C언어로 배우는 프로그래밍 기초

Perfect C 3판

INFINITY BOOKS

제 4 장 전처리와 입출력

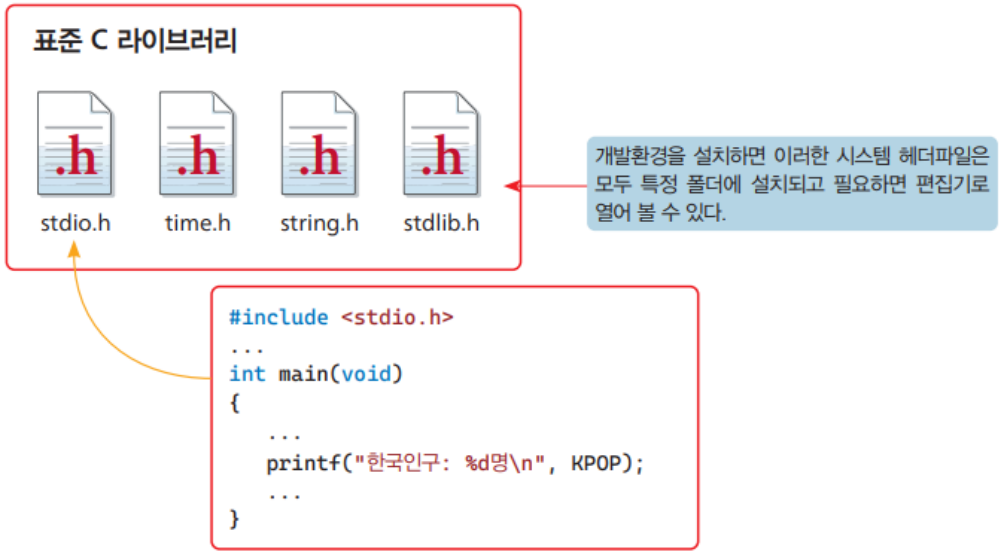
01 전처리
02 출력함수 printf()
03 입력함수 scanf()

학습목표

- 전처리기와 전처리 지시자에 대하여 이해하고 설명할 수 있다.
  - 전처리기 역할
  - 전처리 지시자 #define, #include
- 함수 printf()를 이용한 출력을 이해하고 프로그래밍 가능하다.
  - 여러 자료형에 따른 형식지정 방식
  - 형식 제어문자 및 다양한 출력 방식
  - 출력 폭 지정과 다양한 옵션 1, -, #, 0의 사용
- 함수 scanf()를 이용한 입력을 이해하고 프로그래밍 가능하다.
  - 여러 자료형에 따른 형식지정 방식
  - 형식 제어문자 및 다양한 입력 방식

전처리 개요

- **전처리기의 역할**
  - 컴파일(compile) 전, 전처리기(preprocessor)의 전처리 (preprocess) 과정이 필요
  - 컴파일 이전에 하는 작업, 처리 이후 전처리기가 생성한 소스를 컴파일
    - **전처리 지시자인 #include로 헤더파일을 삽입**
    - **#defin에 의해 정의된 상수를 대체시키는 등**
    - **전처리 결과인 전처리 출력파일을 만들어 컴파일러에게 보내는 작업을 수행**
- **전처리 지시자(preprocess directives)**
  - 전처리 과정에서 처리되는 문장
  - #include <헤더파일>
    - **대표적인 헤더파일인 stdio.h**
      - **printf(), scanf(), putchar() getchar() 등과 같은 입출력 함수의 정보가 정의**



