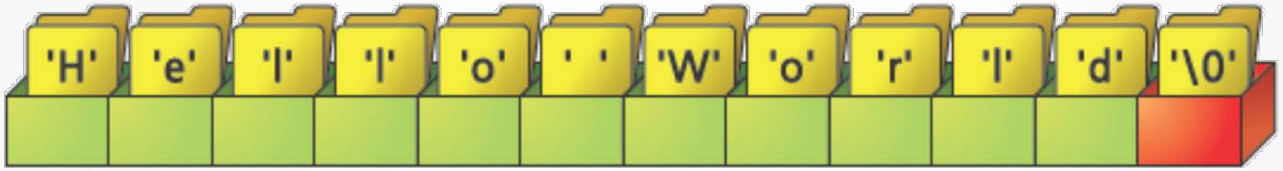
목차

문장을 어떻게 구현할까?

- **Hello World**
  - 문장은 문자들의 연속된 모임, 집합.
  - 연속된 모임을 구현하려면? 배열!

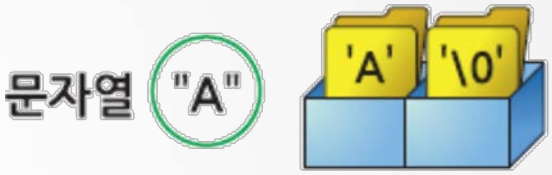


크기가 12인 배열

문자열

문자열이란?

- 연속된 문자들의 모임
- 문자열은 큰 따옴표(“ “)로 표현
- 문자열의 끝에는 널 문자(‘₩0’)를 함께 저장함



문자열 상수와 변수

- 문자열 상수 : 값이 변경되지 않는 문자열
  - “A”나 “Hello World” 등
  - 문자열 리터럴이라고도 함
- 문자열 변수: 프로그램 수행 중에 변경될 수 있는 문자열
  - 문자(char)의 배열

문자 배열의 선언과 사용

문자 배열의 선언

- 문자 배열의 크기는 "저장할 문자열의 길이 + 1"로 할당해야 함

```
char str[10];
```
길이가 9인 문자열을 저장하기 위한 문자 배열을 선언한다.

문자 배열의 사용

- 문자 배열도 인덱스를 이용해서 개별 문자에 접근할 수 있음

```
char str[4];
int i;
str[0] = 'a';
str[1] = 'b';
str[2] = 'c';
str[3] = '\0';
for(i = 0 ; i < 4 ; i++)
    printf("%c", str[i]);
```
문자 배열의 각 원소를 변경한다.

문자 배열의 각 원소를 출력한다.

문자 배열의 초기화

- 문자열 상수를 이용해서 초기화 함

```
char str[4] = "abc";
```

문자열 상수를 이용한 문자 배열의 초기화

- 문자 배열의 크기보다 큰 문자열로 초기화할 수 없음

```
char str[4] = "ab";
char str[4] = "abcde";
```

str 배열은 'a', 'b', '\0', '\0'으로 초기화된다.

컴파일 경고가 발생한다.

- 초기값을 지정할 때는 문자 배열의 크기를 생략할 수 있음

```
char str[ ] = "Hello";
```

str은 크기가 6인 배열로 할당된다.

문자 배열의 초기화 예


```
char str1[4] = "abc";
char str2[10] = "12345";
char str3[ ] = "Hello World";
```

문자 배열의 출력

- 문자 배열 전체에 직접 다른 문자열을 대입할 수 없음

```
char str[10] = "abcdefg";
str = "Hello";
```

컴파일 에러

- 문자 배열에 저장된 각 문자를 하나씩 변경하는 것은 가능함

```
char str[10] = "abcdefg";
str[0] = 'H';
str[1] = 'e';
str[2] = 'l';
str[3] = 'l';
str[4] = 'o';
str[5] = '\0';
```

문자 배열의 각 문자를 하나씩 변경한다.

