



PART 01. C 언어 입문

CHAPTER 02. C 프로그램의 구성

Section 2.1

C 프로그램의 구성 요소

1. C 프로그램의 구성 요소

■ C 프로그램의 기본 구조

- **헤더 함수:** 프로그램에서 사용할 표준 라이브러리나 다른 라이브러리 포함
- **함수 정의:** 하나 이상의 함수로 구성
- **변수 선언:** 프로그램에서 사용될 변수 선언
- **실행문:** 프로그램이 작업하는 수행 과정과 로직으로 구성된 코드 영역

C 언어의 기본 구조

helloworld.c

```
01 #include <stdio.h>           // 헤더 파일 포함
02                               전처리기
03 int main(void)                프로그램 시작: main() 함수
04 {
05
06     printf("Hello World!");    // 프로그램 본문
07                               반환문
08     return 0;                 // 반환
09 }
```

프로그램 끝

Hello World!

1. C 프로그램의 구성 요소

■ 전처리기

```
01 #include <stdio.h>           // 헤더 파일 포함
```

- 대표적인 전처리 지시문으로, 코드를 컴파일 과정에 맞춰 조정하고 변환하는 역할
- 표준 입출력 함수를 사용하는 데 필요
- **#include**
 - 라이브러리 함수, 매크로 정의, 전역변수 등의 정보가 담김 파일을 포함할 때 사용
- **<stdio.h>**
 - 헤더 파일
 - 별도의 파일에 정의하고 필요할 때마다 사용

1. C 프로그램의 구성 요소

■ main() 함수

```
03 int main(void)
```

- C 프로그램의 실질적인 시작점
- 운영체제가 프로그램을 실행할 때 자동으로 호출
- **함수:** 수학에서의 함수와 비슷한 개념으로, 특정한 작업을 수행하도록 설계된 독립적인 코드 블록
- 함수는 7, 8장에서 자세히 설명

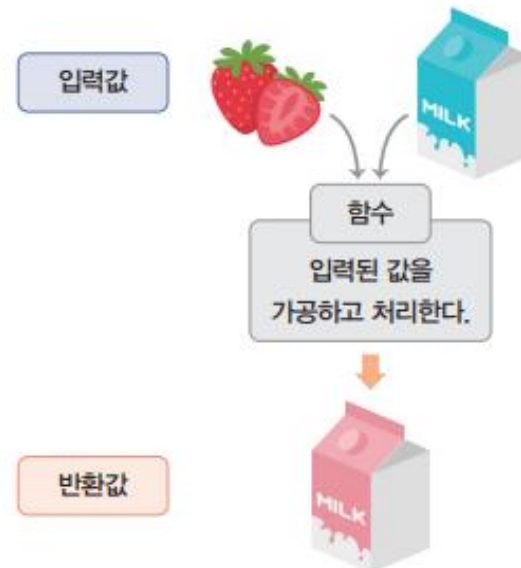


그림 2-1 프로그램에서 함수의 개념

1. C 프로그램의 구성 요소

■ main() 함수

▪ main() 함수의 구조

- 열린 중괄호({) 기호로 함수가 시작 되어 다음 행부터 실행할 코드를 작성하고, 닫힌 중괄호(}) 기호로 함수를 끝맺음
- 중괄호 내부에는 앞으로 배우게 될 C 언어의 모든 문법과 알고리즘을 활용한 코드 삽입

함수의 헤더

```
int main(void) // 함수명, 정의
```

함수의 몸체

```
{  
    // 실행 코드 작성  
  
    return 0;  
}
```

그림 2-2 main() 함수의 구조

1. C 프로그램의 구성 요소

■ main() 함수

■ 소스 코드

```
06 printf("Hello World!");
```

• printf()

- 형식에 맞춰 데이터를 출력하는 함수
- <stdio.h>에 의해 표준 출력 함수로 정의

■ return 문

```
08 return 0;
```

- 모든 함수가 작업을 다 끝내고 나면 실행을 즉시 중단하고, 함수가 호출된 곳으로 값을 반환할 때 사용
- 함수에서 값의 반환 및 종료를 의미하므로 중요한 키워드