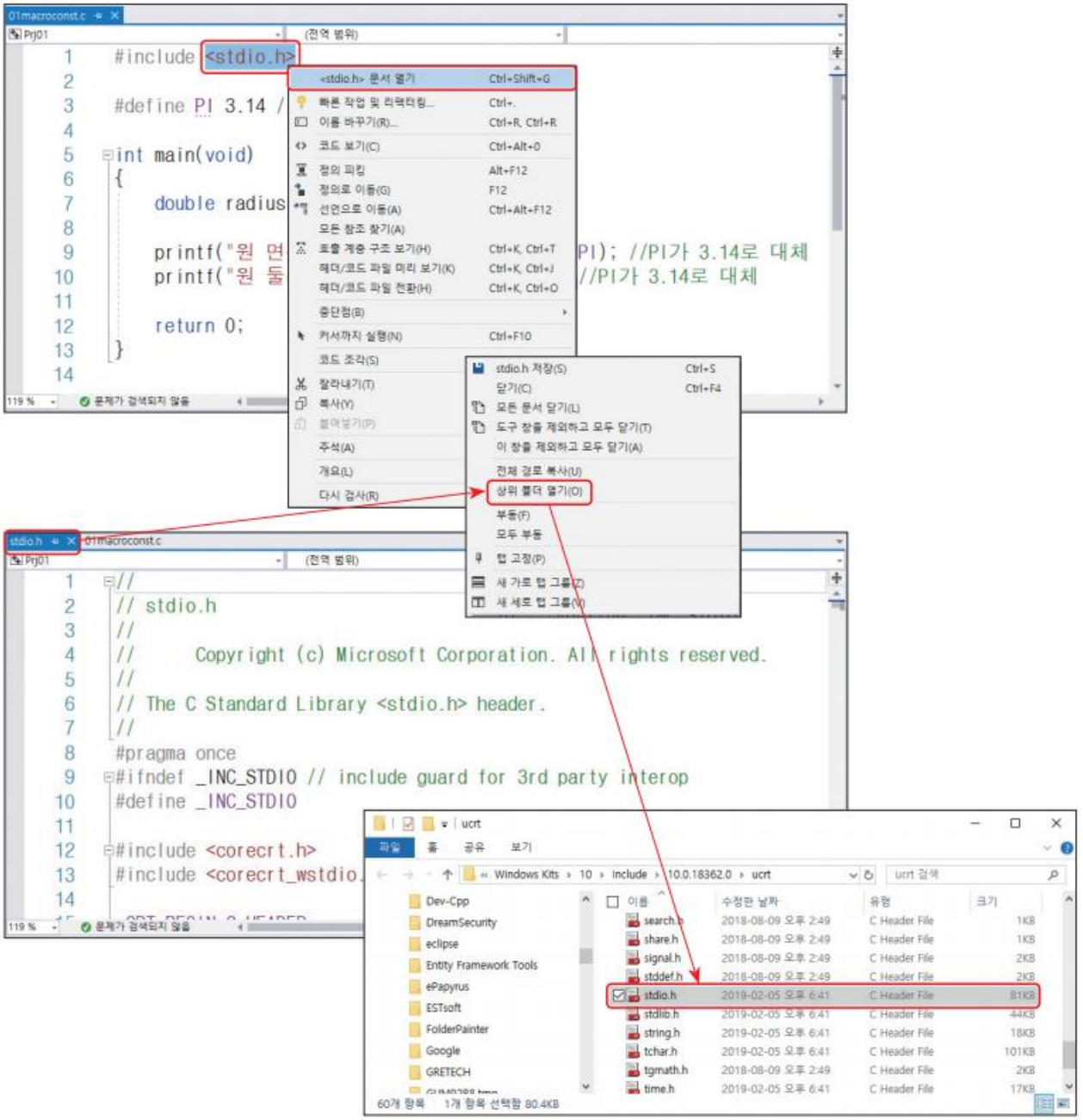
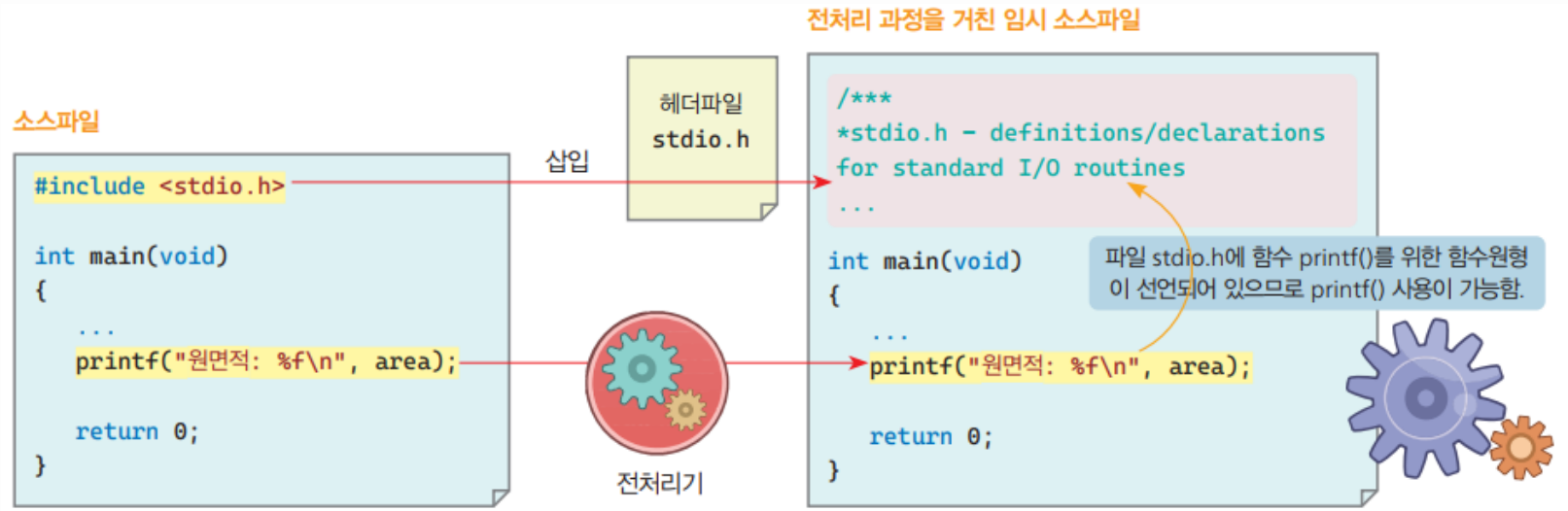


