C언어 멘토링

1주차

이승준 2020년 9월 10일

헤더파일

도서관속 각자의 역할이 있는 책들

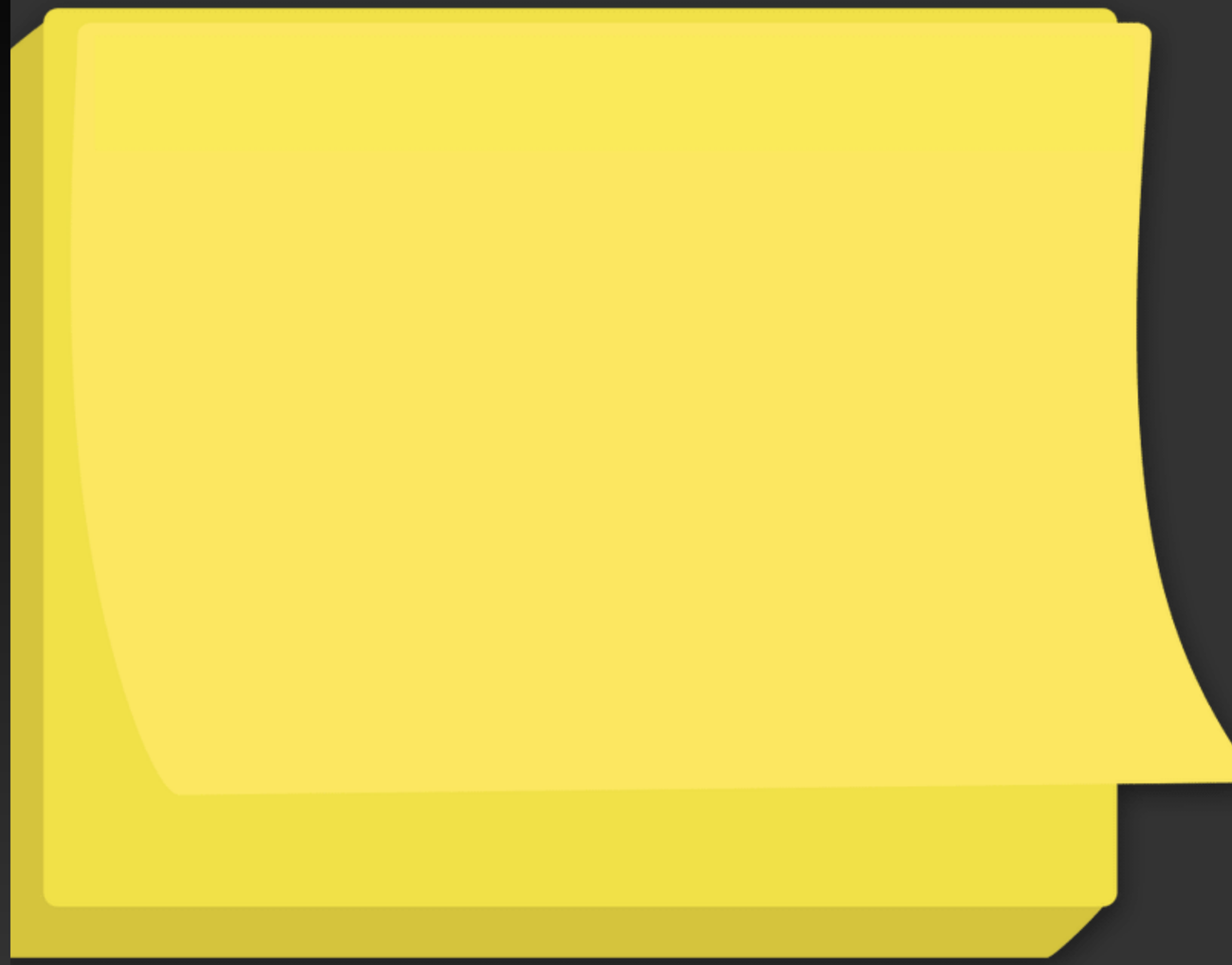
- 헤더파일 (header file)
- 표준함수를 사용하기 위해 해당 함수의 원형이 선언되어야 한다.
- 사용하고자 하는 함수를 쓰기전에 미리 헤더파일을 불러온다고 선언해줘야 한다.

