문자열 다루기

- Chapter 09 -

학습목차

문자열 사용하기

문자열 사용하기

- C언어에서는 자바의 String타입과 같은 문자열을 저장하는 자료형을 지원하지 않음
  - char는 문자 데이터를 저장
  - 문자열은 여러 문자를 나열한 형태의 문자 집합
- 아래 예제는 문자 저장공간에 문자열을 저장할 경우 발생하는 오류
  - 운영체제나 개발 IDE마다 에러 메시지가 다름

```
#include <stdio.h>

int main()
{
    char s1 = "Hello";    // "Hello"는 문자열, 문자열은 " "로 둘러쌈

    printf("%s", s1);     // 실행 에러

    return 0;
}
```

```
0xC0000005: 0xFFFFFFE8 위치를 읽는 동안 액세스 위반이 발생했습니다..
```

문자열 사용하기

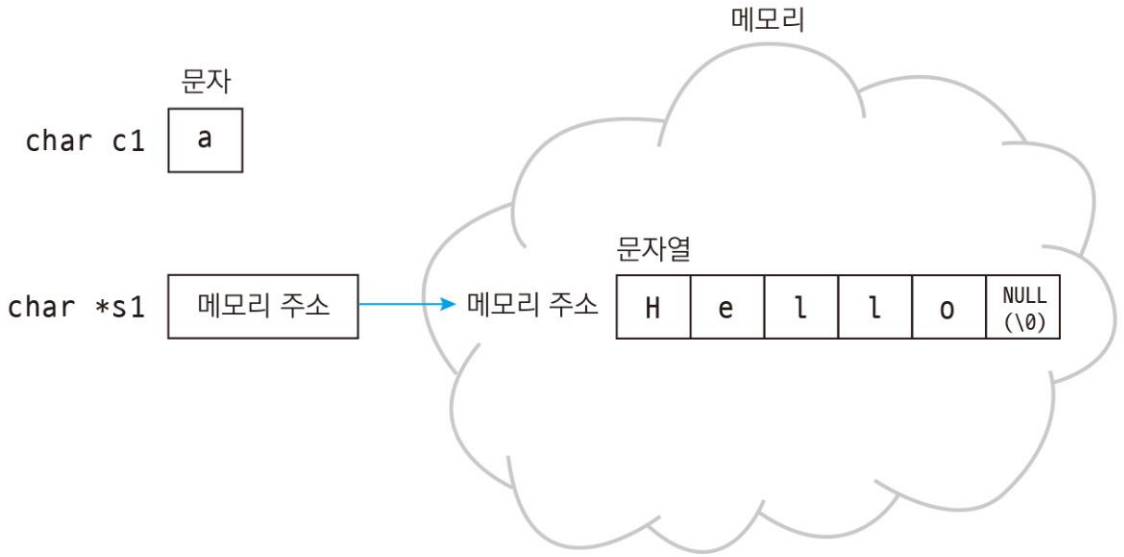
- 문자와 문자열 포인터
  - 문자열은 char포인터 형식으로 사용
  - char *변수이름 = "문자열";

```
#include <stdio.h>

int main()
{
    char c1 = 'a';          // 변수에 문자 'a' 저장
    char* s1 = "Hello";     // 포인터에 문자열 "Hello"의 주소 저장

    printf("%c\n", c1);     // a: %c로 문자 출력
    printf("%s\n", s1);     // Hello: %s로 문자열 출력

    return 0;
}
```

```
a
Hello
```

문자열 사용하기

- 문자와 문자열 포인터
  - 문자는 'a'처럼 글자가 하나만 있는 상태이고 문자열은 "Hello"처럼 글자 여러 개가 계속 이어진 상태
  - 문자는 1바이트 크기의 char형 변수에 저장
  - 문자열은 크기가 1바이트를 넘어서므로 char에 저장할 수 없음
  - 문자열은 변수에 직접 저장하지 않고 포인터를 이용해서 저장
  - 문자와 문자열의 저장 방식



문자열 사용하기

- 문자와 문자열 포인터 알아보기
  - 문자열 리터럴이 있는 메모리 주소는 읽기 전용
  - 다른 문자열을 덮어쓸 수 없음

변수 안에 'a'가 그대로 저장됨

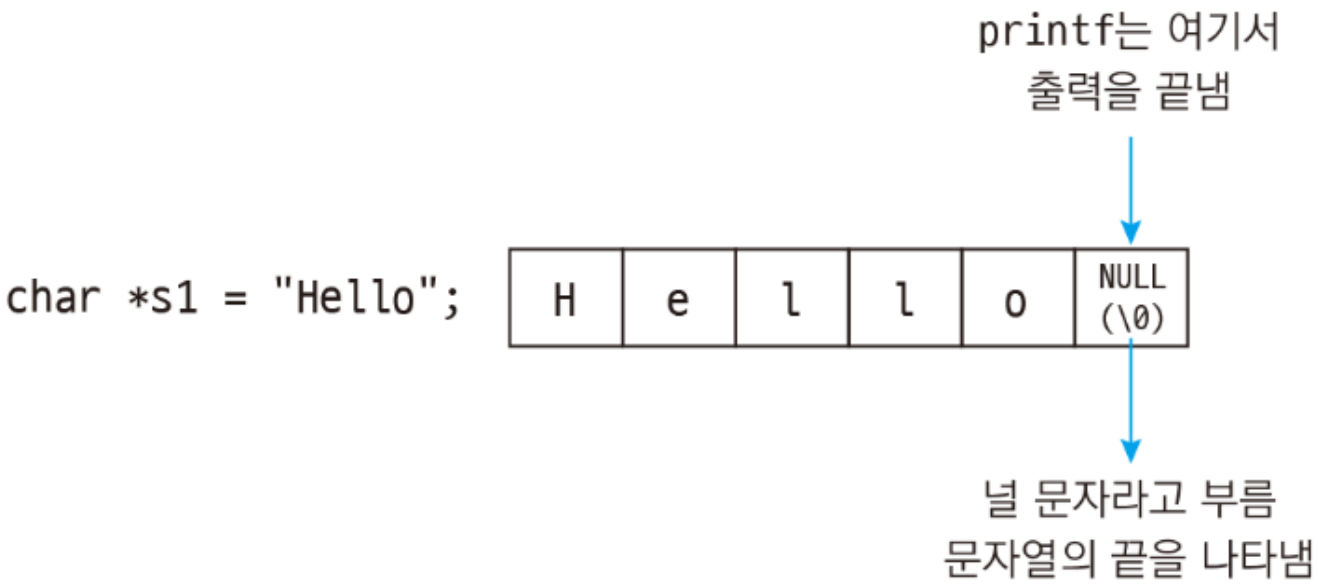
```
char c1 = 'a';
```

```
char *s1 = "Hello";
```

문자열 리터럴이 있는 곳의 메모리 주소만 저장

문자열 사용하기

- 문자와 문자열 포인터 알아보기
  - 문자열은 마지막에 항상 널 문자(NULL)가 붙음
  - NULL은 문자열의 끝을 나타냄
  - 문자열과 널 문자



문자열 사용하기

▶ 문자열 포인터에서 인덱스로 문자에 접근하기

```
#include <stdio.h>

int main()
{
    char* s1 = "Hello";        // 포인터에 문자열 Hello의 주소 저장

    printf("%c\n", s1[1]);     // e: 인덱스 1(두 번째)의 문자 출력
    printf("%c\n", s1[4]);     // o: 인덱스 4(다섯 번째)의 문자 출력
    printf("%c\n", s1[5]);     // 문자열 맨 뒤의 NULL(\0) 출력. NULL은 화면에 표시되지 않음

    return 0;
}
```

```
e
o
```

문자열 사용하기

- 문자열 포인터에서 인덱스로 문자에 접근하기
  - 문자열 리터럴이 있는 메모리주소는 읽기 전용이므로 오류 발생

```
#include <stdio.h>

int main(){
    char* s1 = "Hello";     // 포인터에 문자열 Hello의 주소 저장
                            // Hello가 있는 메모리 주소는 읽기 전용

    s1[0] = 'A';            // 문자열 포인터의 인덱스 0에 문자 A를 할당
                            // 실행 에러

    printf("%c\n", s1[0]);

    return 0;
}
```

문자열 사용하기

배열 형태로 문자열 선언하기

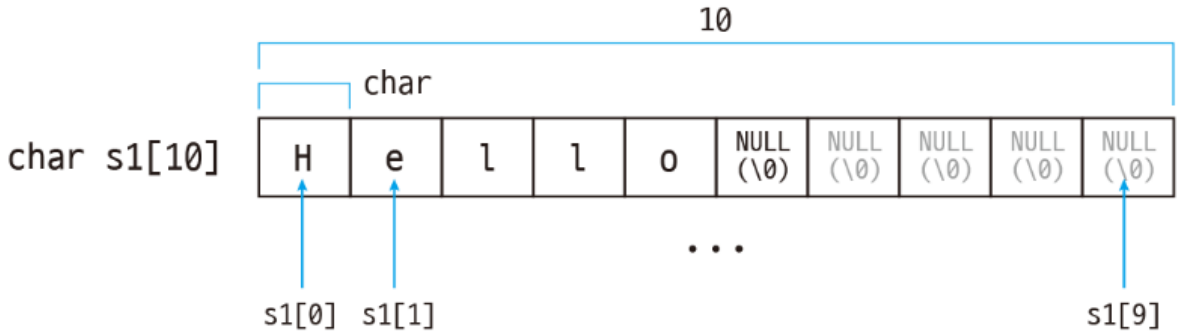
char 배열이름[크기] = "문자열";

```
#include <stdio.h>

int main(){
    char s1[10] = "Hello";  // 크기가 10인 char형 배열을 선언하고 문자열 할당

    printf("%s\n", s1);     // Hello: %s로 문자열 출력

    return 0;
}
```

```
Hello
```

문자열 사용하기

- 배열 형태로 문자열 선언하기
  - 문자열을 배열에 저장하기



문자열 사용하기

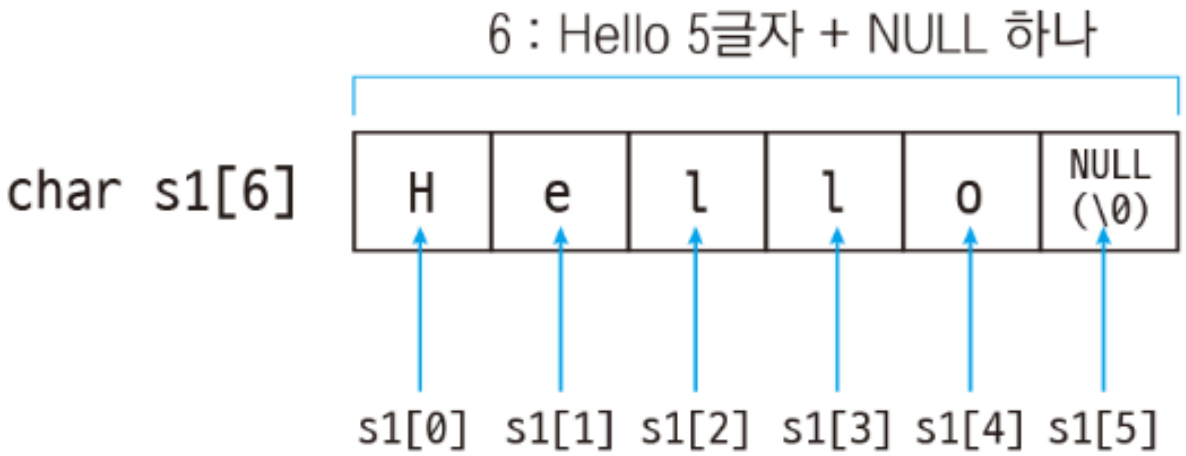
배열 형태로 문자열 선언하기

```
#include <stdio.h>

int main(){
    char s1[10];     // 크기가 10인 char형 배열 선언

    s1 = "Hello";    // 이미 선언된 배열에 문자열을 할당하면 컴파일 에러 발생

    printf("%s\n", s1);    // Hello: %s로 문자열 출력

    return 0;
}
```

```
string_array_assign_error.c(7): error C2106: '=' : 왼쪽 피연산자는 l-value이어야 합니다.
```

문자열 사용하기

- 배열 형태로 문자열 선언하기
  - 문자열을 배열에 저장할 때 배열의 최소 크기



문자열 사용하기

- 배열 형태로 문자열 선언하기
  - char 배열이름[] = "문자열";

```
#include <stdio.h>

int main(){
    char s1[] = "Hello";     // 문자열을 할당할 때 배열의 크기를 생략하는 방법

    printf("%s\n", s1);      // Hello: %s로 문자열 출력

    return 0;
}
```

```
Hello
```

문자열 사용하기

▶ 배열 형태의 문자열에서 인덱스로 문자에 접근하기

```
int main(){
    char s1[10] = "Hello";    // 크기가 10인 char형 배열을 선언하고 문자열 할당

    printf("%c\n", s1[1]);    // e: 인덱스 1(두 번째)의 문자 출력
    printf("%c\n", s1[4]);    // o: 인덱스 4(다섯 번째)의 문자 출력
    printf("%c\n", s1[5]);    // 문자열 맨 뒤의 NULL(\0) 출력. NULL은 화면에 표시되지 않음

    return 0;
}
```

```
실행 결과
e
o
```

문자열 사용하기

▶ 배열 형태의 문자열에서 인덱스로 문자에 접근하기

```
#include <stdio.h>

int main(){
    char s1[10] = "Hello";     // 크기가 10인 char형 배열을 선언하고 문자열 할당
                               // 배열에 문자열이 복사됨

    s1[0] = 'A';               // 문자 배열의 인덱스 0에 문자 A를 할당

    printf("%s\n", s1);        // Aello: 바뀐 문자열이 출력됨

    return 0;
}
```

```
Aello
```

실습문제 01

문자열 만들기

다음 소스 코드를 완성하여 "Beethoven 9th Symphony"가 출력되게 만드세요.

```
#include <stdio.h>

int main(){

   ________________ = "Beethoven 9th Symphony";

   printf("%s\n", s1);

   return 0;
}
```

```
Beethoven 9th Symphony
```

실습문제 02

문자열 요소 출력

다음 소스 코드를 완성하여 9가 출력되게 만드시요.

```
#include <stdio.h>

int main()
{
    char s1[30] = "Beethoven 9th Symphony";

    printf("%c\n", _________);

    return 0;
}
```

```
9
```

입력 값을 문자열에 저장하기

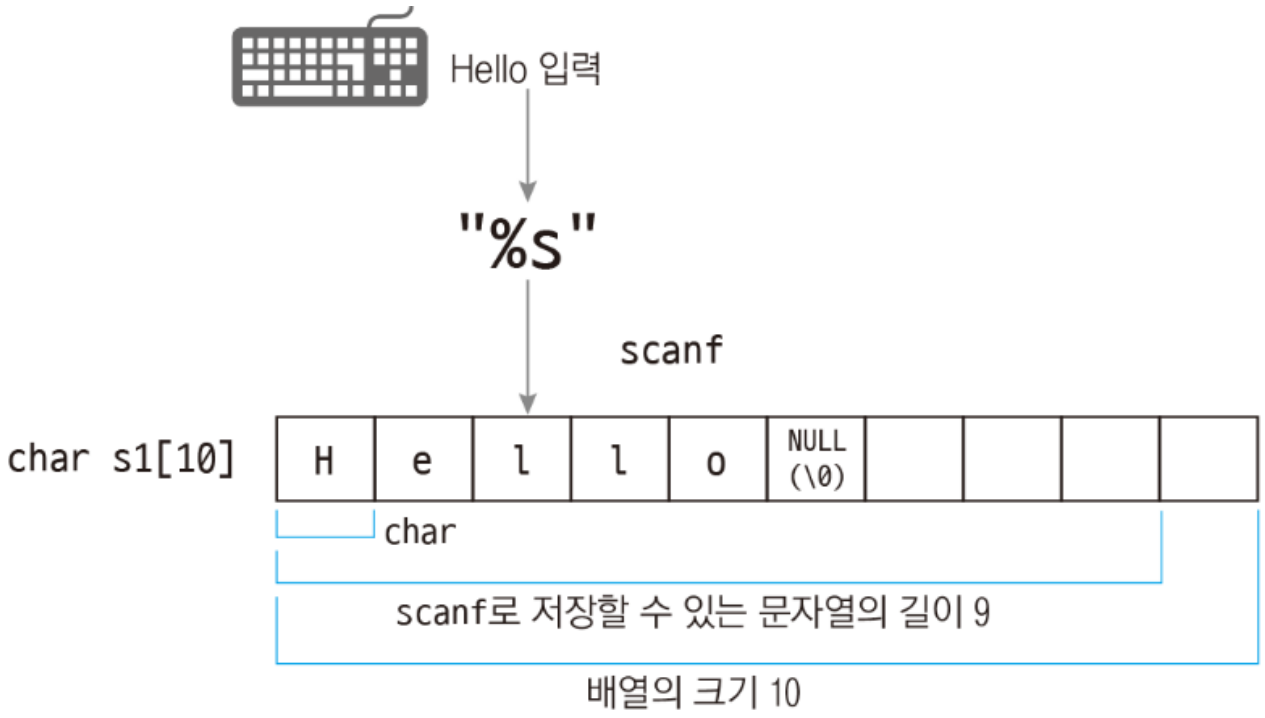
- 입력 값을 배열 형태의 문자열에 저장하기
  - scanf("%s", 배열);
    - int scanf(char const *const _Format, ...);
    - 성공하면 가져온 값의 개수를 반환, 실패하면 EOF(-1)를 반환

```
#define _CRT_SECURE_NO_WARNINGS // scanf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>

int main()
{
    char s1[10];    // 크기가 10인 char형 배열을 선언

    printf("문자열을 입력하세요: ");
    scanf("%s", s1);     // 표준 입력을 받아서 배열 형태의 문자열에 저장

    printf("%s\n", s1);  // 문자열의 내용을 출력

    return 0;
}
```

```
문자열을 입력하세요: Hello (입력)
Hello
```