그림 4-3 비주얼 스튜디오에서 헤더파일 열기

전처리 지시자 #include와 헤더 파일

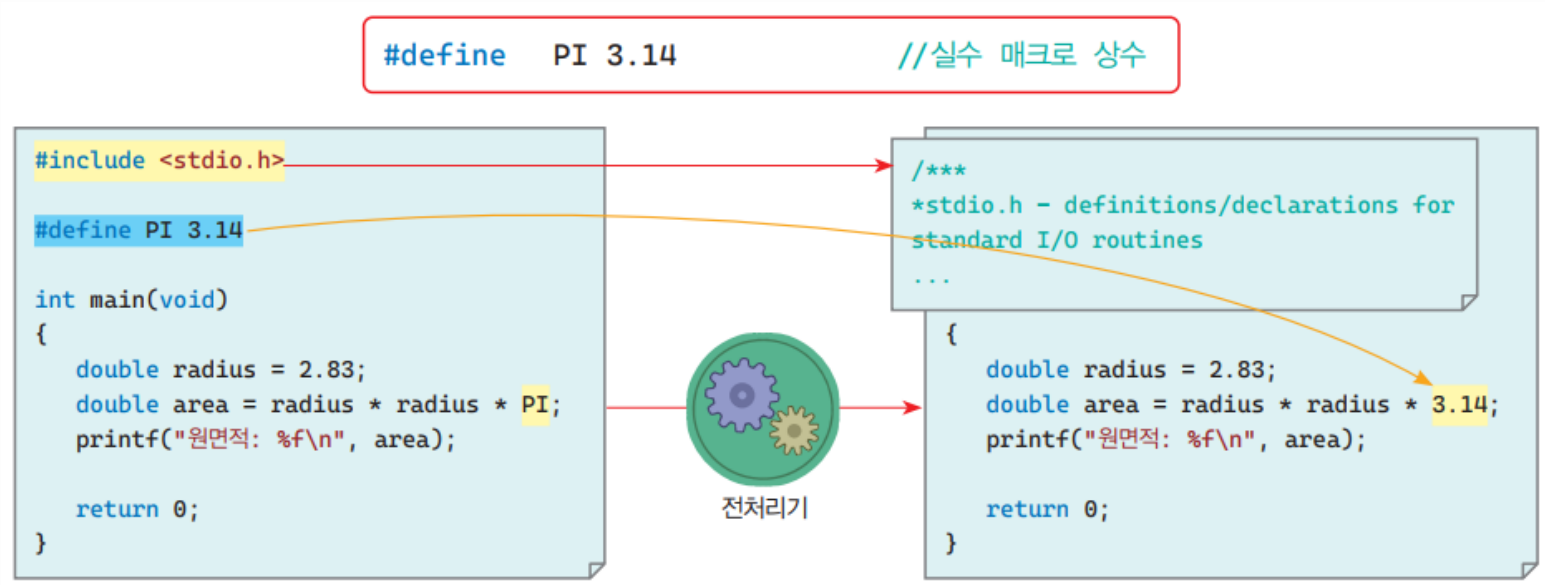
- 주요 헤더파일

헤더파일	파일 이름	파일 내용
stdio.h	STanDard Input Ouput(표준 입출력)	표준 입출력 함수와 상수
stdlib.h	STanDard LIBrary(표준 함수)	주요 메모리 할당 함수와 상수
math.h	math	수학 관련 함수와 상수
string.h	string	문자열 관련 함수와 상수
time.h	time	시간 관련 함수와 상수
ctype.h	Character type	문자 관련 함수와 상수
limits.h	limits	정수 상수 등 여러 상수
float.h	float	부동소수에 관련된 각종 상수



전처리 지시자 #define

- **매크로 상수(macro constant)**
  - 전처리 지시자 #define은 매크로 상수를 정의하는 지시자
  - #define에 의한 심볼릭 상수는 주로 대문자 이름으로 정의
  - 전처리기 (preprocessor)는 소스에서 정의된 매크로 상수를 모두 #define 지시자에서 정의된 문자열로 대체(replace)



#define에 의한 매크로 상수

- **4장을 위한 솔루션 생성**
  - 프로젝트 Prj01 생성

실습예제 4-1 | Prj01 | 01macroconst.c | #define에 의한 매크로 상수 | 난이도: ★

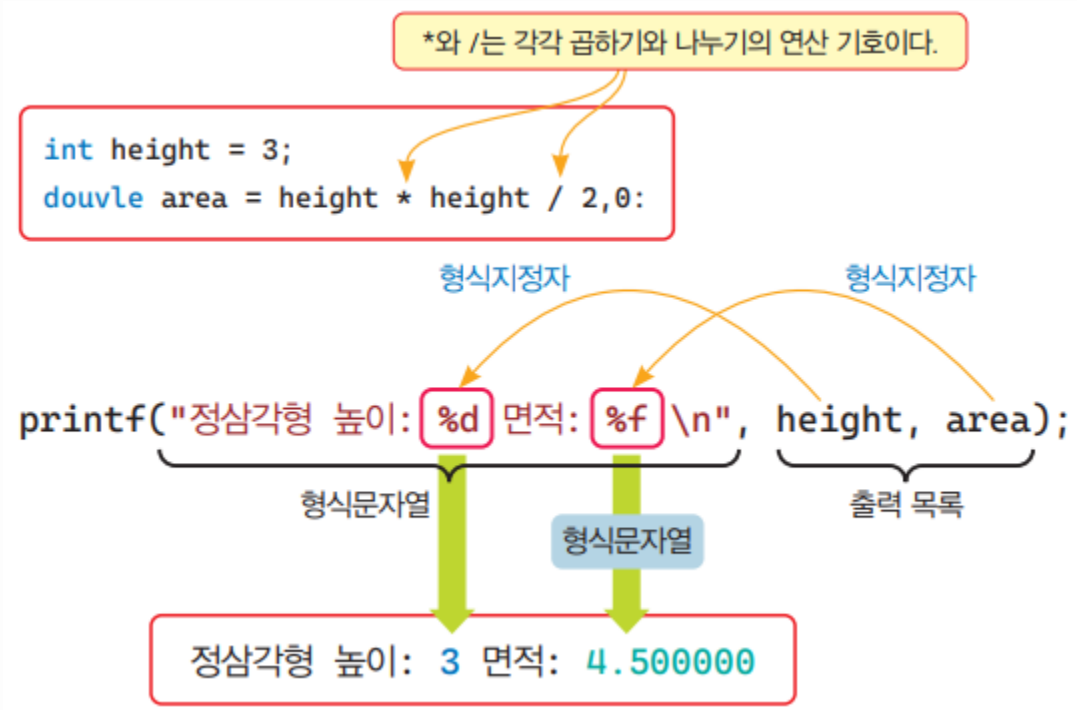
```
01  #include <stdio.h>
02
03  #define PI 3.14    //실수 매크로 상수
04
05  int main(void)
06  {
07     double radius = 2.83;
08
09     printf("원 면적: %f\n", radius * radius * PI);   //PI가 3.14로 대체
10     printf("원 둘레: %f\n", 2 * radius * PI);        //PI가 3.14로 대체
11
12     return 0;
13  }
```