헤더파일의 종류

- <stdio.h>
- <stdlib.h>
- <string.h>
- <math.h>
- <time.h>

주석

코드 속 포스트잇



주석

기억 보단 기록을 (유명한 블로그 이름)

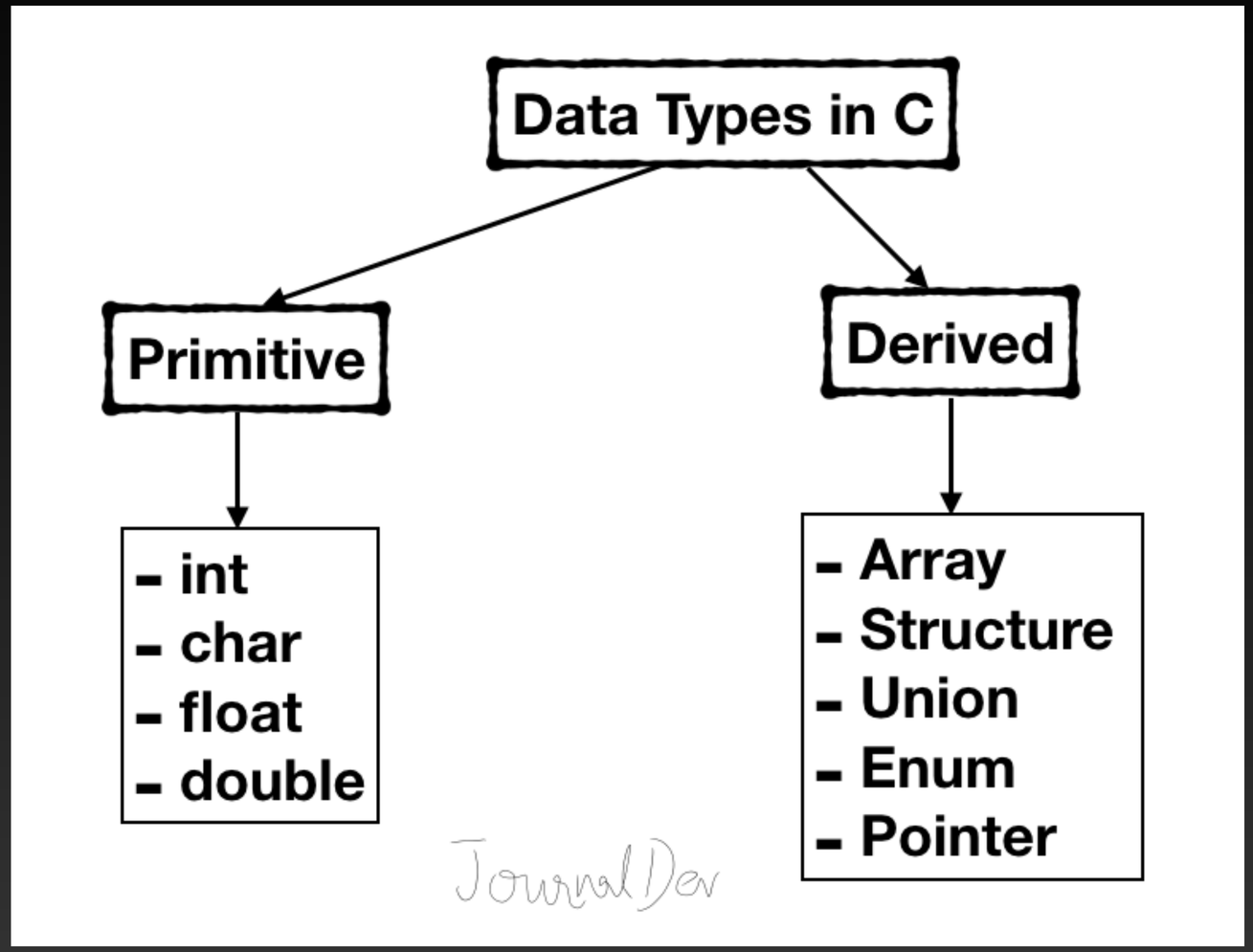
주석을 써야 하는 이유

1. 코딩을 쓰는 중간 중간에 내 생각을 정리할줄 알아야 한다.
2. 아무리 코딩을 잘 했더라도 남이 보거나, 나중에 시간이 흘러 보면 내가 쓴

   코도 한번에 이해가 가지 않는다. 따라서 포스트 잇으로 메모를 남기듯 주석을 다는게 좋다.
3. (교수님께서 해주신말) 프로그래밍은 내가 쓰는것 보다 남이 쓴 코드를 읽을 날이 많아질 것이다