1. C 프로그램의 구성 요소

■ main() 함수

▪ main() 함수와 return 문

- main() 함수는 운영체제의 호출로 실행되고, 마지막에 작성되는 return 문은 프로그램을 종료할 때 사용
- 'return 0'은 프로그램이 성공적으로 실행되어 정상적으로 종료됨을 의미
- 다른 값이 반환된다면 오류 또는 특정 종료 조건을 의미

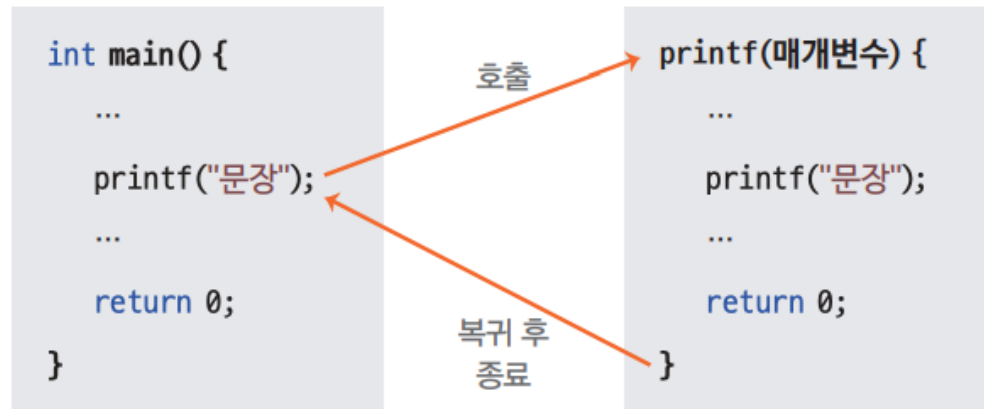


그림 2-3 main() 함수에서의 return 문 수행 과정

1. C 프로그램의 구성 요소

■ main() 함수

▪ 세미콜론(;)

| Syntax | return 문

return 0; 세미콜론

- C 언어로 프로그래밍을 할 때는 모든 문장의 맨 마지막에 세미콜론(;)을 작성
- 문장의 종결을 나타내는 기호
- 한 문장이 끝나고 다음 문장이 시작된다는 것을 컴파일러 에 알려주는 역할
- 세미콜론을 사용하지 않거나 잘못 사용하면 컴파일러가 오류를 발생하므로 주의

1. C 프로그램의 구성 요소

■ 주석

- 코드의 이해를 돕는 설명
- 컴파일러가 무시하기 때문에 프로그램 실행에 직접적인 영향 없음

| Syntax | 주석

// 한 줄 주석: 코드 한 줄이 주석으로 처리된다.

/*

주석 시작

여러 줄 주석: 여러 줄로 작성한 문장이 모두
주석으로 처리된다.

*/

주석 끝

1. C 프로그램의 구성 요소

예제 2-1 주석 사용하기

```
01 #include <stdio.h>
02
03 int main()
04 {
05     // 변수 선언
06     int a = 5;
07     int b = 10;
08
09     /*
10         여기서는 두 변수 a, b의 합을 계산하고
11         결과를 화면에 출력합니다.
12         이 부분은 여러 줄 주석으로 처리되었습니다.
13     */
14
15     // 아래 코드는 현재 비활성화되어 있습니다.
16     // printf("a + b = %d\n", (a+b));
17
18     return 0;
19 }
```

1. C 프로그램의 구성 요소

하나 더 알기 주석의 또 다른 용도

주석의 예

comment.c

```
01  /*****
02  * 파일명: comment.c
03  * 프로그램 설명: 주석에 대한 예시입니다.
04  * 작성일: 2024.01.28
05  * 작성자: 문석재
06  * 버전: v0.1
07  * 설명: comment.c와 같은 코드는 실행 시 결과가 나오지 않는다.
08  *****/
```