문자열 처리 함수

- 문자열을 복사하거나, 문자열의 길이를 구하거나, 문자열을 비교할 때 이용

- 문자열 처리 함수를 사용하려면 <string.h>를 포함해야 함

문자열 복사 - strcpy(string copy) 함수

형식

```
strcpy(dest, src);
```

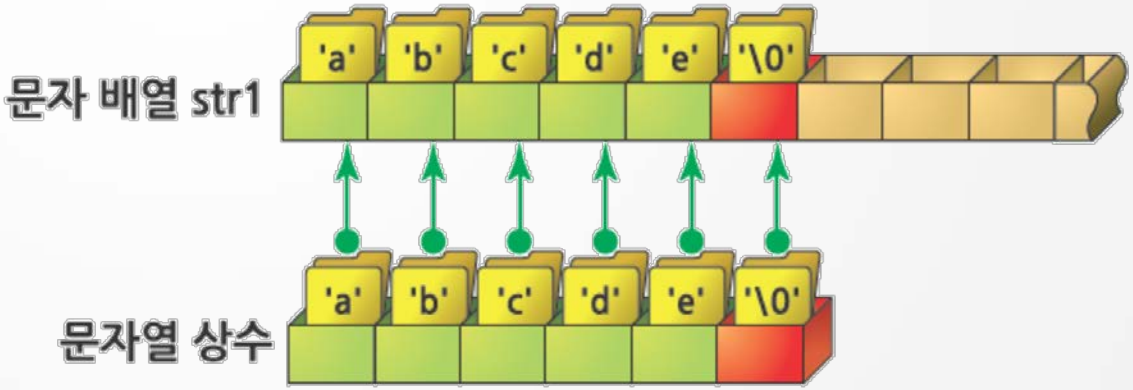
예제

```
char    str1[20], str2 [20] ;

strcpy ( str1,  "abcde"  ) ;
strcpy ( str2,  str1     ) ;
```

dest src

```
strcpy(str1, "abcde");
```



널 문자를 만날 때까지 1:1로 복사합니다.

문자열 길이 - strlen(string length) 함수


```
char str1[ ] = "abcde";
int len;
printf("문자열의 길이 : %d\n", strlen(str1));
printf("문자열의 길이 : %d\n", strlen("Hello World"));
```

문자열의 길이는 5이다.

문자열의 길이는 11이다.

```
len = strlen(str1);
```

문자열 비교 - strcmp(string compare) 함수

- 문자열을 비교할 때는 == 연산자를 사용하지 않음!!!!!

```
if( str1 == "abcde" )
    printf("두 문자열이 같습니다.\n");
else
    printf("두 문자열이 다릅니다.\n");
```

문자 배열의 주소와 문자열 상수의 주소를 비교한다.

- 문자열의 내용을 비교하려면 strcmp 함수를 사용해야 함

```
if( strcmp(str1, "abcde") == 0 )
    printf("두 문자열이 같습니다.\n");
else
    printf("두 문자열이 다릅니다.\n");
```

