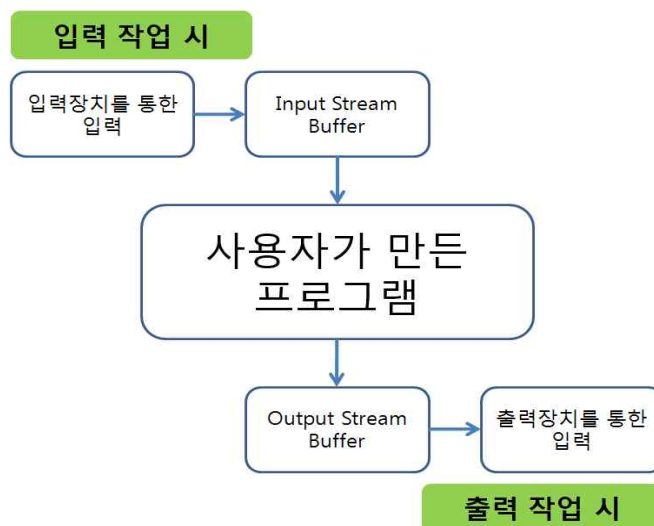


## 2. C언어의 입출력 함수

### ♻️ 입출력스트림

#### stream이란?

흐름 또는 개울을 뜻하는 단어로서 C언어에서 사용될 때는 data가 흘러가는 통로의 의미로 사용된다. 출력을 예로 들자면 10이란 data를 프린터를 통해 출력하려 할 때 사용자가 만든 프로그램에서 프린터까지 data가 전달되어야 한다. 이 때 data가 전달되는 통로를 stream이라 하는 것이다. 이 stream은 실제 존재하는 통로가 아닌 추상적인 개념임으로 눈에 보이지는 않는다. 이러한 stream은 입출력 장치가 data를 다루는 자세한 방식을 알지 못해도 사용자가 입출력을 처리 할 수 있도록 해준다. 또한 stream에는 방향성이 있다. 즉, 입력하기 위해서는 입력 stream을 사용하며 원하는 data를 출력하기 위해서는 출력 stream을 사용해야 한다.



C언어에서는 표준 입출력 스트림을 사용하기 위해 `stdio.h` 라는 헤더파일을 `include` 한다. `stdio.h` 에는 아래의 표와 같이 5개의 스트림이 포함되어 있다.

입출력스트림명	기 능	입출력 장치
<code>stdin</code>	표준입력	키보드
<code>stdout</code>	표준출력	모니터
<code>stderr</code>	표준에러	모니터
<code>stdprn</code>	표준프린터	프린터 (LPT1)
<code>stdaux</code>	표준 보조 입출력	직렬포트 (COM1)

위의 표에는 `stdio`에 있는 각 stream과 해당 stream이 사용하는 장치에 대한 설명이 나와 있다. 키보드로부터 들어온 신호는 `stdin`을 이용해 입력받을 수 있고 모니터로 보낼 신호는 `stdout`을 통해 출력 할 수 있다는 뜻이다.

stdio.h 안에 있는 각 스트림에는 입력 또는 출력을 위한 여러 가지 함수가 포함되어 있다. 그 중 일반적으로 많이 사용하는 함수는 아래와 같다.

함수명	함수의 사용법	설 명
putchar( )	putchar( '출력할문자' );	모니터에 원하는 문자 1개를 출력해 준다. 문자만 출력가능하며 출력할 문자 대신 해당문자의 Ascii 코드값을 넣어 출력하는 것도 가능하다. 출력의 성공 여부에 따라 1 또는 0이 함수의 리턴값이 된다.
getchar( )	변수명 = getchar( );	키보드로부터 원하는 문자 1개를 입력 받아 변수에 보관할 수 있게 해준다. 입력 시 원하는 글자를 입력하고 Enter키를 눌러야 입력이 완료된다. 입력받은 문자의 Ascii 코드값이 리턴되어 변수에 들어가게 된다.
puts( )	puts(출력할문자열);	모니터를 이용해 매개변수로 받아간 문자열을 출력해 준다. 문자열이 출력된 후 자동으로 줄바꿈이 일어나며 함수의 실행 결과로 출력의 성공여부에 따라 1 또는 0이 리턴 된다.
gets( )	gets(문자열을저장할배열명 );	키보드로부터 문자열을 입력받아 char형 배열에 보관해 준다. 입력 가능한 문자열의 크기는 배열의 크기-1 이며 문자열 중간에 공백문자가 있다면 문자열 입력이 종료 된다. 문자열 입력이 완료되면 입력받은 문자열의 시작주소 값이 함수의 실행결과로 리턴 된다.
printf( )	printf("출력할내용 및 서식", ....);	원하는 내용을 서식에 맞추어 출력 할 수 있도록 해준다. 출력서식은 사용자가 원하는 형태로 지정할 수 있으며 출력서식을 사용했을 경우 출력 대상을 “ ” 뒤쪽에 서식의 숫자와 같은 개수로 입력해해야 한다.
scanf( )	scanf("입력서식", &입력받을변수명);	원하는 내용을 서식에 맞추어 입력 할 수 있도록 해준다. 입력 서식은 사용자가 원하는 형태로 지정할 수 있으며 입력서식을 사용했을 경우 입력받은 값을 보관할 변수명을 “ ” 뒤쪽에 &연산자와 함께 기재해야 한다. 단, 문자열은 입력 시 &연산자 없이 배열명만 기재한다.

#### 비표준입출력 헤더파일인 conio.h에 포함되어 있는 함수들

함수명	함수의 사용법	설 명
getche( )	변수명 = getche( );	stdio.h에 있는 getchar()함수와 사용방법과 용도가 동일하다. 단, getche 함수는 문자를 입력 받을 때 엔터키를 입력하지 않아도 키보드를 누르는 순간 자동으로 입력이 완료된다. (입력한 글자는 화면에 표시 된다.)
getch( )	변수명 = getch( );	위의 getche함수와 동일하게 엔터키 없이 입력이 가능하지만 입력한 글자가 화면에 표시되지 않는다.
kbhit( )	kbhit( );	키보드가 눌렸는지 검사해 아무키라도 눌렀다면 1을 리턴하고 그렇지 않다면 0을 리턴한다.

## 🔄 입출력 함수의 사용방법

### 1. putchar( )의 사용방법

putchar( ) 함수 : 매개변수로 받아간 문자를 화면에 출력해 준다.

putchar( ) 함수의 원형 : `int putchar(int x);`

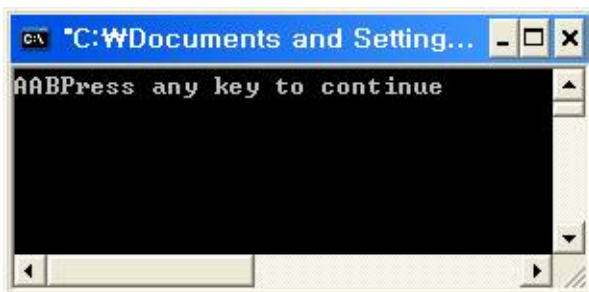
putchar( ) 함수의 특징 : 매개변수에 문자를 넣으면 해당 문자가 출력되고 매개변수에 정수를 넣으면 해당 정수를 아스키 코드값으로 가지는 문자가 출력된다. 오직 문자만 한 개만 출력이 가능하며 문자열이나 한 개 이상의 문자를 동시에 출력할 수는 없다.

예제 2-1 : putchar( ) 함수의 사용 예 - 1

```
#include <stdio.h>

void main ( ) {
    char x = 'A';
    int y = 66;

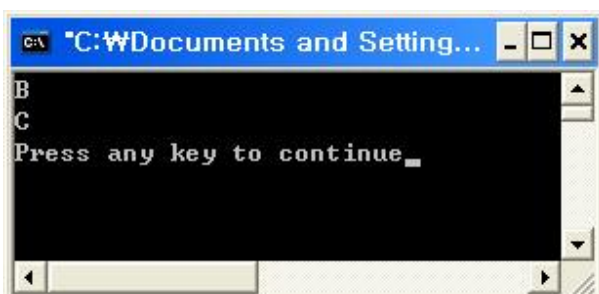
    putchar('A');    // 화면에 대문자 A가 출력된다.
    putchar( x );    // 화면에 char형 변수 x에 들어 있던 대문자 A가 출력된다.
    putchar( 66 );   // 아스키 코드가 66인 대문자 B가 화면에 출력된다.
}
```



예제 2-2 : putchar( ) 함수의 사용 예 - 2

```
#include <stdio.h>

void main ( ) {
    putchar('A'+1);    // 65+1의 결과인 66을 아스키 코드로 하는 문자인 대문자 B가 출력된다.
    putchar('Wn');     // 줄바꿈 문자가 출력되어 줄이 변경된다.
    putchar( 35*2-3 ); // 35*2-3의 결과인 67을 아스키 코드로 하는 문자인 대문자 C가 출력된다.
    putchar('Wn');     // 줄바꿈 문자가 출력되어 줄이 변경된다.
}
```