입력 값을 문자열에 저장하기

- 입력 값을 배열 형태의 문자열에 저장하기
  - 입력 값을 배열 형태의 문자열에 저장하기



입력 값을 문자열에 저장하기

▶ 입력 값을 문자열 포인터에 저장하기

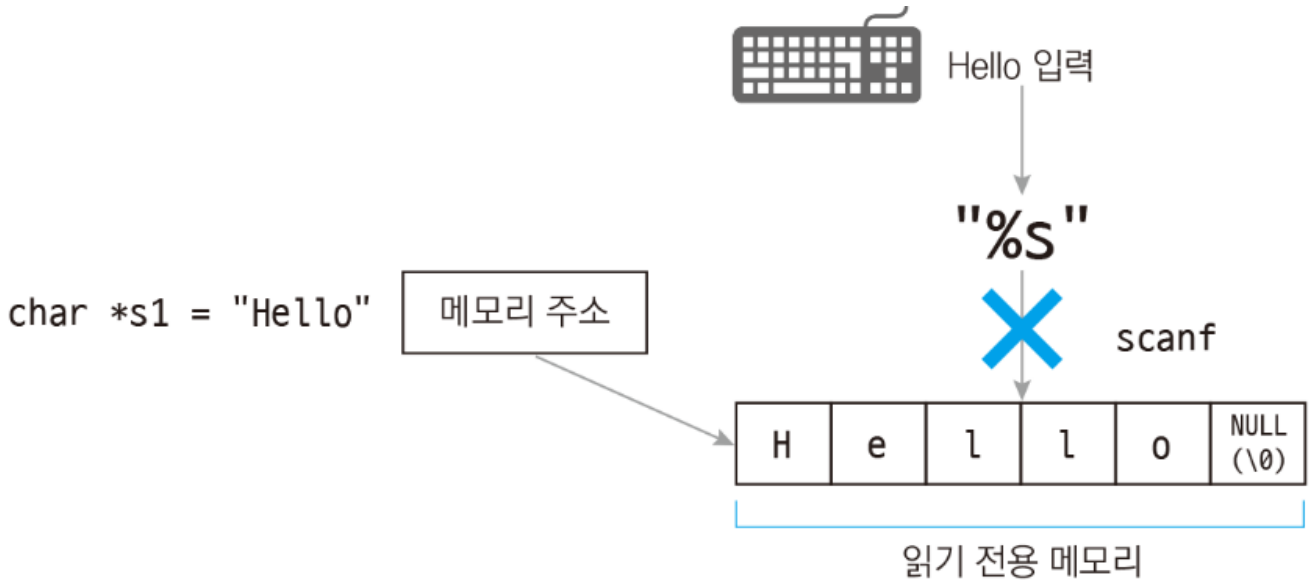
```
#define _CRT_SECURE_NO_WARNINGS    // scanf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>

int main()
{
    char *s1 = "Hello";    // 문자열 포인터를 선언하고 문자열 할당

    printf("문자열을 입력하세요: ");
    scanf("%s", s1);    // 실행 에러

    printf("%s\n", s1);

    return 0;
}
```

```
문자열을 입력하세요: Hello (입력)
```

```
0xC0000005: 0x013A585D 위치를 기록하는 동안 액세스 위반이 발생했습니다.
```

입력 값을 문자열에 저장하기

- 입력 값을 문자열 포인터에 저장하기
  - 읽기 전용 메모리에는 scanf의 입력 값을 저장할 수 없음



입력 값을 문자열에 저장하기

입력 값을 문자열 포인터에 저장하기

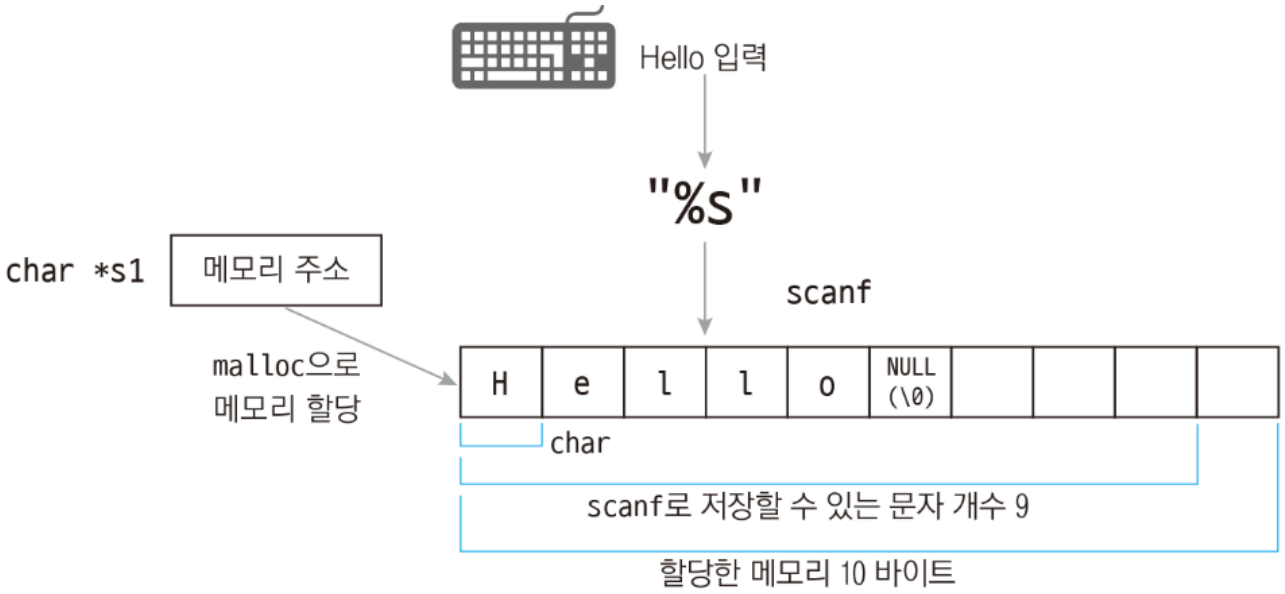
scanf("%s", 문자열포인터);

```
#define _CRT_SECURE_NO_WARNINGS     // scanf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    char *s1 = malloc(sizeof(char) * 10);    // char 10개 크기만큼 동적 메모리 할당

    printf("문자열을 입력하세요: ");
    scanf("%s", s1);      // 표준 입력을 받아서 메모리가 할당된 문자열 포인터에 저장

    printf("%s\n", s1);   // 문자열의 내용을 출력

    free(s1);    // 동적 메모리 해제

    return 0;
}
```

```
문자열을 입력하세요: Hello (입력)
Hello
```

입력 값을 문자열에 저장하기

- 입력 값을 문자열 포인터에 저장하기
  - 입력 값을 문자열 포인터에 저장하기



입력 값을 문자열에 저장하기

- 문자열을 여러 개 입력받기
  - scanf("%s %s ...", 배열1, 배열2, ...);

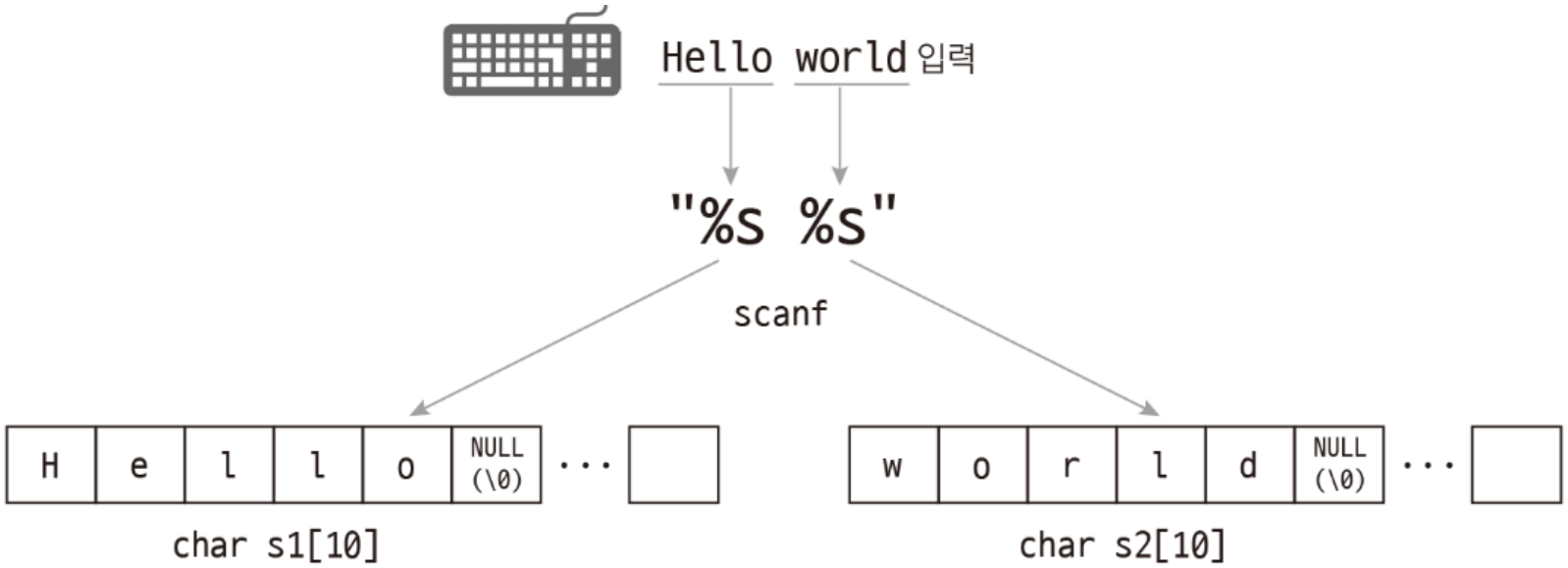
    scanf("%s %s ...", 문자열포인터1, 문자열포인터2, ...);

```
#define _CRT_SECURE_NO_WARNINGS    // scanf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>

int main()
{
    char s1[10];    // 크기가 10인 char형 배열을 선언
    char s2[10];    // 크기가 10인 char형 배열을 선언

    printf("문자열을 두 개 입력하세요: ");
    scanf("%s %s", s1, s2);    // 표준 입력에서 공백으로 구분된 문자열 두 개를 입력받음

    printf("%s\n", s1);    // s1의 내용을 출력
    printf("%s\n", s2);    // s2의 내용을 출력

    return 0;
}
```

```
문자열을 두 개 입력하세요: Hello world (입력)
Hello
world
```

입력 값을 문자열에 저장하기

- 문자열을 여러 개 입력받기
  - scanf로 문자열 여러 개 입력받기



실습문제 03

입력받은 문자열을 배열에 저장하기

다음 소스 코드를 완성하여 표준 입력으로 입력받은 문자열이 그대로 출력되게 만드시오.

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

int main()
{
    char s1[10];

    printf("문자열을 입력하세요: ");
    ________________________

    printf("%s\n", s1);

    return 0;
}
```

```
문자열을 입력하세요: Hello (입력)
Hello
```

실습문제 04

입력받은 문자열을 동적 메모리에 저장하기

다음 소스 코드를 완성하여 표준 입력으로 입력받은 문자열이 그대로 출력되게 만드시오.

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

int main()
{
    ①_____________________

    printf("문자열을 입력하세요: ");
    ②_____________________

    printf("%s\n", s1);

    free(s1);

    return 0;
}
```

```
문자열을 입력하세요: Hello (입력)
Hello
```

실습문제 05

문자열 세 개 입력받기

다음 소스 코드를 완성하여 표준 입력으로 문자열 세 개를 입력받은 뒤 각각 출력되게 만드시오.

```
int main(){
    char *s1 = malloc(sizeof(char) * 10);
    char *s2 = malloc(sizeof(char) * 10);
    char *s3 = malloc(sizeof(char) * 10);

    printf("문자열을 세 개 입력하세요: ");
   ____________________________________

    printf("%s\n", s1);
    printf("%s\n", s2);
    printf("%s\n", s3);
    free(s1);
    free(s2);
    free(s3);
    return 0;
}
```

```
문자열을 세 개 입력하세요: Beethoven 9th symphony(입력)
Beethoven
9th
symphony
```

문자열의 길이를 구하고 비교하기

- 문자열 길이 구하기
  - strlen(문자열포인터);
  - strlen(문자배열);
    - size_t strlen(const *_Str);
    - 문자열의 길이를 반환

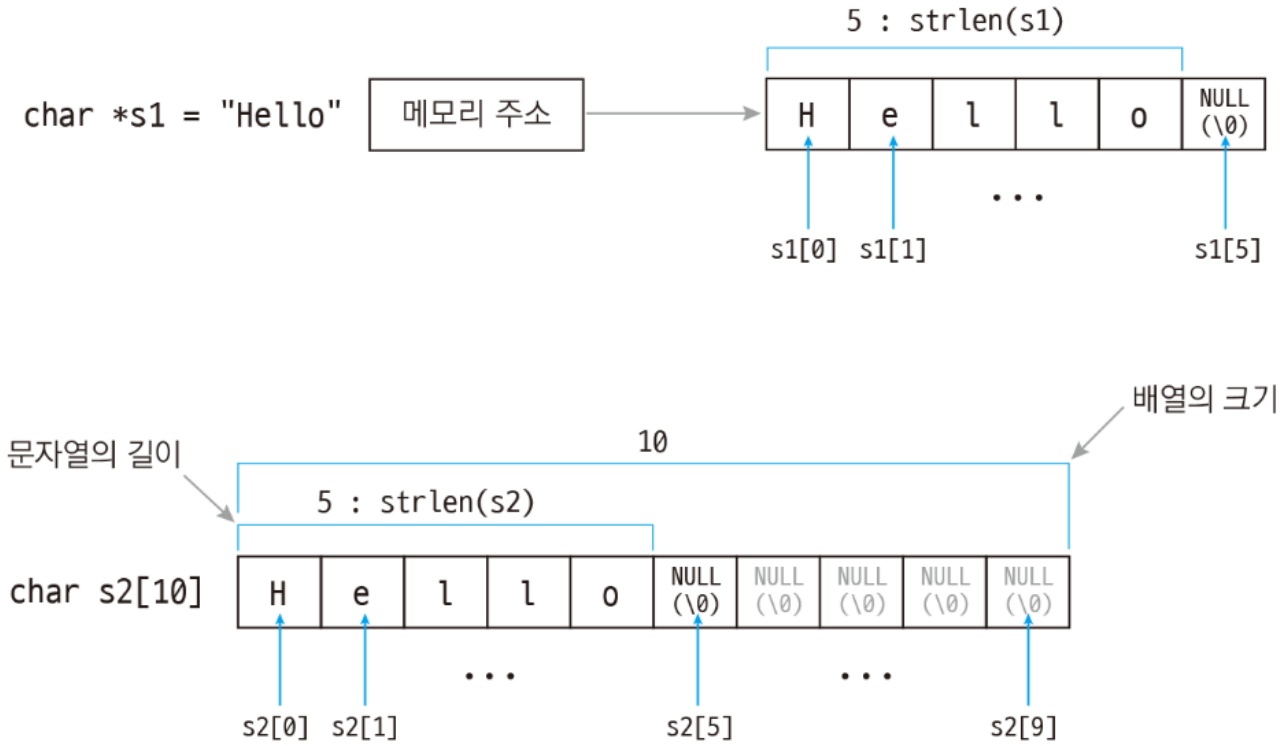
```
#include <stdio.h>
#include <string.h>    // strlen 함수가 선언된 헤더 파일

int main()
{
    char *s1 = "Hello";        // 포인터에 문자열 Hello의 주소 저장
    char s2[10] = "Hello";     // 크기가 10인 char형 배열을 선언하고 문자열 할당

    printf("%d\n", strlen(s1));    // 5: strlen 함수로 문자열의 길이를 구함
    printf("%d\n", strlen(s2));    // 5: strlen 함수로 문자열의 길이를 구함

    return 0;
}
```

```
5
5
```

