

1강. C프로그래밍 개요

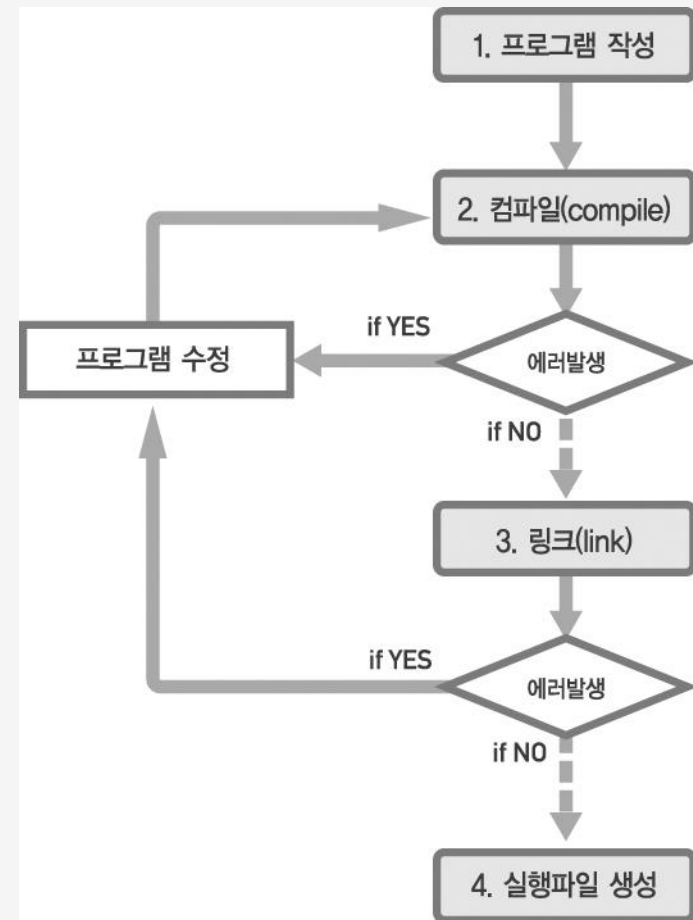
1장. C 언어 개요

- 프로그래밍 언어
 - 사람과 컴파일러가 이해할 수 있는 약속된 형태의 언어
 - C 언어도 프로그래밍 언어 중 하나
- 컴파일
 - 프로그래밍 언어로 작성된 프로그램을 컴퓨터가 이해할 수 있도록 기계어로 번역해 주는 역할

- C 언어의 장점
 - 익숙해지는데 오랜 시간이 걸리지 않는다.
 - 이식성이 좋다.
 - 효율성이 높다.
- C 언어의 단점
 - 프로그래밍 하는데 많은 주의를 요한다.
 - 완전한 고급 언어에 비해 상대적으로 어렵다.