예제 2-3 : putchar ( ) 함수 사용 시 주의사항

```
#include <stdio.h>

void main ( ) {
    char x;
    putchar ( "korea" ); // " "로 묶인 것은 문자열임으로 putchar ( )함수로 출력 할 수 없다. 에러
    putchar ( "A" );     // " "로 묶인 것은 문자열임으로 putchar ( )함수로 출력 할 수 없다. 에러
    putchar ( y );       // y라는 변수는 생성하지 않았음으로 출력 할 수 없다. 에러
    putchar ( x );       // 변수 x는 초기화가 되지 않았음으로 쓰레기 값이 출력된다. 오류
    putchar ( 255 )      // 아스키 코드는 0~127까지 밖에 없음으로 255를 문자로 출력하면 쓰레기
                        // 값이 출력된다. 오류
}
```



## 2. getchar ( )의 사용방법

getchar ( ) 함수 : 키보드로부터 하나의 문자를 입력받아 입력받은 값을 리턴해 준다.

getchar ( ) 함수의 원형 : int getchar ( void );

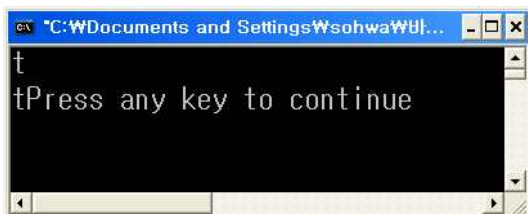
getchar ( ) 함수의 특징 : getchar ( ) 함수는 입력받은 문자가 함수의 실행결과로 리턴 되기 때문에 어떠한 변수에 이 입력받은 값을 보관시켜야 한다. 따라서 사용 시

**입력한 값을 저장할 변수 = getchar();** 와 같은 식으로 사용해야 한다.

예제 2-4 : getchar ( ) 함수의 사용 예 - 1

```
#include <stdio.h>

void main ( ) {
    char x;
    x = getchar ( );
    putchar ( x );
}
```

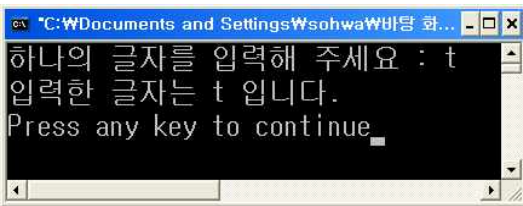


결과설명 : 위의 실행 결과에서 t는 사용자가 입력한 글자이다. 입력을 위해 t키를 누르고 엔터를 입력했다.

예제 2-5 : getchar( ) 함수의 사용 예 - 2

```
#include <stdio.h>

void main ( ) {
    char x;
    printf("하나의 글자를 입력해 주세요 : ");
    x = getchar( );
    printf("입력한 글자는 ");
    putchar( x );
    printf(" 입니다.\n");
}
```

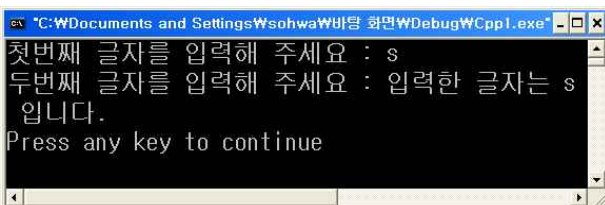


결과설명 : 위의 실행 결과에서 t는 사용자가 입력한 글자이다. 입력을 위해 t키를 누르고 엔터를 입력했다.

예제 2-6 : 연달아 두 개의 문자를 입력받을 때 오류가 발생하는 예제

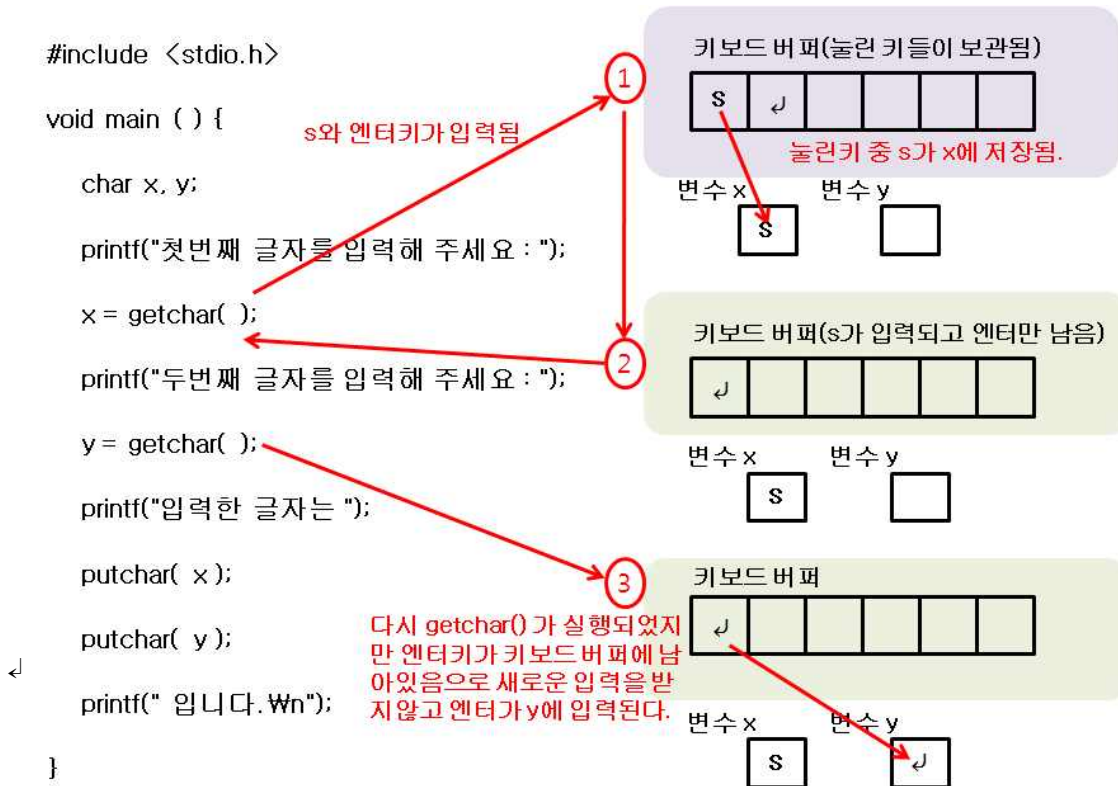
```
#include <stdio.h>

void main ( ) {
    char x, y;
    printf("첫번째 글자를 입력해 주세요 : ");
    x = getchar( );
    printf("두번째 글자를 입력해 주세요 : ");
    y = getchar( );
    printf("입력한 글자는 ");
    putchar( x );
    putchar( y );
    printf(" 입니다.\n");
}
```



결과설명 : 위의 실행 결과에서 s는 사용자가 입력한 글자이다. 입력을 위해 s키를 누르고 엔터를 입력했다. 그랬더니 두 번째 입력은 이루어지지 않고 바로 s와 엔터 문자가 출력된 것을 확인할 수 있다. (s가 출력된 이후 줄이 바뀌었다는 것은 줄바꿈 문자 또는 엔터문자가 출력되었다는 뜻이다.) 즉, 위와 같은 결과가 나온 이유는 문자 s를 입력하기 위해 s키와 엔터키를 입력했기 때문에 키보드 버퍼에 s와 엔터라는 두 개의 글자가 들어오기 때문이다. s는 x에 입력되어 저장되고 키보드 버퍼에 남아있던 엔터문자는 두 번째 getchar( )함수 실행 시 입력된 값으로 인

식되어 y에 전달된다.



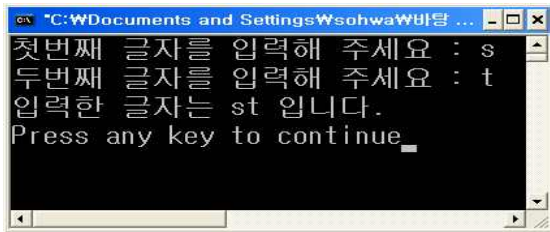
해결방법 : 키보드 버퍼에 남아있는 내용을 삭제하는 `fflush(stdin);` 을 이용해 키보드 버퍼의 내용을 비운 뒤 두 번째 문자를 입력받는다.

※ 모든 char 입력 전 키보드 버퍼에 문자가 남아 있다면 이 문자를 지운 뒤 입력을 받아야 올바른 char 입력이 가능하다. 따라서 char를 입력받기 전 `fflush(stdin);`을 한 번 실행해 주는 것이 좋다. (char를 제외한 다른 자료형 입력 시에는 `fflush(stdin);`이 필요하지 않다.)