결과

```
원 면적: 25.147946
원 둘레: 17.772400
```

출력함수 printf() 개요

- **함수 printf()**
  - 일련의 문자 및 값으로 서식을 지정하여
  - 표준 출력인 명령 프롬프트 콘솔(console)에 다양한 변수와 상수를 출력하는 함수
- **첫 번째 인자인 형식문자열(format string)**
  - 일반 문자와 이스케이프 문자 그리고 형식지정자(format specification)로 구성
    - **%d와 %s와 같이 %로 시작하는 형식지정자는 출력하려는 값의 위치에 배치**



출력함수 printf() 다양한 형식지정자

- **형식지정자는 출력 내용의 자료형에 따라 %d, %i, %c, %s와 같이 %로 시작**
  - 형식지정자 형식에 맞게 콘솔로 출력할 값이 표시
  - 형식지정자는 출력 값의 목록과 순서대로 서로 일치해야 함

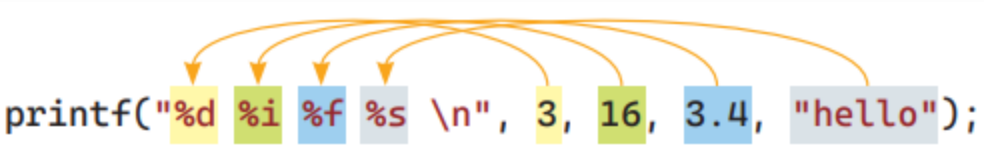


그림 4-7 출력 값 목록의 순서와 형식지정자의 일치

표 4-2 형식지정자 종류

형식지정자	출력 양식
%d %i %lld	정수 출력, 특히 %lld는 long long 자료형 출력에 사용
%f, %lf	실수 출력
%c, %C	문자 출력
%s	문자열 출력

형식지정자의 출력 폭 지정

- **출력 값의 가로 길이인 폭(width)도 지정이 가능**
  - %10d와 같이 %와 d 사이에 폭 길이인 10을 기술
  - 정렬은 기본적으로 오른쪽
    - **출력 폭을 잘 못 지정하여 출력 내용보다 좁으면 폭을 무시하고 원래대로 출력**

- **형식지정자 %8d는 10진수를 8자리 폭에 출력**한다. **지정한 출력 폭이 출력할 내용보다 넓으면 정렬은 기본적으로 오른쪽**이다. 다음은 정수 7629를 %8d로 출력한 모습이다.


```
printf("%8d\n", 7629);
```

- 형식지정자 %10f는 소수점을 포함한 전체 폭은 10이고 소수점 이하 자릿수는 기본으로 6자릿수로 출력한다.


```
printf("%10f\n", 32.369);
```

- 형식지정자가 %10.3f이면 소수점을 포함하여 전체 폭은 10, 그 중에서 3은 소수점 이하 자리수로 출력하며, 다음은 정수 32.369를 %10.3f로 출력한 모습으로, 소수점도 전체 폭 10에 해당한다.


```
printf("%10.3f\n", 32.369);
```

- 형식지정자 %10s로 문자열 "hello"를 출력해도 오른쪽 정렬로 다음과 같이 출력된다.


```
printf("%10s\n", "Hello");
```

printf에서의 다양한 형식지정자

실습예제 4-2 | Prj02 | 02printfbasic.c | printf에서의 다양한 형식지정자 | 난이도: ★

```
01  #include <stdio.h>
02
03  int main(void)
04  {
05      double width = 3. 424, height = 2.718;
06      int shape = 3;    //삼각형 또는 사각형
07
08      printf("가로: %f, 세로: %lf\n", width, height);
09      printf("%d각형 %s: %8.2f\n", shape, "면적", (width * height) / 2);
10      printf("%d각형 %s: %10.4f\n", shape + 1, "면적", width * height);
11
12      return 0;
13  }
```

printf()에서 실수는 %f와 %lf 모두 사용 가능하다.