- 프로그램 작성 및 실행 순서

- 1. 프로그램 작성
- 2. 컴파일
- 3. 링크
- 4. 실행파일 생성

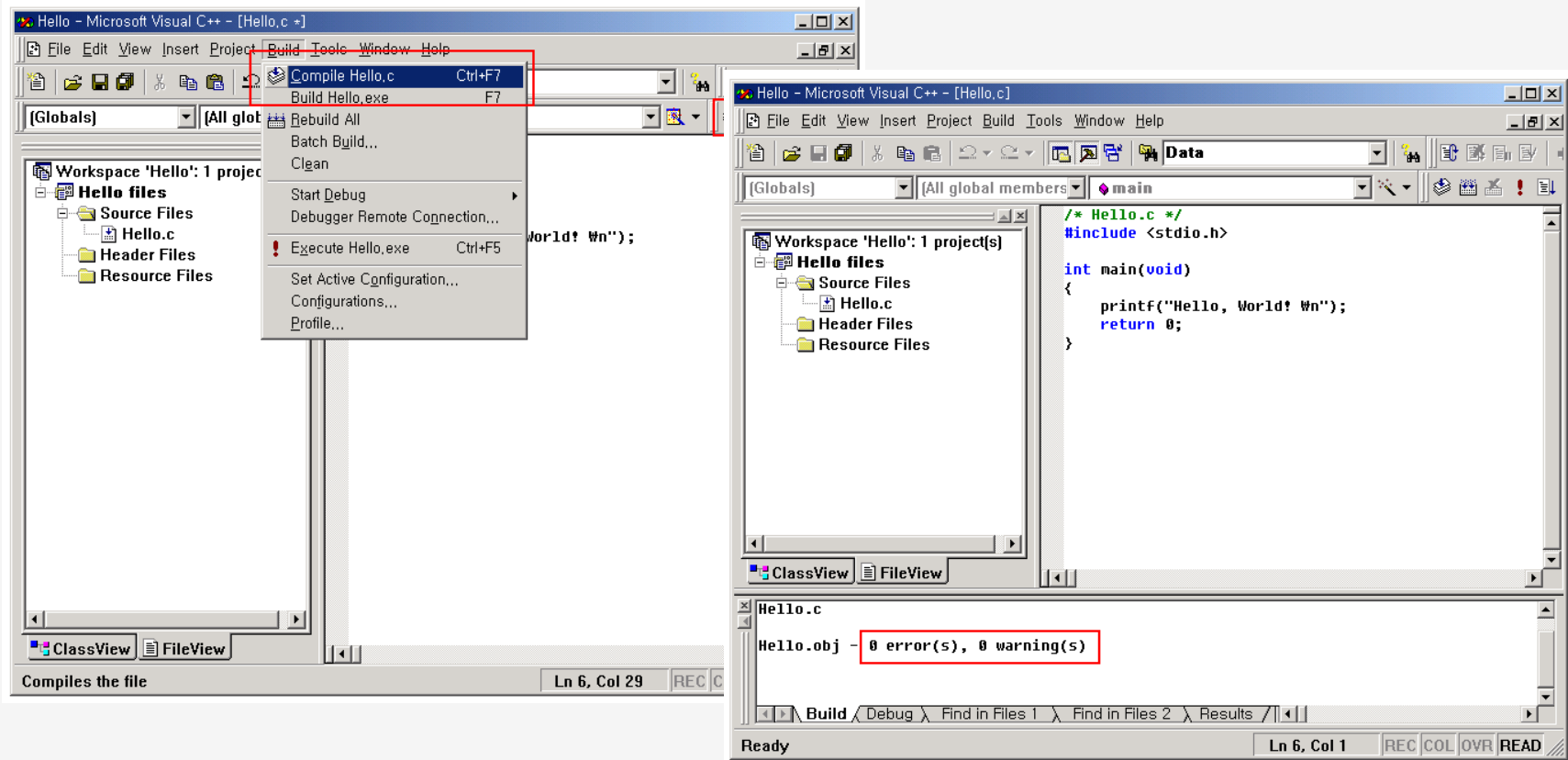


- 프로그램 편집

```
/* Hello.c */  
#include <stdio.h>  
//#include <stdafx.h>  
  
int main(void)  
{  
    printf("Hello, World! \n");  
    return 0;  
}
```