자료형(DATA TYPE)

데이터를 표현하는 방법



자료형이 필요한 이유

컴퓨터 메모리에게 미리 크기와 저장방식(자료형 마다 다름)을 알려주는게 효율적이다.

(자료를 저장하는 방식이 10100110… 일지라도 각 숫자의 위치에 따라 의미하는 바가 다르다.)

자료형 종류를 먼저 한 이유 : 미리 자료형들의 종류를 알아두면 서식문자와 연산자 공부에서 조금 더 잘 알아 들을 수 있어서

Data type

정수형

name	bytes	min~max
char	1	-128~127
short	2	-32768~32767
int	4	-2147483648~ 2147483647
long	4	-2147483648~ 2147483647
long long	8	

실수형

name	bytes	소수점 이하 정밀도
float	4	6
double	8	15
long double	more than 8	more than 15

Telemetry Consent | week2_01.c

```c
#include <stdio.h>

int main (void){

  float numf = 1.0;
  double numd = 1.0;
  long double numld = 1.0;

  printf("size of %s is %lu\n","char" ,sizeof(char));
  printf("size of %s is %lu\n","short" ,sizeof(short));
  printf("size of %s is %lu\n","int" ,sizeof(int));
  printf("size of %s is %lu\n","long" ,sizeof(long));
  //이 컴파일러에서만 8로 나온다.
  printf("size of %s is %lu\n","long long" ,sizeof(long long));
  printf("size of %s is %lu\n","float" ,sizeof(numf));
  printf("size of %s is %lu\n","double" ,sizeof(numd));
  printf("size of %s is %lu\n","long double" ,sizeof(numld));
  return 0;
}
```

```
size of char is 1
size of short is 2
size of int is 4
size of long is 4
size of long long is 8
size of float is 4
size of double is 8
size of long double is 16
```

Unsigned Integer

name	bytes	min~max	unsigned	bytes	min~max
char	1	-128~127	unsigned char	1	0~(128+127)
short	2	-32768~32767	unsigned short	2	0~(32768+32767)
int	4	-2147483648~ 2147483647	unsigned int	4	0~(2147483648+ 2147483647)
long	4	-2147483648~ 2147483647	unsigned long	4	0~(2147483648+ 2147483647)
long long	8		unsigned long long	8	

ASCII 코드와 문자 그리고 정수

ASCII 코드는 10000001 은 무슨 '문자'를 표현하는 가에 대한 약속

65 = 'A' , 90 = 'Z'

97 = 'a' , 122 = 'z'