문자열의 길이를 구하고 비교하기

문자열 길이 구하기

strlen 함수로 문자열 길이 구하기



문자열의 길이를 구하고 비교하기

- 문자열 비교하기
  - strcmp(문자열1, 문자열2);
    - int strcmp(const *_Str1, char const *_Str2);
    - 문자열 비교 결과를 반환
      - 같으면 0, 다른면 1 또는 -1

```
#include <stdio.h>
#include <string.h>    // strcmp 함수가 선언된 헤더 파일
int main(){
    char s1[10] = "Hello";
    char *s2 = "Hello";

    int ret = strcmp(s1, s2);    // 두 문자열이 같은지 문자열 비교

    printf("%d\n", ret);         // 0: 두 문자열이 같으면 0

    return 0;
}
```

```
0
```

문자열의 길이를 구하고 비교하기

문자열 비교하기

strcmp(s1, s2);로 비교

-1: ASCII 코드 기준으로 문자열2(s2)가 클 때

0: ASCII 코드 기준으로 두 문자열이 같을 때

1: ASCII 코드 기준으로 문자열1(s1)이 클 때

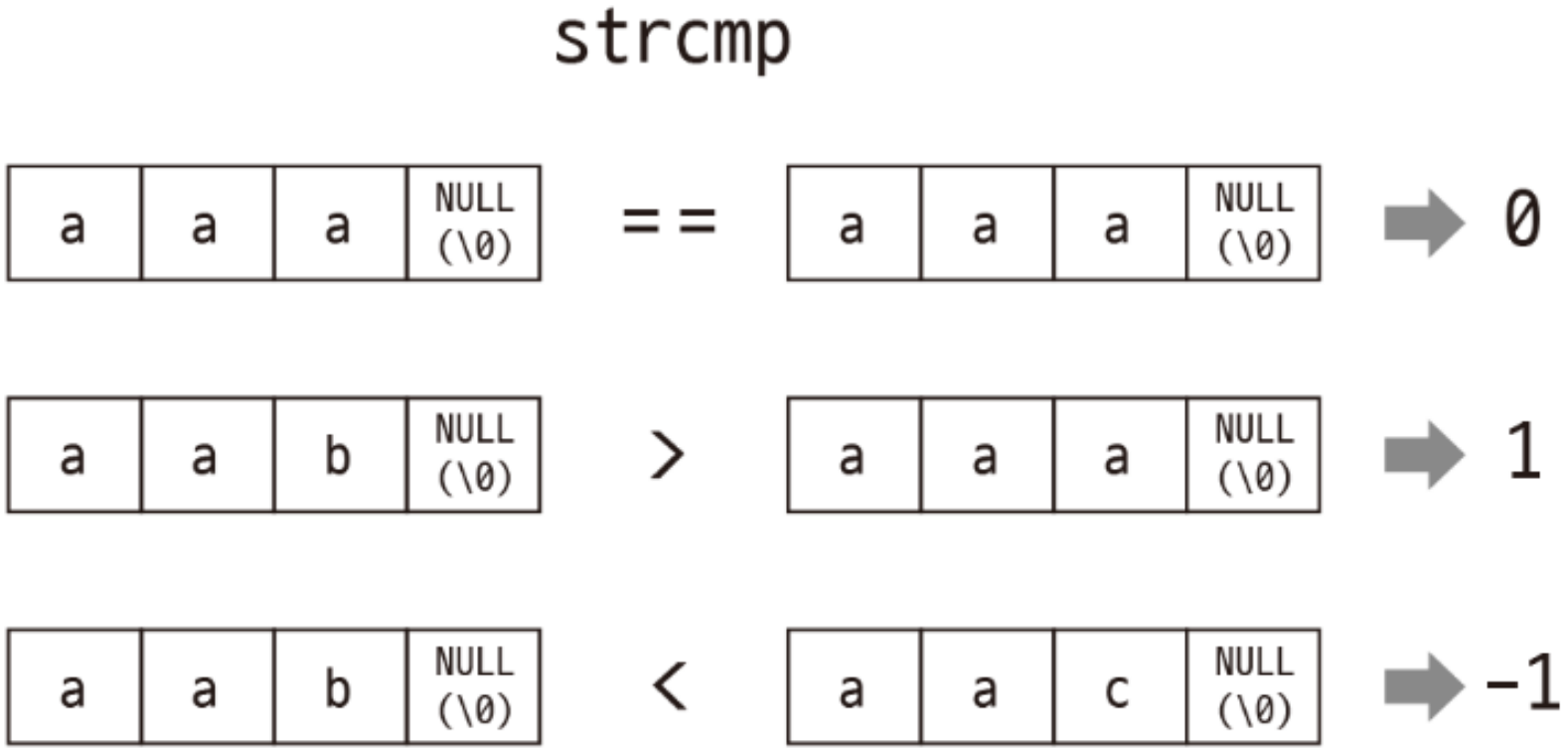
```
#include <stdio.h>
#include <string.h>    // strcmp 함수가 선언된 헤더 파일
int main(){
    // aaa는 ASCII 코드로 97 97 97
    // aab는 ASCII 코드로 97 97 98
    // aac는 ASCII 코드로 97 97 99

    printf("%d\n", strcmp("aaa", "aaa"));    //  0: aaa와 aaa는 같으므로 0
    printf("%d\n", strcmp("aab", "aaa"));    //  1: aab와 aaa 중에서 aab가 크므로 1
    printf("%d\n", strcmp("aab", "aac"));    // -1: aab와 aac 중에서 aac가 크므로 -1

    return 0;
}
```

```
0
1
-1
```

문자열의 길이를 구하고 비교하기

문자열 비교하기

strcmp 함수로 문자열 비교



문자열의 길이를 구하고 비교하기

▶문자열 비교하기

```
#define _CRT_SECURE_NO_WARNINGS    // scanf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcmp 함수가 선언된 헤더 파일
int main(){
    char s1[20];
    char s2[20];
    printf("문자열 두 개를 입력하세요: ");
    scanf("%s %s", s1, s2);
    int ret = strcmp(s1, s2);    // 입력된 문자열 비교
    switch (ret){
    case 0:
        printf("두 문자열이 같음\n");
        break;
    case 1:
        printf("%s보다 %s가 큼\n", s2, s1);
        break;
    case -1:
        printf("%s보다 %s가 큼\n", s1, s2);
        break;
    }
    return 0;
}
```

```
문자열 두 개를 입력하세요: hello world (입력)
hello보다 world가 큼
```

문자열의 길이를 구하고 비교하기

문자열 비교하기

```c
#include <stdio.h>
#include <string.h>    // strcmp 함수가 선언된 헤더 파일
int main(){
    char s1[20];
    char s2[20];
    printf("문자열 두 개를 입력하세요: ");
    scanf("%s %s", s1, s2);
    int ret = strcmp(s1, s2);    // 입력된 문자열 비교
    printf("반환값: %d\n", ret);
    // 리눅스와 OS X에서는 ASCII 코드값의 차이를 반환하므로
    // if 조건문으로 판단
    if (ret == 0){
        printf("두 문자열이 같음\n");
    }
    else if (ret > 0){    // 양수일 때
        printf("%s보다 %s가 큼\n", s2, s1);
    }
    else if (ret < 0){    // 음수일 때
        printf("%s보다 %s가 큼\n", s1, s2);
    }
    return 0;
}
```

```
$ gcc string_scanf_compare_linux_osx.c -o compare
$ ./compare
문자열 두 개를 입력하세요: aaf aaa (입력)
반환값: 5
aaa보다 aaf가 큼
```

실습문제 06

문자열 길이 구하기

다음 소스 코드를 완성하여 10이 출력되도록 작성하시오.

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

int main()

{
    char *s1 = "C Language";

    printf("%d\n", _____________);

    return 0;

}
```

```
10
```

실습문제 07

연습문제: 문자열 비교하기

다음 소스 코드를 완성하여 0이 출력되게 만드세요.

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

int main()
{
    ____________________________
    char *s2 = "Pachelbel Canon";

    int ret = strcmp(s1, s2);

    printf("%d", ret);

    return 0;
}
```

```
0
```

문자열을 복사하고 붙이기

문자열 복사하기

strcpy(대상문자열, 원본문자열);

- char *strcpy(char *_Dest, char const *_Source);
- 대상문자열의 포인터를 반환

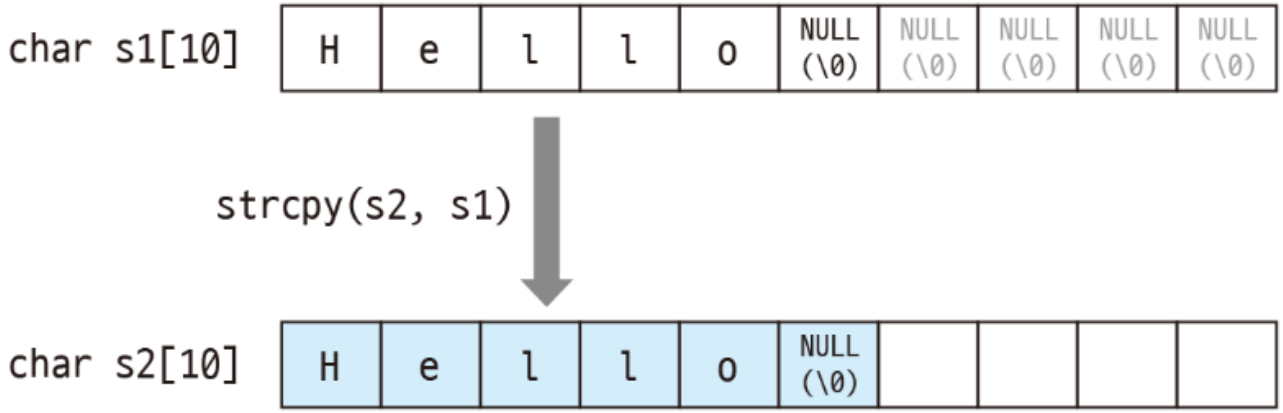
```
#define _CRT_SECURE_NO_WARNINGS    // strcpy 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcpy 함수가 선언된 헤더 파일

int main()
{
    char s1[10] = "Hello";    // 크기가 10인 char형 배열을 선언하고 문자열 할당
    char s2[10];              // 크기가 10인 char형 배열을 선언

    strcpy(s2, s1);         // s1의 문자열을 s2로 복사

    printf("%s\n", s2);    // Hello

    return 0;
}
```

```
Hello
```

문자열을 복사하고 붙이기

문자열 복사하기

strcpy 함수로 문자열 복사



문자열을 복사하고 붙이기

문자열 복사하기

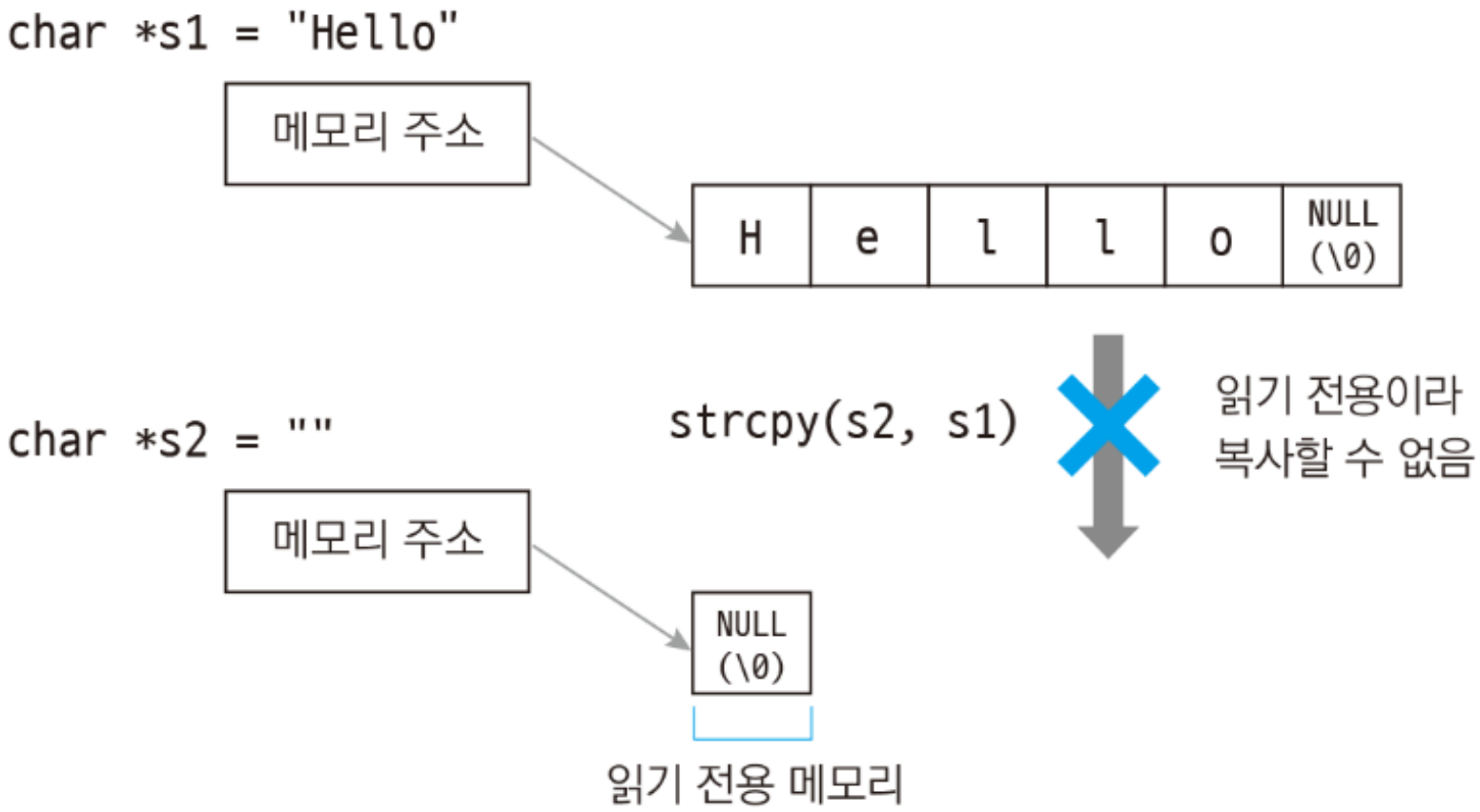
```
#define _CRT_SECURE_NO_WARNINGS    // strcpy 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcpy 함수가 선언된 헤더 파일

int main()
{
    char *s1 = "Hello";    // 문자열 포인터
    char *s2 = "";         // 문자열 포인터

    strcpy(s2, s1);    // 실행 에러

    printf("%s\n", s2);

    return 0;
}
```

```
0xC0000005: 0x013A585D 위치를 기록하는 동안 액세스 위반이 발생했습니다.
```

문자열을 복사하고 붙이기

문자열 복사하기

읽기 전용 메모리에는 문자열을 복사할 수 없음



문자열을 복사하고 붙이기

▶문자열 복사하기

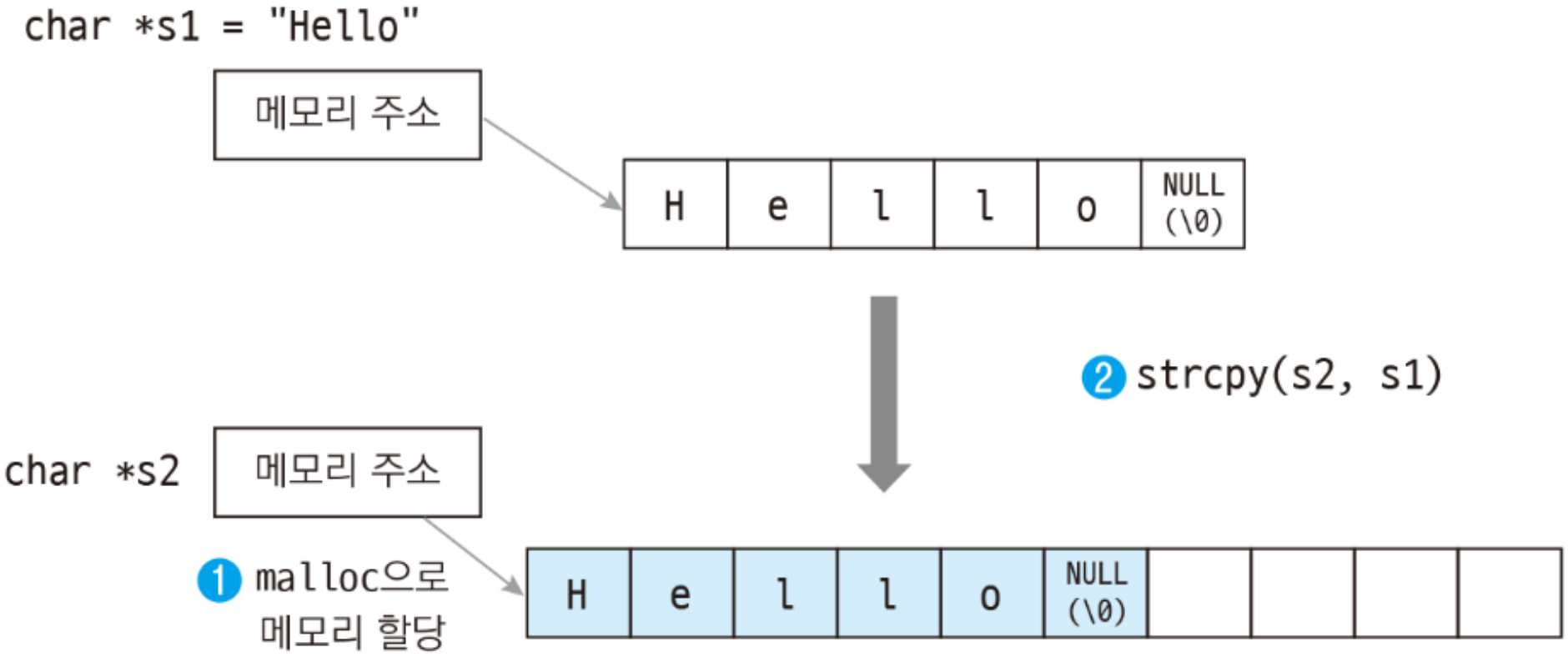
```
#define _CRT_SECURE_NO_WARNINGS    // strcpy 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcpy 함수가 정의된 헤더 파일
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    char *s1 = "hello";                        // 문자열 포인터
    char *s2 = malloc(sizeof(char) * 10);    // char 10개 크기만큼 동적 메모리 할당

    strcpy(s2, s1);        // s1의 문자열을 s2로 복사

    printf("%s\n", s2);    // Hello

    free(s2);    // 동적 메모리 해제

    return 0;
}
```

```
Hello
```