%10.4는 실수 전체의 폭을 10으로 그 중에서 소수점 이하 자릿수를 4로 지정하는 방법이다.

결과

```
가로: 3.424000, 세로: 2.718000
3각형 면적:      4.65
4각형 면적:      9.3064
```

정수의 8진수 16진수 출력

- **정수를 각각 8진수와 16진수로 출력**
  - %o와 %x를 사용
  - 020과 0x1a처럼 0과 0x 등을 표기하려면
    - **#을 중간에 삽입해 %#o와 %#x을 사용**

실습예제 4-3 | Prj03 | 03printocthex.c | printf()에서 10진수를 바로 8진수와 16진수로 출력 | 난이도: ★★

```
01  #include <stdio.h>
02
03  int main(void)
04  {
05      printf("%3o %3d %3x\n", 10, 10, 10);    //10을 8, 10, 16진수로 각각 출력
06      printf("%#3o %3d %#3x\n", 12, 12, 12);
07      printf("%3o %3i %3X\n", 14, 14, 14);
08
09      return 0;
10  }
```

#을 사용한 8진수와 16진수의 출력에서 폭을 지정하려면 %#3o와 %#3x처럼 #이후에 폭을 지정

결과

```
 12   10    a
014   12  0xc
 16   14    E
```

상세 출력 폭과 정렬 방법

- **정렬**
  - 기본이 오른쪽
  - 폭 앞에 –를 붙여 %-8d로 지정하면 왼쪽으로 지정
- **부동소수에서 소수점 이하 자릿수를 지정**
  - %[전체폭].[소수점이하폭]f와 같이 표시

- **형식지정자 %8d는 10진수를 8자리 폭에 출력**한다. **지정한 출력 폭이 출력할 내용보다 넓으면 정렬은 기본적으로 오른쪽**이다. 다음은 정수 7629를 %8d로 출력한 모습이다.


```
printf("%8d\n", 7629);
```

- **출력 폭을 지정하며 정렬을 왼쪽**으로 지정하려면 %-8d처럼 폭 앞에 –를 삽입한다.


```
printf("%-8d\n", 7629);
```

- 형식지정자가 %10.3f이면 소수점을 포함하여 전체 폭은 10, 그 중에서 3은 소수점 이하 자리수로 출력하며, 다음은 정수 32.369를 %10.3f로 출력한 모습으로, 소수점도 전체 폭 10에 해당한다.


```
printf("%10.3f\n", 32.369);
```

- 또한 %10f는 전체 폭은 10이고 소수점 이하 자릿수는 기본으로 6자릿수로 출력한다.


```
printf("%10f\n", 32.369);
```

printf()에서 정수와 실수의 다양한 출력

실습예제 4-5 | Prj05 | 05printfnote.c | printf()에서 정수와 실수의 다양한 출력 | 난이도: ★★

```
01  #include <stdio.h>
02
03  int main(void)
04  {
05     printf("%d * %4d = %#5o\n", 2, 2, 2 * 2);
06     printf("%d * %04d = %#5o\n", 2, 3, 2 * 3);
07     printf("%d * %+04d = %-#5x\n", 2, 4, 2 * 4);
08     printf("%d * %+-4d = %+#5X\n", 2, 5, 2 * 5);
09
10     printf("%15.3f\n", 123456.789);
11     printf("%e\n", 123456.789);
12     printf("%g\n", 12.34e-5);
13     printf("%G\n", 12.34e-6);
14
15     return 0;
16  }
```

+는 + 표시, 0은 0 채움, -는 좌측정렬

%f로 출력해도 될 정도로 정밀도가 낮아 알아서 소수점 형태로 출력

정밀도가 높아 알아서 %E로 출력

결과

```
2 *    2 =     0 4
2 * 0003 =     0 6
2 * +004 = 0 x 8
2 * +5   =     0 X A
     123456.789
1.234568e+05
0.0001234
1.234E-05
```

문자열 출력에서의 출력폭 지정

- **심화학습: %[-][전체폭].[출력할 문자수]s**

  - 형식지정자 %10.3s는 전체폭이 10, 출력할 문자 수는 3개이므로 Hel까지 출력되며, 우측정렬이 기본이다.


```
printf("%10.3s\n", "Hello!");
```

  - 만약 좌측정렬을 하려면 %-10.3s를 하면 된다.


```
printf("%10.3s\n", "Hello!");
```

  - 형식지정자에서 주의할 것은 지정한 전체폭이 출력할 문자열의 수보다 작으면 무시하고, 원래 문자열의 폭만큼 모두 출력한다는 것이다.


```
printf("%4s\n", "Hello!");
```

  - 또한 형식지정자의 전체폭과 정밀도는 형식지정자에 *를 이용한 후, 그에 대응하는 정수를 목록 값으로 지정할 수 있다. 다음은 *이 5으로 대체되어 %10.5s로 문자열 "Hello"를 출력한다.


```
printf("%10.*s\n", 5, "Hello!");
```

LAB 정수와 실수, 문자와 문자열의 출력

- 개인의 성별과 이름, 나이, 성적 등 개인 정보를 출력하는 프로그램

Lab 4-1 lab1basicoutput.c 난이도: ★

```
01  #include <stdio.h>
02
03  int main(void)
04  {
05      int age = 20;
06      double gpa = 3.88;
07      char gender = 'M';
08      float weight = 62.489F;
09
10      printf("성별: %c\n", gender);
11      printf("이름: \n", "안 병훈");
12      printf("나이: %d\n", age);
13      printf("몸무게: %.2f\n", weight);
14      printf("평균평점(GPA): \n", gpa);
15
16      return 0;
17  }
```