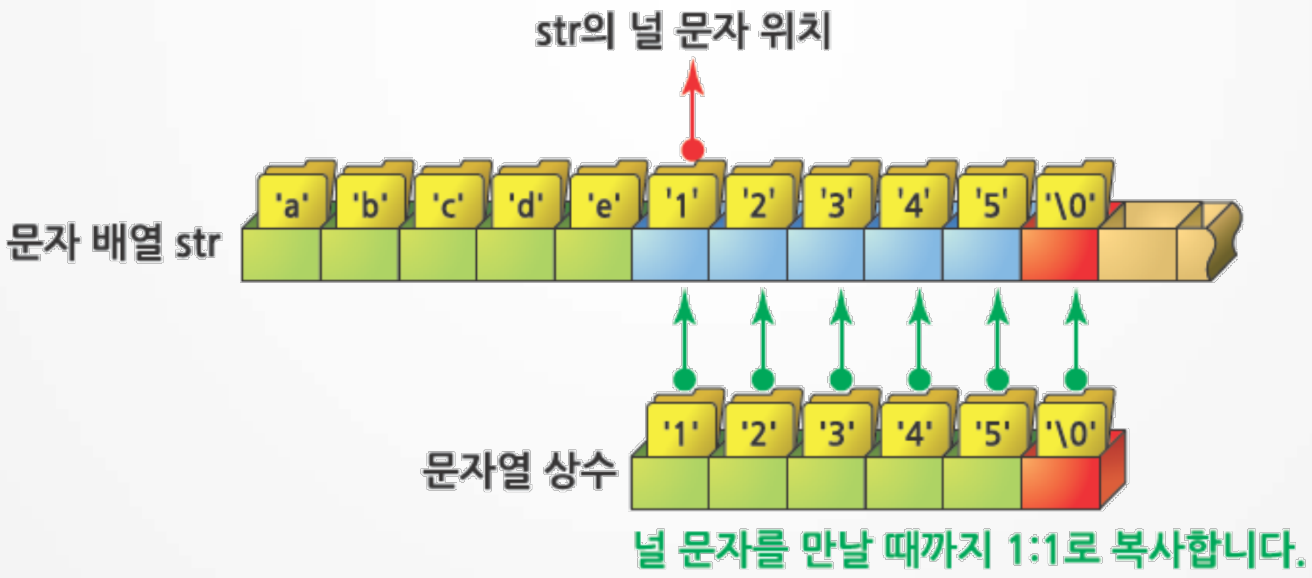
문자열의 내용을 비교한다.

문자열 연결 – strcat(string catnate) 함수


```
char str[20] = "abcde";
strcat(str, "12345");
printf("str = %s\n", str);
```

str은 "abcde12345"가 된다.

```
strcat(str, "12345");
```



문자열 입출력

- scanf와 printf 함수를 이용한 문자열 입출력
  - 형식 문자열로 %s를 사용
  - 입력된 내용 중 항상 공백 문자(„ ‟, „₩t‟, „₩n‟ 등)까지만을 읽음

문자열의 입출력 예

```
01: /* Ex07_12.c */
02: #include <stdio.h>
03: #include <string.h>
04:
05: int main(void)
06: {
07:         char name[20];
08:
09:         printf("이름을 입력하세요 : ");
10:         scanf("%s", name);
11:         printf("%s씨, 안녕하세요?₩n", name);
12:
13:         return 0;
14: }
```

문자 배열의 선언

문자열의 입력

문자열의 출력

실 행 결 과

```
이름을 입력하세요 : 김모모
김모모씨, 안녕하세요?
```

실 행 결 과

```
이름을 입력하세요 : 김 모모
김씨, 안녕하세요?
```

빈칸을 포함한 문자열을 입력하면 빈칸까지만 입력으로 처리된다.

gets 함수 & puts 함수

gets

- 빈칸을 포함한 한 줄의 문자열을 입력받는 데 사용

puts

- 한 줄의 문자열을 출력하는 데 사용
- 출력 후에 자동으로 줄 바꿈 문자(₩n)를 출력

```
char name[20];
char msg[40];
printf("이름을 입력하세요 : ");
gets(name);
sprintf(msg, "%s씨, 안녕하세요?");   ← 형식 문자열을 이용해서 문자 배열의 내용을 변경한다.
puts(msg);   ← 한 줄의 문자열을 출력한다.
```