문자열을 복사하고 붙이기

문자열 복사하기

strcpy 함수로 문자열 포인터에 문자열 복사



문자열을 복사하고 붙이기

문자열 붙이기

strcat(최종문자열, 붙일문자열);

- char *strcat(char *_Destination, char const *_Source);
- 최종 문자열의 포인터를 반환

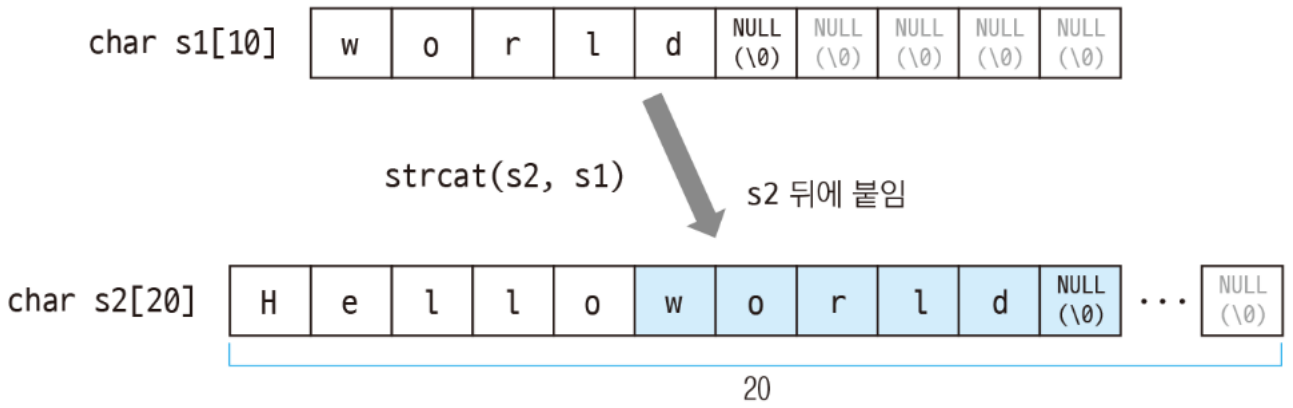
```
#define _CRT_SECURE_NO_WARNINGS    // strcat 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcat 함수가 선언된 헤더 파일

int main()
{
    char s1[10] = "world";
    char s2[20] = "Hello"; // s2 뒤에 붙일 것이므로 배열 크기를 크게 만듦

    strcat(s2, s1);        // s2 뒤에 s1를 붙임

    printf("%s\n", s2);    // Helloworld

    return 0;
}
```

```
Helloworld
```

문자열을 복사하고 붙이기

문자열 붙이기

strcat 함수로 문자열 붙이기



문자열을 복사하고 붙이기

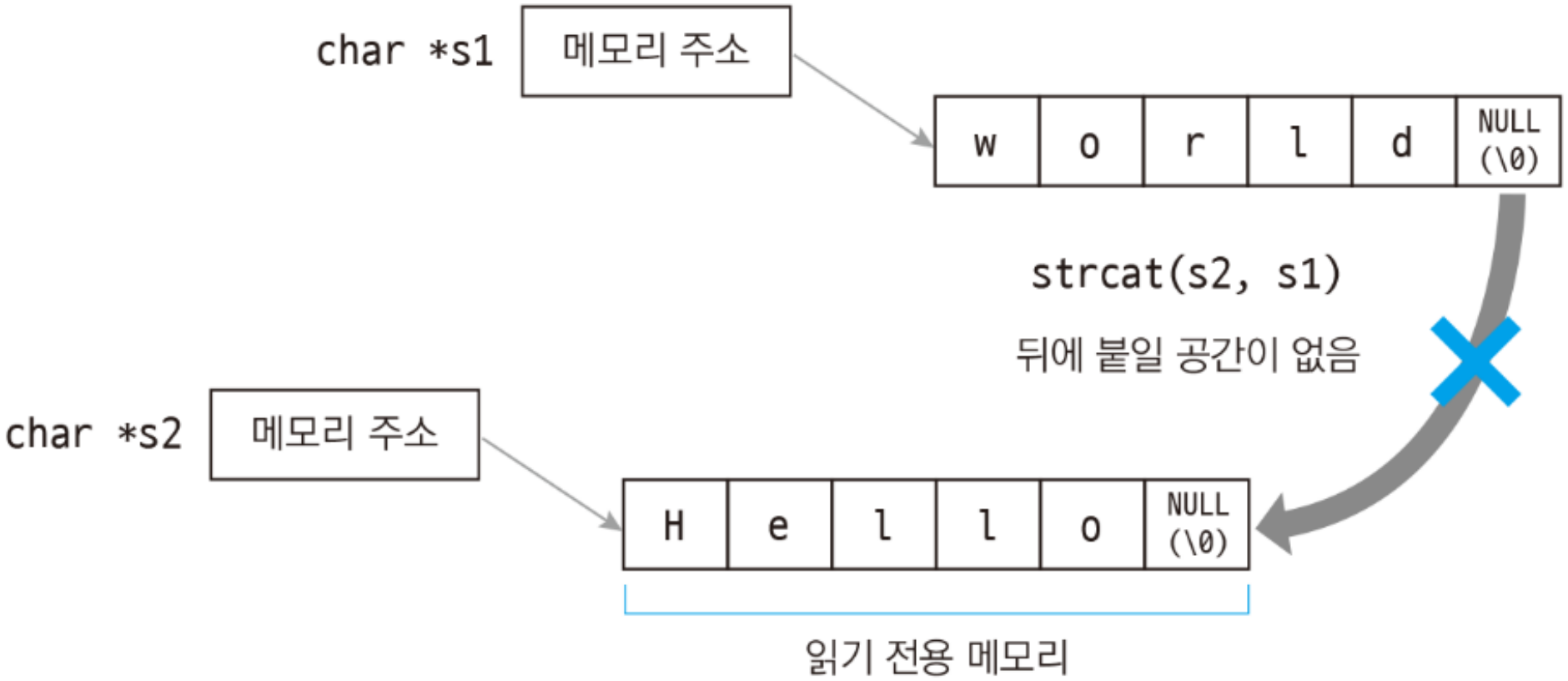
문자열 붙이기

```
#define _CRT_SECURE_NO_WARNINGS    // strcat 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcat 함수가 선언된 헤더 파일

int main()
{
    char *s1 = "world";    // 문자열 포인터
    char *s2 = "Hello";    // 문자열 포인터

    strcat(s2, s1);    // 실행 에러

    printf("%s\n", s1);

    return 0;
}
```

```
0xC0000005: 0x013A585D 위치를 기록하는 동안 액세스 위반이 발생했습니다.
```

문자열을 복사하고 붙이기

- 문자열 붙이기
  - 문자열 포인터에는 문자열을 붙일 공간이 없음



문자열을 복사하고 붙이기

문자열 붙이기

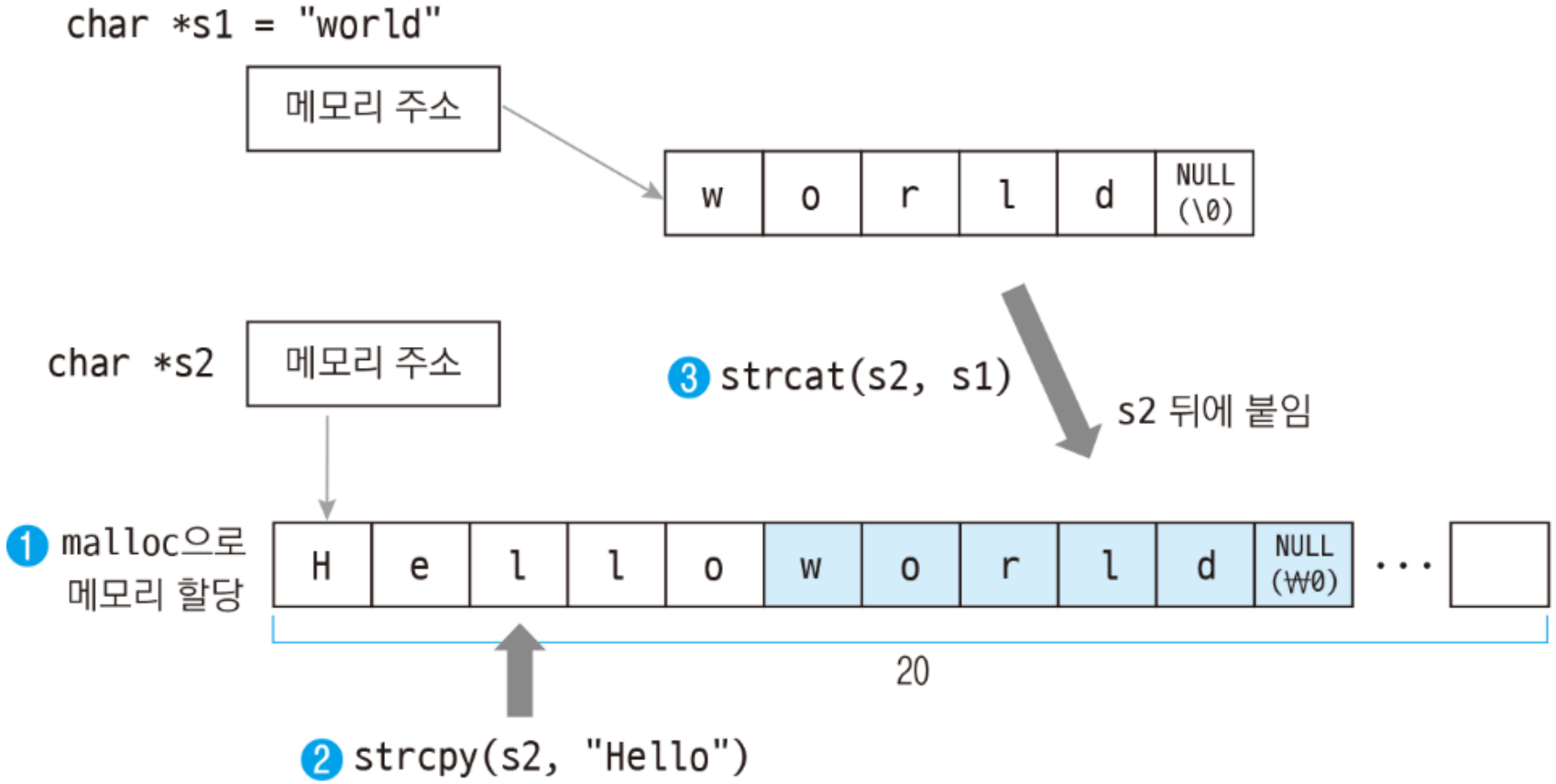
```
#define _CRT_SECURE_NO_WARNINGS    // strcpy 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcat 함수가 선언된 헤더 파일
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    char *s1 = "world";                        // 문자열 포인터
    char *s2 = malloc(sizeof(char) * 20);    // char 20개 크기만큼 동적 메모리 할당

    strcpy(s2, "Hello");   // s2에 Hello 문자열 복사

    strcat(s2, s1);        // s2 뒤에 s1을 붙임

    printf("%s\n", s2);   // Helloworld

    free(s2);    // 동적 메모리 해제

    return 0;
}
```

```
Helloworld
```

문자열을 복사하고 붙이기

▶문자열 붙이기

▶문자열 포인터에 메모리를 할당한 뒤 문자열을 붙임



문자열을 복사하고 붙이기

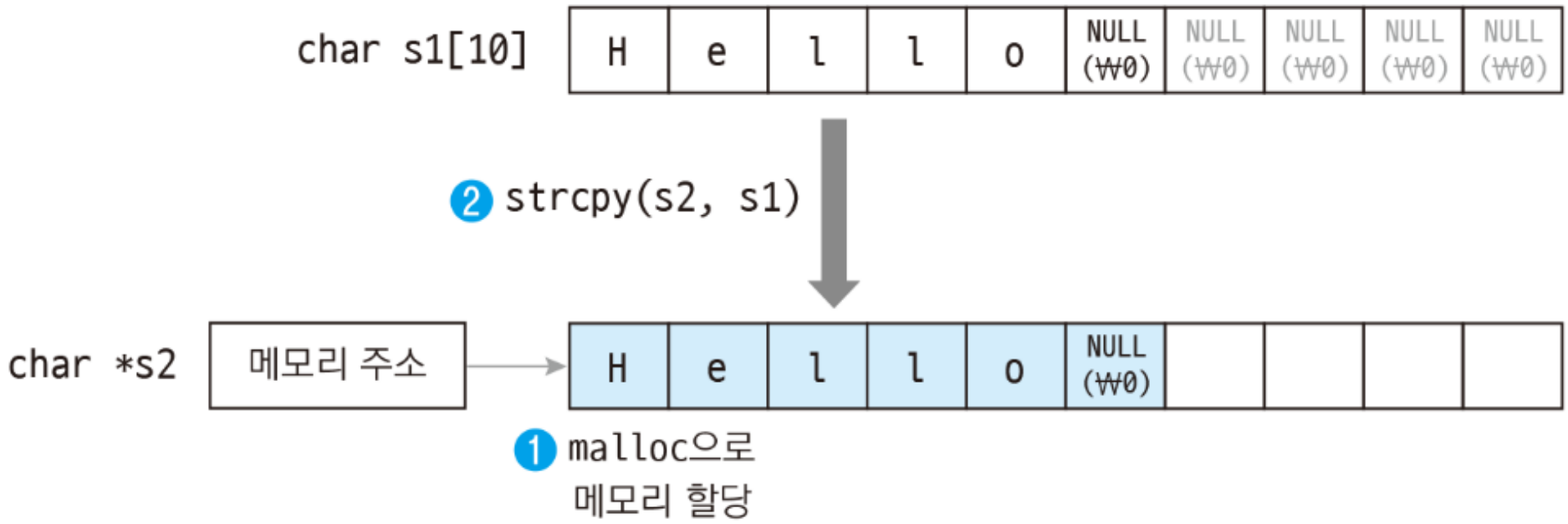
- 배열 형태의 문자열을 문자열 포인터에 복사하기
  - strcat(최종문자열, 붙일문자열);
    - char *strcat(char *_Destination, char const *_Source);
    - 최종 문자열의 포인터를 반환

```
#define _CRT_SECURE_NO_WARNINGS    // strcpy 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcpy 함수가 선언된 헤더 파일
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일
int main()
{
    char s1[10] = "Hello";                    // 크기가 10인 char형 배열을 선언하고 문자열 할당
    char *s2 = malloc(sizeof(char) * 10);    // char 10개 크기만큼 동적 메모리 할당
    strcpy(s2, s1);         // s1의 문자열을 s2로 복사
    printf("%s\n", s2);    // Hello
    free(s2);    // 동적 메모리 해제
    return 0;
}
```

```
Hello
```

