
C 언어

- 전처리기
- 표준 출력 함수
- 표준 입력 함수

전처리기

전처리기 (Preprocessor)

- 프로그래머가 원하는 사항을 컴파일러에게 직접 지시하는 문법
- 전처리기 문법은 # 기호로 시작하며 기계어로 번역되는 명령문이 아니기 때문에
- 문법 끝에 ; 세미콜론을 사용하지 않는다.

#include 전처리기

#include 전처리기는 컴파일러에 자신이 명시한 파일을 읽도록 지시한다.

#include

읽을 파일 이름

#include <헤더 파일 이름>

비주얼 스튜디오에서 제공하는 헤더 파일을 포함할 때

#include "헤더 파일 이름"

프로그래머가 정의해 사용하는 헤더 파일을 포함할 때

#include "MyMath.h"

#include <stdio.h>

#include "C:\download\MyMath.h"

#include <string.h>

전처리기

#define 전처리는 상수나 명령문을 교체하는 문법이다.

#define 문법으로 상수 치환하기

#define	MAX_COUNT	3
	치환할 이름	치환할 내용

```
#define MAX_COUNT 3 /* 3이라는 상수를 MAX_COUNT 로 치환 */  
int data = MAX_COUNT;
```

#define 문법으로 명령 치환하기

#define	POW_VALUE(a)	(a * a)
		명령

매크로 함수
(Macro Function)

```
int data = POW_VALUE(3); /* int data = (3 * 3); 으로 번역됨 */
```

C 표준 라이브러리

운영체제 별로 기술적 기반이 다르기 때문에 화면에 문자 A 를 출력하는 과정과 방법도 다르다!

C 언어가 문자 출력 기능을 문법으로 제공한다면?

환경이 변화했을 때 이를 수용하기 위해 C 언어 문법까지 변경해야 하는 문제가 발생
한다!

C 언어를 기계어로 번역하는 컴파일러의 소스코드를 변경하는 것!

C 언어는 문자를 출력하는 것과 같이 시스템에 영향을 받는
요소들은 고정된 문법으로 제공하지 않는다.



C 언어는 자신의 기능을 확장하기 위해 함수라는 개념을 제공한다!

하지만 프로그래머가 직접 문자 출력 함수를 만드는 것은 어려운 일!

C 언어 표준 함수

C 언어 컴파일러를 만든 회사에서 운영체제 별로 다양한 기능을 구현하여 제공하는 함수

C 표준 라이브러리

C 언어 표준 함수들이 정의된 라이브러리

표준 출력 함수

표준 출력

컴퓨터의 표준 출력은 모니터이다!

- 표준 출력이란 해당 시스템이 가장 기본으로 사용하는 출력 방식을 말한다.
- 프로그래머가 프로그램을 만들 때 해당 시스템에서 표준인 출력 방식 중 하나를 선택하여 원하는 정보를 전달할 수 있다!

표준 출력 함수

putchar, putc : 단일 문자 출력 함수

```
putchar(65); /* 아스키 값 65에 해당하는 문자 A 가 출력됨 */  
putchar('A'); /* 문자 A 가 출력됨 */
```

puts : 문자열 출력 함수

```
puts("Hi~"); /* "Hi~" 출력 후에 줄 바꿈이 일어남 */
```

문자열 출력 함수 printf

printf 함수

- 표준 출력 함수 중에서 가장 많이 사용하는 함수
- 형식화된 문자열을 출력할 수 있다. `print + format = printf`

printf 함수를 사용하는 방법

% 서식 지정 키워드를 사용하여 변수 값을 일정한 형식으로 출력할 수 있다.

키워드	%d	%f	%c	%s
출력 형식	정수(10진수)	실수	문자	문자열


```
int data = 5;  
printf("%d", data); /* data 변수에 저장된 5 라는 정수를 출력한다. */
```

출력하려는 형식과 일치하는 키워드

문자열 출력 함수 printf


printf 함수를 사용하는 방법

```
int data1 = 3;  
int data2 = 5;  
printf("%d:%d", data1, data2); /* data1, data2 변수에 저장된 정수를 출력 */
```



출력 결과 : 3 5

```
int step = 5;  
int value = 3 * step;  
  
/* 첫 번째 %d 는 step 과 짝이 되고 두 번째 %d 는 value 와 짝을 이룸 */  
printf(" 3 * %d = %d", step, value);
```



출력 결과 : 3 * 5 = 15

문자열 출력 함수 printf

printf 함수

- %c 는 변수가 가지고 있는 값을 아스키 표에 대응하는 문자로 출력한다.

```
printf( "65의 ASCII 값은 %c입니다.", 65);
```

출력 결과 : 65의 ASCII 값은 A입니다.

```
char data3 = 65;
```

```
/* 첫 번째 data는 %c 와 짝이되고 두 번째 data는 %d와 짝을 이룸 */
```

```
printf( "%c의 ASCII 값은 %d입니다.", data3, data3);
```

출력 결과 : A의 ASCII 값은 65입니다.