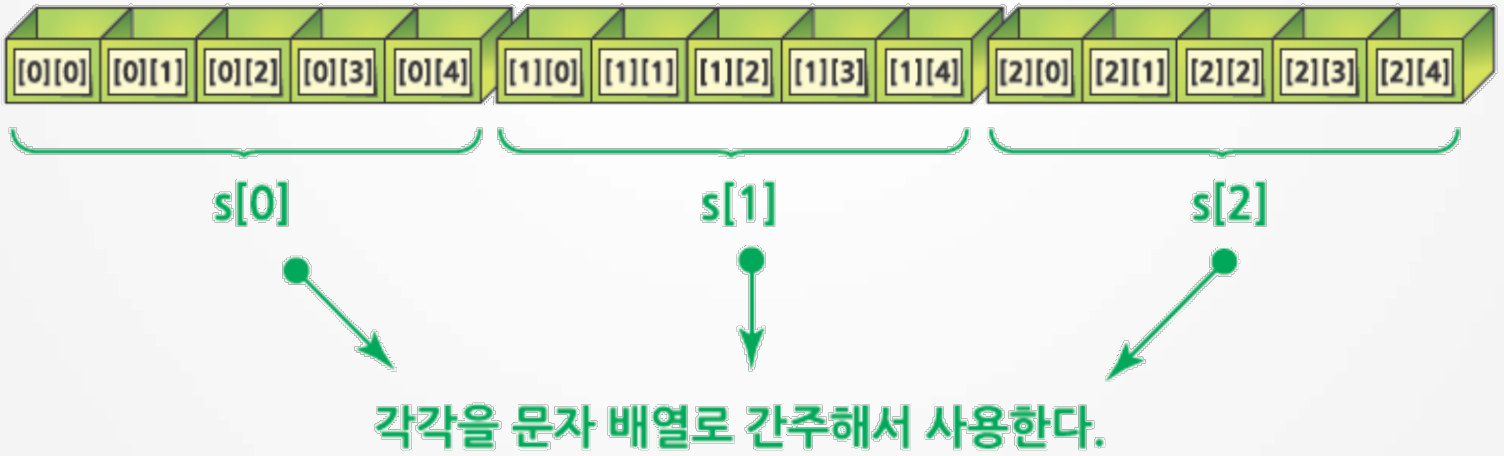
문자열 배열의 (문자 배열 X)

- 문자열을 여러 개 저장하려면 문자 배열을 이차원 배열로 선언해야 함

- 문자열 배열의 선언
  - 제2크기는 저장할 문자열의 길이+1로 지정
  - 제1크기는 필요한 문자열의 개수로 지정

```
char s [3][5];
```

최대 길이가 4인 문자열이 3개



문자열 배열의 사용(1/2)

- 인덱스를 하나만 사용하면 배열에 보관된 문자열에 접근할 수 있음

```
char s[3][5];
int i;
strcpy(s[0], "ab");
strcpy(s[1], "cdef");
strcpy(s[2], "ghi");
for(i = 0 ; i < 3 ; i++)
    printf("문자열의 길이 : %d\n", strlen(s[i]));
```

두 번째 인덱스 없이 첫 번째 인덱스만 지정해서 사용한다.