문자열을 복사하고 붙이기

- 배열 형태의 문자열을 문자열 포인터에 복사하기
  - 배열 형태의 문자열을 문자열 포인터에 복사



문자열을 복사하고 붙이기

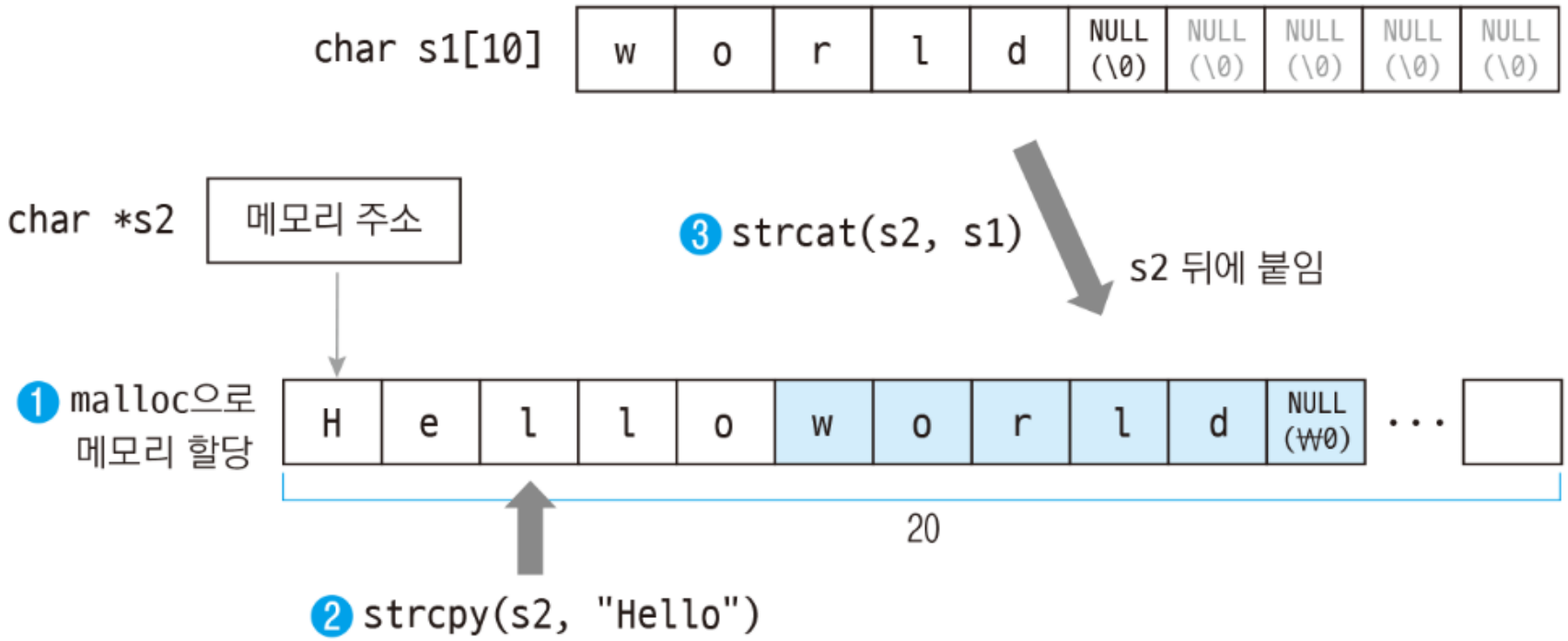
▶ 배열 형태의 문자열을 문자열 포인터에 붙이기

```
#define _CRT_SECURE_NO_WARNINGS    // strcpy, strcat 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strcpy, strcat 함수가 선언된 헤더 파일
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    char s1[10] = "world";                  // 크기가 10인 char형 배열을 선언하고 문자열 할당
    char *s2 = malloc(sizeof(char) * 20);    // char 20개 크기만큼 동적 메모리 할당

    strcpy(s2, "Hello");   // s2에 Hello 문자열 복사

    strcat(s2, s1);        // s2 뒤에 s1을 붙임

    printf("%s\n", s2);    // Helloworld

    free(s2);  // 동적 메모리 해제

    return 0;
}
```

```
Helloworld
```

문자열을 복사하고 붙이기

- 배열 형태의 문자열을 문자열 포인터에 복사하기
  - 배열 형태의 문자열을 문자열 포인터에 붙이기



실습문제 08

문자열 포인터를 배열에 복사하기

다음 소스 코드를 완성하여 "C Language"가 출력되게 만드시오.

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

int main()
{
    char *s1 = "C Language";
    char s2[20];

    ____________________________

    printf("%s\n", s2);

    return 0;
}
```

```
C Language
```

실습문제 09

문자열 포인터를 동적 메모리에 복사하기

다음 소스 코드를 완성하여 "The Little Prince"가 출력되게 만드세요.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main()
{
    char *s1 = "The Little Prince";
    char *s2 = ①_____________________

    ②_____________________________

    printf("%s\n", s2);

    free(s2);

    return 0;
}
```

```
The Little Prince
```

실습문제 10

문자 배열을 붙이기

다음 소스 코드를 완성하여 "Beethoven 9th Symphony"가 출력되게 만드시오.

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

int main()
{
    char s1[20] = " 9th Symphony";
    char s2[40] = "Beethoven";

    ________________________

    printf("%s\n", s2);

    return 0;
}
```

```
Beethoven 9th Symphony
```

실습문제 11

문자열 리터럴과 동적 메모리 붙이기

다음 소스 코드를 완성하여 "Alice in Wonderland"가 출력되게 만드시오.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main()
{
    char *s1 = " Wonderland";
    char *s2 = malloc(sizeof(char) * 30);

    ①____________________

    ②____________________

    printf("%s\n", s2);

    free(s2);

    return 0;
}
```

```
Alice in Wonderland
```

문자열 만들기

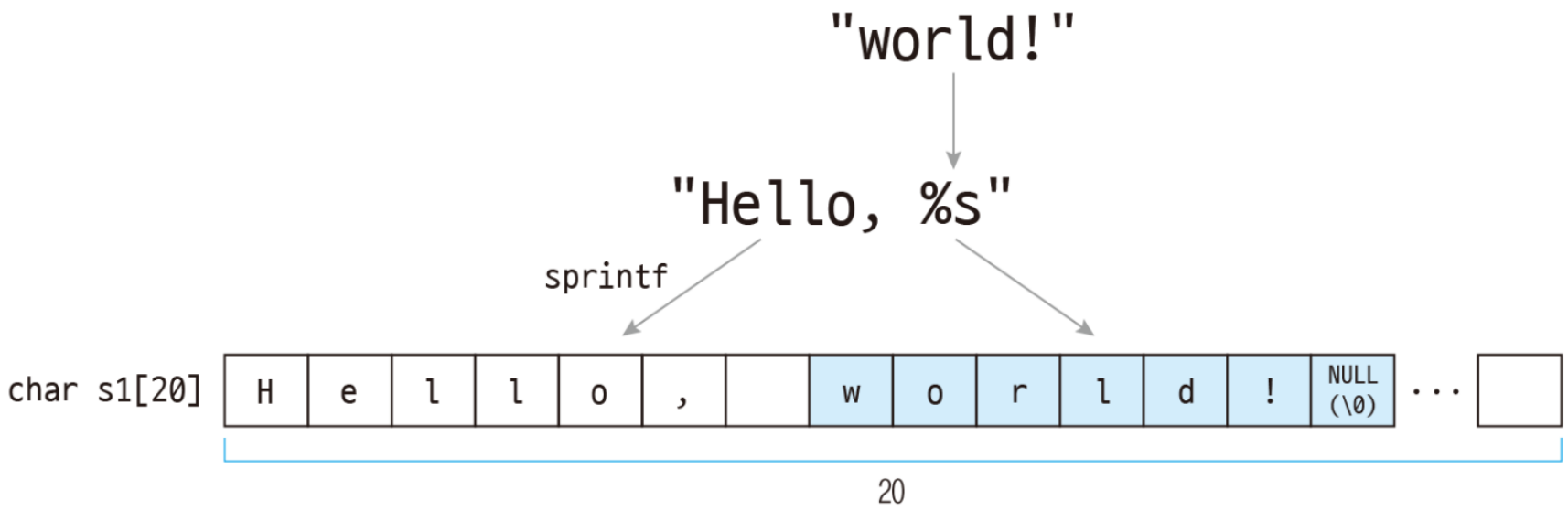
- 서식을 지정하여 배열 형태로 문자열 만들기
  - sprintf(배열, 서식, 값);
  - sprintf(배열, 서식, 값1, 값2, …);
    - int sprintf(char *const _Buffer, char const *const _Format, ...);
    - 성공하면 만든 문자열의 길이를 반환, 실패하면 음수를 반환

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일

int main()
{
    char s1[20];    // 크기가 20인 char형 배열을 선언

    sprintf(s1, "Hello, %s", "world!");    // "Hello, %s"로 서식을 지정하여 s1에 저장

    printf("%s\n", s1);    // Hello, world!: 문자열 s1 출력

    return 0;
}
```

```
Hello, world!
```

문자열 만들기

- 서식을 지정하여 배열 형태로 문자열 만들기
  - sprintf 함수를 사용하여 배열 형태로 문자열 만들기



문자열 만들기

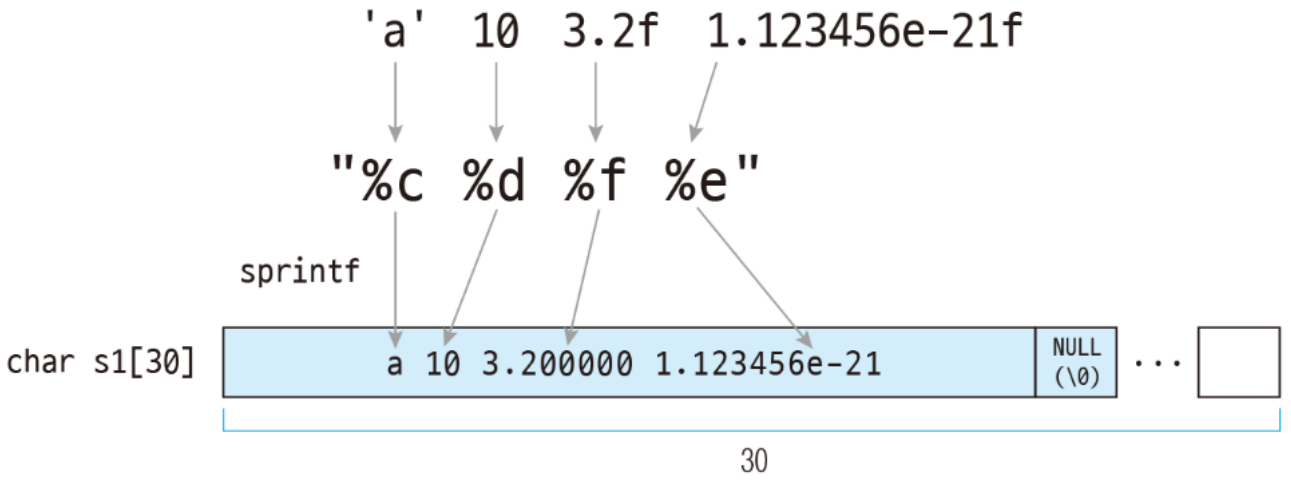
▶ 서식을 지정하여 배열 형태로 문자열 만들기

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일

int main()
{
    char s1[30];    // 크기가 30인 char형 배열을 선언

    sprintf(s1, "%c %d %f %e", 'a', 10, 3.2f, 1.123456e-21f);    // 문자, 정수, 실수를 문자열로 만듦

    printf("%s\n", s1);    // a 10 3.200000 1.123456e-21: 문자열 s1 출력

    return 0;
}
```

```
a 10 3.200000 1.123456e-21
```

문자열 만들기

- 서식을 지정하여 배열 형태로 문자열 만들기
  - sprintf 함수를 사용하여 다양한 서식의 값을 문자열로 만들기



문자열 만들기

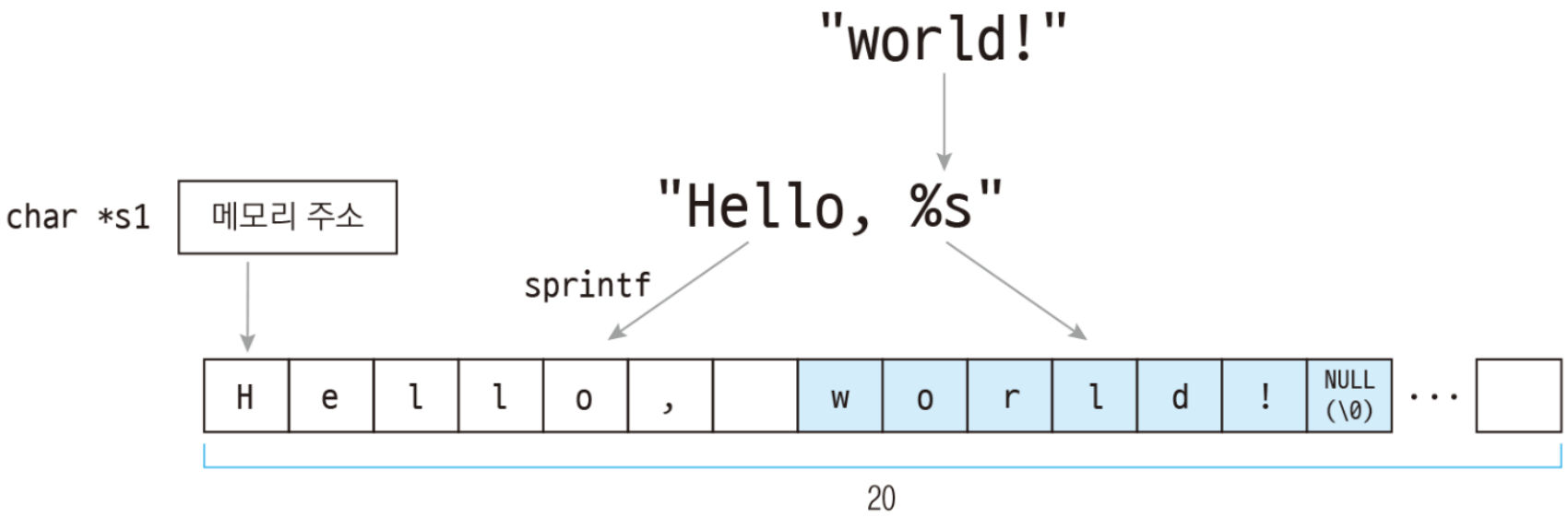
- 서식을 지정하여 문자열 포인터에 문자열 만들기
  - sprintf(문자열포인터, 서식, 값);
  - sprintf(문자열포인터, 서식, 값1, 값2, ...);

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    char *s1 = malloc(sizeof(char) * 20);  // char 20개 크기만큼 동적 메모리 할당

    sprintf(s1, "Hello, %s", "world!");    // "Hello, %s"로 서식을 지정하여 s1에 저장

    printf("%s\n", s1);    // Hello, world!: 문자열 s1 출력

    free(s1);    // 동적 메모리 해제

    return 0;
}
```

```
Hello, world!
```

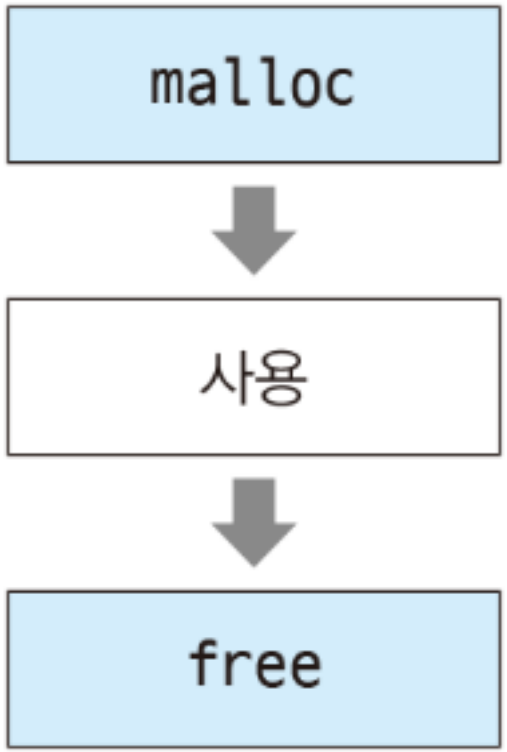
문자열 만들기

- 서식을 지정하여 문자열 포인터에 문자열 만들기
  - printf 함수를 사용하여 문자열 포인터에 문자열 만들기



문자열 만들기

- 서식을 지정하여 문자열 포인터에 문자열 만들기
  - 문자열 포인터 사용 패턴



문자열 만들기

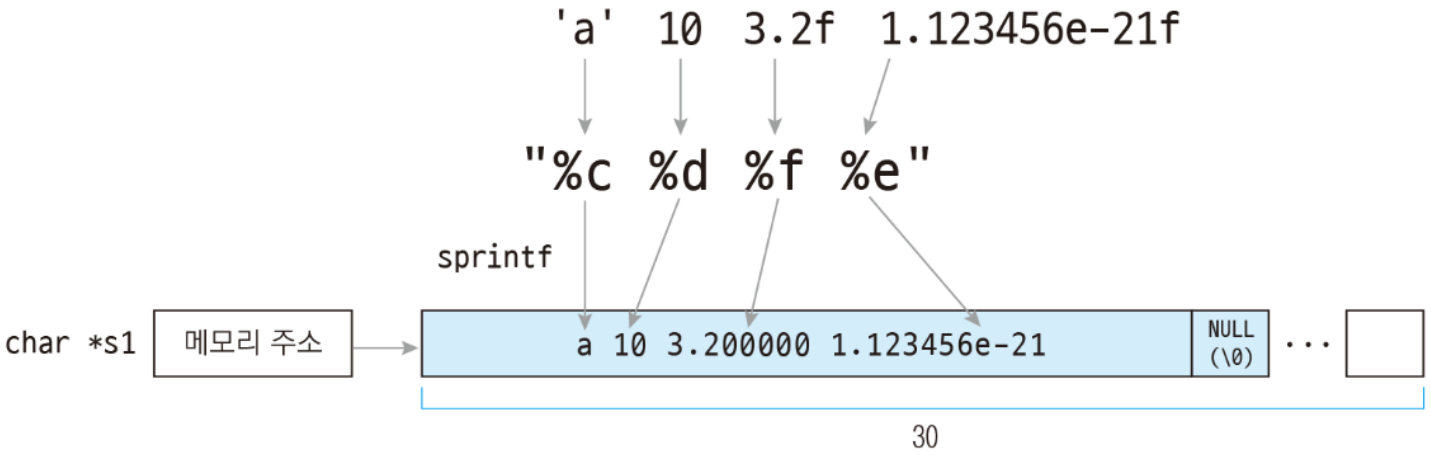
▶ 서식을 지정하여 문자열 포인터에 문자열 만들기

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일
#include <stdlib.h>    // malloc, free 함수가 선언된 헤더 파일

int main()
{
    char *s1 = malloc(sizeof(char) * 30);    // char 30개 크기만큼 동적 메모리 할당

    sprintf(s1, "%c %d %f %e", 'a', 10, 3.2f, 1.123456e-21f);    // 문자, 정수, 실수를 문자열로 만듦

    printf("%s\n", s1);    // a 10 3.200000 1.123456e-21: 문자열 s1 출력

    free(s1);    // 동적 메모리 해제

    return 0;
}
```

```
a 10 3.200000 1.123456e-21
```

문자열 만들기

- 서식을 지정하여 문자열 포인터에 문자열 만들기
  - sprintf 함수를 사용하여 다양한 서식의 값을 문자열로 만들기



실습문제 12

숫자와 문자열을 조합하여 문자열 만들기

다음 소스 코드를 완성하여 "9th Symphony"가 출력되게 만드시오.