예제 2-7 : 예제 2-6의 문제를 해결한 예제

```
#include <stdio.h>

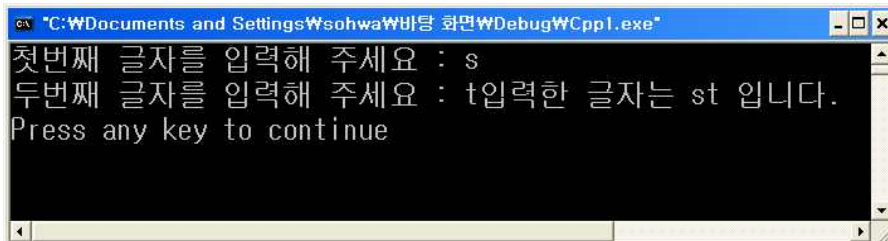
void main ( ) {
    char x, y;
    printf("첫번째 글자를 입력해 주세요 : ");
    x = getchar( );
    printf("두번째 글자를 입력해 주세요 : ");
    fflush(stdin); // 키보드 버퍼에 남아있는 문자를 비운다.
    y = getchar( );
    printf("입력한 글자는 ");
    putchar( x );
    putchar( y );
    printf(" 입니다.\n");
}
```



예제 2-8 : getchar( ) 함수와 유사하지만 엔터키 없이 입력이 가능한 getche( )함수의 사용 예제

```
#include <stdio.h>
#include <conio.h>          // getche( ) 함수를 사용하기 위한 헤더파일 선언

void main ( ) {
    char x, y;
    printf("첫번째 글자를 입력해 주세요 : ");
    x = getchar( );
    printf("두번째 글자를 입력해 주세요 : ");
    fflush(stdin);           // 키보드 버퍼에 남아있는 문자를 비운다.
    y = getche( );
    printf("입력한 글자는 ");
    putchar( x );
    putchar( y );
    printf(" 입니다.\n");
}
```

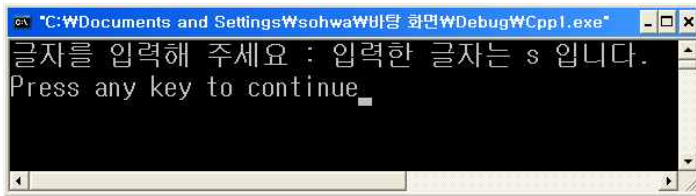


결과설명 : 위의 실행 결과에서 s는 getchar( ) 함수로 입력받았기 때문에 엔터키를 눌러야 입력이 완료되지만 t는 getche( ) 함수로 입력받았기 때문에 엔터키 없이 키보드를 누르자마자 입력이 완료된다. getche( ) 함수는 문자 입력 시 입력한 문자가 화면에 표시된다. getche( ) 함수를 사용하기 위해서는 conio.h를 include 해야 한다.

예제 2-8 : getche( ) 함수와 유사하지만 입력한 글자가 표시안되는 getch( )함수의 사용 예제

```
#include <stdio.h>
#include <conio.h>          // getche( ) 함수와 getch( ) 함수를 사용하기 위한 헤더파일 선언

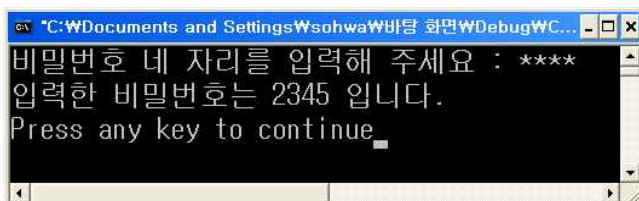
void main ( ) {
    char x;
    printf("글자를 입력해 주세요 : ");
    x = getch( );
    printf("입력한 글자는 ");
    putchar( x );
    printf(" 입니다.\n");
}
```



결과설명 : 위의 실행 결과에서 s를 입력했지만 getch( ) 함수는 입력한 글자를 화면에 보여주지 않기 때문에 출력 결과에만 s가 출력되는 것을 확인 할 수 있다. getch( ) 함수 역시 엔터키를 누르지 않아도 입력이 완료된다. getch( ) 함수를 사용하기 위해서는 conio.h를 include 해야 한다.

예제 2-9 : getch( ) 함수를 활용한 패스워드 입력 프로그램

```
#include <stdio.h>
#include <conio.h>          // getch( ) 함수를 사용하기 위한 헤더파일 선언
void main ( ) {
    char a, b, c, d;
    printf("비밀번호 네 자리를 입력해 주세요 : ");
    a = getch( );
    putchar('*');
    b = getch( );
    putchar('*');
    c = getch( );
    putchar('*');
    d = getch( );
    putchar('*');
    printf("\n입력한 비밀번호는 ");
    putchar( a );
    putchar( b );
    putchar( c );
    putchar( d );
    printf(" 입니다.\n");
}
```



결과설명 : 위의 실행 결과에서 비밀번호 네 자리를 입력 시 getch( ) 함수로 입력 받았기 때문에 입력한 글자가 화면에 출력되지 않고 한 번 입력이 완료될 때마다 putchar( ) 함수로 \*을 출력했다. 따라서 입력한 숫자가 화면에서는 \*로 출력되는 것처럼 보이게 된다.



### 3. puts( )의 사용방법

puts( ) 함수 : 매개변수로 받아간 문자열을 모니터에 출력해 주는 함수

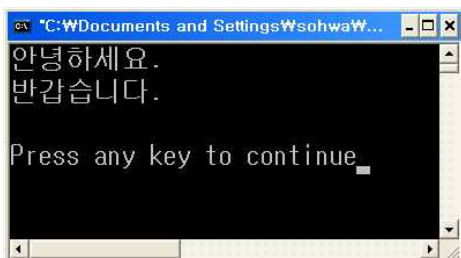
puts( ) 함수의 원형 : int puts( char \*str );

puts( ) 함수의 특징 : puts( ) 함수는 매개변수로 받아간 문자열을 출력한다. puts( ) 함수의 매개변수는 반드시 " "(큰따옴표)로 묶여있는 문자열 상수 이거나 문자열이 있는 곳의 시작주소를 보관하고 있는 char형 포인터 변수이어야 한다.

예제 2-10 : puts( ) 함수의 사용방법 - 1

```
#include <stdio.h>

void main ( ) {
    puts("안녕하세요.");
    puts("반갑습니다.\n");
}
```

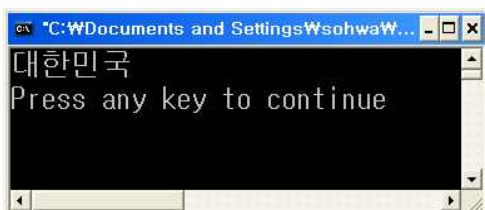


결과설명 : 위의 실행 결과에서 puts( ) 함수는 문자열 출력 후 자동으로 줄바꿈이 되는 것을 확인할 수 있다. 문자열 "반갑습니다.\n"의 경우에는 별도로 줄바꿈 문자인 \n을 출력했으므로 두 줄이 바뀐 것을 확인할 수 있다.

예제 2-10 : puts( ) 함수의 사용방법 - 2

```
#include <stdio.h>

void main ( ) {
    char name[]="대한민국"; // 문자열을 보관할 수 있는 char형 배열을 만들어 "대한민국"으로
                             // 초기화 하였다. 배열의 크기는 9로 만들어 진다.
    puts(name);
}
```



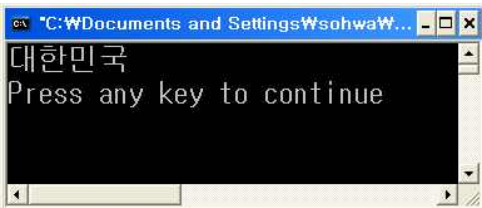
결과설명 : 배열에 보관되어 있는 문자열을 출력하는 방법이다. 배열에 있는 문자열을 puts( ) 함수로 출력할 경우 배열의 이름을 puts( ) 함수의 매개변수로 건네주면 되며 자세한 설명은 차후 배열 챕터 및 포인터 챕터에서 자세하게 설명한다.

예제 2-11 : puts( ) 함수의 사용방법 - 3

```
#include <stdio.h>

void main ( ) {
    char *name="대한민국";    // 문자열의 시작위치를 보관할 수 있는 char형 포인터 변수 name을
                               // 만들어 문자열 "대한민국"의 시작 주소값으로 초기화 하였다.

    puts(name);
}
```



결과설명 : char형 포인터 변수가 가리키고 있는 문자열을 출력하는 방법이다. 포인터 변수가 가리키는 문자열을 puts( ) 함수로 출력할 경우 포인터 변수를 puts( ) 함수의 매개변수로 건네주면 되며 자세한 설명은 차후 배열 챕터 및 포인터 챕터에서 자세하게 설명한다.

#### 4. gets( )의 사용방법

gets( ) 함수 : 키보드로부터 문자열을 입력받아 char형 배열에 보관 할 수 있도록 해주는 함수

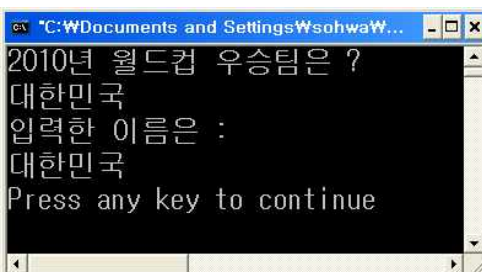
gets( ) 함수의 원형 : char\* gets( char \*str );

gets( ) 함수의 특징 : gets( ) 함수는 키보드로부터 문자열을 입력받아 매개변수로 받아간 배열에 입력받은 문자열을 저장 해준다. 따라서 입력받은 문자열을 보관하기 위해서는 gets( ) 함수의 매개변수에 char형 배열을 넣어야 한다. 사용방법은 "gets(char형 배열명);" 이다.