Section 2.2

표준 출력 함수

2. 표준 출력 함수

■ printf() 함수

- 모니터에 데이터를 출력할 때 사용하는 표준 출력 함수
- <stdio.h> 헤더 파일에 정의되어 있으며, 매우 다양한 형식으로 사용 가능



그림 2-4 표준 출력 버퍼를 이용한 `printf()` 함수 수행 과정

2. 표준 출력 함수

■ printf() 함수

■ 문자열 출력

| Syntax | 문자열 출력

```
printf("출력할 문자열 입력");
```

```
printf("Hello C language");
```

■ 정수와 실수 출력

| Syntax | 정수, 실수 출력

```
printf("%d", 정숫값);
```

정수 출력 형식 지정자

```
printf("%f", 실숫값);
```

실수 출력 형식 지정자

```
printf("%d", 100); // 정수 100 출력
```

```
printf("%f", 2.5); // 실수 2.5 출력(실행하면 2.500000이 출력됨)
```

2. 표준 출력 함수

■ printf() 함수

▪ 정수와 실수 출력

• 형식 지정자

- printf(), scanf()와 같은 입출력 함수에서 쓰이는 특수 문자
- 출력 데이터의 형식을 지정하는 데 사용

100을 %d 위치에 출력
↓
printf("%d", 100);

2.5를 %f 위치에 출력
↓
printf("%f", 2.5);

표 2-1 출력 형식 지정자

형식 지정자	설명	예	출력 결과
%d	정수를 출력한다.	printf("%d", 10);	10
%f	실수를 출력한다.	printf("%f", 3.14);	3.140000
%c	단일 문자를 출력한다.	printf("%c", 'K');	K
%s	문자열을 출력한다.	printf("%s", "Master");	Master

그림 2-5 printf() 함수에서 정수와 실수가 출력되는 방식

2. 표준 출력 함수

예제 2-2 printf() 함수를 사용하여 정수, 실수, 문자, 문자열 출력하기

```
01 #include <stdio.h>
02
03 int main()
04 {
05     printf("정수: %d\n", 100);           // 정수 100 출력
06     printf("실수: %f\n", 10.123);       // 실수 10.123 출력
07     printf("문자: %c\n", 'A');          // 문자 A 출력
08     printf("문자열: %s\n", "Hello, C!"); // 문자열 Hello, C 출력
09
10     return 0;
11 }
```

문자열을 출력한 뒤 줄 바꿈

```
정수: 100
실수: 10.123000
문자: A
문자열: Hello, C!
```

2. 표준 출력 함수

하나 더 알기 이스케이프 시퀀스

표 2-2 이스케이프 시퀀스의 종류

기호	의미	예
\n	줄 바꿈	<code>printf("Hello,\nworld!\n");</code>
\t	수평 탭	<code>printf("Hello,\tworld!\n");</code>
\"	이중 인용 부호	<code>printf("She said, \"Hello!\"\n");</code>
\'	작은따옴표	<code>printf("I\'m here.\n");</code>
\\	역슬래시	<code>printf("directory is c:\\Program Files\\.\\n");</code>
\b	백스페이스	<code>printf("Back\bSpace\n");</code>
\r	캐리지 리턴	<code>printf("Carriage\rReturn\n");</code>
\f	페이지 넘기기	<code>printf("Page\fFeed\n");</code>
\v	수직 탭	<code>printf("Vertical\vTab\n");</code>
\a	벨 소리	<code>printf("Be\aAlert\n");</code>
\?	물음표	<code>printf("What\?!n");</code>

SELF STUDY

이스케이프 시퀀스의 예시 코드를 참고하여 C 코드를 작성하고 결과를 확인해보자.


2. 표준 출력 함수

■ printf() 함수

■ 변수 출력

| Syntax | 변수 출력

printf("%d", 변수);



정수 출력 형식 지정자
변수 입력

```
printf("%d", total);    // 변수 total에 저장된 값 '89' 출력
```

```
printf("%d %f", total, average);    // 변수 total과 average 출력
```