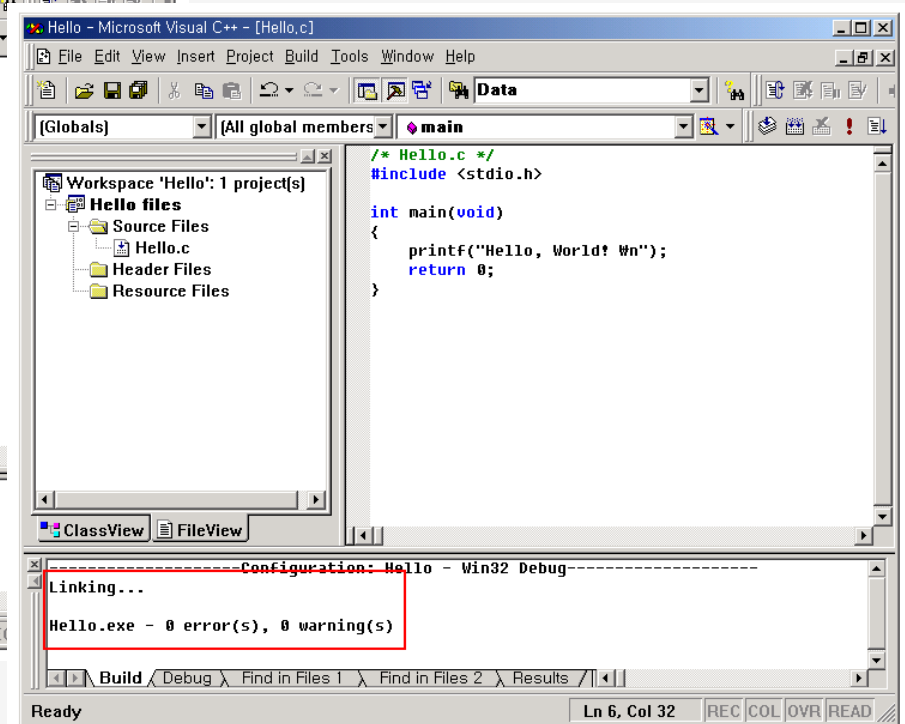
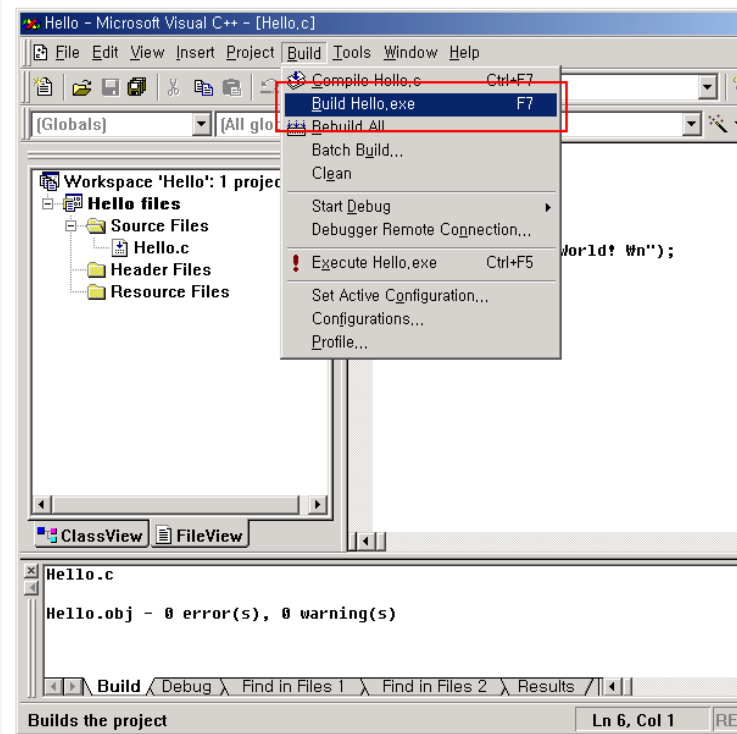
C/C++

- 컴파일(Compile)



