

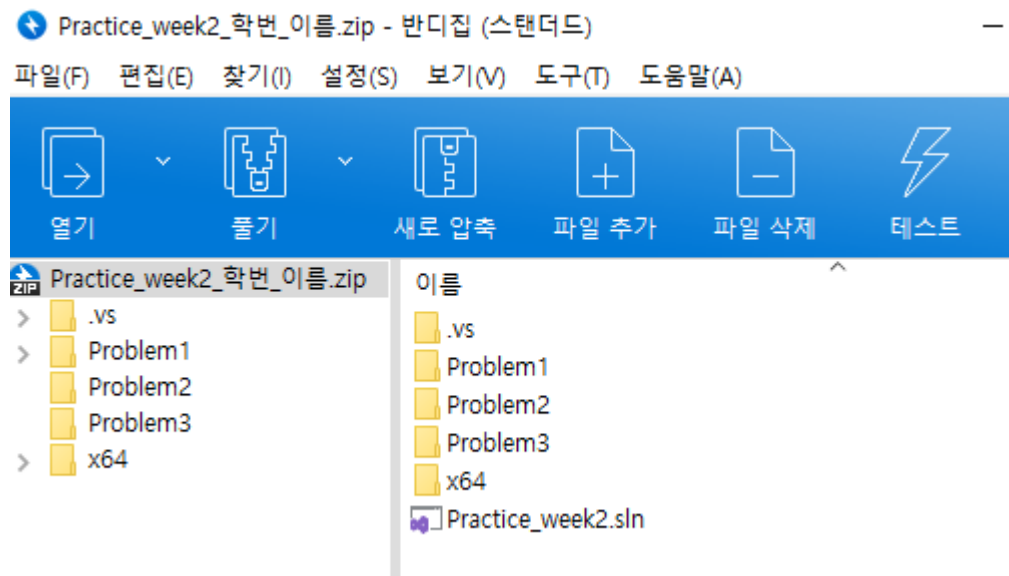
컴퓨터 프로그래밍 및 실습

2주차. 변수와 자료형

1. 변수와 상수

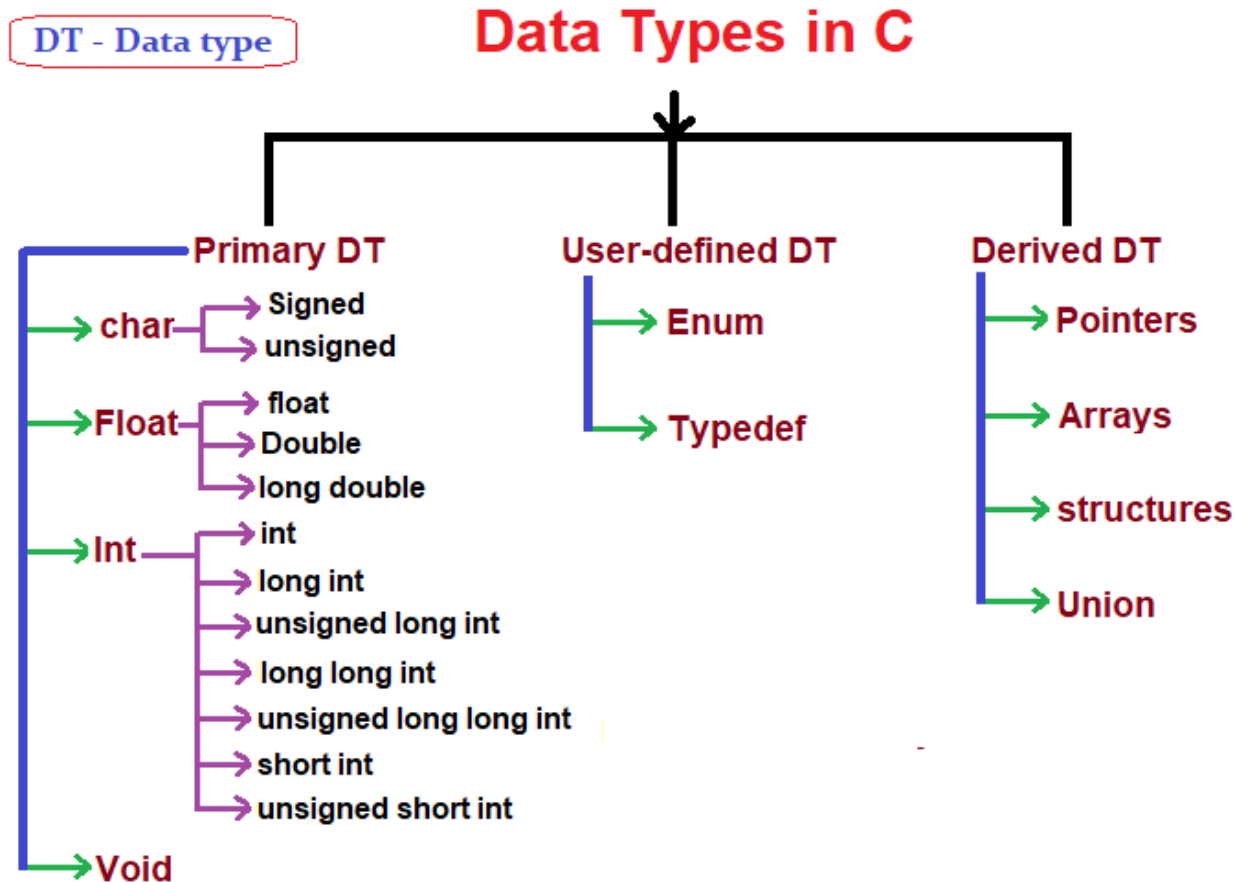
■ 실습 제출 안내

- 솔루션 이름은 Practice_week2
- 프로젝트 이름은 Problem1, Problem2, ...
- 솔루션 폴더를 압축하여 Practice_week2_학번_이름.zip 으로 제출



1. 변수와 상수

■ 자료형(Data type)



1. 변수와 상수

■ 자료형(Data type)

```
#include <stdio.h>

int main(void) {
    // (자료형) (변수명); - 변수의 선언
    // (변수명) = (값);      - 변수의 초기화

    // (자료형) (변수명) = (값); - 변수의 선언과 초기화를 동시에

    // 선언과 초기화를 동시에
    int a = 7;
    double k = 1.5;
    char str[] = "Hello";
    char c = 'Y';

    // 선언먼저, 초기화는 나중에
    int b;
    scanf("%d", &b);
}
```

1. 변수와 상수

■ 자료형(Data type)

```
#include <stdio.h>

int main(void) {
    int x = 5;    // 변수의 초기화
    int y;        // 변수 선언은 했으나, 초기화는 안함
    int z = x + y; // 초기화 된 변수와 초기화 되지 않은 변수의 연산

    printf("%d", z);
}
```

오류 목록				검색 오		
전제 솔루션	1 오류	1 경고	0/1 메시지	빌드 + IntelliSense		
코드	설명	프로젝트	파일	줄		