2. 표준 출력 함수

■ printf() 함수

■ 변수 출력

- 형식 지정자와 자료형이 반드시 일치해야 하고 출력되는 위치도 동일

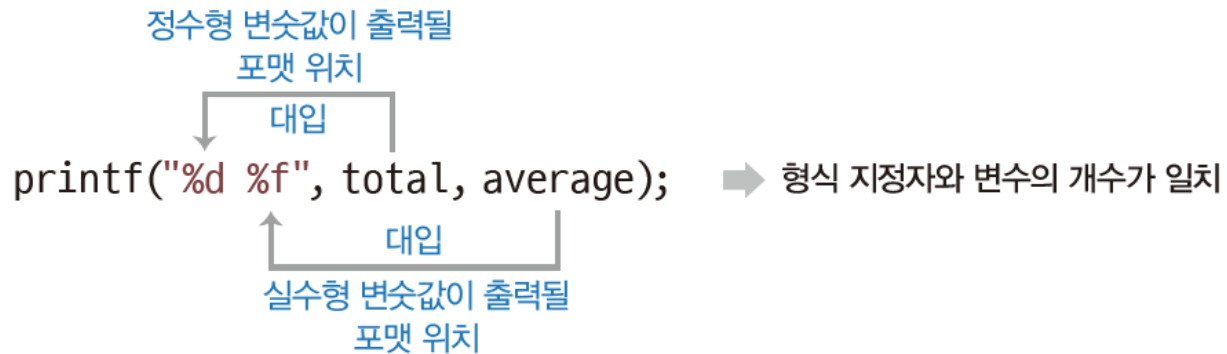


그림 2-6 printf() 함수에서 변수 total, average가 출력되는 방식

2. 표준 출력 함수

예제 2-3 printf() 함수를 사용하여 변수 출력하기

```
01 #include <stdio.h>
02
03 int main()
04 {
05     int total = 89;           // 총점
06     float average = 95.2f;    // 평균
07
08     printf("총점: %d\n", total);    // 총점 출력
09     printf("평균: %.2f\n", average); // 평균 출력
10     printf("총점: %d, 평균: %.2f\n", total, average);
11
12     return 0;
13 }
```

정수형 변수를 선언하고 변수값 89 저장

부동 소수점(float) 변수를 선언하고
변수값 95.2f 저장

%d 형식 지정자는 변수 total의 정수값을
문자열에 포함함(위치 지정)

%.2f 형식 지정자는 변수 average의
실수값을 소수점 아래 둘째 자리까지
출력

총점: 89

평균: 95.20

총점: 89, 평균 95.20

2. 표준 출력 함수

■ printf() 함수의 변환 서식

■ 정수 출력 형식 지정자

- 정수 출력 형식 지정자 %d로 정수 98765를 출력하는 코드와 실행 결과

```
printf("%d\n", 98765);  
printf("%-d\n", 98765);  
printf("%10d\n", 98765);  
printf("%-10d\n", 98765);
```

```
98765          // 일반 출력  
98765          // 왼쪽 정렬  
____98765      // 너비 10, 오른쪽 정렬  
98765____      // 너비 10, 왼쪽 정렬
```

2. 표준 출력 함수

■ printf() 함수의 변환 서식

■ 실수 출력 형식 지정자

- 실수 출력 형식 지정자 %f로 실숫값을 출력하는 코드와 실행 결과

```
printf("%f\n", 95.12345);  
printf("%.2f\n", 95.12345);  
printf("%10f\n", 95.12345);  
printf("%-10f\n", 95.12345);  
printf("%10.2f\n", 95.12645);  
printf("%-10.2f\n", 95.12645);
```

```
95.123450    // 소수점 아래 여섯 자리  
95.12        // 소수점 아래 두 자리(반올림하지 않음)  
_95.123450   // 너비 10, 오른쪽 정렬  
95.123450_   // 너비 10, 왼쪽 정렬  
____95.13    // 소수점 아래 두 자리(반올림함)  
95.13____    // 소수점 아래 두 자리(반올림함)
```

2. 표준 출력 함수

■ printf() 함수의 변환 서식

표 2-3 printf() 함수의 변환 형식 지정자

float f = 123.45f, int x = 52342			
형식	설명	예	출력 결과
숫자.숫자	소수점 아래 자릿수를 의미한다.	printf("%.2f", f);	123.45
-	서식 문자를 왼쪽부터 채운다.	printf("%-10d", x);	52342 _ _ _ _ _
.	최대 출력 너비와 문자 수를 구분한다.	printf("%5.2s", "abcd"); printf("%.2f", f);	_ _ _ ab 123.45
	long형으로 출력한다.	printf("%ld", x);	long형 출력