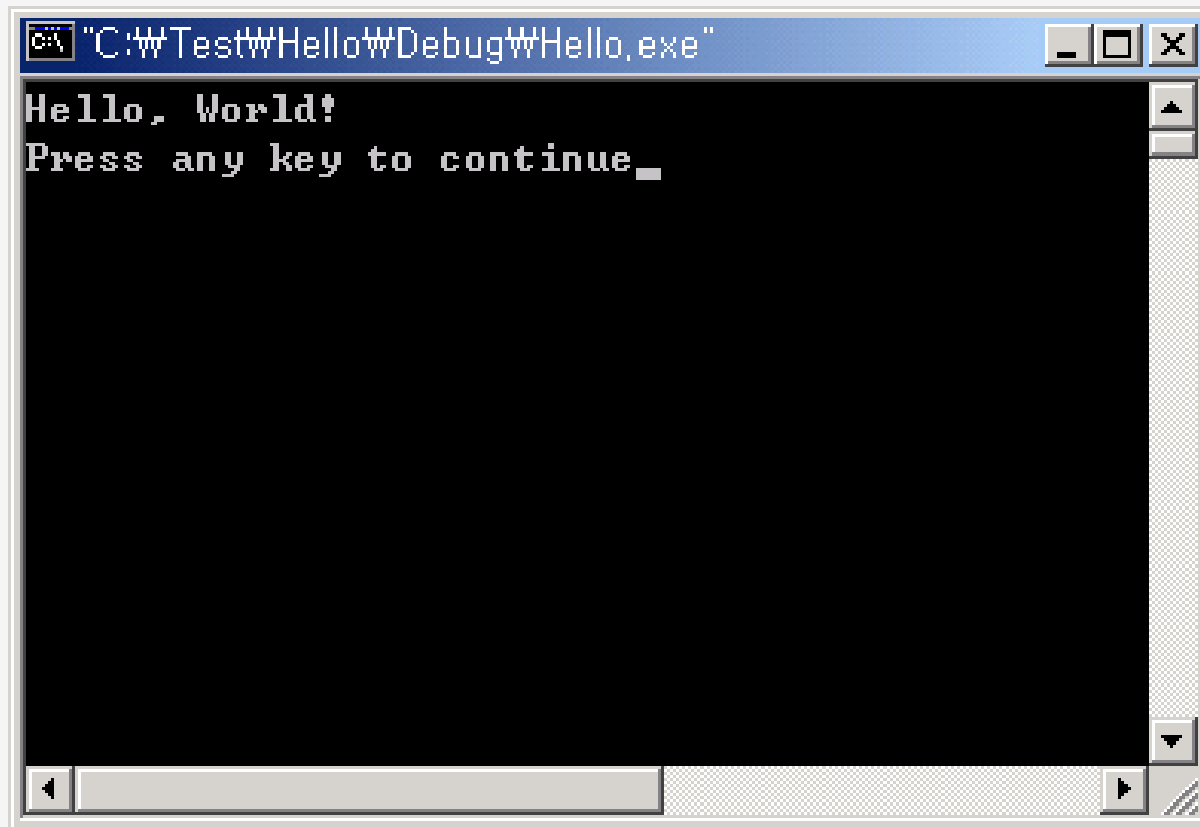
C/C++

- 링크(Link)



C/C++

- 프로그램의 실행

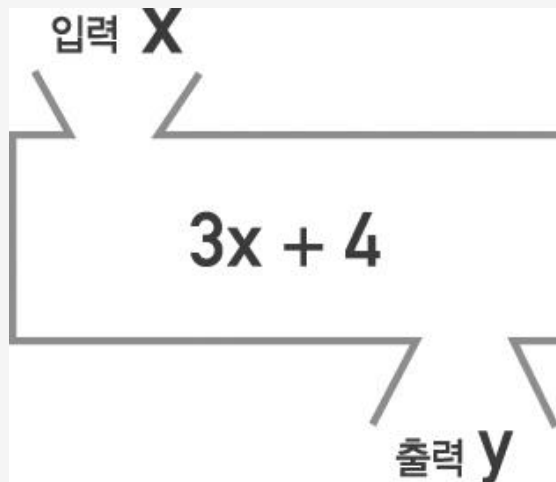


2장. 프로그램의 기본 구성

C/C++ "Hello, World!"

```
/* Hello.c */  
#include <stdio.h>  
// #include <stdafx.h>  
  
int main(void)  
{  
    printf("Hello, World! \n");  
    return 0;  
}
```

- 함수에 대한 이해
 - 적절한 입력과 그에 따른 출력이 존재 하는 것을 가리켜 함수라 한다.
 - C 언어의 기본 단위는 함수



- 함수 호출과 인자 전달
 - 인자 전달 : 입력 x 를 전달하는 행위
 - 함수 호출 : 인자를 전달하면서 함수의 실행을 요구하는 행위
- C 언어의 함수 특성
 - 입력과 출력 존재
 - 순차적으로 실행
 - 함수의 기능을 정의하는 몸체 부분 존재

- 예제 Hello.c에서의 함수



- 세미콜론이 필요한 문장
 - 연산을 수행하는 문장 : 시간의 흐름에 따라서 컴퓨터에게 "이러 이러한 일을 해라"라고 명령을 하는 문장
- 표준 라이브러리에 대한 이해
 - 이미 표준화 해서 만들어 놓은 함수들의 집합을 가리켜 표준 라이브러리라 한다.
 - 헤더 파일을 포함해야 사용이 가능하다.