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

int main()
{
    char s1[20];

    sprintf(s1, "%dth %s", _________________);

    printf("%s\n", s1);

    return 0;
}
```

```
9th Symphony
```

실습문제 13

서식에 맞게 문자열 만들기

다음 소스 코드를 완성하여 10, 20, 30, c, 99가 출력되게 만드시오.

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

int main()
{
    char s1[20];

    sprintf(s1, _________________, 10, 20, 30, 'c', 99);

    printf("%s\n", s1);

    return 0;
}
```

```
10 20 30 c 99
```

문자열 검색하기

- 문자열 안에서 문자로 검색하기
  - strchr(대상 문자열, 검색할 문자);
    - char *strchr(char *const _String, int const _Ch);
    - 문자를 찾았으면 문자로 시작하는 문자열의 포인터를 반환, 문자가 없으면 NULL을 반환

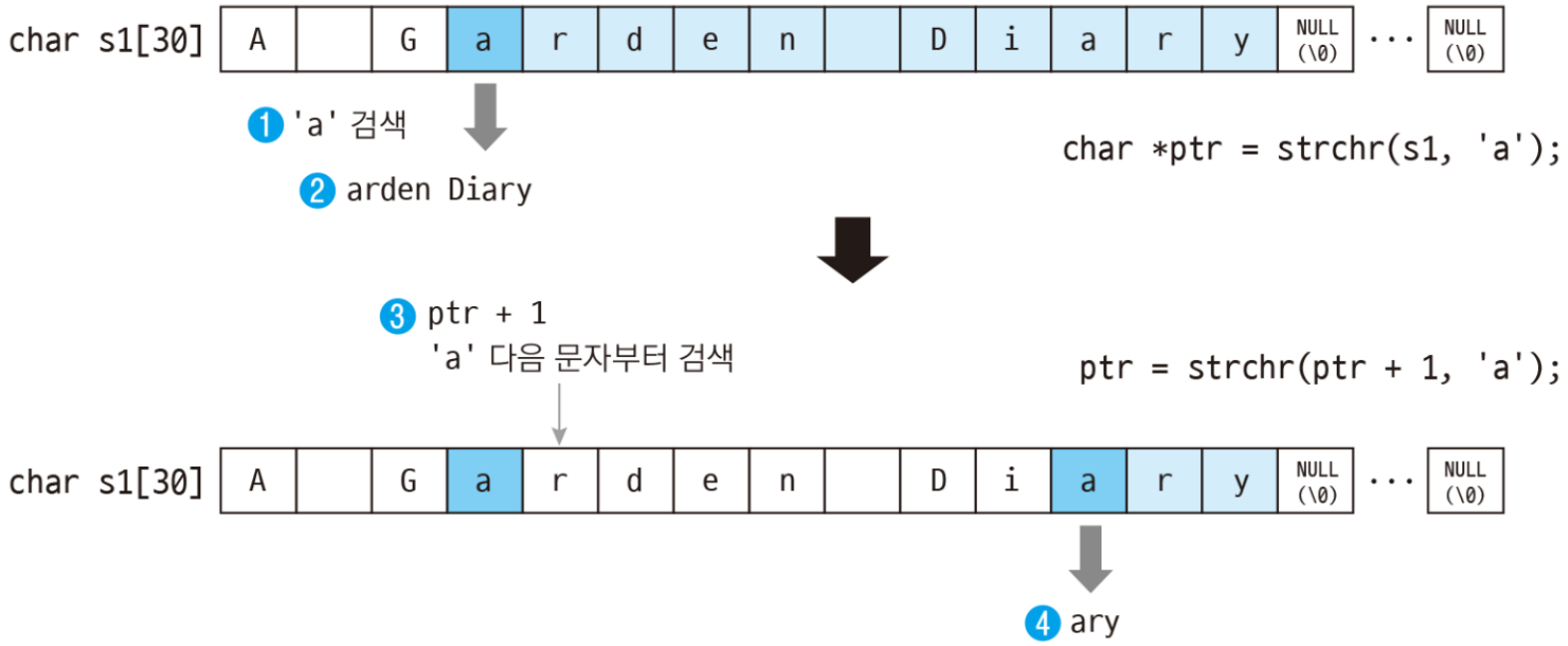
```
#include <stdio.h>
#include <string.h>    // strchr 함수가 선언된 헤더 파일
int main()
{
    char s1[30] = "A Garden Diary";  // 크기가 30인 char형 배열을 선언하고 문자열 할당

    char *ptr = strchr(s1, 'a');     // 'a'로 시작하는 문자열 검색, 포인터 반환

    while (ptr != NULL)              // 검색된 문자열이 없을 때까지 반복
    {
        printf("%s\n", ptr);         // 검색된 문자열 출력
        ptr = strchr(ptr + 1, 'a');  // 포인터에 1을 더하여 a 다음부터 검색
    }
    return 0;
}
```

```
arden Diary
ary
```

문자열 검색하기

문자열 안에서 문자로 검색하기

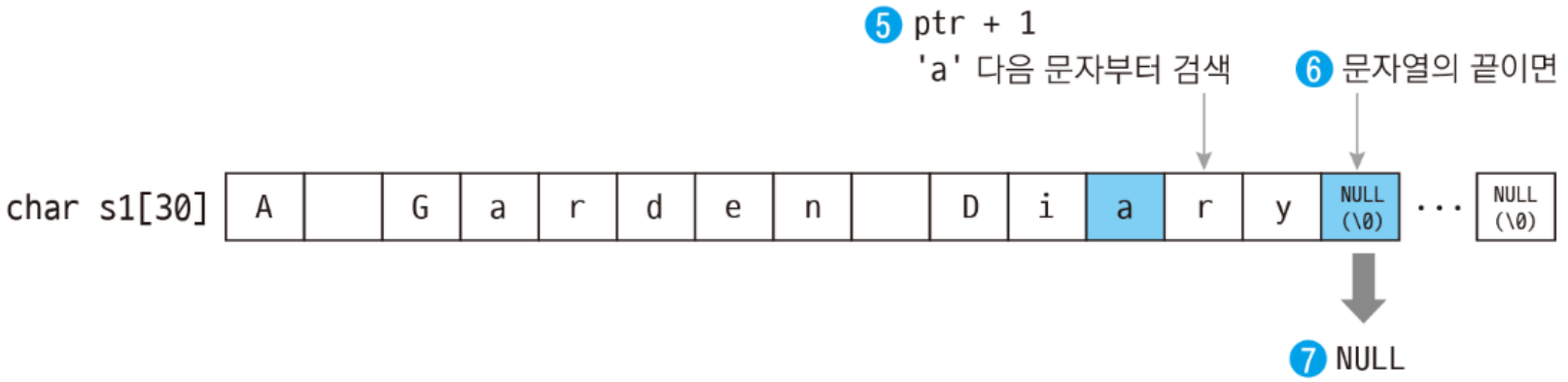
strchr 함수로 문자 검색 1



문자열 검색하기

문자열 안에서 문자로 검색하기

strchr 함수로 문자 검색 2




```
ptr = strchr(ptr + 1, 'a');
```

문자열 검색하기

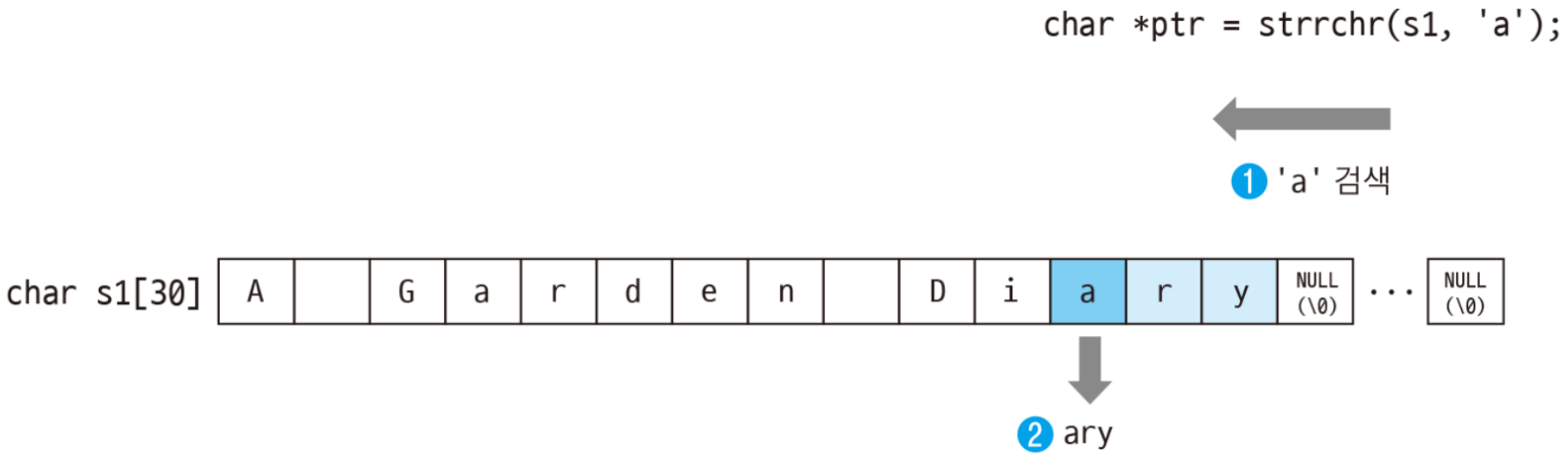
- 문자열의 오른쪽 끝부터 문자로 검색하기
  - strrchr(대상문자열, 검색할문자);
    - char *strrchr(char *const _String, int const _Ch);
    - 문자열의 끝에서부터 역순으로 검색해서 문자를 찾았으면 해당 문자로 시작하는 문자열의 포인터를 반환, 문자가 없으면 NULL을 반환

```
#include <stdio.h>
#include <string.h>   // strrchr 함수가 선언된 헤더 파일

int main()
{
    char s1[30] = "A Garden Diary";    // 크기가 30인 char형 배열을 선언하고 문자열 할당

    char *ptr = strrchr(s1, 'a');      // 문자열 끝에서부터 'a'로 시작하는 문자열 검색. 포인터 반환

    printf("%s\n", ptr);   // ary

    return 0;
}
```

```
ary
```

문자열 검색하기

- 문자열의 오른쪽 끝부터 문자로 검색하기
  - strrchr 함수로 끝에서부터 문자 검색



문자열 검색하기

문자열 안에서 문자열로 검색하기

strstr(대상 문자열, 검색할 문자열);

- char *strstr(char *const _String, char const *const _SubString);
- 문자열을 찾았으면 문자열로 시작하는 문자열의 포인터를 반환, 문자열이 없으면 NULL을 반환

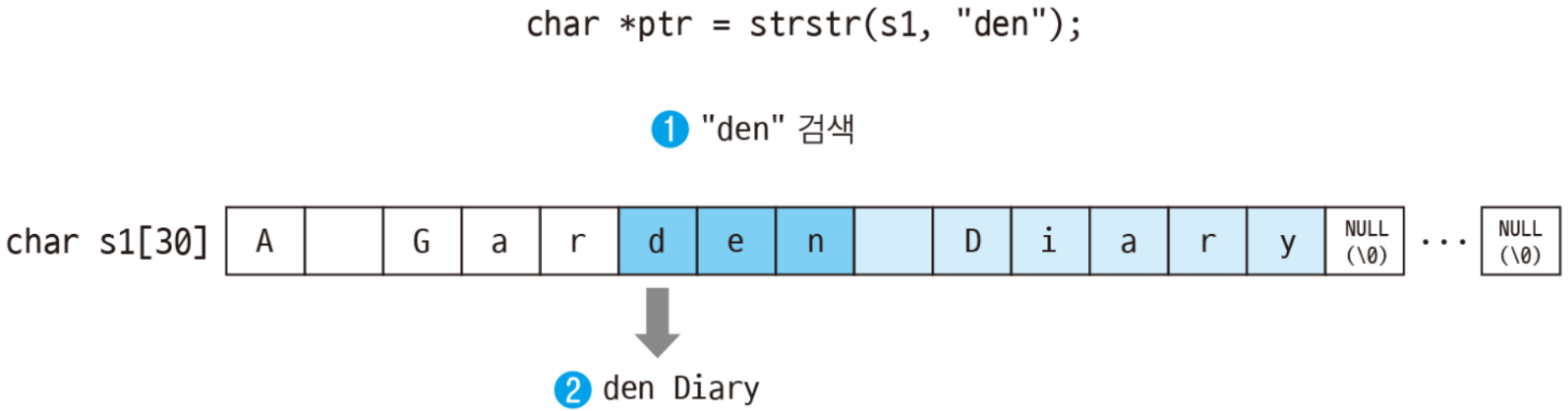
```
#include <stdio.h>
#include <string.h>    // strstr 함수가 선언된 헤더 파일

int main()
{
    char s1[30] = "A Garden Diary";     // 크기가 30인 char형 배열을 선언하고 문자열 할당

    char *ptr = strstr(s1, "den");      // den으로 시작하는 문자열 검색, 포인터 반환

    printf("%s\n", ptr);    // den Diary

    return 0;
}
```

```
den Diary
```

문자열 검색하기

▶문자열 안에서 문자열로 검색하기

▶strstr 함수로 문자열 검색



문자열 검색하기

문자열 안에서 문자열로 검색하기

문자열 끝까지 검색

```
char s1[100] = "A Garden Diary A Garden Diary A Garden Diary";

char *ptr = strstr(s1, "den");      // den으로 시작하는 문자열 검색, 포인터 반환

while (ptr != NULL)                 // 검색된 문자열이 없을 때까지 반복
{
    printf("%s\n", ptr);            // 검색된 문자열 출력
    ptr = strstr(ptr + 1, "den");  // den 포인터에 1을 더하여 e부터 검색
}
```

```
den Diary A Garden Diary A Garden Diary
den Diary A Garden Diary
den Diary
```

실습문제 14

문자열의 오른쪽 끝부터 문자로 검색하기

다음 소스 코드를 완성하여 "ince"가 출력되게 만드시오.