문자열 출력 함수 printf

printf 함수

- 실수와 정수는 숫자를 표현하는 방식이 다르다

```
float value = 2.1f;  
printf("%f", value);
```

출력 결과 : 2.100000

```
/* value 변수 값을 실수 형식과 정수 형식으로 출력함 */  
printf("%f", %d", value, value);
```

출력 결과 : 2.100000, -107374182

실수 데이터를 정수 표현 키워드인 %d
로 출력하면 엉뚱한 값이 출력된다

문자열 출력 함수 printf

printf 함수

- %d와 %u는 변수 크기를 **4바이트 값으로 변환해서 출력한다.**



부호가 있는 1바이트(8비트) char 형에서 음수 최댓값 -1의 비트 패턴



부호가 없는 1바이트(8비트) char 형에서 최댓값 255의 비트 패턴

```
char data4 = -1;  
printf( "%d, %u", data4, data4);
```

예상 출력 결과 : -1, 255

실제 출력 결과 : -1, 4294967295

예상 출력 결과와 실제 출력 결과가 다른 이유?

%d 또는 %u 가 4바이트(32비트) 크기의 정수 기반으로 값을 출력하기 때문!

문자열 출력 함수 printf

- %o는 8진수, %x는 16진수 형태로 정수를 출력한다.

```
int data11 = 10; /* 10진수 → 10진수로 10 */  
int data12 = 012; /* 8진수 → 10진수로 10 */  
int data13 = 0x10; /* 16진수 → 10진수로 16 */
```

```
/* data1은 16진수, data2는 10진수, data3은 8진수로 출력함 */  
printf("%x, %d, %o", data11, data12, data13);
```

출력 결과 : a, 8, 20

- %e 는 실수를 지수 형태로 출력한다.

```
float data = 12.34;  
printf("%f, %e, %E", data, data, data);
```

출력 결과 : 12.340000, 1.234000e+-1,
1.234000E+01

- 콘솔 모드에서 출력되는 문자의 개수와 위치
- 출력할 문자의 개수보다 칸을 더 사용하여 다음과 같이 출력하면 빈칸과 숫자가 마치 **그래프**처럼 표시되어 값을 정확히 확인하지 않고도 어느 정도 범위에 있는 값인지 알 수 있다.

9	9	9	9	9
			5	0
	1	9	3	2

[illegible]

- 출력할 문자의 개수보다 칸을 더 사용하여 다음과 같이 출력하면 빈칸과 숫자가 마치 그래프처럼 표시되어 값을 정확히 확인하지 않고도 어느 정도 범위에 있는 값인지 알 수 있다.

9	9	9	9	9
			5	0
	1	9	3	2

문자열 출력 함수 printf

출력 칸 수 조절하기

```
int data21 = 7;  
/* 자릿수 확인을 위해 [ ] 문자를 사용함 */  
printf( "[%d] [%5d]", data21, data21);
```

출력 결과 : [7] [7]

오른쪽 정렬과 왼쪽 정렬

```
/* 자릿수 확인을 위해 [ ] 문자를 사용함 */  
printf( "[%5d] [%05d] [%-5d]", data21, data21, data21 );
```

오른쪽 공백대신 왼쪽
정렬 0채움 정렬

출력 결과 : [7] [00007] [7]

문자열 출력 함수 printf

실수의 소수점 자릿수 지정하기

실수는 출력한 칸을 지정하는 것 외에도 . 마침표를 사용하여 소수점 이하 자릿수를 몇 자리까지 출력할 것인지를 명시할 수 있다.

%

전체 칸 수

.

소수점 자릿수

f

```
double data31 = 3.141592;
```

```
/* 자릿수를 확인하기 위해 [ ] 문자를 사용함 */
```

```
printf("[%f] [%.4f] [%8.4f] [%-8.4f]", data31, data31, data31, data31 );
```

출력 결과 : [3.141592] [3.1416] [3.1416] [3.1416]

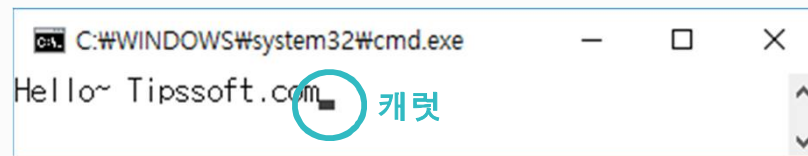
```
/* printf 함수에서 % 자체를 출력하고 싶다면 %%라고 쓰면 된다 */
```

```
printf("95%%");
```

문자열 출력 함수 printf

제어 코드 사용하기

아스키 표에는 소리를 내거나 콘솔의 출력과 입력의 현재 위치를 알려주는 **캐럿(Caret, 문자로 깜빡임)** 위치를 변경할 수 있는 제어코드가 있다.