이것만큼은 꼭 외우자

이진법	팔진법	십진법	십육진법	모양	85진법 (아스키 85)	이진법	팔진법	십진법	십육진법	모양	85진법 (아스키 85)	이진법	팔진법	십진법	십육진법	모양	85진법 (아스키85)
0100000	040	32	20	␣		1000000	100	64	40	@	31	1100000	140	96	60	`	63
0100001	041	33	21	!	0	1000001	101	65	41	A	32	1100001	141	97	61	a	64
0100010	042	34	22	"	1	1000010	102	66	42	B	33	1100010	142	98	62	b	65
0100011	043	35	23	#	2	1000011	103	67	43	C	34	1100011	143	99	63	c	66
0100100	044	36	24	$	3	1000100	104	68	44	D	35	1100100	144	100	64	d	67
0100101	045	37	25	%	4	1000101	105	69	45	E	36	1100101	145	101	65	e	68
0100110	046	38	26	&	5	1000110	106	70	46	F	37	1100110	146	102	66	f	69
0100111	047	39	27	'	6	1000111	107	71	47	G	38	1100111	147	103	67	g	70
0101000	050	40	28	(	7	1001000	110	72	48	H	39	1101000	150	104	68	h	71
0101001	051	41	29	)	8	1001001	111	73	49	I	40	1101001	151	105	69	i	72
0101010	052	42	2A	*	9	1001010	112	74	4A	J	41	1101010	152	106	6A	j	73
0101011	053	43	2B	+	10	1001011	113	75	4B	K	42	1101011	153	107	6B	k	74
0101100	054	44	2C	,	11	1001100	114	76	4C	L	43	1101100	154	108	6C	l	75
0101101	055	45	2D	-	12	1001101	115	77	4D	M	44	1101101	155	109	6D	m	76
0101110	056	46	2E	.	13	1001110	116	78	4E	N	45	1101110	156	110	6E	n	77
0101111	057	47	2F	/	14	1001111	117	79	4F	O	46	1101111	157	111	6F	o	78
0110000	060	48	30	0	15	1010000	120	80	50	P	47	1110000	160	112	70	p	79
0110001	061	49	31	1	16	1010001	121	81	51	Q	48	1110001	161	113	71	q	80
0110010	062	50	32	2	17	1010010	122	82	52	R	49	1110010	162	114	72	r	81
0110011	063	51	33	3	18	1010011	123	83	53	S	50	1110011	163	115	73	s	82
0110100	064	52	34	4	19	1010100	124	84	54	T	51	1110100	164	116	74	t	83
0110101	065	53	35	5	20	1010101	125	85	55	U	52	1110101	165	117	75	u	84
0110110	066	54	36	6	21	1010110	126	86	56	V	53	1110110	166	118	76	v	
0110111	067	55	37	7	22	1010111	127	87	57	W	54	1110111	167	119	77	w	
0111000	070	56	38	8	23	1011000	130	88	58	X	55	1111000	170	120	78	x	
0111001	071	57	39	9	24	1011001	131	89	59	Y	56	1111001	171	121	79	y	
0111010	072	58	3A	:	25	1011010	132	90	5A	Z	57	1111010	172	122	7A	z	
0111011	073	59	3B	;	26	1011011	133	91	5B	[	58	1111011	173	123	7B	{	
0111100	074	60	3C	<	27	1011100	134	92	5C	\	59	1111100	174	124	7C	|	
0111101	075	61	3D	=	28	1011101	135	93	5D	]	60	1111101	175	125	7D	}	
0111110	076	62	3E	>	29	1011110	136	94	5E	^	61	1111110	176	126	7E	~	
0111111	077	63	3F	?	30	1011111	137	95	5F	_	62						