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

int main()
{
    char s1[30] = "The Little Prince";

    ____________________________________

    printf("%s\n", ptr);

    return 0;
}
```

```
ince
```

실습문제 15

문자열 안에서 문자로 검색하기

다음 소스 코드를 완성하여 "n Wonderland", "nderland", "nd"이 각 줄마다 출력되게 만드세요.

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

int main()
{
    char s1[30] = "Alice in Wonderland";

    ①_____________________________________

    ②_____________________________________
    {
        printf("%s\n", ptr);
        ③_____________________________________
    }

    return 0;
}
```

```
n Wonderland
nderland
nd
```

문자열 자르기

문자를 기준으로 문자열 자르기

strtok(대상문자열, 기준문자);

- char *strtok(char *_String, char const *_Delimiter);
- 자른 문자열을 반환, 더 이상 자를 문자열이 없으면 NULL을 반환

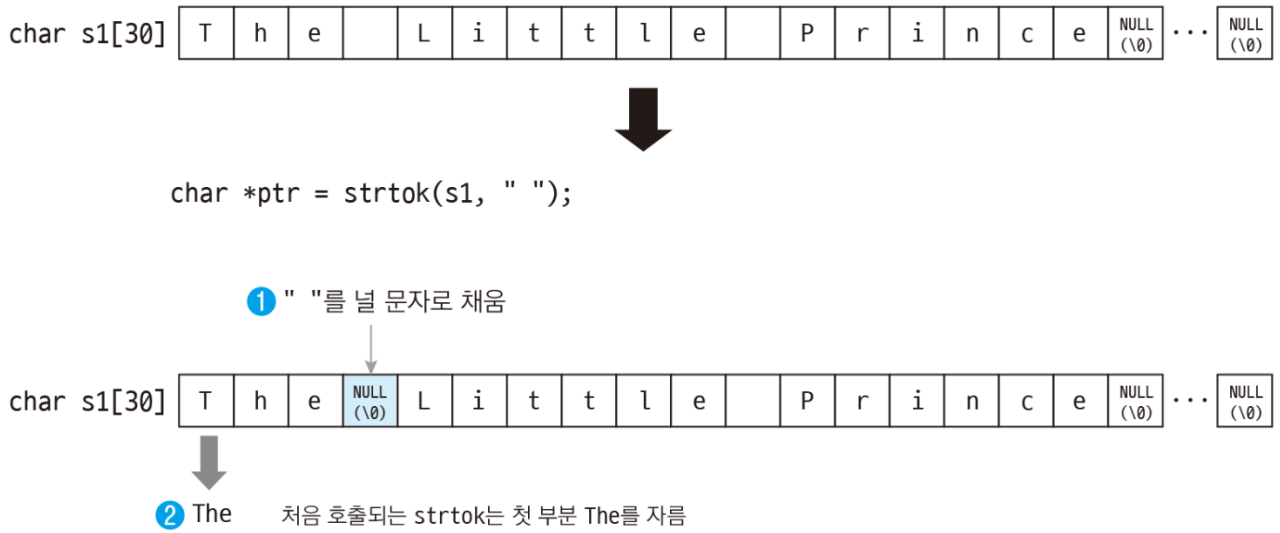
```
#define _CRT_SECURE_NO_WARNINGS    // strtok 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strtok 함수가 선언된 헤더 파일
int main(){
    char s1[30] = "The Little Prince";  // 크기가 30인 char형 배열을 선언하고 문자열 할당

    char *ptr = strtok(s1, " ");      // " " 공백 문자를 기준으로 문자열을 자름, 포인터 반환

    while (ptr != NULL){              // 자른 문자열이 나오지 않을 때까지 반복
        printf("%s\n", ptr);          // 자른 문자열 출력
        ptr = strtok(NULL, " ");      // 다음 문자열을 잘라서 포인터를 반환
    }
    return 0;
}
```

```
The
Little
Prince
```

문자열 자르기

▶문자를 기준으로 문자열 자르기

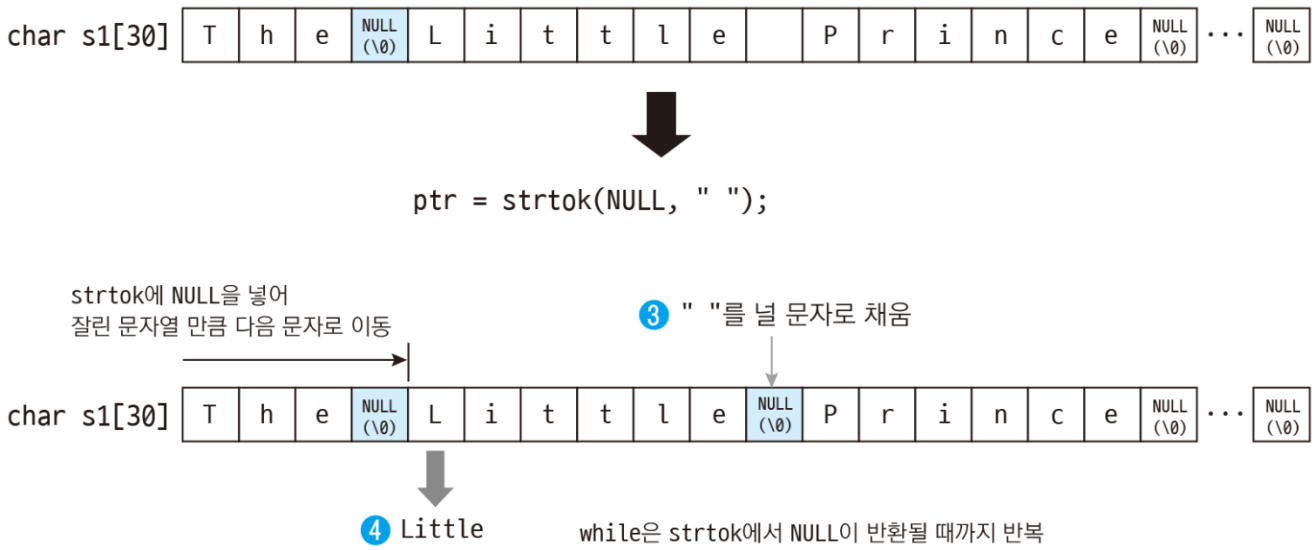
▶strtok 함수로 문자열 자르기 1



문자열 자르기

▶문자를 기준으로 문자열 자르기

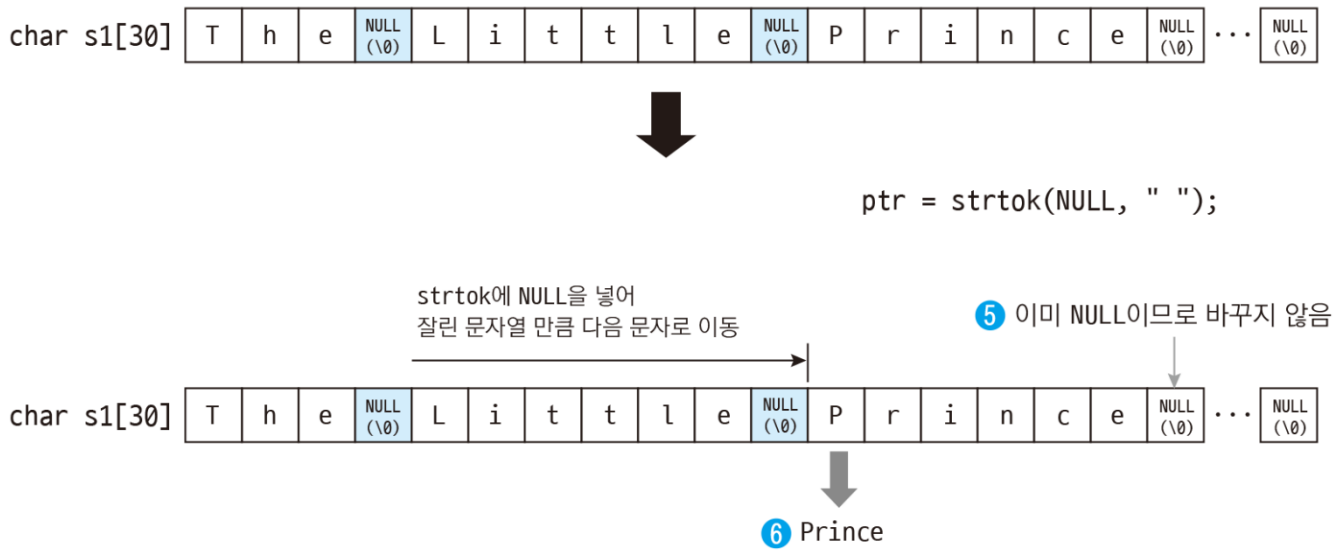
▶strtok 함수로 문자열 자르기 2



문자열 자르기

문자를 기준으로 문자열 자르기

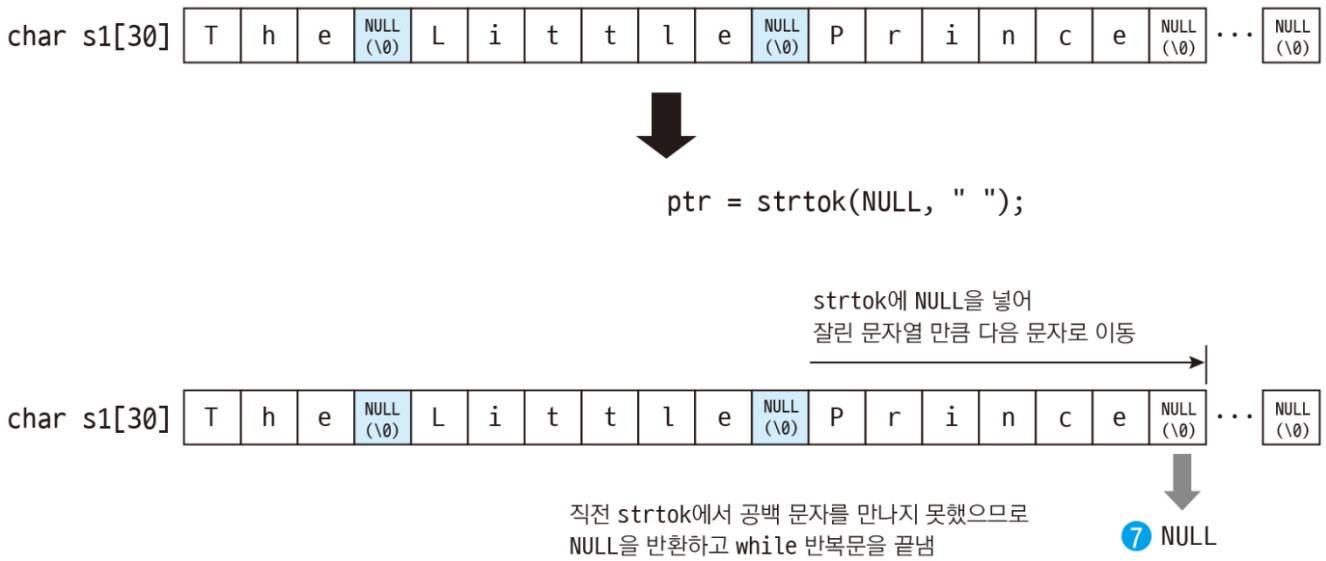
strtok 함수로 문자열 자르기 3



문자열 자르기

▶문자를 기준으로 문자열 자르기

▶strtok 함수로 문자열 자르기 4



문자열 자르기

문자열 포인터 자르기

문자열 포인터에 문자열 리터럴이 들어있어서 읽기 전용인 상태라면 strtok함수를 사용할 수 없음

```
char *s1 = "The Little Prince";   // 포인터에 문자열 리터럴 "The Little Prince"의 주소 저장

char *ptr = strtok(s1, " ");      // 실행 에러

while (ptr != NULL)
{
   printf("%s\n", ptr);
   ptr = strtok(NULL, " ");
}
```

```
0xC0000005: 0x013A585D 위치를 기록하는 동안 액세스 위반이 발생했습니다.
```

문자열 자르기

문자열 포인터 자르기

문자열 포인터에 문자열 리터럴을 할당하는 대신 동적 메모리를 할당하고, 문자열을 복사하면 이 문제를 해결 가능

```
char *s1 = malloc(sizeof(char) * 30);    // char 30개 크기만큼 동적 메모리 할당

strcpy(s1, "The Little Prince");    // s1에 문자열 복사

char *ptr = strtok(s1, " ");    // 동적 메모리에 들어있는 문자열은 자를 수 있음

while (ptr != NULL)
{
   printf("%s\n", ptr);
   ptr = strtok(NULL, " ");
}

free(s1);    // 동적 메모리 해제
```

문자열 자르기

날짜와 시간값 자르기

```
#define _CRT_SECURE_NO_WARNINGS    // strtok 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strtok 함수가 선언된 헤더 파일
int main(){
    char s1[30] = "2019-06-10T15:32:19";    // 크기가 30인 char형 배열을 선언하고 문자열 할당

    char *ptr = strtok(s1, "-T:");    // -, T, 콜론을 기준으로 문자열을 자름
                                      // 포인터 반환

    while (ptr != NULL){              // 자른 문자열이 나오지 않을 때까지 반복
        printf("%s\n", ptr);          // 자른 문자열 출력
        ptr = strtok(NULL, "-T:");    // 다음 문자열을 잘라서 포인터를 반환
    }

    return 0;
}
```

```
2019
06
10
15
32
19
```

문자열 자르기

자른 문자열 보관하기

```
#define _CRT_SECURE_NO_WARNINGS    // strtok 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>
#include <string.h>    // strtok 함수가 선언된 헤더 파일
int main(){
    char s1[30] = "The Little Prince";    // 크기가 30인 char형 배열을 선언하고 문자열 할당
    char *sArr[10] = { NULL, };    // 크기가 10인 문자열 포인터 배열을 선언하고 NULL로 초기화
    int i = 0;                     // 문자열 포인터 배열의 인덱스로 사용할 변수
    char *ptr = strtok(s1, " ");   // 공백 문자열을 기준으로 문자열을 자름
    while (ptr != NULL){            // 자른 문자열이 나오지 않을 때까지 반복
        sArr[i] = ptr;             // 문자열을 자른 뒤 메모리 주소를 문자열 포인터 배열에 저장
        i++;                       // 인덱스 증가
        ptr = strtok(NULL, " ");   // 다음 문자열을 잘라서 포인터를 반환
    }
    for (int i = 0; i < 10; i++){
        if (sArr[i] != NULL)           // 문자열 포인터 배열의 요소가 NULL이 아닐 때만
            printf("%s\n", sArr[i]);   // 문자열 포인터 배열에 인덱스로 접근하여 각 문자열 출력
    }
    return 0;
}
```

```
The
Little
Prince
```

실습문제 16

문자열 자르기

다음 소스 코드를 완성하여 "Alice's", "Adventures", "in", "Wonderland"가 각 줄마다 출력되게 만드시오.

```
ractice_string_tokensize_array.c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
int main(){
    char s1[40] = "Alice's Adventures in Wonderland";

    ①_____________________________________

    ②_____________________________________
    {
        printf("%s\n", tok);
        ③_____________________________________
    }

    return 0;
}
```

```
Alice's
Adventures
in
Wonderland
```

문자열과 숫자를 서로 변환하기

- 문자열과 숫자를 서로 변환하기
  - 프로그램을 만들다 보면 내용은 숫자이지만 형태는 문자열인 경우가 있음
  - 문자열을 int, float형으로 변환하고, int, float형 숫자를 문자열로 변환 가능

Hello, world!

10

35.28

문자열과 숫자를 서로 변환하기

- 포인터 사용하기
  - atoi(문자열);
    - int atoi(char const *_String);
    - 성공하면 변환된 정수를 반환, 실패하면 0을 반환

```
#include <stdio.h>
#include <stdlib.h>    // atoi 함수가 선언된 헤더 파일

int main(){
    char *s1 = "283";   // "283"은 문자열
    int num1;

    num1 = atoi(s1);        // 문자열을 정수로 변환하여 num1에 할당

    printf("%d\n", num1);   // 283

    return 0;
}
```

```
283
```

문자열과 숫자를 서로 변환하기

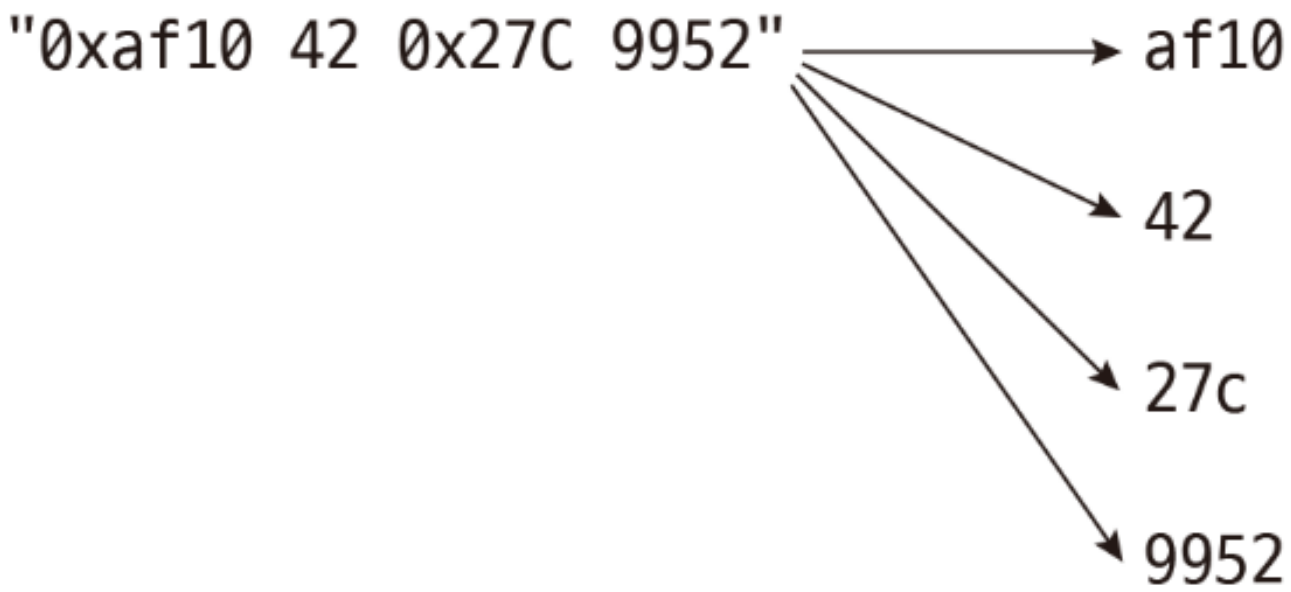
- 특정 진법으로 표기된 문자열을 정수로 변환하기
  - strtol(문자열, 끝포인터, 진법);
    - long strtol(char const *_String, char **_EndPtr, int _Radix);
    - 성공하면 변환된 정수를 반환, 실패하면 0을 반환

```
#include <stdio.h>
#include <stdlib.h>    // strtol 함수가 선언된 헤더 파일

int main()
{
    char *s1 = "0xaf10";    // "0xaf10"은 문자열
    int num1;

    num1 = strtol(s1, NULL, 16);    // 16진법으로 표기된 문자열을 정수로 변환

    printf("%x %d\n", num1, num1);  // af10 44816

    return 0;
}
```

```
af10 44816
```