예제 2-10 : gets( ) 함수의 사용방법

```
#include <stdio.h>

void main ( ) {
    char arr[100];                // 입력받은 문자열을 보관시키기 위한 배열 arr를 만든다.
    puts("2010년 월드컵 우승팀은 ? ");
    gets( arr );                  // gets( ) 함수를 이용해 문자열을 입력받는다.
    puts("입력한 이름은 : ");
    puts( arr );
}
```



결과설명 : char형 배열 arr에 gets( ) 함수로 입력받은 문자열을 보관하고 출력하는 예제이다. 문자열 입력에 대한 작업 방법은 차후 배열 챕터 및 포인터 챕터에서 자세하게 설명한다.

## 5. printf( )의 사용방법

printf( ) 함수 : 매개변수로 문자열과 그안의 서식에 맞게 원하는 내용을 출력할 수 있도록 해주는 함수  
 printf( ) 함수의 원형 : int printf("string 또는 cotrolstring", ... );

printf( ) 함수의 특징 : printf( ) 함수는 여러 가지 종류의 data를 원하는 서식에 맞추어 출력 할 수 있도록 해주는 함수이다. 따라서 printf( ) 함수의 가로안에 있는 문자열 내부에 사용할 서식을 기술해야 하며 " " 뒤쪽에 출력 대상체를 기재해야 한다. 사용 할 수 있는 서식의 종류는 아래의 표에 정리되어 있으며 사용가능한 변환지정자 역시 아래쪽 표에 별도로 정리되어 있다. 따라서 실제 사용되는 printf( ) 함수의 형태는 printf("출력할 문자열 또는 출력서식", 출력대상체들); 이 된다. printf( ) 함수에 사용한 출력서식과 출력 대상체의 수는 같아야 하며 " " 안에 기재해 문제가 발생할 수 있는 문자는 \를 붙여 출력 하면 된다.

printf( ) 함수에서 사용가능한 출력 서식

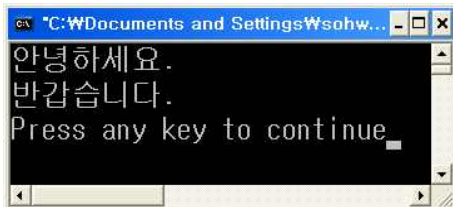
서식	용도	서식	용도
%d	10진 정수형으로 data를 출력한다.	%u	부호없는 10진 정수형으로 data를 출력한다.
%o	8진 정수형으로 data를 출력한다.	%x	16진 정수형으로 data를 출력한다.
%f	고정소숫점 형태로 data를 출력한다.	%lf	긴 고정소숫점 형태로 data를 출력한다.
%e	부동소숫점 형태(지수형)로 data를 출력한다.	%le	긴 부동소숫점 형태(긴 지수형)로 data를 출력한다.
%c	문자 형태로 data를 출력한다.	%s	문자열 형태로 data를 출력한다.
%p	주소 형태(포인터형)로 data를 출력한다.	%ld	긴 10진 정수형으로 data를 출력한다.

printf( ) 함수에서 사용가능한 변환 서식 지정자

변환서식	용도	사 용 예 제	
%숫자d %숫자f 등등	내용이 출력될 공간의 크기를 "숫자"의 크기만큼 만큼 확보	사용예	printf("+++%5d+++Wn", 123);
		결 과	+++ 123+++
%.숫자f %.숫자lf	실수가 출력될 소수점이하의 자리수를 "숫자"의 크기로 지정	사용예	printf("+++%.2lf+++Wn", 123.456789);
		결 과	+++123.45+++
%.숫자s	출력될 문자열의 길이를 "숫자"의 크기로 지정	사용예	printf("+++%.2s+++Wn", "korea");
		결 과	+++ko+++
%-숫자d %-숫자f 등등	"숫자"의 크기만큼 만큼 확보한 공간내에 자료 출력 시 왼쪽정렬	사용예	printf("+++%-5d+++Wn", 123);
		결 과	+++123 +++
%0숫자d %0숫자f 등등	"숫자"의 크기만큼 만큼 확보한 공간내에 자료 출력 시 공백을 0으로 채움	사용예	printf("+++%05d+++Wn", 123);
		결 과	+++00123+++

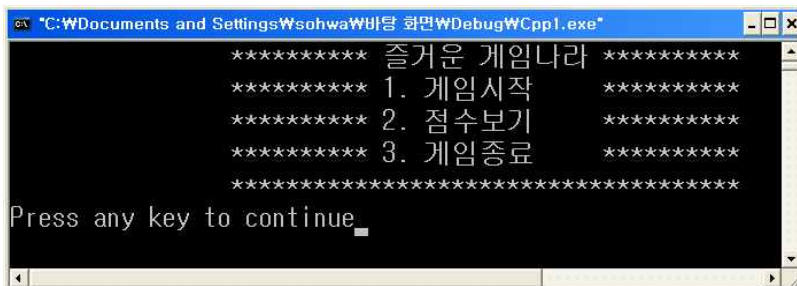
예제 2-11 : printf( ) 함수를 이용한 문자열 상수 출력 방법 - 1

```
#include <stdio.h>
void main ( ) {
    printf("안녕하세요.\n");
    printf("반갑습니다.\n");
}
```



예제 2-12 : printf( ) 함수를 이용한 문자열 상수 출력 방법 - 2

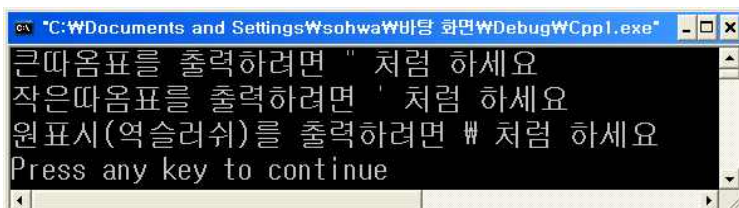
```
#include <stdio.h>
void main ( ) {
    printf("WtWt***** 즐거운 게임나라 *****\n");
    printf("WtWt***** 1. 게임시작 *****\n");
    printf("WtWt***** 2. 점수보기 *****\n");
    printf("WtWt***** 3. 게임종료 *****\n");
    printf("WtWt*****\n");
}
```



결과설명 : Wt는 탭문자를 의미한다. 따라서 Wt를 출력하면 탭이 출력된다.

예제 2-13 : printf( )의 ""에 입력해 출력 시 문제가 발생하는 문자의 입력방법

```
#include <stdio.h>
void main ( ) {
    printf("큰따옴표를 출력하려면 \"\" 처럼 하세요\n");
    printf("작은따옴표를 출력하려면 \"'\" 처럼 하세요\n");
    printf("원표시(역슬러쉬)를 출력하려면 \"\\\" 처럼 하세요\n");
}
```



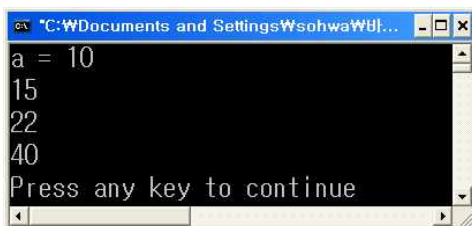
결과설명 : 출력 시 문제가 될 수 있는 ", ', W 와 같은 문자는 앞에 W를 붙여 출력하면 된다.

출력 시 자주 사용하는 확장 문자 상수

문자상수	Ascii코드	출력되는 문자
W0	0	NULL문자를 출력함 (문자열이 더 이상 출력되지 않는다.)
Wa	7	alarm을 출력함 (스피커에서 삐 소리가 출력된다.)
Wb	8	Backspace를 출력함 (커서가 한 칸 앞으로 이동된다.)
Wt	9	tab을 출력함 (tab을 누른 것과 동일하게 커서가 이동한다.)
Wn	10	줄바꿈 문자가 출력됨 (줄이 바뀐다.)
W"	34	쌍따옴표를 출력함
W'	39	따옴표를 출력함
WW	92	W 문자를 출력함

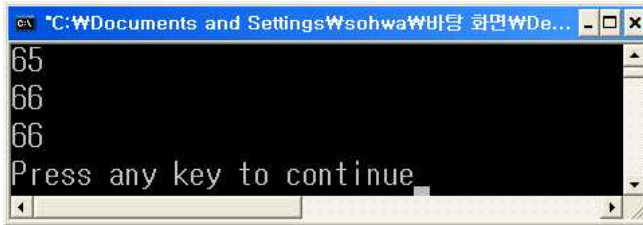
예제 2-14 : 서식 지정자 %d를 이용한 int형 정수의 출력 방법

```
#include <stdio.h>
void main ( ) {
    int a = 10;
    int b = 30;
    printf("a = %d\n", a);           // 변수 a에 있는 10이 %d(10진 정수형)로 출력된다.
    printf("%d\n", 15);              // 출력 대상체인 15가 %d(10진 정수형)로 출력된다.
    printf("%d\n", 15+7);            // 15+7의 결과인 22가 %d(10진 정수형)로 출력된다.
    printf("%d\n", a+b);            // 10+30의 결과인 40이 %d(10진 정수형)로 출력된다.
}
```