2. 표준 출력 함수

예제 2-4 printf() 함수 변환 형식 지정자 사용하기 1

```
01 #include <stdio.h>
02
03 int main()
04 {
05     printf("%d %c %f\n", 10, 'z', 11.23);    // 각 필드에 공백 삽입
06     printf("%d\t%c\t%f\n", 10, 'z', 14.23);  // 탭(\t) 문자, 줄 바꿈(\n)
07     printf("슬램덩크 강백호는 %d 등번호\n", 10);  \t 탭만큼 너비 이동
08     printf("등번호 %6d\n", 10);
09     printf("등번호 %-6d\n", 10);
10     printf("강백호의 올해 필드 성공률이 %8.2f으로 오른 것으로 보인다.\n", 11.23);
11     printf("강백호의 올해 자유투 성공률이 %08.2f으로 오른 것으로 보인다.\n", 14.23);
12
13     return 0;
14 }
```

```
10 z 11.230000
10      z      14.230000
슬램덩크 강백호는 10 등번호
등번호      10
등번호 10
강백호의 올해 필드 성공률이      11.23으로 오른 것으로 보인다.
강백호의 올해 자유투 성공률이 00014.23으로 오른 것으로 보인다.
```

2. 표준 출력 함수

예제 2-5 printf() 함수 변환 형식 지정자 사용하기 2

```
01 #include <stdio.h>
02
03 int main()
04 {
05     int a = 1234;
06     // 최소 너비 8칸 출력
07     printf("[%d]\n", a);      // 정수 출력
08     printf("[%8d]\n", a);    // 오른쪽 정렬
09     printf("[%8d]\n", a);    // 왼쪽 정렬
10     printf("[%+8d]\n", a);   // 오른쪽 정렬, 부호 표시
11     printf("[%08d]\n", a);   // 공백 채움
12     printf("[%+08d]\n", a);  // 부호 표시, 공백 채움
13
14     char b = 'A';
15     double c = 3.14159;
16
17     printf("[%c]\n", b);     // 문자 출력
18     printf("[%8c]\n", b);    // 오른쪽 정렬
19     printf("[%8c]\n", b);    // 왼쪽 정렬
20     printf("[%f]\n", c);     // 소수점 아래 여섯 자리 실수 출력
21     printf("[%8.3f]\n", c);  // 오른쪽 정렬, 소수점 아래 세 자리 출력
22     printf("[%8.3f]\n", c);  // 왼쪽 정렬, 소수점 아래 세 자리 출력
23     printf("[%+8.3f]\n", c); // 부호 표시, 소수점 아래 세 자리 출력
24     printf("[%08.3f]\n", c); // 공백 채움, 소수점 아래 세 자리 출력
25     printf("[%+08.3f]\n", c); // 부호 표시, 공백 채움, 소수점 아래 세 자리 출력
26
27 }
```

```
[1234]
[ 1234]
[1234 ]
[ +1234]
[00001234]
[+0001234]
[A]
[  A]
[A ]
[3.141590]
[ 3.142]
[3.142 ]
[ +3.142]
[0003.142]
[+003.142]
```

2. 표준 출력 함수

LAB 형식 지정자 필드 옵션 지정하기

문제 해결

LAB_2-1.c

```
01 #include <stdio.h>
02
03 int main()
04 {
05     printf("### 형식 지정자 적용 ###\n");
06     printf("12345678901234567890\n"); // 숫자로 구성된 문자열
07     printf("%10c\n", 'A'); // 문자, 오른쪽 정렬
08     printf("%10d\n", 128); // 정수, 너비 10, 오른쪽 정렬
09     printf("%10lf\n", 3.1415926); // 실수, 너비 10, 오른쪽 정렬
10     printf("%10.3lf\n", 3.1415926); // 소수점 아래 셋째 자리까지 출력, 오른쪽 정렬
11     printf("%10.4s\n", "Love is"); // 문자열 네 자리 출력, 오른쪽 정렬
12
13     return 0;
14 }
```