week2_01.c | week2_02.c | Telemetry Consent | Welcome | Settings

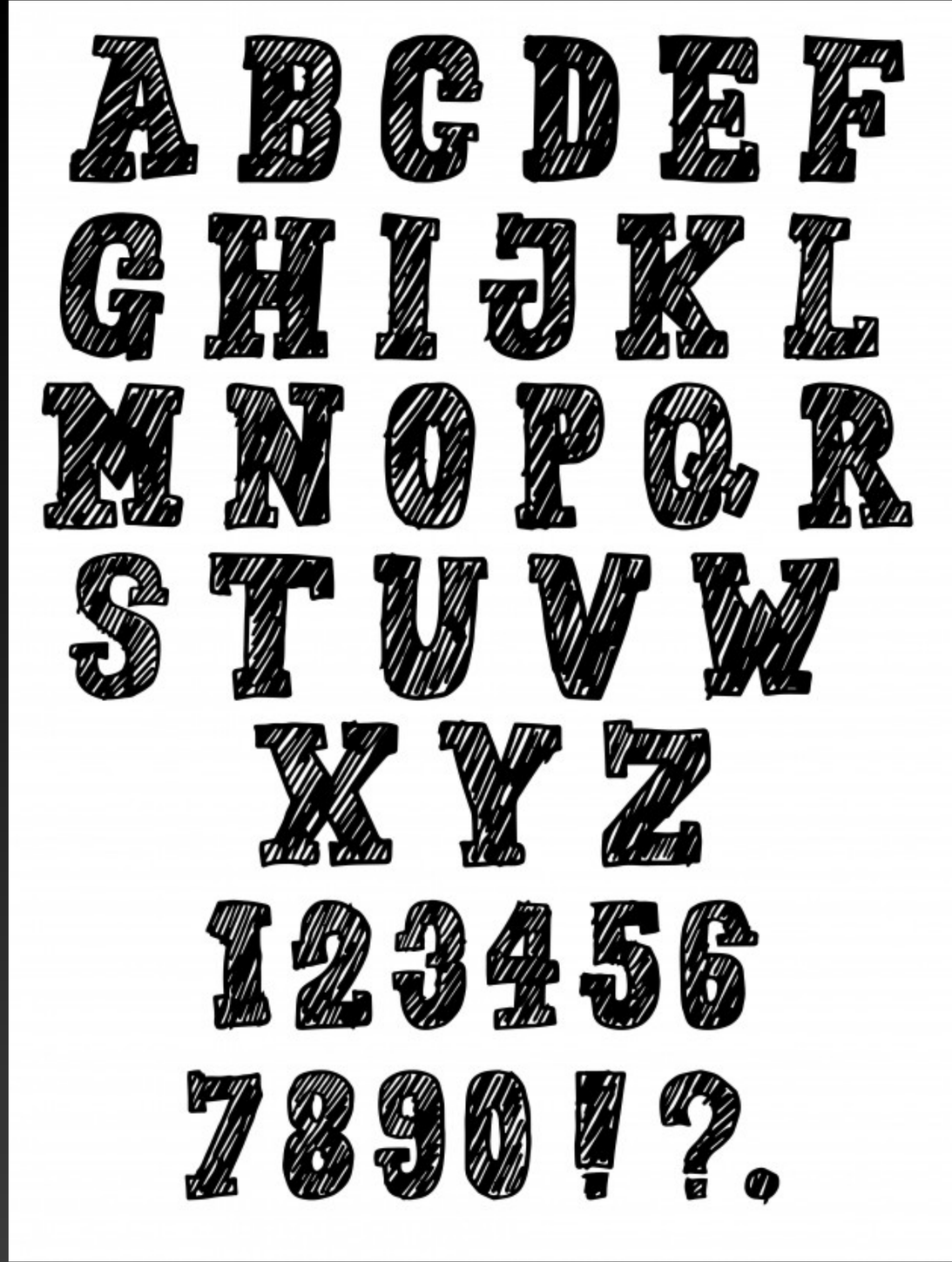
```c
#include <stdio.h>

int main (void){

  char ch1 = 'A'; char ch2 = 65;
  char ch3 = 'Z'; char ch4 = 90;

  printf("%c %d\n",ch1,ch1);
  printf("%c %d\n",ch2,ch2);
  printf("%c %d\n",ch3,ch3);
  printf("%c %d\n",ch4,ch4);

//컴퓨터는 문자를 문자가 아닌 0과 1로 저장한다.
//그리고 65와 A를 저장할때 1000001로 같은 숫자로 저장한다
//출력할때 %c로 출력하면 ASCII 코드 규칙에 의해 A 가 출력되고
//%d로 출력하면 int로 인식해서 정수인 65로 출력한다.

  return 0;
}

```

```
A 65
A 65
Z 90
Z 90
```

LF UTF-8 C GitHub Git (0)

리터럴 상수
(Literal Constant)

상수에 대한 이해



```c
#include <stdio.h>

int main(void) {
    printf("literal int size: %d\n", sizeof(7));
    printf("literal double size: %d\n",
           sizeof(7.14));
    printf("literal char size: %d\n",
           sizeof('A'));
    return 0;
}
```