예제 2-15 : 서식 지정자 %d를 이용한 char형 문자의 출력

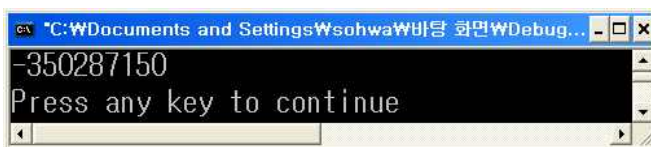
```
#include <stdio.h>
void main ( ) {
    char x = 'A';
    char y = 'B';
    printf("%d\n", x);               // 출력 대상체인 x에 들어있는 'A'의 ascii코드인 65가
                                    // %d(10진 정수형)로 출력된다.
    printf("%d\n", y);               // 출력 대상체인 y에 들어있는 'B'의 ascii코드인 66이
                                    // %d(10진 정수형)로 출력된다.
    printf("%d\n", x+1);             // x+1의 결과 즉, 65+1의 결과값인 66이 %d(10진 정수형)로 출력된다.
}
```



```
C:\Windows\system32\cmd.exe
65
66
66
Press any key to continue
```

예제 2-16 : 크기가 큰 정수를 %d로 출력

```
#include <stdio.h>
void main ( ) {
    printf("%d\n", 12345678901234567890);
}
```

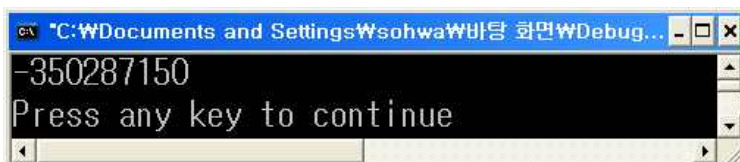


```
C:\Windows\system32\cmd.exe
-350287150
Press any key to continue
```

결과설명 : 출력 가능한 범위를 넘어서는 수를 출력하면 쓰레기 값이 출력된다. %d는 int의 크기를 넘어서는 수는 출력 할 수 없다.

예제 2-16 : 서식을 지정한 뒤 출력 대상체를 미 입력한 경우

```
#include <stdio.h>
void main ( ) {
    printf("%d\n"); // 출력대상이 없음으로 쓰레기 값이 출력된다.
}
```

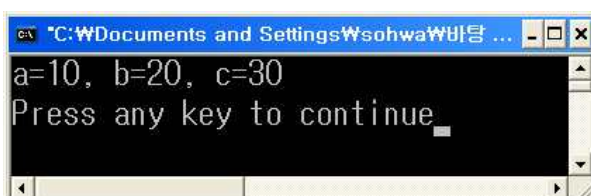


```
C:\Windows\system32\cmd.exe
-350287150
Press any key to continue
```

결과설명 : 출력 대상체가 없다면 쓰레기 값이 출력된다.

예제 2-16 : 한 번에 여러개의 변수를 출력하는 방법

```
#include <stdio.h>
void main ( ) {
    int a=10, b=20, c=30;
    printf("a=%d, b=%d, c=%d\n", a, b, c); // 순서대로 a, b, c의 값이 출력된다.
}
```



```
C:\Windows\system32\cmd.exe
a=10, b=20, c=30
Press any key to continue
```

결과설명 : 서식을 지정한 순서대로 출력 대상체가 하나씩 출력된다.

예제 2-17 : 출력서식 보다 출력 대상체가 많은 경우

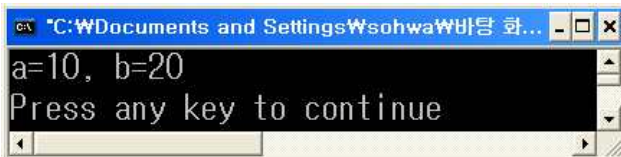
```
#include <stdio.h>
void main ( ) {
    int a=10, b=20, c=30;
    printf("a=%d, b=%d \n", a, b, c);    // 서식이 두 개임으로 순서대로 a와 b까지만 출력된다.
}
```



결과설명 : 서식이 지정되지 않은 출력 대상체는 출력되지 않는다.

예제 2-18 : %d를 이용한 short형 변수와 long형 변수의 출력 방법

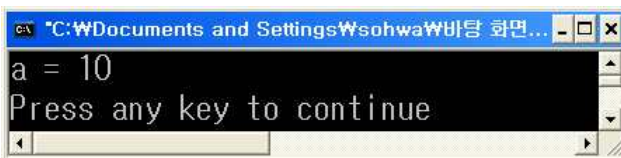
```
#include <stdio.h>
void main ( ) {
    short a = 10;
    long b = 20;
    printf("a=%d, b=%d \n", a, b);    // short와 long 모두 int와 같은 방법으로 출력 할 수 있다.
}
```



결과설명 : int, short, long은 출력 방법이 동일하다.

예제 2-18 : %u를 이용한 부호없는 정수의 출력

```
#include <stdio.h>
void main ( ) {
    unsigned int a = 10;
    printf("a = %u \n", a);    // unsigned short와 long 모두 같은 방법으로 출력 할 수 있다.
}
```



결과설명 : unsigned int, unsigned short, unsigned long 형 변수는 %u로 출력한다.

예제 2-18 : %u를 이용해 음수를 출력한 모양

```
#include <stdio.h>
void main ( ) {
    short a = -1;
    printf("a = %u \n", a);    // %u는 부호없는 수만 출력함으로 음수를 출력하지 못한다.
}
```

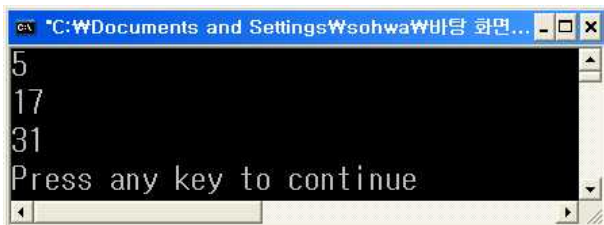




결과설명 : -1은 음수이지만 %u는 양수만 출력하는 출력 서식임으로 a에 들어 있는 -1을 양수로 인식하여 출력한다. -1은 메모리에 보관될 때 1111111111111111 로 보관됨으로 이걸 양수로 인식하면 4294967295가 된다.

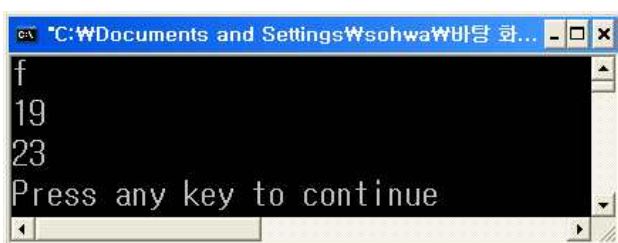
예제 2-19 : %o를 이용해 10진 정수를 8진수 형태로 출력한 예제

```
#include <stdio.h>
void main ( ) {
    printf("%o\n", 5);           // 10진수 5가 8진수로 변경되어 5로 출력된다.
    printf("%o\n", 15);          // 10진수 15가 8진수로 변경되어 17로 출력된다.
    printf("%o\n", 25);          // 10진수 25가 8진수로 변경되어 31로 출력된다.
}
```



예제 2-20 : %x를 이용해 10진 정수를 16진수 형태로 출력한 예제

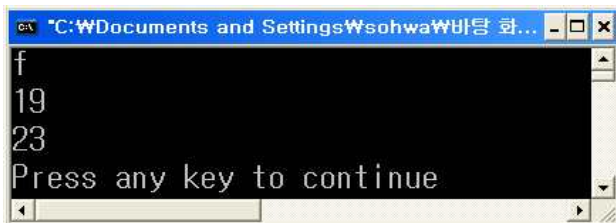
```
#include <stdio.h>
void main ( ) {
    printf("%x\n", 15);           // 10진수 5가 16진수로 변경되어 f로 출력된다.
    printf("%x\n", 25);          // 10진수 15가 16진수로 변경되어 19로 출력된다.
    printf("%x\n", 35);          // 10진수 25가 16진수로 변경되어 23으로 출력된다.
}
```





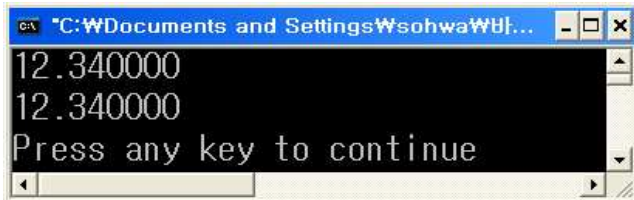
예제 2-20 : %x를 이용해 10진 정수를 16진수 형태로 출력한 예제

```
#include <stdio.h>
void main ( ) {
    printf("%x\n", 5);           // 10진수 5가 16진수로 변경되어 f로 출력된다.
    printf("%x\n", 25);          // 10진수 15가 16진수로 변경되어 19로 출력된다.
    printf("%x\n", 35);          // 10진수 25가 16진수로 변경되어 23으로 출력된다.
}
```



예제 2-21 : %f를 이용해 float형 변수와 상수의 값을 출력하는 예제

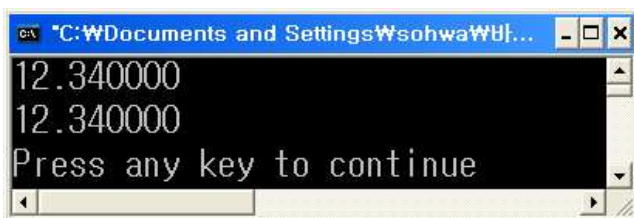
```
#include <stdio.h>
void main ( ) {
    float x = 12.34f;
    printf("%f\n", x);           // x에 들어 있는 123.45가 고정소수점 형태로 출력된다.
    printf("%f\n", 12.34f);      // 23.45가 고정소수점 형태로 출력된다.
}
```



결과설명 : 실수를 표기하면서 뒤에 f를 붙이면 float형 실수라는 것을 의미하게 된다. %f는 출력 시 자동으로 소수점 이하 6번째 자리까지 출력된다.