▶ C6001	초기화되지 않은 메모리 'y'를(들) 사용하고 있습니다.	Variable1	index.c	6		
✖ C4700	초기화되지 않은 'y' 지역 변수를 사용했습니다.	Variable1	index.c	6		

1. 변수와 상수

■ printf, scanf에서 사용되는 format specifiers

Format Specifier	Type
%c	Character
%d	Signed integer
%e or %E	Scientific notation of floats
%f	Float values
%g or %G	Similar as %e or %E
%hi	Signed integer (short)
%hu	Unsigned Integer (short)
%i	Unsigned integer
%l or %ld or %li	Long
%lf	Double
%Lf	Long double
%lu	Unsigned int or unsigned long
%lli or %lld	Long long
%llu	Unsigned long long
%o	Octal representation
%p	Pointer
%s	String
%u	Unsigned int
%x or %X	Hexadecimal representation
%n	Prints nothing
%%	Prints % character

```
#include <stdio.h>

int main(void) {
    printf("%d\n", 5); // 정수 출력
    printf("%c\n", 'H'); // 문자 출력
    printf("%s\n", "Hello"); // 문자열 출력

    int score = 94;
    char name[] = "장인호";
    int age = 23;
    printf("\n\n%d살인 %s 학생이 %d점을 맞았다.\n", age, name, score);
    return 0;
}
```