제어 코드는 키보드에 있는 문자가 아니기 때문에 제어코드를 직접 입력할 수 없기 때문에 C 언어는 **\(백슬래시)** 와 키워드로 제어코드를 사용할 수 있는 문법을 제공한다.

제어 코드	기능
\n	캐럿을 다음 줄로 이동(Line Feed)
\r	캐럿을 해당 줄의 처음으로 이동(Carriage Return)
\t	캐럿을 한 탭만큼 이동
\b	캐럿을 바로 앞 칸으로 이동
\a	시스템 스피커로 경고음 발생
\"	큰따옴표 출력
\'	작은따옴표 출력

문자열 출력 함수 printf. ex05-02-caret.c

Wn 제어코드 : 캐럿을 다음 줄로 이동

```
/* Wn 을 사용하면 현재 출력된 문자열이 있는 다음 줄로 캐럿이 이동한다 */  
printf( "Hello~ Tipssoft.comWn" );
```

Hello~ Tipssoft.com



캐럿

Wr 제어코드 : 캐럿을 해당 줄의 처음으로 이동

```
/* Wr 을 사용하면 현재 줄의 처음으로 캐럿이 이동한다 */  
printf( "Hello~ Tipssoft.comWn" );  
printf( "Hello~ Tipssoft.comWrHi~~~~" );
```

Hello~ Tipssoft.com

Hi~~~~~

캐럿

문자열 출력 함수 printf

Wt 제어코드 : 캐럿을 한 탭만큼 이동

```
/* Wt 탭의 위치 이동을 확인하기 위해서 8의 배수로 출력함 */  
printf("12345678123456781234567812345678Wn");  
printf("aWtbcWtdefWtg");
```

12345678123456781234567812345678
a bc def g  캐럿

Wb 제어코드 : 캐럿을 바로 앞 칸으로 이동

```
/* Wb 의 b 는 back 을 의미한다. 캐럿이 한 칸 앞으로 이동한다. */  
printf("1234567Wb");
```

1234567  캐럿

문자열 출력 함수 printf

Wa 제어코드 : 경고음을 발생

Alert 의 약자로, 이 제어문을 사용하면 스피커로 짧은 경고음이 발생한다.

그 밖의 제어 코드

```
/* “ 큰 따옴표가 출력 형식의 일부이기 때문에 오류가 발생한다 */  
printf(“ “Hello” ”);
```

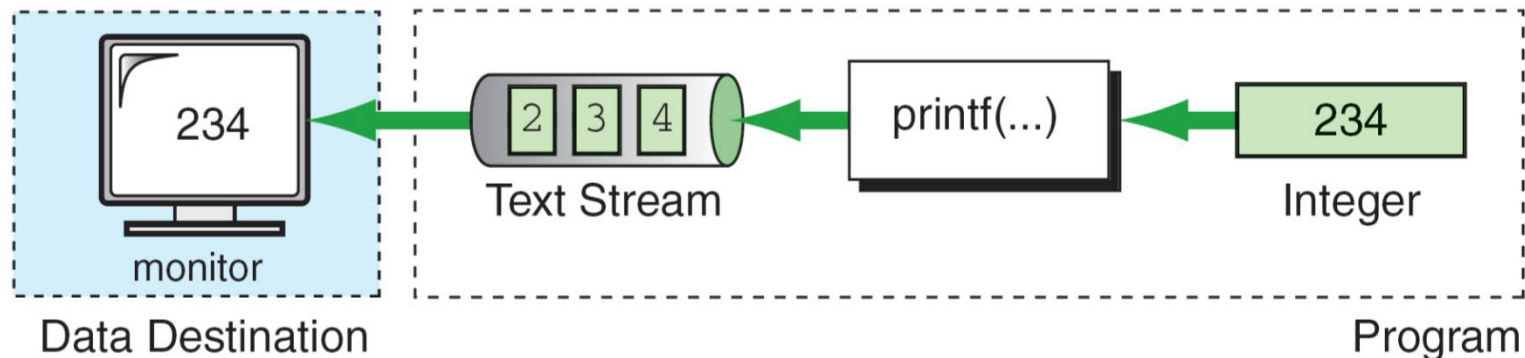
```
/* ₩를 사용하여 큰따옴표의 의미를 단순 문자로 변경한다 */  
printf(“ ₩“ Hello ₩” ”);
```

```
/* ₩를 사용하여 작은 따옴표의 의미를 단순 문자로 변경한다 */  
printf(“₩’ Tipsware ₩’ ”);
```

```
/* % 를 출력하고 싶을 때에는 %를 두 번 사용하면 된다*/  
printf(“ 작년 대비 생산률이 20%% 증가”);
```