예제 2-22 : %lf를 이용해 double형 변수와 상수의 값을 출력하는 예제

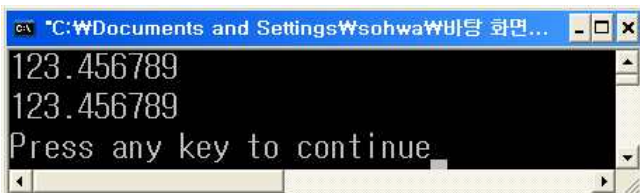
```
#include <stdio.h>
void main ( ) {
    double x = 12.34;
    printf("%lf\n", x);          // x에 들어 있는 123.45가 긴 고정소수점 형태로 출력된다.
    printf("%lf\n", 12.34);      // 23.45가 긴 고정소수점 형태로 출력된다.
}
```



결과설명 : 실수를 표기하면 자동으로 double로 인식한다. %f는 출력 시 자동으로 소수점 이하 6번째 자리까지 출력된다.

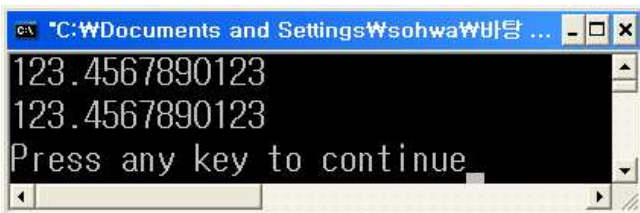
예제 2-23 : %f와 %lf를 이용해 크기가 큰 실수를 출력했을 때의 모양

```
#include <stdio.h>
void main ( ) {
    printf("%f\n", 123.4567890123);    // 소수점 이하 6자리 이상은 출력되지 않는다.
    printf("%lf\n", 123.4567890123);  // 소수점 이하 6자리 이상은 출력되지 않는다.
}
```



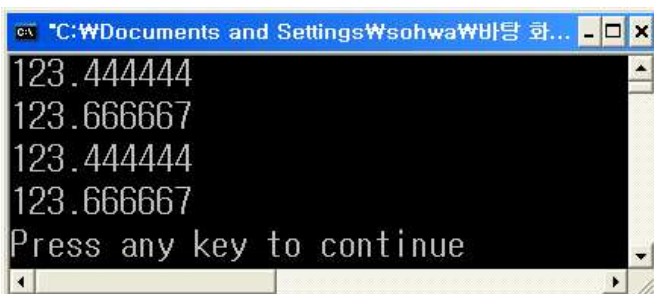
예제 2-24 : %f와 %lf를 이용해 크기가 큰 실수를 자리수를 정해 출력했을 때의 모양

```
#include <stdio.h>
void main ( ) {
    printf("%.10f\n", 123.4567890123); // 길이가 긴 실수라도 자리수를 지정하면 출력 할 수 있다.
    printf("%.10lf\n", 123.4567890123); // 길이가 긴 실수라도 자리수를 지정하면 출력 할 수 있다.
}
```



예제 2-24 : %f와 %lf의 반올림 형태

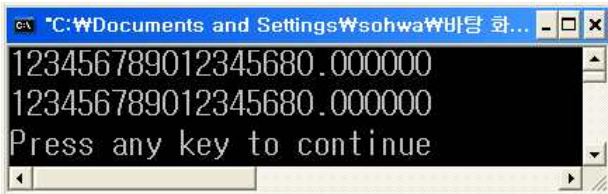
```
#include <stdio.h>
void main ( ) {
    printf("%f\n", 123.4444444); // 6자리 이하의 잘린 자리의 수가 4임으로 반올림 되지 않는다.
    printf("%f\n", 123.6666666); // 6자리 이하의 잘린 자리의 수가 6임으로 반올림 된다.
    printf("%lf\n", 123.4444444); // 6자리 이하의 잘린 자리의 수가 4임으로 반올림 되지 않는다.
    printf("%lf\n", 123.6666666); // 6자리 이하의 잘린 자리의 수가 6임으로 반올림 된다.
}
```



결과설명 : 실수를 출력 시 자리수가 잘려 출력된다면 잘려서 출력되지 않은 수는 반올림된다.

예제 2-25 : %f와 %lf의 앞자리수가 길어질 때의 예제

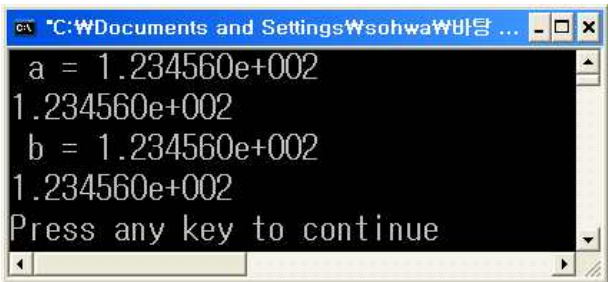
```
#include <stdio.h>
void main ( ) {
    printf("%f\n", 123456789012345679.6666666); // 앞자리가 지나치게 길면 출력 불가능한 자리
    printf("%lf\n", 123456789012345679.6666666); // 에서 반올림 처리되고 나머지는 0으로 출력된다.
}
```



결과설명 : 실수를 출력 시 자리수가 잘려 출력된다면 잘려서 출력되지 않은 수는 반올림된다.

예제 2-26 : %e와 %le서식으로 float형 변수와 상수, double형 변수와 상수를 출력한 예제

```
#include <stdio.h>
void main ( ) {
    float a = 123.456f;
    double b = 123.456;
    printf(" a = %e\n", a); // a에 들어있는 123.456이 %e(지수형)로 출력된다.
    printf("%e\n", 123.456f); // 123.456f 가 %e(지수형)로 출력된다.
    printf(" b = %le\n", b); // b에 들어있는 123.456이 %le(긴지수형)로 출력된다.
    printf("%le\n", 123.456); // 123.456f 가 %le(긴지수형)로 출력된다.
}
```



결과설명 : 지수형은 실수를 "0이 아닌수 1개만 소수점 이상에 오는 실수 \* 10의 n승" 형태로 실수를 출력하는 방식이다. 위의 실행결과인 1.234560e+002는 1.234560 \* 10의 2승을 의미한다.

예제 2-27 : %e와 %le서식의 지수가 -가 되는 예제

```
#include <stdio.h>
void main ( ) {
    printf("%e\n", 0.0012f); // 0.0012f 가 %e(지수형)로 출력된다.
    printf("%le\n", 0.0012); // 0.0012 가 %le(긴지수형)로 출력된다.
}
```

결과설명 : 1.200000e-003은  $1.200000 \times 10^{-3}$ 승이란 의미이다. 즉, 0.0012를 의미한다.

예제 2-28 : %c를 이용해 char형 변수와 상수를 출력하는 예제

```
#include <stdio.h>
void main ( ) {
    char x = 'A';
    printf("x = %c \n", x);    // 변수 x에 들어있는 대문자 A가 %c(문자형태)로 화면에 출력된다.
    printf("%c\n", 'A');      // 대문자 'A'가 %c(문자형태)로 화면에 출력된다.
}
```

결과설명 : 문자 상수는 반드시 표시할 때 ' '로 묶어서 표시해야 한다.

예제 2-28 : %c를 이용해 연산의 결과와 정수를 출력하는 예제

```
#include <stdio.h>
void main ( ) {
    char x = 'A';
    printf("x+2 = %c\n", x+2); // 변수 x에 들어있는 A+2의 결과인 67를 ascii코드로 하는 문자 C가
                                // 출력된다.
    printf("%c\n", 68);       // ascii코드 68에 해당하는 문자 D가 출력된다.
}
```

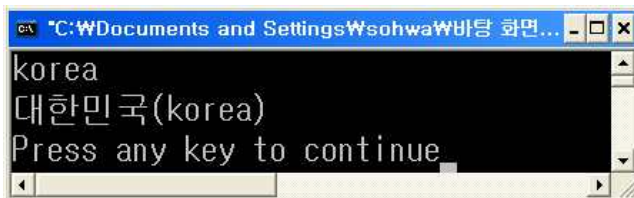
예제 2-28 : %c를 이용해 문자 출력 시 문자상수를 ' '로 묶지 않으면 에러가 발생한다.

```
#include <stdio.h>
void main ( ) {
    printf("%c\n", Z);        // Z를 변수로 인식해 해당 변수가 없다는 에러가 발생한다.
}
```

Compiling...  
 Cpp1.cpp  
 C:\Documents and Settings\sohwa\바탕 화면\Cpp1.cpp(3) : error C2065: 'Z' : undeclared identifier  
 Error executing cl.exe.  
 Cpp1.exe - 1 error(s), 0 warning(s)