정답

```
11  printf("이름: %s\n", "안 병훈");
14  printf("평균평점(GPA): %.3f\n", gpa);
```

입력함수 scanf()

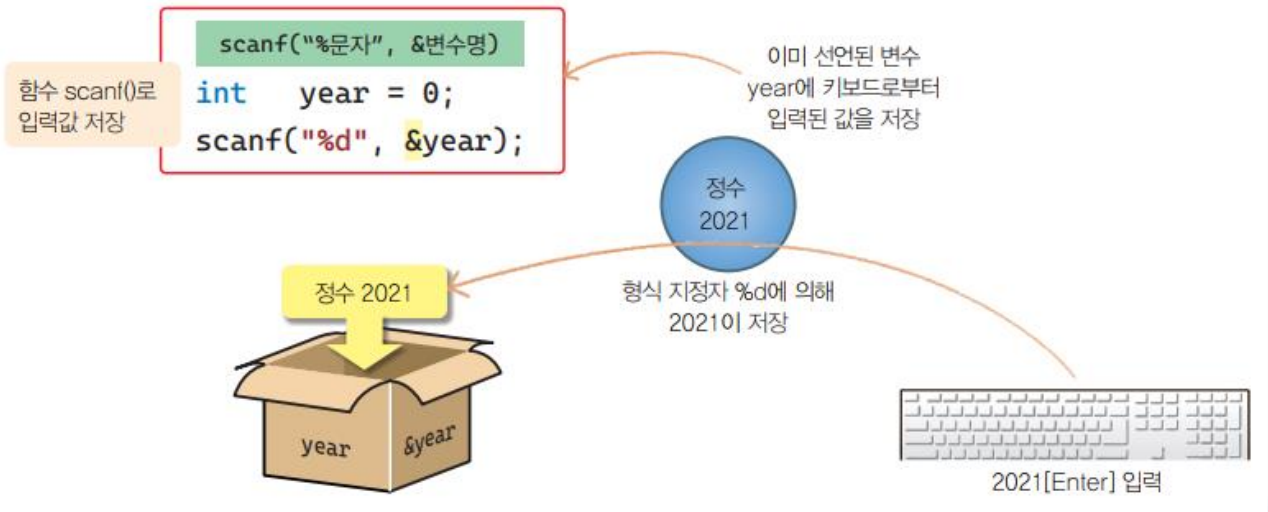
- **대표적인 입력 함수**
  - %d와 %c, %lf 등의 형식 지정자를 사용
  - 반드시 변수 앞에 주소 를 의미하는 &을 붙여 '&변수이름'으로 사용
    - **입력 값이 저장되는 변수의 주소 위치를 찾는다는 의미**

```
int point;
float value;
double data;
scanf(" %d %f %lf ", &point, &value, &data );
```

형식지정자 형식지정자 형식지정자

형식문자열

입력변수 목록



printf()에서 10진수를 바로 8진수와 16진수로 출력

실습예제 4-6 | Prj06 | 06scanf.c | printf()에서 10진수를 바로 8진수와 16진수로 출력 | 난이도: ★★

```
01  #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
02
03  #include <stdio.h>
04
05  int main(void)
06  {
07     int   month = 0;
08     printf("1년은 몇 달? ");
09     scanf("%d", &month);
10     printf("1년은 %d달\n\n", month);
11
12     int snum, credit;
13     printf("당신의 학번과 신청 학점은? ");
14     scanf("%d%d", &snum, &credit);
15     printf("학번: %d 신청학점: %d\n", snum, credit);
16
17     return 0;
18  }
```

간혹 scanf("1년은 몇 달? %d", &month); 문장으로 값을 입력 받으려 하는데, 잘못된 문장으로 scanf()의 형식문자열에 표시된 문자는 꼭 입력이 되어야 하는 형식이며, 콘솔에서 정수를 입력한 후 반드시 [return] 키 입력이 필요하다.

"%d%d"는 "%d %d"처럼 수를 위한 형식지정자 사이의 빈 공백은 아무 의미가 없다.

결과

```
1년은 몇 달? 12
1년은 12달

당신의 학번과 신청 학점은? 2023 24
학번: 2023 신청학점: 24
```

정수를 입력한 후 [Enter] 키를 눌러야 프로그램이 진행됨

두 정수를 스페이스 또는 [Enter] 키로 구분하여 입력한 후 [Enter] 키를 눌러야 프로그램이 진행됨

여러 값 입력에서 구분자 문자 사용

- **년, 월, 일을 2023-4-29과 같이 중간에 –를 넣어 입력 받으려면**
  - 함수 scanf("%d - %d - %d", ...) 처럼 형식문자열에 입력 형식을 명시