줄 바꿈

%c는 문자를 출력하는 형식 지정자, 10은 출력 너비

%d는 정수를 출력하는 형식 지정자, 10은 출력 너비

%s는 문자열을 출력하는 형식 지정자, 10은 출력 너비, 4는 최대 4개의 문자 출력

%lf는 double형 변수를 출력하는 형식 지정자, 10은 출력 너비

Section 2.3

표준 입력 함수

3. 표준 입력 함수

■ scanf() 함수

- 키보드로 다양한 값을 입력받을 때 사용하는 함수
- 입력 함수 중에서도 가장 기본적인 함수이며, 헤더 파일에 정의



그림 2-7 표준 입력 버퍼를 이용한 scanf() 함수 수행 과정

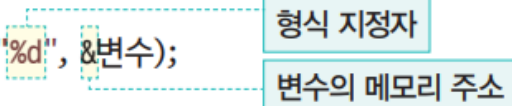
3. 표준 입력 함수

■ 키보드 입력

- 정수를 입력받는 scanf() 함수의 기본 구조

| Syntax | 정수, 실수 입력

```
scanf("%d", &변수);
```



형식 지정자

변수의 메모리 주소

- 정수형 변수 num에 정수 23을 입력받는 코드

```
scanf("%d", &num);
```

3. 표준 입력 함수

하나 더 알기

주소 연산자 &

변수는 모두 메모리에 저장된다. 그런데 우리는 실제 메모리 주소를 알 수가 없으므로 C 언어에서는 변수명 앞에 주소 연산자 &를 붙여서 변수의 메모리 주소를 나타낸다.