- 헤더 파일의 이해
 - `stdio.h` 라는 이름의 헤더 파일
 - 헤더 파일의 포함을 알리는 선언은 제일 먼저 등장해야 한다.

- return의 의미
 - 함수를 종료(빠져 나온다).
 - 함수를 호출한 영역으로 값을 반환
- return의 특징
 - return은 함수 내에서 존재 하지 않을 수도 있다.
 - 둘 이상의 return문이 존재하는 것도 가능

- 주석이란?
 - 프로그래머에게 메모(memo)의 기능을 부여
 - 컴파일러는 주석을 없는 것으로 간주
 - 주석을 삽입 함으로 인해 프로그램의 가독성 증가
 - 선택이 아닌 필수!

- 주식의 두 가지 형태
 - 여러 줄에 걸친 주식 처리

```
/* 한 줄 짜리 주식 */
```

```
/*  
여러 줄에  
걸친 주식  
*/
```

단일 행 주식 처리

```
// 주식 하나.  
// 주식 둘.  
// 주식 셋.
```

- 주석의 예

```
#include <stdafx.h>                // stdafx.h 헤더 파일 포함

int main(void)                      // main 함수의 시작
{
    /*
        printf 함수는 모니터로 출력을 하는 경우에 쓴다.
        인자로 문자열을 전달하면 문자열을 출력한다.
    */
    printf("Hello World! \n");      //모니터로 문자열 출력
    return 0;                      // 0을 반환한다.
}                                  // main 함수의 끝
```

- 주석 처리에 있어서의 주의점
 - 주석을 나타내는 기호는 중복될 수 없다.

```
/* 주석의 시작, 여러 행에 걸쳐서
   /* 단일 행 주석 처리 */
*/
```

- 단, 단일 행 주석은 중복 가능하다.

```
/* 주석의 시작, 여러 행에 걸쳐서
   // 단일 행 주석 처리
*/
```

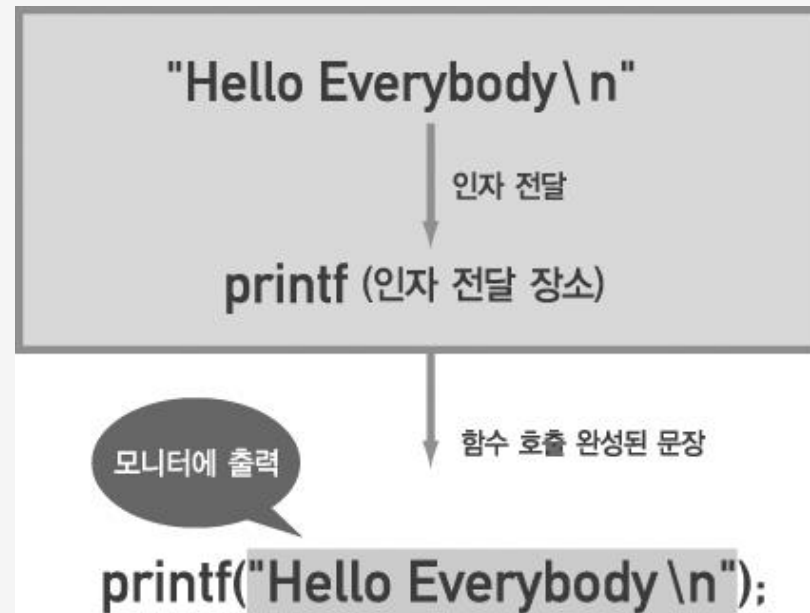
- printf 함수 사용의 예 1

```
#include <stdio.h>
// #include <stdafx.h>

int main(void)
{
    printf("Hello Everybody \n");
    printf("%d \n", 1234);
    printf("%d %d \n", 10, 20);
    return 0;
}
```