문자열과 숫자를 서로 변환하기

- 특정 진법으로 표기된 문자열을 정수로 변환하기
  - 여러 개의 정수로 된 문자열을 변환



문자열과 숫자를 서로 변환하기

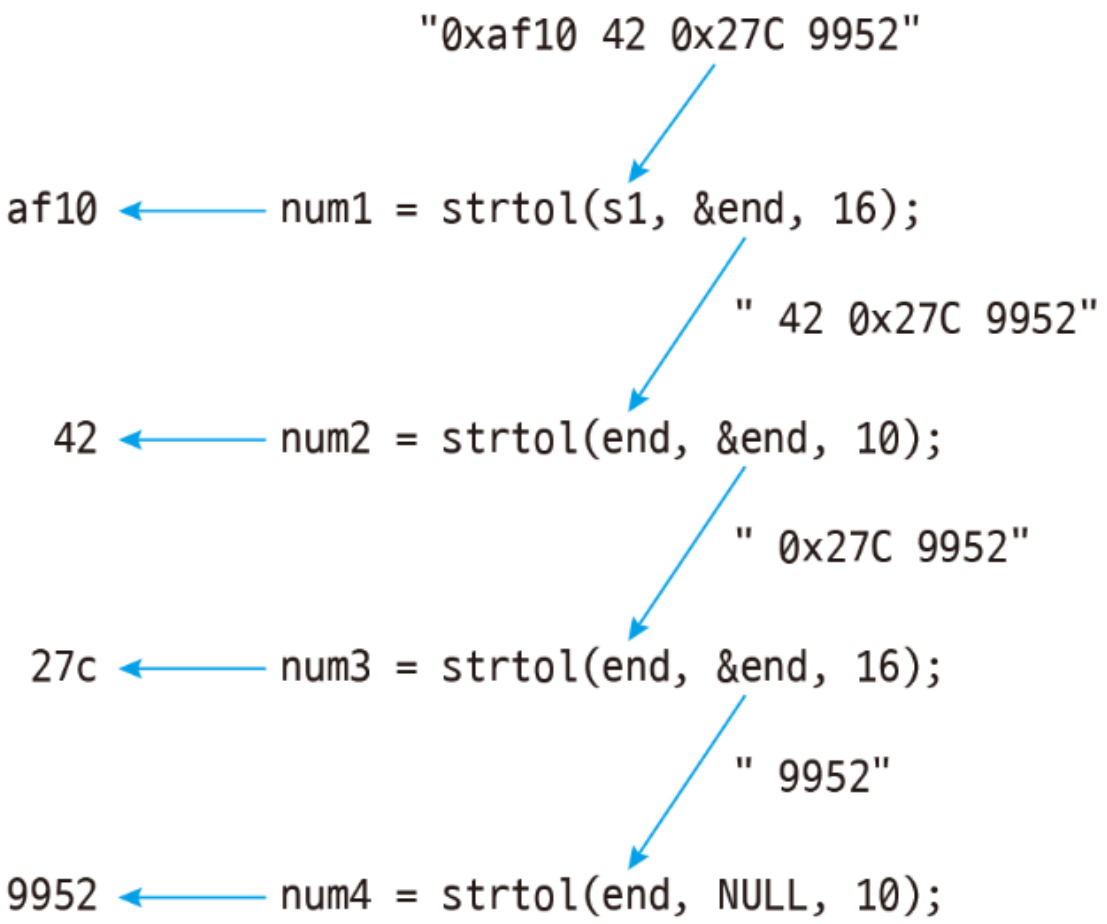
▶ 특정 진법으로 표기된 문자열을 정수로 변환하기

```
#include <stdio.h>
#include <stdlib.h> // strtol 함수가 선언된 헤더 파일
int main(){
    char *s1 = "0xaf10 42 0x27C 9952"; // "0xaf10 42 0x27C 9952"는 문자열
    int num1;
    int num2;
    int num3;
    int num4;
    char *end;    // 이전 숫자의 끝 부분을 저장하기 위한 포인터
    num1 = strtol(s1, &end, 16);     // 16진법으로 표기된 문자열을 정수로 변환
    num2 = strtol(end, &end, 10);    // 10진법으로 표기된 문자열을 정수로 변환
    num3 = strtol(end, &end, 16);    // 16진법으로 표기된 문자열을 정수로 변환
    num4 = strtol(end, NULL, 10);    // 10진법으로 표기된 문자열을 정수로 변환
    printf("%x\n", num1);    // af10
    printf("%d\n", num2);    // 42
    printf("%X\n", num3);    // 27C
    printf("%d\n", num4);    // 9952
    return 0;
}
```

```
af10
42
27C
9952
```

문자열과 숫자를 서로 변환하기

- 특정 진법으로 표기된 문자열을 정수로 변환하기
  - strtol로 여러 개의 숫자로 된 문자열을 변환하기



문자열과 숫자를 서로 변환하기

- 문자열을 실수로 변환하기
  - atof(문자열);
    - double atof(char const *_String);
    - 성공하면 변환된 실수를 반환, 실패하면 0을 반환

```
#include <stdio.h>
#include <stdlib.h>    // atof 함수가 선언된 헤더 파일

int main(){
    char *s1 = "35.283672";  // "35.283672"은 문자열
    float num1;

    num1 = atof(s1);         // 문자열을 실수로 변환하여 num1에 할당

    printf("%f\n", num1);    // 35.283672

    return 0;
}
```

```
35.283672
```

문자열과 숫자를 서로 변환하기

문자열을 실수로 변환하기

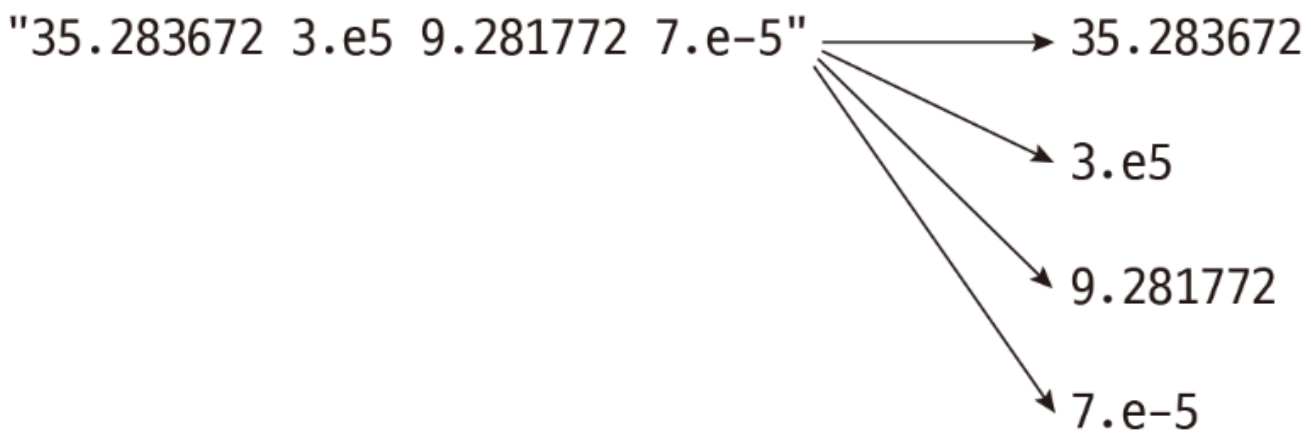
다음과 같이 알파벳 e를 사용하여 지수 표기법으로 된 문자열도 실수로 바꿀 수 있음

```
#include <stdio.h>
#include <stdlib.h>    // atof 함수가 선언된 헤더 파일

int main(){
    char *s1 = "3.e5";   // "3.e5"는 문자열
    float num1;

    num1 = atof(s1);     // 문자열을 실수로 변환하여 num1에 할당

    printf("%e %f\n", num1, num1);    // 3.000000e+05 300000.000000

    return 0;
}
```

```
3.000000e+05 300000.000000
```

문자열과 숫자를 서로 변환하기

- 문자열을 실수로 변환하기
  - 여러 개의 실수로 된 문자열 변환



문자열과 숫자를 서로 변환하기

- 문자열을 실수로 변환하기
  - strtof(문자열, 끝포인터);
    - float strtof(char const* _String, char** _EndPtr);
    - 성공하면 변환된 실수를 반환, 실패하면 0을 반환

```
#include <stdio.h>
#include <stdlib.h>    // strtof 함수가 선언된 헤더 파일
int main(){
    char *s1 = "35.283672 3.e5 9.281772 7.e-5";    // "35.283672 3.e5f 9.2817721 7.e-5f"는 문자열
    float num1, num2, num3, num4;
    char *end;    // 이전 숫자의 끝 부분을 저장하기 위한 포인터
    num1 = strtof(s1, &end);     // 문자열을 실수로 변환
    num2 = strtof(end, &end);    // 문자열을 실수로 변환
    num3 = strtof(end, &end);    // 문자열을 실수로 변환
    num4 = strtof(end, NULL);    // 문자열을 실수로 변환
    printf("%f\n", num1);    // 35.283672
    printf("%e\n", num2);    // 3.000000e+05
    printf("%f\n", num3);    // 9.281772
    printf("%e\n", num4);    // 7.000000e-05
    return 0;
}
```

```
35.283672
3.000000e+05
9.281772
7.000000e-05
```

문자열과 숫자를 서로 변환하기

- 정수를 문자열로 변환하기
  - sprintf(문자열, "%d", 정수);
  - sprintf(문자열, "%x", 정수);
  - sprintf(문자열, "%X", 정수);

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일

int main()
{
    char s1[10];        // 변환한 문자열을 저장할 배열
    int num1 = 283;     // 283은 정수

    sprintf(s1, "%d", num1);     // %d를 지정하여 정수를 문자열로 저장

    printf("%s\n", s1);          // 283

    return 0;
}
```

```
283
```

문자열과 숫자를 서로 변환하기

정수를 문자열로 변환하기

정수를 16진법으로 표기된 문자열로 변환하려면 어떻게 해야 할까?

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일

int main()
{
    char s1[10];       // 변환한 문자열을 저장할 배열
    int num1 = 283;    // 283은 정수

    sprintf(s1, "0x%x", num1);     // %x를 지정하여 정수를 16진법으로 표기된 문자열로 저장
                                   // 16진수라는 것을 나타내기 위해 앞에 0x를 붙임

    printf("%s\n", s1);            // 0x11b

    return 0;
}
```

```
0x11b
```

문자열과 숫자를 서로 변환하기

실수를 문자열로 변환하기

sprintf(문자열, "%f", 실수);

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일

int main(){
    char s1[10];                // 변환한 문자열을 저장할 배열
    float num1 = 38.972340f;    // 38.972340은 실수

    sprintf(s1, "%f", num1);    // %f를 지정하여 실수를 문자열로 저장

    printf("%s\n", s1);         // 38.972340

    return 0;
}
```

```
38.972340
```

문자열과 숫자를 서로 변환하기

실수를 문자열로 변환하기

sprintf함수에 서식 지정자로 %e를 지정하면 실수를 지수 형태의 문자열로 변환 가능

```
#define _CRT_SECURE_NO_WARNINGS    // sprintf 보안 경고로 인한 컴파일 에러 방지
#include <stdio.h>     // sprintf 함수가 선언된 헤더 파일

int main(){
    char s1[20];                // 변환한 문자열을 저장할 배열
    float num1 = 38.972340f;    // 38.972340은 실수

    sprintf(s1, "%e", num1);    // %e를 지정하여 실수를 지수 표기법으로 된 문자열로 저장

    printf("%s\n", s1);         // 3.897234e+01

    return 0;
}
```

```
3.897234e+01
```

실습문제 17

문자열을 10진 정수로 변환하기

다음 소스 코드를 완성하여 정수 20972가 출력되게 만드세요.

```
#include <stdio.h>
①_____________________

int main()
{
    char *s1 = "20972";
    int num1;

    ②_____________________

    printf("%d\n", num1);

    return 0;
}
```

```
20972
```

실습문제 18

문자열을 16진 정수로 변환하기

다음 소스 코드를 완성하여 16진 정수 0x1FACEFEE가 출력되게 만드세요.

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

int main(){
    char *s1 = "0x1facefee";
    int num1;

    ①_____________________________

    printf(②_________, num1);

    return 0;
}
```

```
0x1FACEFEE
```

실습문제 19

여러 개의 실수로 된 문자열을 실수로 변환하기

다음 소스 코드를 완성하여 실수 97.527824가 출력되도록 작성하시오.

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

int main()
{
    char *s1 = ①__________;
    float num1;

    num1 = atof(s1);

    printf("%f\n", ②_________);

    return 0;
}
```

```
97.527824
```

실습문제 20

▶ 여러 개의 실수로 된 문자열을 실수로 변환하기

▶ 다음 소스 코드를 완성하여 실수 29.977213, 4528.112305가 각 줄 마다 출력되게 만드시오.

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

int main(){
    char *s1 = "29.977213 4528.112305";
    float num1;
    float num2;
    char *end;

    ①_____________________
    ②_____________________

    printf("%f\n", num1);
    printf("%f\n", num2);

    return 0;
}
```

```
29.977213
4528.112305
```

실습문제 21

숫자를 문자열로 변환하기

다음 소스 코드를 완성하여 문자열 "98.415237 0x3fce1920"이 출력되게 만드세요.

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

int main()
{
    ①_____________
    float num1 = 98.415237f;
    int num2 = 0x3fce1920;

    ②_____________________________

    printf("%s\n", s1);

    return 0;
}
```

```
98.415237 0x3fce1920
```

연습문제 1

- 문자열을 교환하는 프로그램 작성
  - 함수를 사용하여 두 개의 문자배열에 저장된 문자열을 서로 바꾸는 프로그램을 작성
    - char str1[20] = “apple”;
    - char str2[20] = “banana”;
  - 함수는 문자열을 바꾸는 작업만 수행하며 문자열의 출력은 main 함수에서 수행
    - swap_string(str1, str2); //함수를 호출하여 문자열을 교환
  - 실행 결과

```
str1 : banana
str2 : apple
```

연습문제 2

- 두 개의 문자 배열에 저장된 문자열 중에서 길이가 긴 문자열을 출력
  - 배열의 선언과 초기화는 아래와 같음
    - char str1[20] = “Long time no see.”;
    - char str2[20] = “What’s up?”;
  - 실행 결과

```
Long time no see.
```

연습문제 3

- 키보드로부터 성과 이름을 따로 입력 받아서 하나의 문자열로 붙여서 출력하는 프로그램 작성
  - 배열은 다음과 같이 세 개를 선언하여 작성
    - char last_name[20]; //성을 입력할 배열
    - char first_name[20]; //이름을 입력할 배열
    - char full_name[20]; //성과 이름을 모두 저장할 배열
  - 실행 결과

```
성을 입력하시오. : 이
이름을 입력하시오. : 순신
성을 포함한 이름은 이 순신 입니다.
```

연습문제 4

- strcmp 함수와 동일한 역할을 하는 my_strcmp 함수를 작성하시오.
  - main()에서 입력 받은 두 개의 문자열을 my_strcmp함수를 통하여 비교하고 출력
  - quit가 입력될 때까지 실행을 반복

```
int my_strcmp(const char *str1, const char *str2);
```

연습문제 5

- strcat 함수와 동일한 역할을 수행하는 my_starcat 함수를 작성하시오.
  - dest가 가리키는 공간은 src 문자열을 이어붙이기에 충분하다고 가정
  - main()에서 두 개의 문자열을 입력받은 후, 이 함수를 호출하여 결과를 출력
  - dest 문자열에 quit가 입력될 때까지 실행을 반복
  - main()에서 루프를 실행하며 if(strcmp(dest, “quit”)==0)break;에서 루프를 빠져나옴

```
char* my_strcat(char *dest, char *src);
```

연습문제 6

- strcpy 함수와 동일한 동작을 수행하는 my_strcpy 함수를 구현하시오.
  - main()에서 gets로 입력을 받고 src를 채운 뒤 이 함수를 호출하고 main으로 되돌아와서 puts로 dest배열을 출력
  - 단, dest 배열의 크기는 src 배열과 동일한 크기라고 가정
  - index는 현재 복사해야 할 문자의 인덱스를 의미
  - 따라서 src[index]가 '₩0'이면 문자열의 끝이므로 그것을 복사한 후 곧바로 빠져나와야 하고, 그렇지 않다면 복사 후에 계속 index를 증가시켜서 재귀호출을 수행

```
char * my_strcpy(char * dest, const char * src, int index);
```

Q & A