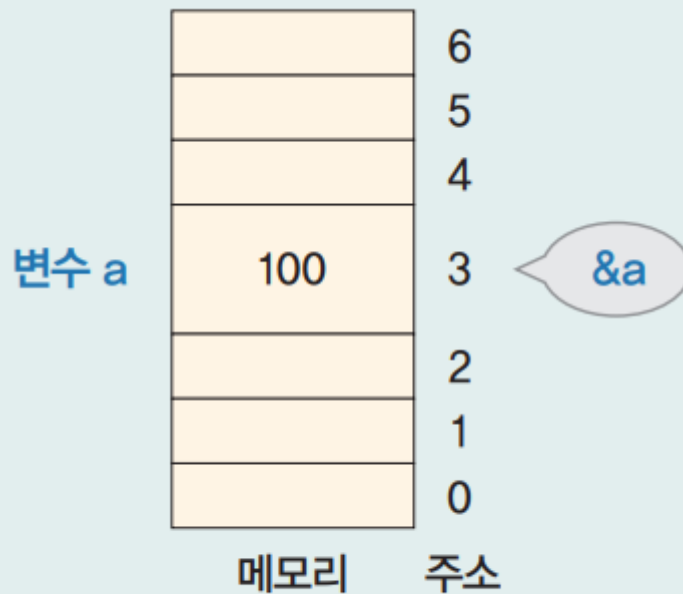


그림 2-8 메모리 변수 주소

3. 표준 입력 함수

■ 키보드 입력

▪ 정수와 실수 입력

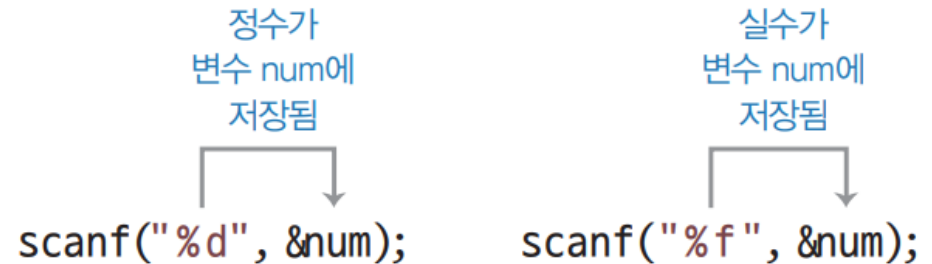


표 2-4 입력 형식 지정자

형식 지정자	설명	예
%d	정수(int)를 입력한다.	<pre>int num; scanf("%d", &num);</pre>
%f	실수(float)를 입력한다.	<pre>float pi; scanf("%f", &pi);</pre>
%lf	실수(double)를 입력한다.	<pre>double pi; scanf("%lf", &pi);</pre>
%c	단일 문자(char)를 입력한다.	<pre>char ch; scanf("%c", &ch);</pre>
%s	문자열(string)을 입력한다.	<pre>char str[10]; scanf("%s", str);</pre>

그림 2-9 scanf() 함수에서 정수와 실수가 입력되는 방식

3. 표준 입력 함수

■ 키보드 입력

▪ 문자와 문자열 입력

- 사용자가 1개의 문자 또는 최대 10개의 문자로 구성된 문자열을 입력하는 코드

```
char ch;           // 문자 1개를 저장할 변수 선언
scanf("%c", &ch);

char str[10];      // 문자 10개를 저장할 변수 선언(배열)
scanf("%s", str);  // 사용자로부터 문자열을 입력받아 변수 str에 저장
```

- 배열 변수 str에는 문자열 끝에 자동으로 널 (null) 종료 문자('\0')가 추가

3. 표준 입력 함수

예제 2-6 scanf() 함수를 사용하여 정수, 실수, 문자, 문자열 입력하기

```
01 #include <stdio.h>
02
03 int main()
04 {
05     char ch;      // 문자 1개를 저장할 변수 선언
06     char str[10];  // 문자 10개를 저장할 문자열(배열) 변수 선언
07     int i;         // 정수를 저장하기 위한 변수 i 선언
08     float f;       // 실수를 저장하기 위한 변수 f 선언
09
10     printf("문자를 입력하세요: ");
11     scanf("%c", &ch);
12
13     printf("문자열을 입력하세요: ");
14     scanf("%s", str);
15
16     printf("입력받은 문자: %c\n", ch);    // 입력받은 문자 ch 출력
17     printf("입력받은 문자열: %s\n", str); // 입력받은 문자열 str 출력
18
19     printf("정수를 입력하세요: ");
20     scanf("%d", &i);
21     // %d는 정수를 입력받는 형식 지정자, &i는 변수 i의 메모리 주소를 의미
22
23     printf("실수를 입력하세요: ");
24     scanf("%f", &f);
25     // %f는 실수를 입력받는 형식 지정자, &f는 변수 f의 메모리 주소를 의미
26
27     printf("입력받은 정수: %d\n", i);    // 입력받은 정수 i 출력
28     printf("입력받은 실수: %f\n", f);    // 입력받은 실수 f 출력
29
30     return 0; // 정상적인 프로그램 종료를 나타내며, main() 함수에서 0을 반환
31 }
```

문자를 입력하세요: H 키보드로 값을 입력한 후 [Enter+J]

문자열을 입력하세요: Hello

입력받은 문자: H

입력받은 문자열: Hello

정수를 입력하세요: 63 키보드로 값을 입력한 후 [Enter+J]

실수를 입력하세요: 8.3

입력받은 정수: 63

입력받은 실수: 8.300000

3. 표준 입력 함수

■ 키보드 입력

■ 여러 개의 변수 입력

- 프로그램을 작성하다 보면 scanf() 함수를 사용하여 1개 이상의 변수에 데이터를 동시에 입력받는 경우
- 들어 정수형 변수 a, 문자형 변수 op, 정수형 변수 b를 순차적으로 입력받으려면 다음과 같이 작성