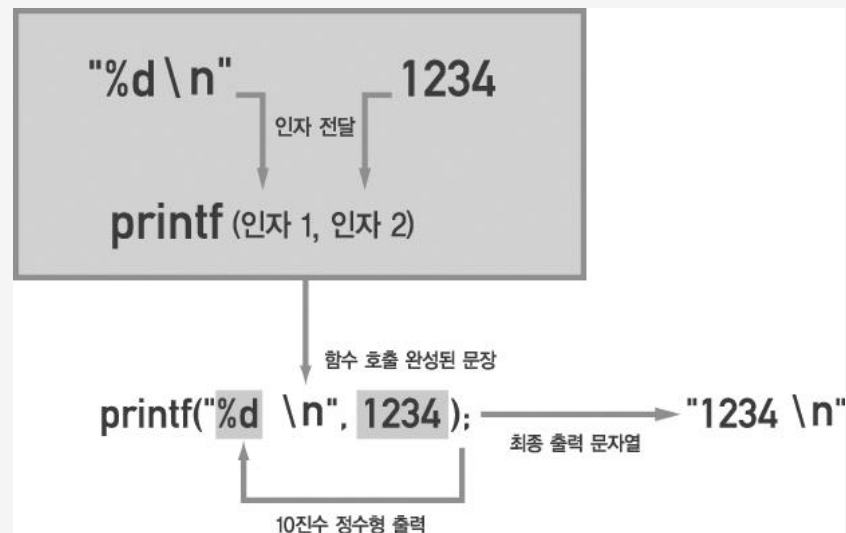
- printf 함수 호출의 이해 1

```
printf("Hello Everybody \n");
```



- printf 함수 호출의 이해 2 : 서식 문자
 - 서식 문자(Conversion specifier)란 출력 대상의 출력 형태를 지정하기 위한 문자

```
printf("%d \n", 1234);
```



- printf 함수 호출의 이해 3

```
printf("%d %d \n", 10, 20);
```



- printf 함수 사용의 예 2

```
#include <stdio.h>
// #include <stdafx.h>

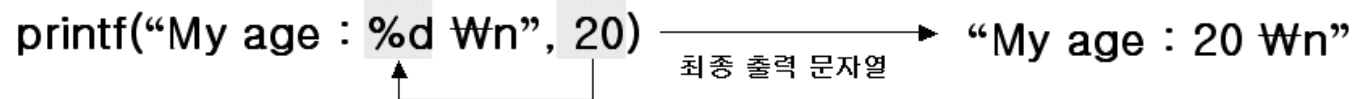
int main(void)
{
    printf("My age : %d \n", 20);
    printf("%d is my point \n", 100);
    printf("Good \nmorning \neverybody\n");

    return 0;
}
```

- printf 함수 호출의 이해 4

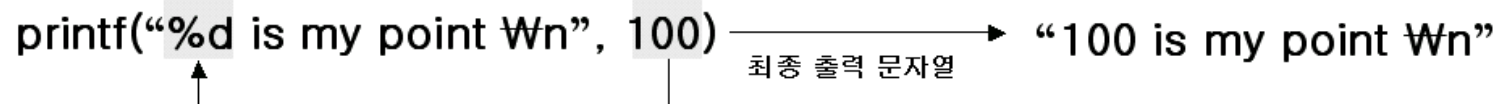
printf("My age : %d \n", 20) → "My age : 20 \n"

최종 출력 문자열



printf("%d is my point \n", 100) → "100 is my point \n"

최종 출력 문자열

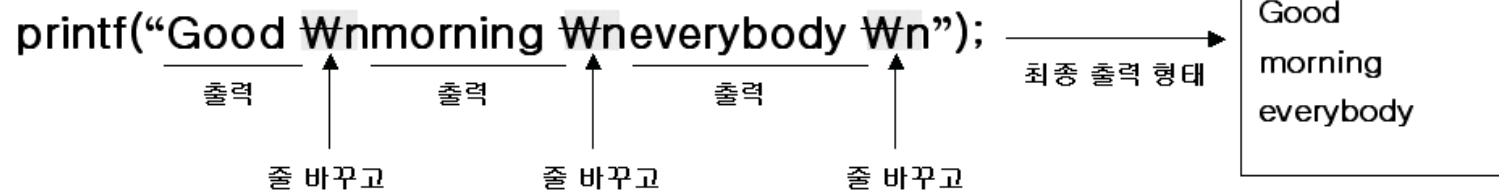


printf("Good \n morning \n everybody \n");

출력 줄 바꾸고 출력 줄 바꾸고 출력 줄 바꾸고

최종 출력 형태

Good
morning
everybody



부록. printf와 scanf 함수에 대한 고찰

C/C++ printf 함수 이야기

- printf는 문자열을 출력하는 함수이다.
- printf는 특수 문자 출력이 가능하다.