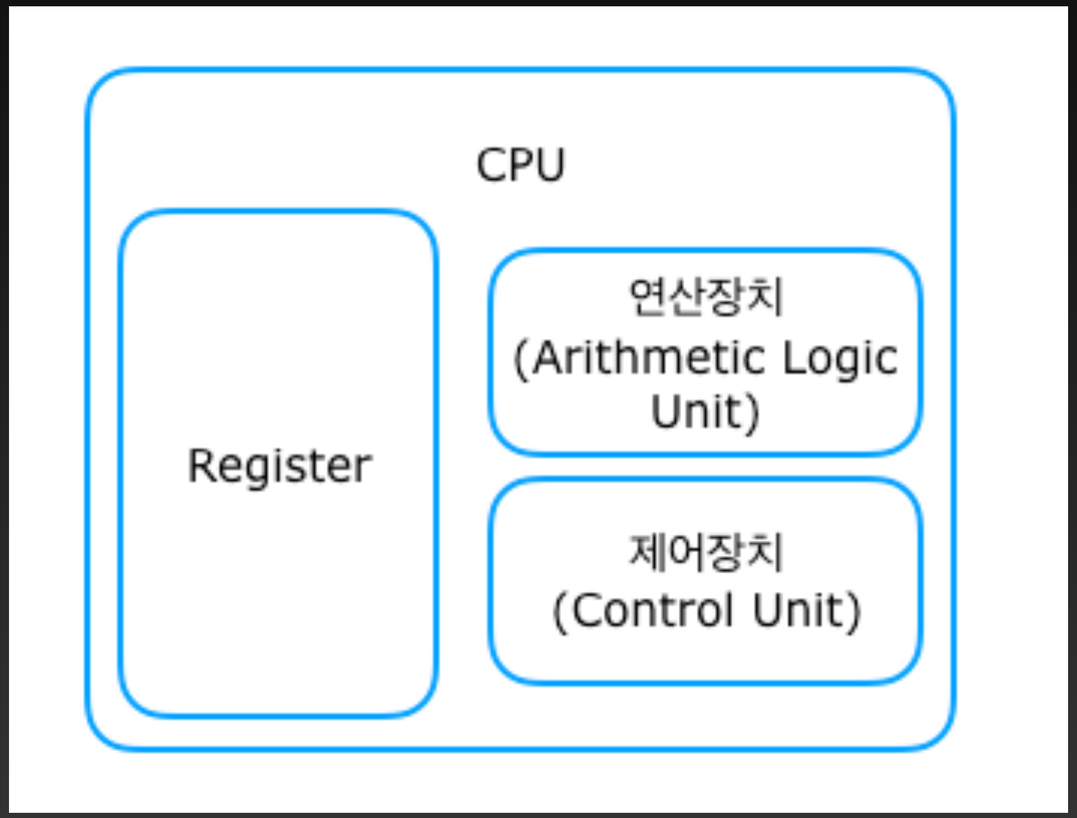
```
literal int size: 4
literal double size: 8
literal char size: 4
```

정수는 기본적으로 int 형

실수는 기본적으로 double 형

문자는 기본적으로 int 형

CPU 연산은 변수에 저장된 상수가 CPU의 연산장치로 이동하고
연산장치에서 연산이 이루어진 뒤 결과 값이 지정된 변수에 저장된다.




```c
#include <stdio.h>

int main (void){
  int num1 = 30;
  int num2 = 40;
  int result = num1 + num2;

  printf("num1 + num2 = %d\n",result);
return 0;
}

```

접미사를 통해서 float임을 명시해주는 방법

```c
#include <stdio.h>

int main(void){

    float numf1 = 2.34;
    float numf2 = 3.423 + 5.233;

    float numf3 = 3.532f;
    float numf4 = 7.323f + 8.323f;

    return 0;
}

```

접미사를 이용한 다양한 상수의 표현

접미사	자료형	사용 예
U	unsigned int	unsigned n = 523U
L	long	Long n = 523L
UL	unsigned long	unsigned long n = 523UL
LL	long long	long long n = 523LL
ULL	unsigned long long	unsigned long long n = 523ULL
F or f	float	float f = 5.23f
L or l	long double	long double d = 5.23L

직접 상수를 만들기

대문자만 사용 & 띄어쓰기는 '_' 로 표현해주기

```c
#include <stdio.h>

int main (void){

  const int MAX = 100;
  const double PI = 3.1415;
  const double E = 2.7182;
  const int HELLO_WORLD = 1;

  return 0;
}
```