1. 변수와 상수

■ 실습 1

- 화씨온도를 받아서 섭씨온도로 변환
- $C = (F - 32) * 5 / 9$
- 변환된 섭씨온도를 아래와 같이 출력

연산자	기호	사용예	결과값
덧셈	+	$7 + 4$	11
뺄셈	-	$7 - 4$	3
곱셈	*	$7 * 4$	28
나눗셈	/	$7 / 4$	1
나머지	%	$7 \% 4$	3

$$\begin{aligned} y &= mx + b & \rightarrow y &= m * x + b; \\ y &= ax^2 + bx + c & \rightarrow y &= a * x * x + b * x + c; \\ m &= \frac{x + y + z}{3} & \rightarrow m &= (x + y + z) / 3; \end{aligned}$$

- "(화씨온도)는 섭씨온도로 (섭씨온도)입니다."

1. 변수와 상수

■ 실습 2

- 반지름을 입력 받아, 원의 넓이를 출력하는 프로그램
- Pi 값은 3.14
- 출력 시 "r=(반지름), A=(넓이)" 형태로 출력

1. 변수와 상수

- 실습 3
 - 실습 1, 실습 2를 scanf를 사용해서 개발해보자

2. Number data


■ 정수형

❖ 각 자료의 최대값과 최소값은 [limits.h](#)에 정의되어 있음

자료형			비트	범위
정수형	short	부호있는 정수	16비트	-32768~32767
	int		32비트	-2147483648~2147483647
	long			-2147483648~2147483647
	long long		64비트	-9,223,372,036,854,775,808 ~9,223,372,036,854,775,807
	unsigned short	부호없는 정수	16비트	0~65535
	unsigned int		32비트	0~4294967295
	unsigned long			0~4294967295
	unsigned long long		64비트	0~18,446,744,073,709,551,615

2. Number data

- 정수형



```
#include <stdio.h>
#include <limits.h>

int main(void) {
    int val = INT_MAX;
    int overflow_test = val + 1;

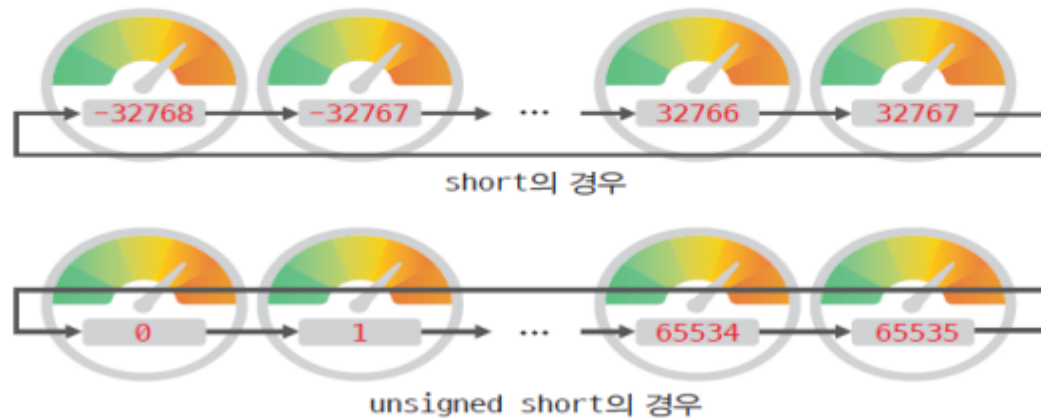
    printf("val = %d\n", val);
    printf("overflow_test = %d\n", overflow_test);
    return 0;
}
```

2. Number data

■ 정수형

오버플로우 (1/2)