특수 문자	의 미
\a	경고음 소리 발생
\b	백스페이스(backspace)
\f	폼 피드(form feed)
\n	개행
\r	캐리지 리턴(carriage return)
\t	수평 탭
\v	수직 탭
\\	백슬래시(\)
\'	작은 따옴표
\"	큰 따옴표

- 특수 문자가 필요한 이유
 - 잘못된 문자열 출력

```
#include <stdafx.h>
```

```
int main(void)
```

```
{
```

```
    printf("앞집 강아지가 말했다. "멍! 멍!" 정말 귀엽다.");
```

```
    return 0;
```

```
}
```

printf("앞집 강아지가 말했다. " 멍! 멍! " 정말 귀엽다.");

문자열 1 정체불명 문자열 2

- printf 함수는 서식 지정이 가능하다.
 - printf의 f는 "formatted"를 의미한다.
 - 서식 지정 : 출력의 형태를 지정한다는 의미
(ex : 문자열 안에 숫자 삽입)
 - 서식 지정의 예

```
#include <stdio.h>
//#include <stdafx.h>

int main(void)
{
    int age=12;
    printf("10진수로 %d살이고 16진수로 %x살 입니다.", age, age);
    return 0;
}
```

서식 문자	출력 형태
%c	단일 문자
%d	부호 있는 10진 정수
%f	부호 있는 10진 실수
%s	문자열
%o	부호 없는 8진 정수
%u	부호 없는 10진 정수
%x	부호 없는 16진 정수, 소문자 사용
%X	부호 없는 16진 정수, 대문자 사용
%e	e 표기법에 의한 실수
%E	E 표기법에 의한 실수
%g	값에 따라서 %f, %e 둘 중 하나를 선택
%G	값에 따라서 %f, %E 둘 중 하나를 선택
%%	% 기호 출력

- 필드 폭을 지정하여 멋진 출력을!
 - 아래 서식 문자를 이용해서 임의의 실수를 전체 7자리, 소수점이하 2 자릿수로 (숫자 양쪽에 [], 줄바꿈하여 출력.
 - (예) 1.234,
 20.0567, 35.0345 ,
 420.123
 - 출력 문자열 :
 “%7.2f Wn”

- %c, %d, %f, %s
 - 가장 많이 쓰이는 서식 문자들
- %o, %u, %x, %X
 - 부호 없는 정수형 출력
- %e, %E
 - '부동소수점 표현 방식'에 의한 출력

$3.1245e+2 \rightarrow 3.1245 \times 10^{+2}$

$2.45e-4 \rightarrow 2.45 \times 10^{-4}$

- %g, %G

- 표현하고자 하는 실수의 값이 소수점 이하 6자리인 경우 %f의 형태로 출력
- 이 범위를 넘길 경우 %e의 형태로 출력

```
#include <stdio.h>
//#include <stdafx.h>

int main(void)
{
    printf("%g \n", 0.00123);           // 0.00123 출력
    printf("%G \n", 0.000123);         // 0.000123 출력
    printf("%g \n", 0.0000123);        // 1.23e-005 출력
    printf("%G \n", 0.00000123);       // 1.23E-006 출력

    return 0;
}
```

- 필드 폭을 지정하여 멋진 출력을!
 - 서식 문자를 이용해서 출력의 폭 지정 가능

서식 문자	출력의 형태
%8d	필드 폭을 8칸 확보하고 오른쪽 정렬해서 출력하라.
%-8d	필드 폭을 8칸 확보하고 왼쪽 정렬해서 출력하라.
%+8d	필드 폭을 8칸 확보하고 오른쪽 정렬한 상태에서 양수는 +, 음수는 -를 붙여서 출력하라.