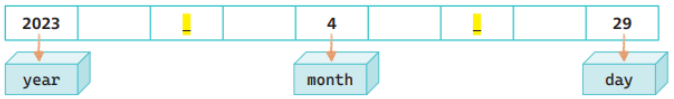
실습예제 4-7 Prj07 07scanfsep.c printf()에서 10진수를 바로 8진수와 16진수로 출력 난이도: ★★

```
01  #define _CRT_SECURE_NO_WARNINGS //
02
03  #include <stdio.h>
04
05  int main(void)
06  {
07      int year, month, day;
08      printf("당신의 생년월일은? ");
09      scanf("%d - %d - %d", &year, &month, &day);
10      printf("생년월일: %d %d %d\n", year, month, day);
11
12      return 0;
13  }
```

수를 입력 받는 경우, 형식문자열에서 빈 공간(space)은 아무 의미가 없으므로 빼도 상관없다. 실제 실행 시, 콘솔에서 년과 월 사이, 월과 일 사이에 빈 공간은 상관없이 –만 입력하면 된다.

```
scanf("%d - %d - %d", &year, &month, &day);
```

구분지인 – 반드시 필요



수와 수 사이에 –을 반드시 입력해야 한다.

2005와 4 사이, 4와 3 사이에는 반드시 -를 입력해야 함

결과

정상 입력

```
당신의 생년월일은? 2005-4-3
생년월일: 2005-4-3
```

잘못된 입력

```
당신의 생년월일은? 2015 3 9
생년월일: 2015 -858993460 -858993460
```

실수와 다양한 자료형의 입력

- **제어문자 %f와 %lf, %c**
  - printf()에서 실수의 출력을 위한 형식 지정자로 %f와 %lf를 모두 사용
- **입력 scanf()**
  - 저장될 자료형이 float이면 %f
  - double이면 %lf로 구분해 사용

실습예제 4-8 | Prj08 | 08floatscan.c | printf()에서 10진수를 바로 8진수와 16진수로 출력 | 난이도: ★

```c
#define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의

#include <stdio.h>

int main(void)
{
   float mile = 0;
   printf("100 킬로미터(km)는 몇 마일(mile)? "); //0.621
   scanf("%f", &mile);
   printf("80 킬로미터: %.2f 마일\n\n", mile * 80.);

   double liter = 0;
   printf("1 갤론(gallon)은 몇 리터(liter)? "); //3.785
   scanf("%lf", &liter);
   printf("18 갤론: %.2f 리터\n", liter * 18);

   return 0;
}
```

결과

```
100 킬로미터(km)는 몇 마일(mile)? 0.621
80 킬로미터: 49.68 마일

1 갤론(gallon)은 몇 리터(liter)? 3.785
18 갤론: 68.13 리터
```

다양한 형식지정자

- **함수 scanf()에서 정수의 콘솔 입력 값을 8진수로 인지하려면**
  - %o를 사용
  - %x는 16진수로 인지

표 4-5 함수 scanf()의 **형식지정자** (변환명세)

형식지정자	콘솔 입력 값의 형태	입력 변수 인자 유형
%d	10진수로 인식	정수형 int 변수에 입력 값 저장
%i	10진수로 인식하며, 단 입력 값에 0이 앞에 붙으면 8진수로 0x가 붙으면 16진수로 인식하여 저장	정수형 int 변수에 입력 값 저장
%u	unsigned int로 인식	정수형 unsigned int 변수에 입력 값 저장
%o	8진수로 인식	정수형 int 변수에 입력 값 저장
%x, %X	16진수로 인식	정수형 int 변수에 입력 값 저장
%f	부동소수로 인식	부동소수형 float 변수에 입력 값 저장
%lf	부동소수로 인식	부동소수형 double 변수에 입력 값 저장
%e, %E	지수 형태의 부동소수로 인식	부동소수형 float 변수에 입력 값 저장
%c	문자로 인식	문자형 char 변수에 입력 값 저장
%s	일련의 문자인 문자열(string)로 인식	문자열을 저장할 배열에 입력 값 저장
%p	주소(address) 값으로 인식	정수형 unsigned int 변수에 입력 값 저장

scanf()로 16진수를 입력 받아 8, 10, 16진수로 출력

실습예제 4-9 | Prj09 | 09scanfhex.c | scanf()로 16진수를 입력 받아 8, 10, 16진수로 출력 | 난이도: ★★

```c
#define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의

#include <stdio.h>

int main(void)
{
    int hex;
    printf("10진수 정수(1A 등)를 입력하세요 >> ");
    scanf("%x", &hex);
    printf("%o %d %x\n\n", hex, hex, hex);

    printf("10진수 정수(0리딩 표시방식으로 0x1a 등)를 입력하세요 >> ");
    scanf("%i", &hex);
    printf("%#o %d %#x\n\n", hex, hex, hex);

    return 0;
}
```