- ❖ 변수가 나타낼 수 있는 범위(최댓값, 최솟값)를 넘는 숫자를 저장하려고 하면 오버플로우(overflow) 발생
- ❖ 한계를 벗어나면 다시 처음으로 돌아 감
- ❖ 오버플로우가 발생하더라도 컴파일러는 아무 경고도 하지 않음



2. Number data

- 정수형

	2,147,483,647
HEX	7FFF FFFF
DEC	2,147,483,647
OCT	17 777 777 777
BIN	0111 1111 1111 1111 1111 1111 1111 1111

	$2147483647 + 1 =$
	-2,147,483,648
HEX	8000 0000
DEC	-2,147,483,648
OCT	20 000 000 000
BIN	1000 0000 0000 0000 0000 0000 0000 0000

- signed 자료형은 앞자리가 부호를 표시하는데 사용된다.

2. Number data

■ 정수형

■ 정수를 표현하는 다른 방법

정수형 상수 (1/2)

- ❖ 소수점 없이 정수형으로 선언한 변수에 할당(대입)되는 값
- ❖ 모든 정수형 상수 앞에 +, - 부호를 붙일 수 있음 *없으면 양의 정수
- ❖ 상수의 자료형을 명시할 수 있음 (예) sum = 123L;
- ❖ 정수형 상수는 10진수(decimal)뿐만 아니라 8진수(octal), 16진수(hexa)

정수형도 표현할 수 있음

접미사	자료형	예
u 또는 U	unsigned int	123u 또는 123U
l 또는 L	long	123l 또는 123L
ul 또는 UL	unsigned long	123ul 또는 123UL

10진수	8진수	16진수
0	00	0x0
1	01	0x1
2	02	0x2
3	03	0x3
4	04	0x4
5	05	0x5
6	06	0x6
7	07	0x7
8	010	0x8
9	011	0x9
10	012	0xa
11	013	0xb
12	014	0xc
13	015	0xd
14	016	0xe
15	017	0xf
16	020	0x10
17	021	0x11
18	022	0x12

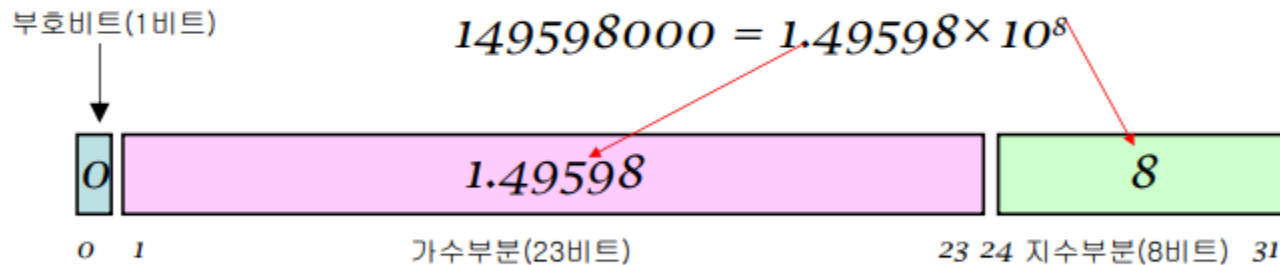
- 이진수는 0b숫자 형식으로 표현할 수 있다. (ex: 0b11010)

2. Number data

■ 실수형

❖ 실수를 표현하는 방법 #2: 부동 소수점 방식

- 소수점이 떠서 움직인다는 것을 의미
- 표현할 수 있는 범위가 대폭 늘어남
- 10^{-38} 에서 10^{+38}




2. Number data

■ 실수형

오버플로우

- ❖ 변수가 나타낼 수 있는 범위(최댓값, 최솟값)를 넘는 숫자를 저장하려고 하면 오버플로우(overflow) 발생
- ❖ 한계를 벗어나면 다시 처음으로 돌아 감
- ❖ 오버플로우가 발생하더라도 컴파일러는 아무 경고도 하지 않음

```
#include <stdio.h>
int main(void)
{
    float x = 1e39;
    float y = 1.23456e-46;
    printf("x=%e\n", x);
    printf("y=%e\n", y);
    return 0;
}
```