- scanf 함수의 입력 형태 정의
 - 데이터를 입력받는 형태를 지정할 수 있다.
즉 입력 서식을 지정하는 것이다.
 - 예 : "%d %o %x"
- 실수 입력에 있어서 주의사항
 - 정밀도 생각!
 - 소수 6자리 이하의 실수 입력 시 %f 사용
 - 소수 6자리를 넘는 실수 입력 시 %e 사용
 - 단! double형 변수를 사용하는 경우에는 서식 문자 %le를 사용

- scanf 함수를 이용한 정수의 입력

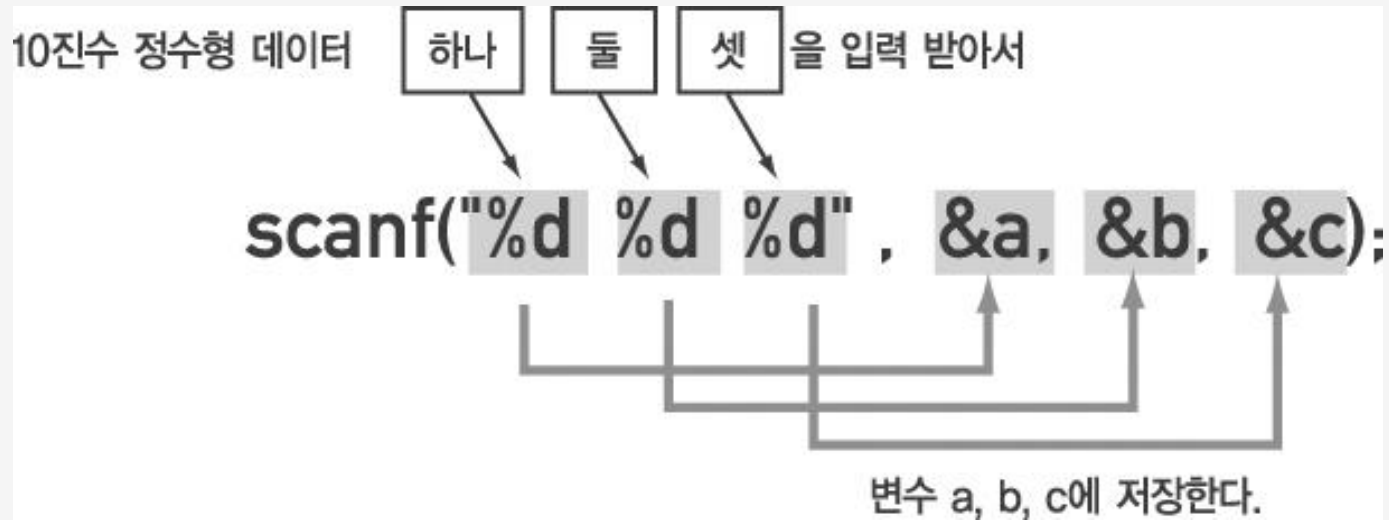
```
int main(void)
{
    int val;
    scanf("%d", &val);
    . . . . .
```

변수 val에 저장하라.

scanf("%d" , &val);

10진수 정수형으로 입력 받아서

- scanf 함수를 이용한 다중 변수 입력
 - 여러 개의 변수 입력 가능



- scanf 함수의 사용시 주의점
 - 입력받을 데이터의 주소에 데이터를 저장할 공간이 반드시 확보되어 있어야 한다. (미리 선언되어야 함)
 - 특히 문자열을 입력받는 경우 입력가능한 최대 범위를 예상하여 미리 문자배열을 선언하거나, 메모리를 확보해야 한다.
 - 변수가 아닌 변수의 주소를 넘겨 주어야 한다.
- scanf 함수 사용을 위한 사전 설정(vs200x~)
 - #define _CRT_SECURE_NO_WARNINGS
 - 메뉴 프로젝트 - 속성 - C/C++ - SDL 검사 No로 설정
 - #pragma warning(disable:4996)

- 필드 폭을 지정하여 멋진 출력을!
 - 서식 문자를 이용해서 정수, 실수, 문자열을 출력하세요.