| Syntax | 1개 이상의 변수를 순차적으로 입력

```
scanf("%d %c %d", &a, &op, &b);
```

형식 지정자

변수 a, op, b의 메모리 주소

3. 표준 입력 함수

예제 2-7 scanf() 함수를 사용하여 덧셈 프로그램 만들기

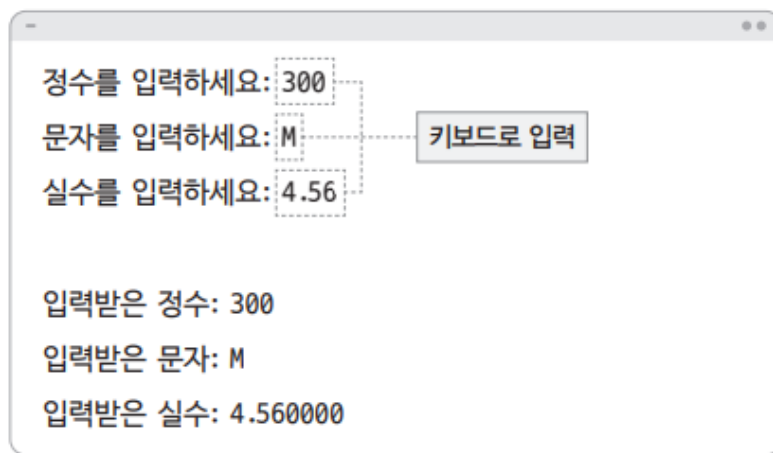
```
01 #pragma warning(disable: 4996) // 컴파일러 경고 비활성화
02 #include <stdio.h> // 표준 라이브러리 헤더 포함
03 // scanf() 함수 보안 경고 회피
04 int main()
05 {
06     int a, b;
07     char op;
08
09     printf("두 정수 덧셈 (ex 10 + 20): ");
10     scanf("%d %c %d", &a, &op, &b);
11     // 문자를 입력받는 형식 지정자, 연산자 입력
12     printf("%d %c %d = %d\n", a, op, b, a + b);
13
14     return 0;
15 }
```

```
두 정수 덧셈 (ex 10 + 20): 10 + 20
10 + 20 = 30
```

3. 표준 입력 함수

LAB 변숫값 입력받고 출력하기

■ 실행 결과



A screenshot of a C program execution window. It shows three input prompts: '정수를 입력하세요:' with '300', '문자를 입력하세요:' with 'M', and '실수를 입력하세요:' with '4.56'. A box labeled '키보드로 입력' (Input from keyboard) is connected to these inputs. Below the inputs, the program outputs: '입력받은 정수: 300', '입력받은 문자: M', and '입력받은 실수: 4.560000'.

```
정수를 입력하세요: 300
문자를 입력하세요: M
실수를 입력하세요: 4.56

입력받은 정수: 300
입력받은 문자: M
입력받은 실수: 4.560000
```



■ 힌트

- 문법: `scanf("형식 문자열", 변수들의 주소);`
- 형식 문자열: 입력받을 자료형을 지정하는 형식 지정자를 포함한다.
- 변수들의 주소: 입력받은 데이터를 저장하도록 변수 앞에 `&` 연산자를 사용한다

3. 표준 입력 함수

LAB 변수값 입력받고 출력하기

문제 해결

LAB_2-2.c

```
01 #pragma warning(disable: 4996)
02 #include <stdio.h>
03
04 int main()
05 {
06     int integerVal;
07     char charVal;
08     double doubleVal;
09
10     // 정수 입력받기
11     printf("정수를 입력하세요: ");
12     scanf("%d", &integerVal);
13
14     // 문자 입력받기
```

컴파일러 경고(4996) 비활성화

변수 선언

3. 표준 입력 함수

LAB 변숫값 입력받고 출력하기

```
15  printf("문자를 입력하세요: ");
16  scanf("%c", &charVal);
17
18  // 실수 입력받기
19  printf("실수를 입력하세요: ");
20  scanf("%lf", &doubleVal);
21
22  // 입력받은 값 출력
23  printf("\n입력받은 정수: %d\n", integerValue);
24  printf("입력받은 문자: %c\n", charVal);
25  printf("입력받은 실수: %lf\n", doubleVal);
26
27  return 0;
28 }
```

공백은 scanf() 함수 호출 후 남은 줄 바꿈 문자(Enter)를 무시: 비주얼 스튜디오가 아닌 gcc 컴파일러로 실행할 때 사용하는 방법(1행 코드 적용 제외)