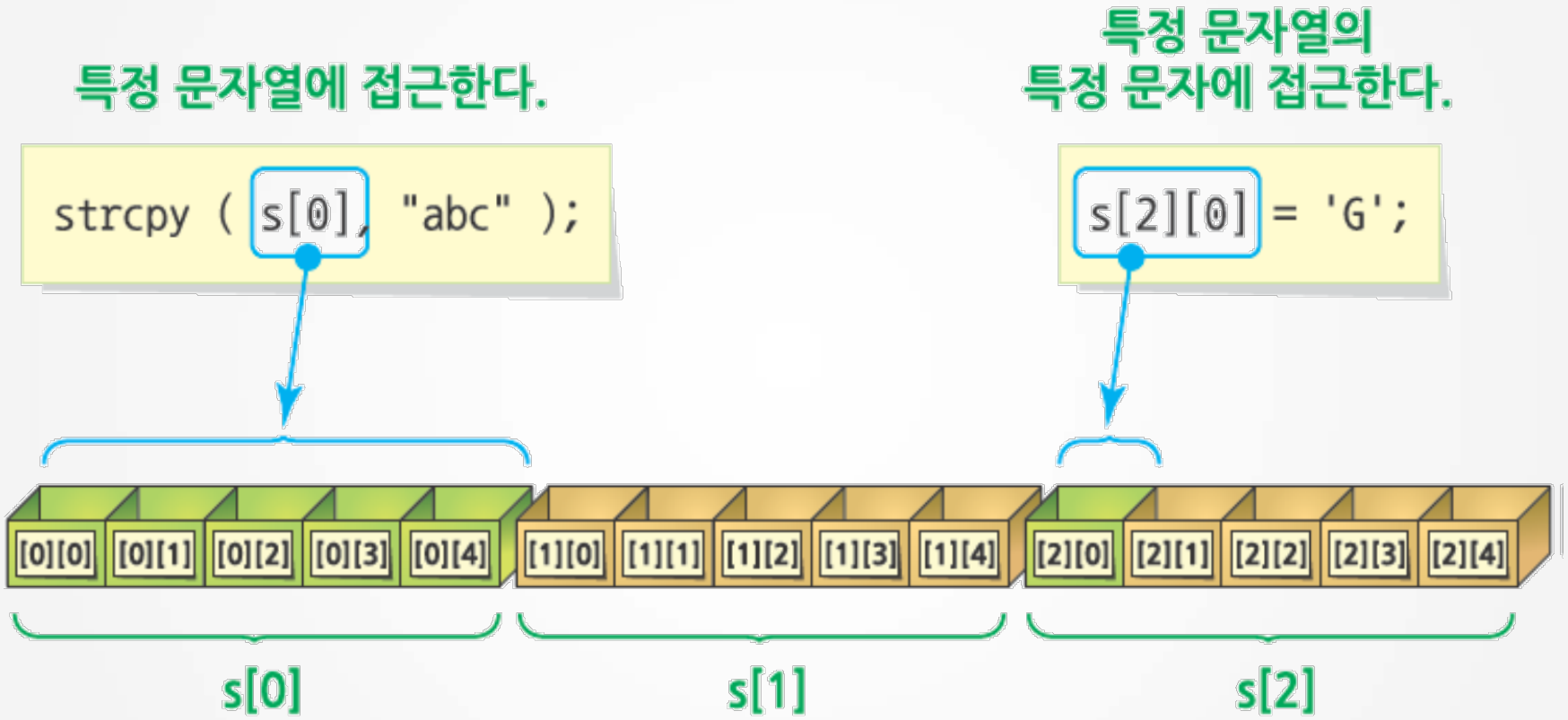
- 인덱스를 두 개 사용하면 특정 문자열의 특정 위치에 있는 문자에 접근

```
char s[3][5];
strcpy(s[0], "abc");
s[2][0] = 'G';
```

인덱스를 하나만 사용하면 문자열에 접근한다.

인덱스를 두 개 사용하면 문자열의 특정 문자에 접근한다.

문자열 배열의 사용(2/2)


특정 문자열에 접근한다.
strcpy (s[0], "abc");
특정 문자열의
특정 문자에 접근한다.
s[2][0] = 'G';
[0][0]
[0][1]
[0][2]
[0][3]
[0][4]
[1][0]
[1][1]
[1][2]
[1][3]
[1][4]
[2][0]
[2][1]
[2][2]
[2][3]
[2][4]
s[0]
s[1]
s[2]

문자열 배열의 초기화

- 문자열 배열을 초기화하려면 { } 안에 문자열 상수들을 나열

```
char weekdays[7][4] = {          문자열 상수로 문자열 배열을 초기화한다.
    "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"
};
```

- 초기값을 지정할 때는 배열의 제1크기를 생략할 수 있음

```
char fruits[ ][20] = {           char fruits[5][20]으로 할당된다.
    "apple", "banana", "melon", "kiwi", "strawberry"
};
```

실습 1. 문자열의 입출력

- 사용자로부터 공백이 포함된 문자열을 입력 받은 다음에 공백을 제거한 문자열을 출력함.
  - (입력) I Love You.
  - (출력) ILoveYou.

- 파일명: strInOut.c

실습 2. 문자열의 배열

- 아래의 전화번호를 검색하는 프로그램을 구현하시오.

이름	전화번호
유시진	010-0000-0001
강모연	010-0000-0002
서대영	010-0000-0003
윤명주	010-0000-0004

- 파일명: phonebook.c