Output Formatting : printf

- C에서 출력을 하는 방법은 여러가지가 있지만 가장 쉽게 쓸 수 있는 것은 printf(print formatted)이다.

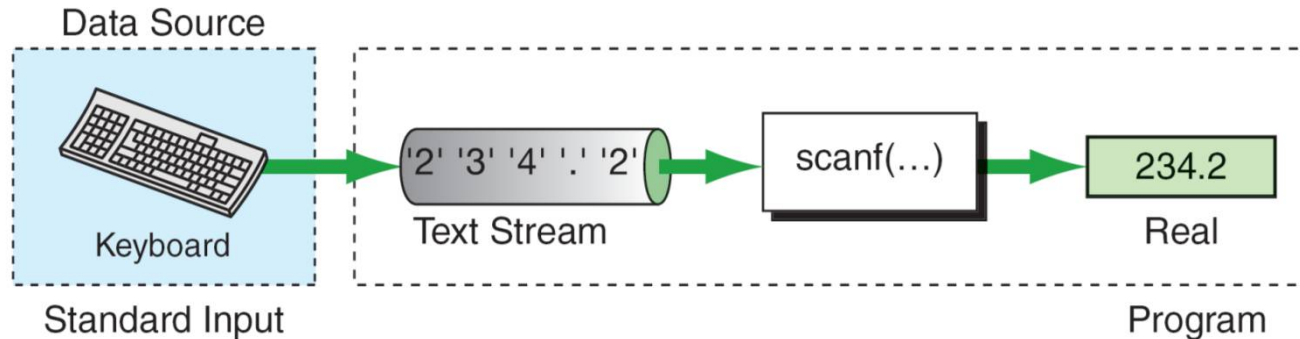


- “ ” 안에 출력할 내용을 적는다.
- 변수의 값을 출력할 때는 “ ” 안의 해당 위치에 형식문자(%d, %c, %f, %s)를 적고, “ ” 뒤에 해당하는 변수를 순서대로 적는다.

```
printf (" ... %d ... %f ...", int형변수, float형변수);
```

Input Formatting: scanf

- C에서 입력을 받는 방법 중 가장 쉽게 쓸 수 있는 것은 scanf()이다.



- `scanf()`는 도스창인 표준 입력으로부터 여러 종류의 자료 값을 입력 받을 수 있음
- 사용법은 `printf()`와 유사
- 단, 입력 값을 저장할 변수의 이름 앞에 '`&`'를 반드시 붙여줘야 한다.
(`&`은 해당 변수의 주소를 의미하는 연산자)

```
scanf("%d%f", &age, &weight);
```

실습예제. ex05-11.c

- 한개의 정수값을 입력 받아 그 값을 출력하시오

- INPUT

- ◆ 입력값: 3

- OUTPUT

- ◆ 출력값: 3

- 실행결과예시

입력값: 3
출력값: 3

실습예제. ex05-12.c

- 정수 입력 받고 그 값을 아스키코드 값으로 출력하시오.
- 단, 입력되는 정수값은 128 이하의 값으로 한다.

■ 실행결과예시

```
input : 65  
output : A
```

```
input : 100  
output : d
```

ASCII Table

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

Source: www.asciitable.com

실습예제. ex05-13.c

- 생년 월일을 입력 받아서 3개의 정수 변수에 저장하고 출력하시오.

- 실행결과예시

```
input year   : 1988
input month  : 3
input date   : 22
Your birthday is 1988. 3. 22.
```


실습예제. ex05-14.c

- 다섯 개의 double 를 입력 받고, 이 숫자들을 한 줄에 하나씩 출력하시오. 단, 모든 숫자는 오른쪽 정렬되며 소수점 아래로는 3자리까지만 출력해야 합니다.
- 실행결과예시

```
input : 1.2 34.567 890.12345 67.8 9012.4567
[      1.200]
[     34.567]
[    890.123]
[     67.800]
[   9012.457]
```