Microsoft Visual Studio 디버그 콘솔

x=inf
y=0.000000e+00

2. Number data

■ 실습 4

- 부동소수점 형식의 연산을 알아보자.
- 사용자로부터 실수 두개를 입력 받는다.
- 입력 받은 두 실수의 합을 출력한다.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void) {
    double a, b;

    scanf("%lf %lf", &a, &b);
    printf("%.20lf\n", a + b);
    return 0;
}
```

C# Microsoft Visual Studio 디버그

```
0.1 0.2
0.3000000000000000004441
```

3. Character

■ Character – 문자 타입

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

3. Character

■ Character - 문자 타입

- C언어는 문자를 ASCII 코드로 표현한다.
 - 컴퓨터는 이진수로 데이터를 표현
 - 이진수는 숫자??
 - 문자도 당연히 숫자(이진수)로 각각 알파벳을 대치하여 저장 및 처리

```
#include <stdio.h>

int main(void) {
    char a = 72;
    char b = 69;
    char c = 76;
    char d = 79;

    printf("%c%c%c%c%c\n", a, b, c, c, d);
    return 0;
}
```

3. Character

■ 실습 5

- 사용자로부터 알파벳 두개를 입력 받는다.
- 입력 받은 두 알파벳 간의 거리를 출력하는 프로그램을 개발

- A B 두개가 입력되었을 경우, 답은 1
- B D \rightarrow 2

4. Type Casting

■ 자료형 변환

- ❖ 자료형 변환은 변수명 앞에 선언해 놓은 기존 자료형을 다른 자료형으로 재정의하는 것을 의미함
- ❖ 자동 자료형 변환과 강제 자료형 변환이 있음

변수의 타입이 변경되는
것이 아니고 변수에
저장되는 데이터의
타입이 변경됨

유형	설 명
자료형	char → short → unsigned int → long → unsigned long → float → double → long double
수식	<ul style="list-style-type: none">• char형과 short형은 int형으로 변환• int형과 float형은 double형으로 변환• 피연산자 둘 중 하나라도 double형이면 다른 연산자와 결과값도 double형으로 변환• 피연산자 둘 중 하나라도 unsigned형이면 다른 연산자와 결과값도 double형으로 변환
대입문	<ul style="list-style-type: none">• 모든 문자형 상수는 int형으로 변환되어 저장• 오른쪽 값이 왼쪽 자료형으로 변환되어 저장• 실수형이 정수형으로 변환될 때 소수점 이하는 버린 후 저장

[Visual Studio에서 적용되는 자동 자료형 변환 규칙]

4. Type Casting

■ 자료형 변환

- a / b 는 정수와 정수의 나눗셈
- $a / (\text{double})b$ 는 정수와 실수의 나눗셈
- 정수 끼리의 연산에는 자동 형변환이 개입하지 않는다.
- int와 double 끼리의 연산에서, int가 double로 바뀌고 결과값도 double로 출력

```
#include <stdio.h>

int main(void) {
    int a = 7;
    int b = 4;

    printf("%lf\n", a / b);
    printf("%lf\n", a / (double)b);
}
```

5. Visual Studio 사용법 설명

■ 코드 작성 시 유의사항

- **Ctrl+space**: 코드힌트 적극 활용 - 개발 시간 단축 및 오타자 예방
- **Ctrl+k+d**: 문서 정렬 적극 활용 - 코드 가독성이 좋아진다
- **괄호나 따옴표는 tab으로 빠져나오기**: 손가락이 편하다
- **오류 목록 확인**: 컴퓨터는 거짓말을 하지 않는다. 컴파일이 안될 때는 빨간 줄, 오류 목록 확인
- **변수명은 가능한 영어로, 의미를 알 수 있게**
- **본인에게 맞는 타입의 폰트, 글자 크기, 테마 찾아서 설정하기**