형변환

자동 형변환과 강제 형변환

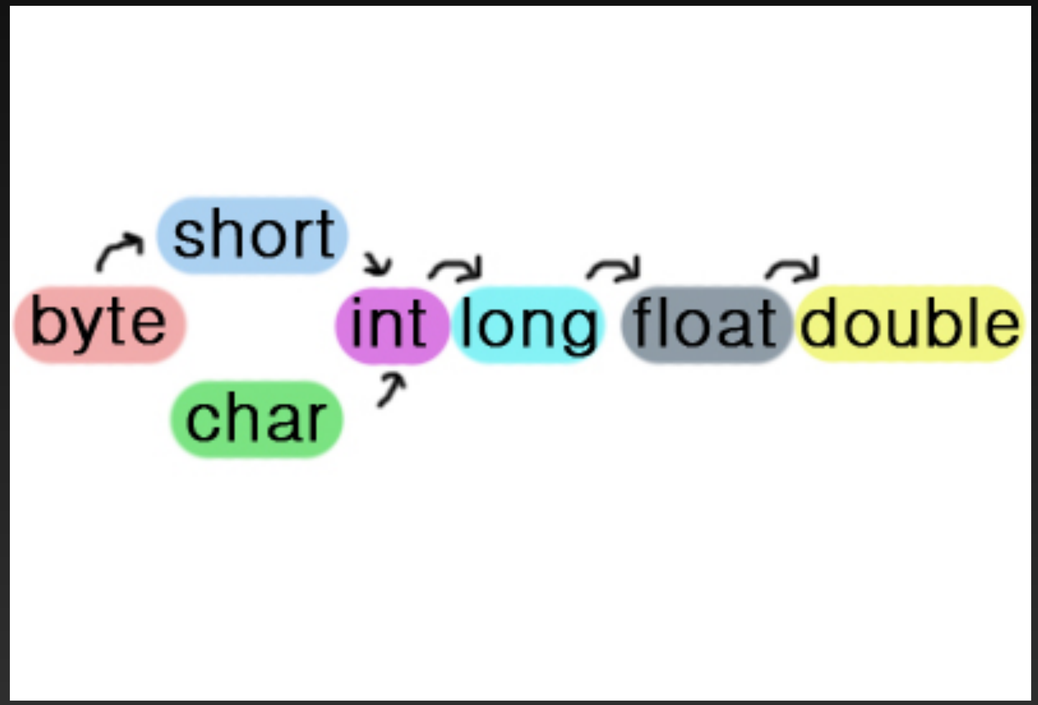
대입연산자에서 발생하는 자동형변환

```c
#include <stdio.h>

int main(void) {
    double num1 = 245;
    int num2 = 3.1415;
    int num3 = 129; char ch = num3;
    double num5 = 5.15 + 19; //자료형 불일치로 인한 자동 형변환
    int num6 = 3.424 + 244;
    printf("정수 245를 실수로: %f\n", num1);
    printf("실수 3.1415를 정수로: %d\n", num2);
    printf("큰 정수 129를 작은 정수로: %d\n", ch);
    printf("자료형 불일치로 인한 자동 형변환: %f\n", num5);
    printf("자료형 불일치로 인한 자동 형변환: %d\n", num6);
    return 0;
}
```

```
정수 245를 실수로: 245.000000
실수 3.1415를 정수로: 3
큰 정수 129를 작은 정수로: -127
자료형 불일치로 인한 자동 형변환: 24.150000
자료형 불일치로 인한 자동 형변환: 247
```

자동형변환

정수를 실수로 형 변환
  3은 3.0으로 5.는 5.0으로 (오차 발생 가능)
실수를 정수로 형 변환
  소수점 이하의 값이 소멸됨
큰 정수를 작은 정수로 형 변환
  작은 정수의 크기에 맞춰 상위 바이트 소멸



  정수승격에 의한 자동 형변환
  short , char 보다 int가 CPU의 연산(+-/*%)
  을 수행할때 더욱 적합해서 정수의 승격이 일어난다

강제형변환

```c
#include <stdio.h>

int main(void) {
    int num1 = 3, num2 = 4;
    double divResult, divResult2;
    divResult = num1 / num2;
    printf("나눗셈 결과: %f\n", divResult);
    divResult2 = (double)num1 / num2;
    printf("나눗셈 결과: %f\n", divResult2);
    return 0;
}
```

```
나눗셈 결과: 0.000000
나눗셈 결과: 0.750000
```

비트

정수 표현(2, 8,10,16 진수, MSB), (비트, 바이트)

, 실수 표현 , 연산자 (&, |, ^, ~, >>, <<)