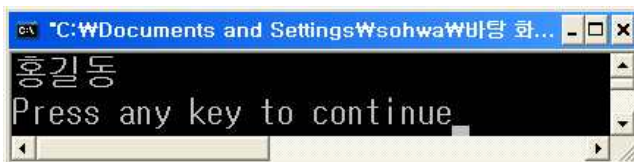
예제 2-29 : %s를 이용해 문자열을 출력하는 방법

```
#include <stdio.h>
void main ( ) {
    printf("%s\n", "korea");           // korea가 %s(문자열형태)로 출력된다.
    printf("대한민국(%s)\n", "korea"); // 대한민국(korea)가 출력된다.
}
```



예제 2-30 : %s를 이용해 배열 안에 있는 문자열을 출력하는 방법

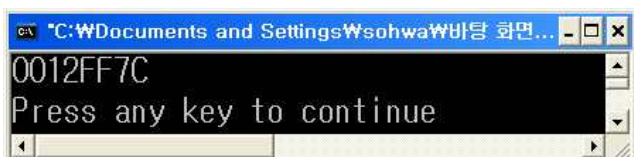
```
#include <stdio.h>
void main ( ) {
    char name[]="홍길동";
    printf("%s\n", name);           // char형 배열의 이름을 %s로 출력하면 배열안에 보관되어
                                   // 있는 문자열이 화면에 %s(문자열 형태)로 출력된다.
}
```



결과설명 : 배열에 보관되어 있는 문자열을 출력하는 방법이다. 배열에 있는 문자열을 printf( ) 함수로 출력할 경우 배열의 이름을 %s서식으로 출력하면 된다. 자세한 설명은 차후 배열 챕터 및 포인터 챕터에서 다시 설명한다.

예제 2-31 : %p를 이용해 주소값을 출력하는 방법

```
#include <stdio.h>
void main ( ) {
    int a = 10;
    printf("%p\n", &a);           // 변수 a의 주소값이 %p(포인터형 - 16진 정수8자리)로 출력된다.
}
```



결과설명 : 변수의 주소값을 printf( ) 함수를 이용해 출력하는 방법이다. 자세한 설명은 차후 포인터 챕터에서 다시 설명한다.

예제 2-32 : printf( ) 함수의 각종 변환 서식 지정자 사용방법

```
#include <stdio.h>
void main ( ) {

    printf("+++%d+++Wn", 12);        // 12를 %d로 +++사이에 출력한다.
    printf("+++%3d+++Wn", 12);       // 공간을 3칸 확보하고 확보된 공간 안에 12를 출력한다.
    printf("+++%5d+++Wn", 12);       // 공간을 5칸 확보하고 확보된 공간 안에 12를 출력한다.

    printf("Wn");
    // 공간확보 작업을 하는 이유
    printf("떡볶이 = %7d 원Wn", 15000);
    printf("오  뎡 = %7d 원Wn", 700);
    printf("순  대 = %7d 원Wn", 4500);
    printf("도너츠 = %7d 원Wn", 60);
    printf("김  밥 = %7d 원Wn", 218000);

    printf("Wn");
    printf("+++%-3d+++Wn", 12);      // 공간을 3칸 확보하고 공간 안에 왼쪽 정렬해 12를 출력한다.
    printf("+++%-5d+++Wn", 12);      // 공간을 5칸 확보하고 공간 안에 왼쪽 정렬해 12를 출력한다.

    printf("Wn");
    printf("+++%03d+++Wn", 12);      //공간을 3칸 확보하고 그 안에 12를 출력하되 빈칸을 0으로 채운다.
    printf("+++%05d+++Wn", 12);      //공간을 5칸 확보하고 그 안에 12를 출력하되 빈칸을 0으로 채운다.

    printf("Wn");
    printf("+++%f+++Wn", 12.3);      // 12.3을 출력한다.
    printf("+++%5f+++Wn", 12.3);     // 공간을 5칸 확보하고 그 안에 12.3을 출력한다.
    printf("+++%-5f+++Wn", 12.3);    // 공간을 5칸 확보하고 그 안에 12.3을 왼쪽 정렬해 출력한다.
    printf("+++%05f+++Wn", 12.3);    // 공간을 5칸 확보하고 그 안에 12.3을 출력한뒤 빈칸을 0으로
                                    // 채운다.

    printf("Wn");
    printf("+++%.0lf+++Wn", 12.3456); // 12.3456을 소수점이하 0자리까지 출력한다.
    printf("+++%.2lf+++Wn", 12.3456); // 12.3456을 소수점이하 2자리까지 출력한다.
    printf("+++%.4lf+++Wn", 12.3456); // 12.3456을 소수점이하 4자리까지 출력한다.
    printf("+++%.8lf+++Wn", 12.3456); // 12.3456을 소수점이하 8자리까지 출력한다.

    printf("Wn");
    printf("+++%-7.3lf+++Wn", 12.3456); // 12.3456을 공간을 7칸 확보하고 확보된 공간내에서 왼쪽
                                    // 정렬해 출력하되 소수점 이하 3자리 까지만 출력한다.

    printf("Wn");
    printf("%sWn", "korea");         // "korea"를 문자열로 출력해라
    printf("%.2sWn", "korea");       // "korea"를 앞에서부터 두 글자만 문자열로 출력해라
    printf("%.4sWn", "korea");       // "korea"를 앞에서부터 네 글자만 문자열로 출력해라
}
```

```

C:\Documents and Settings\Wsohwa\바탕 화면\WDebug...
+++12+++
+++ 12+++
+++ 12+++

떡볶이 = 15000 원
오 덴 = 700 원
순 대 = 4500 원
도너츠 = 60 원
김 밥 = 218000 원

+++12 +++
+++12 +++

+++012+++
+++00012+++

+++12.300000+++
+++12.300000+++
+++12.300000+++
+++12.300000+++

+++12+++
+++12.35+++
+++12.3456+++
+++12.34560000+++

+++12.346 +++

korea
ko
kore
Press any key to continue

```

결과설명 : 주석을 참조한다.

## 6. scanf( )의 사용방법

scanf( ) 함수 : 함수내에서 지정한 서식에 맞게 원하는 내용을 입력 할 수 있도록 해주는 함수

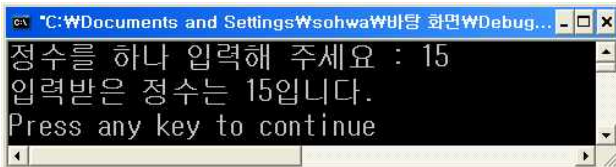
scanf( ) 함수의 원형 : `int scanf("cotrolstring", ... );`

scanf( ) 함수의 특징 : scanf( ) 함수는 여러 가지 종류의 data를 원하는 서식에 맞추어 입력 할 수 있도록 해주는 함수이다. 따라서 scanf( ) 함수의 가로안에 있는 문자열 내부에 사용할 서식을 기술해야 하며 " " 뒤쪽에 입력받을 값을 저장할 메모리의 주소값을 기재해야 한다. 사용 할 수 있는 서식의 종류는 printf( ) 함수의 서식표와 동일하지만 모든 변환서식 지정자는 사용불가하며 입력서식을 제외한 어떠한 내용도 " " 안에 기재하면 안 된다. 따라서 실제 사용되는 scanf( ) 함수의 형태는 scanf("입력서식", &저장할변수명); 이 된다. 단, 문자열의 경우 배열에 이름에 입력을 받아야 하며 배열명 앞쪽에 &를 쓰지 않아야 한다. (자세한 설명은 포인터 챕터 참조) printf( ) 함수와 마찬가지로 scanf( ) 함수에 사용한 입력서식과 입력받은 값을 저장할 변수의 수는 같아야 한다.



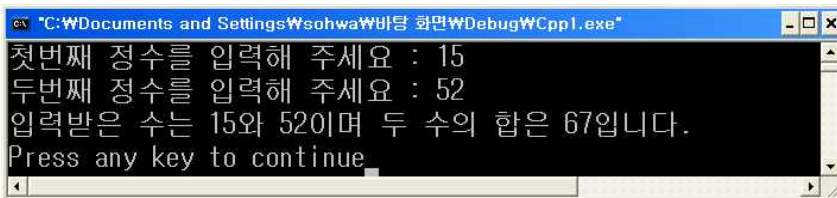
예제 2-33 : scanf( ) 함수를 이용해 정수를 입력받는 방법 - 1

```
#include <stdio.h>
void main ( ) {
    int a;                // 입력받은 정수를 보관할 변수 a를 선언
    printf("정수를 하나 입력해 주세요 : ");
    scanf("%d", &a);      // int, short, long 과 같은 정수는 입력 받을 때 %d를 이용해
                        // 입력받아야 하며 변수명 a를 쓸 때에는 앞에 &를 붙여야 한다.
    printf("입력받은 정수는 %d입니다.\n", a );
}
```