형식지정자 %i인 경우, 입력 값이 03과 같이 0이 리딩하는 수는 8진수로 인식하며, 0x1f과 같이 0x로 리딩하는 수는 16진수로 인식

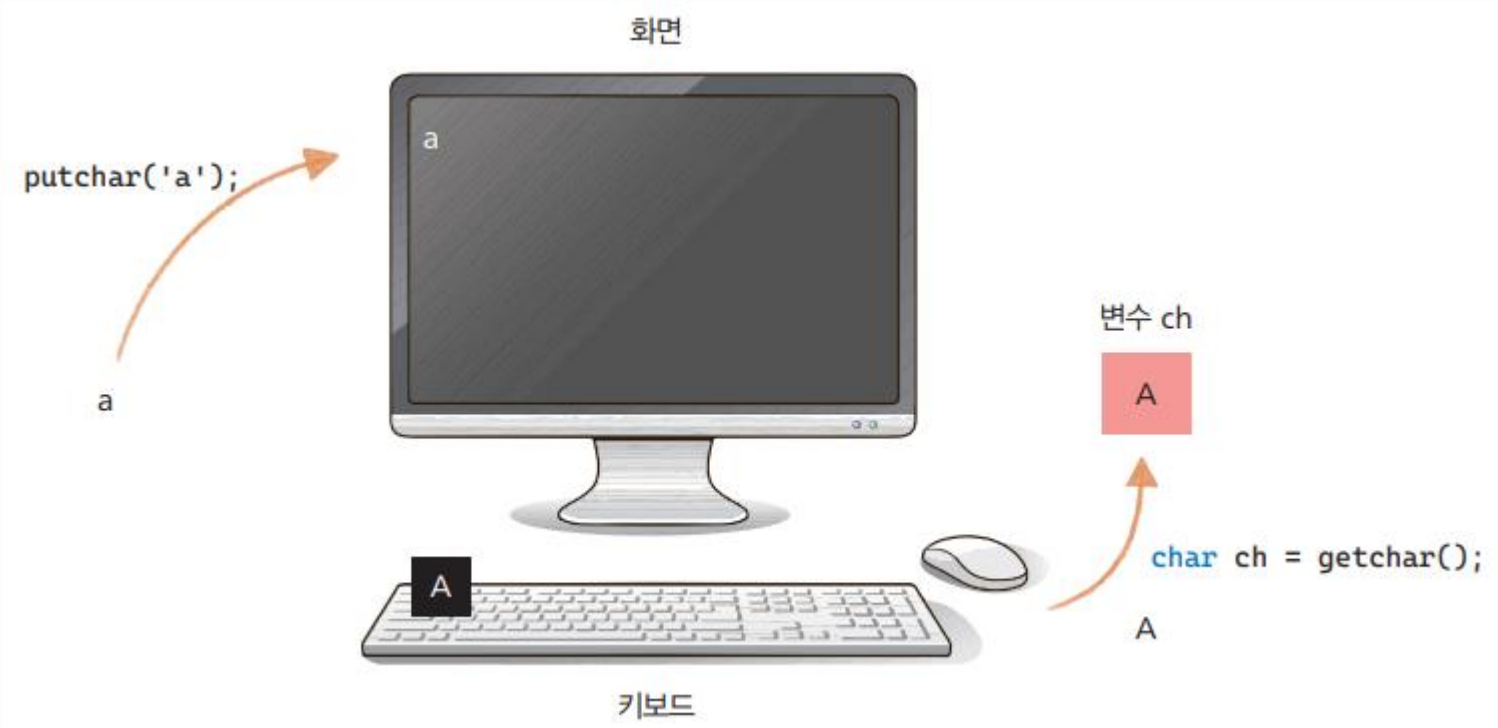
결과

```
10진수 정수(1A 등)를 입력하세요 >> 1b
33 27 1b

10진수 정수(0리딩 표시방식으로 0x1a 등)를 입력하세요 >> ox1B
033 27 0x1b
```

문자 입력과 출력 함수 getchar()와 putchar()

- **함수 getchar()**
  - 문자 하나를 입력하는 함수
- **함수 putchar()**
  - 문자를 출력하기 위한 함수
  - 헤더파일 stdio.h 가 필요



문자 입력과 출력 함수 getchar()와 putchar()

실습예제 4-10 | Prj10 | 10inputgrade.c | getchar()와 scanf()로 문자 입력 | 난이도: ★★

```
01  #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
02
03  #include <stdio.h>
04
05  int main(void)
06  {
07      char abc, plus;
08
09      printf("C 프로그래밍 언어의 원하는 학점(A+, A0처럼)을 입력 >> ");
10      //문자 두 개를 연이어서 입력 받음
11      abc = getchar();
12      scanf("%c", &plus);
13
14      printf("학점 %c%c\n", abc, plus);
15
16      return 0;
17  }
```

함수 getchar()는 인자가 필요 없으며 반환 값으로 입력 값을 저장한다.

scanf()로 두 번째 문자 +를 입력하여 plus에 저장한다.

결과

```
C 프로그래밍 언어의 원하는 학점(A+, A0처럼)을 입력 >> A+
학점 A+
```

LAB 8진수 정수를 입력 받아 8진수와 10진수, 16진수로 출력

- **8진수 입력을 위한 형식지정자 %o**
  - 출력은 두 줄로 각각 8진수와 10진수, 16진수로 출력
  - 출력 두번째 줄은 8진수와 16진수는 0이나 0X를 표시

Lab 4-2 | lab2octinput.c | 난이도: ★★

```
01  #define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한 상수 정의
02
03  #include <stdio.h>
04
05  int main(void)
06  {
07     int oct;
08     printf("8진수 정수 입력 >> ");
09     scanf("  ", &oct);
10
11     //입력 받은 정수를 각각 8, 10, 16진수로 출력
12     printf("%o %d %x\n", oct, oct, oct);
13     printf("                ", oct, oct, oct);
14
15     return 0;
16  }
17
```

정답

```
09  scanf("%o", &oct);
13  printf("%#o %i %#X\n", oct, oct, oct);
```

감사합니다.