 반환된다.

반환값	s1과 s2의 관계
< 0	s1이 s2보다 앞에 있다.
0	s1 == s2
> 0	s1이 s2보다 뒤에 있다.

예제

```c
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>


int main(void)
{
    char s1[80];
    char s2[80];
    int result=0;

    printf("첫번째 단어 입력>>");
    scanf("%s", s1);
    printf("두번째 단어 입력>>");
    scanf("%s", s2);
    result = strcmp(s1, s2);
    printf("비교결과 %d₩n", result);

    if (result < 0) printf("a1<s2");
    else if (result == 0) printf("a1=s2");
    else printf("a1>s2");

    return 0;
}
```

문자 검색

Syntax: strchr()

예

```
char *p = strchr("dog", 'g');
```

'g' 문자의 주소를 반환한다.

문자 검색

```c
#include<stdio.h>

int main(void)
{
    char s[] = "language";
    char c = 'g';
    char* p;
    int loc;

    p = strchr(s, c);
    printf("g가 있는 주소는 %d₩n", p);
    printf("p주소에 있는 내용은 %c₩n", *p);

    //위치 계산하기
    loc = (int)(p - s);
    if (p != NULL) printf("%d 번째 있음₩n", loc);
    else printf("발견되지 않음");


    return 0;
}
```

문자열 검색

Syntax: strchr()

예

```
char *p = strstr("dog", "og");
```

strstr() 함수는 문자열 s안에서 부분 문자열(substring) sub를 검색하는 함수이다. 만약 부분 문자열이 발견되면 그 위치의 주소를 반환한다. 만약 부분 문자열을 찾지 못하면 NULL값이 반환된다.

문자열 검색

```c
#include<stdio.h>

int main(void)
{
    char s[] = "language";
    char c[] = "an";
    char* p;
    int loc;

    p = strstr(s, c);
    printf("g가 있는 주소는 %d₩n", p);
    printf("p주소에 있는 내용은 %c₩n", *p);

    //위치 계산하기
    loc = (int)(p - s);
    if (p != NULL) printf("%d 번째 있음₩n", loc);
    else printf("발견되지 않음");


    return 0;
}
```

문자열 토큰 분리

Syntax:

예

```
char s[] = "Hello World";
char delimit[] = " ";
char *p = strtok(s, delimit);
```

문자열을 스페이스문자를 사용하여 단어들로 분리한다.

문자열 토큰 분리

```c
#include<stdio.h>
int main(void)
{
    char s[] = "Man is immotal, because he has a soul";
    char seps[] = " ,";
    char* token;

    token = strtok(s, seps);
    while (token != NULL)
    {
        printf("%s₩n", token);
        token = strtok(NULL, seps);   // NULL을 주면 이전에 찾았던 구문자 뒤에서 부터 찾아라

    }
}
```

문제 1

문자열을 입력받아 알파벳 문자만 모두 대문자로 출력하는 프로그램을 작성하시오.
문자열의 길이는 100이하이다.

입력 예

```
1988-Seoul-Olympic!!!
```

출력 예

```
SEOULOLYMPIC
```

문제2

공백을 포함한 100글자 이하의 문자열을 입력받아 문장을 이루는 단어의 개수를 출력하는 프로그램을 작성하시오.

입력 예

```
My name is Kimchulsoo
```

출력 예

```
4
```

문제3

문자열을 입력 받아서 문자수만큼 오른쪽으로 한 바퀴 회전하여 출력하는 프로그램을 작성하시오. 문자열의 길이는 100이하이다.

입력 예

```
PROGRAM
```

출력 예

```
MPROGRA
AMPROGR
RAMPROG
GRAMPRO
OGRAMPR
ROGRAMP
PROGRAM
```