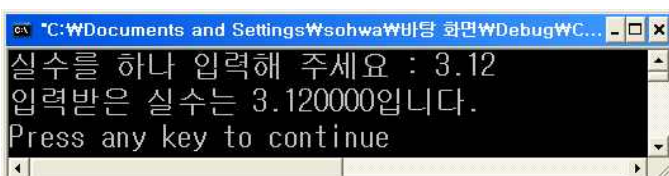
예제 2-34 : scanf( ) 함수를 이용해 정수를 입력받는 방법 - 2

```
#include <stdio.h>
void main ( ) {
    int a, b;              // 입력받은 정수를 보관할 변수 a와 b를 선언
    printf("첫번째 정수를 입력해 주세요 : ");
    scanf("%d", &a);       // 입력받은 정수를 변수 a에 보관시킨다.
    printf("두번째 정수를 입력해 주세요 : ");
    scanf("%d", &b);       // 입력받은 정수를 변수 a에 보관시킨다.
    printf("입력받은 수는 %d와 %d이며 두 수의 합은 %d입니다.\n", a, b, a+b );
}
```



예제 2-35 : scanf( ) 함수를 이용해 float형 변수에 값을 입력받는 방법

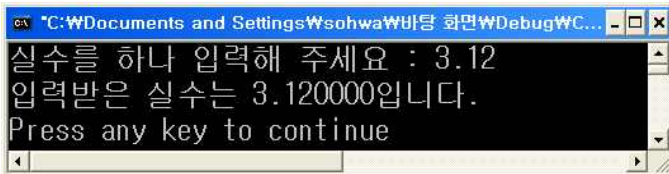
```
#include <stdio.h>
void main ( ) {
    float a;               // 입력받은 실수를 보관할 변수 a를 선언
    printf("실수를 하나 입력해 주세요 : ");
    scanf("%f", &a);       // float은 입력 받을 때 %f를 이용해 입력받아야 하며
                        // 변수명 a를 쓸 때에는 앞에 &를 붙여야 한다.
    printf("입력받은 실수는 %f입니다.\n", a );
}
```





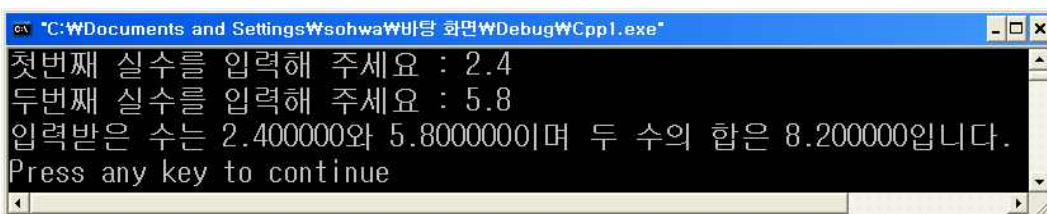
예제 2-36 : scanf( ) 함수를 이용해 double형 변수에 값을 입력받는 방법

```
#include <stdio.h>
void main ( ) {
    double a;                // 입력받은 실수를 보관할 변수 a를 선언
    printf("실수를 하나 입력해 주세요 : ");
    scanf("%lf", &a);        // double형은 입력 받을 때 %lf를 이용해 입력받아야 하며
                                // 변수명 a를 쓸 때에는 앞에 &를 붙여야 한다.
    printf("입력받은 실수는 %lf입니다.\n", a );
}
```



예제 2-34 : scanf( ) 함수를 이용해 float형 변수와 double형 변수에 값을 입력받는 방법

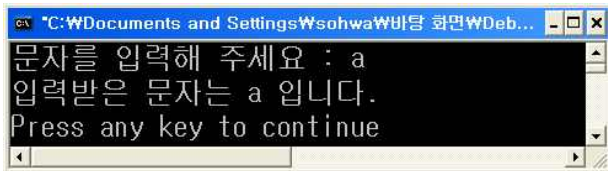
```
#include <stdio.h>
void main ( ) {
    float a;
    double b;                // 입력받은 실수를 보관할 변수 a와 b를 선언
    printf("첫번째 실수를 입력해 주세요 : ");
    scanf("%f", &a);          // float은 입력받을 때 %f로 서식을 지정해야 한다.
    printf("두번째 실수를 입력해 주세요 : ");
    scanf("%lf", &b);          // double은 입력받을 때 %lf로 서식을 지정해야 한다.
    printf("입력받은 수는 %f와 %lf이며 두 수의 합은 %lf입니다.\n", a, b, a+b );
}
```



결과설명 : 정수나 실수의 경우에는 연달아 여러개의 자료를 입력받을 경우에도 fflush(stdin)을 실행할 필요가 없다.

예제 2-35 : scanf( ) 함수를 이용해 char형 변수에 문자를 입력받는 방법

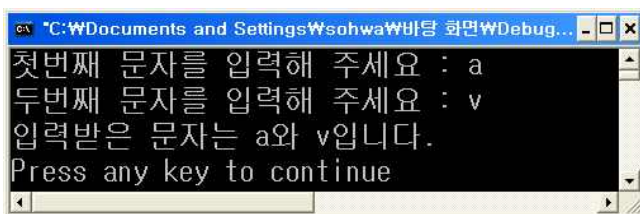
```
#include <stdio.h>
void main ( ) {
    char a;
    printf("문자를 입력해 주세요 : ");
    scanf("%c", &a);          // 문자는 입력받을 때 입력서식을 %c로 지정해야 한다.
    printf("입력받은 문자는 %c 입니다.\n", a);
}
```



예제 2-36 : scanf ( ) 함수를 이용해 char형 변수를 여러개 입력받는 방법

```
#include <stdio.h>
void main ( ) {
    char a, b;                // 입력받은 문자를 보관할 변수 a와 b를 선언
    printf("첫번째 문자를 입력해 주세요 : ");
    scanf("%c", &a);          // char형 변수는 %c로 서식을 지정하여 입력 받는다.
    printf("두번째 문자를 입력해 주세요 : ");
    fflush(stdin);            // char를 입력 받기 전 키보드 버퍼에 기존에 입력된 문자가 남아
                                // 있다면 반드시 fflush(stdin)을 실행시켜 주어야 한다.

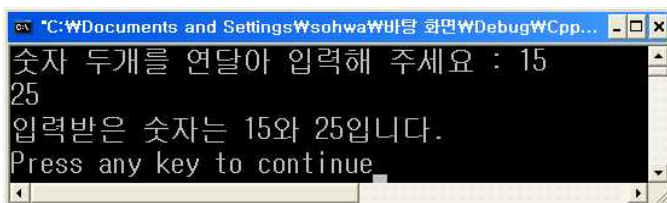
    scanf("%c", &b);
    printf("입력받은 문자는 %c와 %c입니다.\n", a, b);
}
```



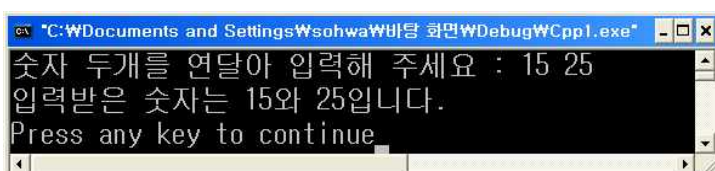
예제 2-37 : scanf ( ) 함수를 이용해 여러개의 변수를 한 번에 입력받는 방법

```
#include <stdio.h>
void main ( ) {
    int a, b;                 // 입력받은 정수를 보관할 변수 a와 b를 선언
    printf("숫자 두개를 연달아 입력해 주세요 : ");
    scanf("%d%d", &a, &b);    // 연달아 입력받기 위해 서식과 변수명을 두 개 사용한다.
    printf("입력받은 숫자는 %d와 %d입니다.\n", a, b);
}
```

하나씩 입력하고 엔터를 쳤을 경우에 나타난 프로그램의 실행 결과



스페이스바를 이용해 연달아 숫자를 입력했을 경우에 나타난 프로그램의 실행 결과



결과설명 : 자료를 연달아 입력할 경우 스페이스바를 이용해 자료를 구분하여 한 번에 입력할 수 있다.

예제 2-38 : scanf( ) 함수를 사용할 경우 자주 틀리는 것에 대해 알아보는 예제

```
#include <stdio.h>
void main ( ) {
    int a, b;
    float c;
    double d;
    char e;

    printf("정수를 입력해 주세요 : ");
    scanf("a를 입력 : %d", &a);    // scanf( ) 함수의 “ ” 내에는 입력서식을 제외한 어떠한 문장도
                                   // 쓸 수 없다. 올바르게 자료가 입력되지 않는다.

    printf("정수를 입력해 주세요 : ");
    scanf("%d\\n", &a);    // scanf( ) 함수내의 서식에 \\n을 쓰면 올바른 데이터를 입력 할 수 없다.

    printf("정수를 입력해 주세요 : ");
    scanf("%d", a);        // 변수명 앞에 &를 붙이지 않으면 에러가 발생한다.

    printf("실수를 입력해 주세요 : ");
    scanf("%d", &c);        // 실수를 %d로 입력받으면 쓰레기 값이 입력된다.

    printf("실수를 입력해 주세요 : ");
    scanf("%lf", &c);    // float을 %lf로 입력받으면 쓰레기 값이 입력 될 수 있다.

    printf("실수를 입력해 주세요 : ");
    scanf("%f", &d);        // double을 %f로 입력받으면 쓰레기 값이 입력 될 수 있다.

    printf("실수를 입력해 주세요 : ");
    scanf("%.2lf", &d);    // 입력서식에는 .2와 같은 소수점 자리수 지정이 불가하다.

    printf("문자를 입력해 주세요 : ");
    scanf("%c", &e);    // char 입력 전에는 fflush(stdin)을 한번 해주는 것이 좋다.
}
```