변수와 자료

1. 강의

▼ 형식지정자

scanf 형식 지정자

Aa 형식 문자열	의미	사용 예
%d	정수 입력	int n; scanf("%d", &n);
%f, %lf	실수 입력	float f; double d; scanf("%f", &f); scanf("%lf", &d);
%c	문자 입력	char ch; scanf("%c", &ch);
%s	문자열 입력	char str[20]; scanf("%s", str);
제목 없음		

printf 형식 지정자

Aa 형식 문자열	의미	사용 예
%d	정수 출력	int n; n = 123; prnitf("n=%d \n", n);
%f	실수 출력	float f; f=3.14; printf("f=%f \n", f);
%c	문자 출력	char ch; ch = 'A'; printf("ch = %c \n", ch);
%s	문자열 출력	char str[20]; strcpy(str, "Hello"); printf("str = %s \n", str);

▼ 백슬래시(\) 문자

제어문자	기능
₩n	(리턴문자) 줄바꿈
₩t	Tab 문자
₩b	Back space
₩r	캐리지 리턴
₩0(zero)	Null 문자
₩f	폼 피드
₩a	Bell 소리
₩₩	역 슬래시
₩’	단 인용 부호 (‘)
₩”	이중 인용 부호 (“)

▼ 변수

- 변수란 어떤 값을 저장하는 공간이다.
- 저장을 하기 위해서는 메모리에 일정한 공간을 확보해야 하는데, 이때 1byte (=8 bit) 단위로 확보하게 된다.
- 저장 하게 되는 최종 크기는 저장할 값의 자료형식에 따라 다르다.
- 따라서, 어떤 변수를 적절히 저장하기 위해서는 자료형식을 결정하는 자료형 (=Data type)에 대해 알아야 한다.
- Data type 의 정의 = 자료형(資料形) 또는 데이터 타입(영어: data type)은 컴퓨터 과학과 프로그래밍 언어에서 실수치, 정수, 불린 자료형 따위의 여러 종류의 데이터를 식별하는 분류로서, 더 나아가 해당 자료형에 대한 가능한 값, 해당 자료형에서 수행을 마칠 수 있는 명령들, 데이터의 의미, 해당 자료형의 값을 저장하는 방식을 결정한다. <https://ko.wikipedia.org/wiki/자료형>
- float num; 에서 이름이 num인 변수에 자료형이 실수임을 알려주는(선언) 하는 것이다.
- 변수는 변경할 수 있는 값이고, 상수는 변경할 수 없는 값이다.
- 변수 선언과 초기화

```
char ch1;
short s_num;
int num = 10;
long l_num;
float real_number= 3.141592 ;
double d_real_number;
```

▼ 상수

- Literal : 값 자체를 사용 하는 것을 말함
- 문자형 상수 : 일반문자 ('a', 'b', 'c'), 특수 문자('\t', '\n'), 유니코드문자
- 정수형 상수 : 10진수, 16진수, 8진수, unsigned형 정수, long형 정수, unsigned long형 정수
- 실수형 상수 : 부동소수점 표기 실수, 지수 표기 실수, float형 실수
- 8진수를 표기 할때는 0 을 숫자(0~7) 앞에 붙인다.
- 16진수를 표기 할때는 0x 나 0X를 숫자(0~ F) 앞에 붙인다.
- 일반적으로 말하는 실수형은 모두 부동소수점 표기 상수로 double (8byte) 형이다. float 형인 경우에는 뒤에 f, F를 써준다.

상수형	구분	예
문자형 상수	일반 문자 상수	'a', 'b', 'c'
	특수 문자 상수	'₩t', '₩n', '₩₩', '₩007', '₩xa'
	유니코드 문자 상수	L'a', L'b', L'c'
정수형 상수	10진수 상수	10, -10
	16진수 상수	0xabcd, 0X12EF
	8진수 상수	012, 0234
	unsigned 형 상수	123u, 123U
	long 형 상수	123456l, 123456L
	unsigned long 형 상수	12345678ul, 12345678UL
실수형 상수	부동소수점 표기 상수	3.1425, -0.12345
	지수 표기 상수	3.5e13, 4.5E-30
	float 형 상수	3.14f, 3.14F

▼ 매크로 상수

- 매크로 상수는 #define문으로 정의되는 상수이다. (프로그램에서 자주 사용되는 상수를 등록)
- 지정 형식

  #define 매크로명 값

  #define PI 3.141592 //예시
- 매크로 상수는 전처리기가 처리하며, PI 로 되어 있는 곳을 모두 3.141592로 변경한다.
- " PI " 안에 있는 PI 는 전처리기에서 변경하지 않는다.
- 매크로 상수 값을 프로그램 안에서 변경하면 error 가 발생한다.

▼ const 변수

- 변수 선언시 const 를 앞에 적으면 값을 변경할 수 없게 된다.
- const 데이터형 변수명 = 초기값;

```
const double PI = 3.14;
const int    MAX_COUNT = 100;
const char   DEFAULT_CHOICE = 'Y';
```

- const 변수의 값을 변경하면 error 발생 → 그래서, 선언시 초기화 해야 한다. 그렇지 않으면 쓰레기 값을 갖게 되는데, 한번 입력된 쓰레기 값은 변경할 수 없다.

▼ 자료형

c 프로그램에서 사용되는 모든 변수나 상수는 정해진 데이터형을 갖는다.

자료 형식

Aa 데이터의 유형	데이터 유형
기본형	문자형 : char 정수형 : short, int, long 실수형 : float, double
파생 데이터 형	배열, 포인터
사용자 정의형	구조체

▼ 문자형

- char 형으로 제공, 1byte 크기
- 예> char 형의 변수에 'A'를 저장하면 65 에 해당하는 값이 저장 된다.

▼ 정수형

- short, int, long 형을 제공한다.
- 크기 순 : short < int < long 로 "정의" 하고 있다.
- 32비트 플랫폼에서는 short - 2 byte, int - 4 byte, long - 4byte 를 사용한다.
- signed int 는 부호가 있는 정수형으로, signed 를 생략할 수 있다.
- +, - 를 나타내는 부호 표시는 최상위에 있는 비트를 1로 하여 음수를 0 일때는 양수를 표시한다.

▼ 양수 정수에 대해 음수 정수를 구하는 법 (2의 보수 이용)

1. 양수의 2진수를 구한다.
2. 구해진 2진수의 0은 1로, 1은 0으로 바꾼다. (~ 비트 NOT 연산자)
3. 그리고 맨 마지막 비트에 1을 더해 준다.

- unsigned 부호 없는 정수형으로, signed 형에서 + 에서 - 까지의 영역을 모두 양수로 사용하기 때문에 표현 할 수 있는 양수 자료의 유효 범위가 2배가 된다.

▼ 변수의 자료형식이 변수에 저장된 값의 의미를 결정한다.

이진수 0100 0001 은 char 형으로 해석하면 A 이고, 십진수 정수형으로 생각하면 65이다.

따라서, 문자형인 char도 1바이트 크기의 정수형인것처럼 signed, unsigned 를 사용 할 수 있다.

▼ 데이터형의 유효범위 → limits.h 출력해보기

- 유효 범위들은 참고 사항 참조
- 오버플로우도 참고 사항 참조

▼ 실수형

- 고정소수점 방식과 부동소수점 방식으로 표현
- 부동소수점 방식에서 float (4 byte) 32 비트 중 8 비트를 지수부로 사용
- double(8 byte) 형은 64 비트 중 11 비트를 지수부로 사용
- 실수형 오버플로우는 최대값 이상에서는 무한대, 최소값 이하에서는 0 이 된다.

▼ 참고사항

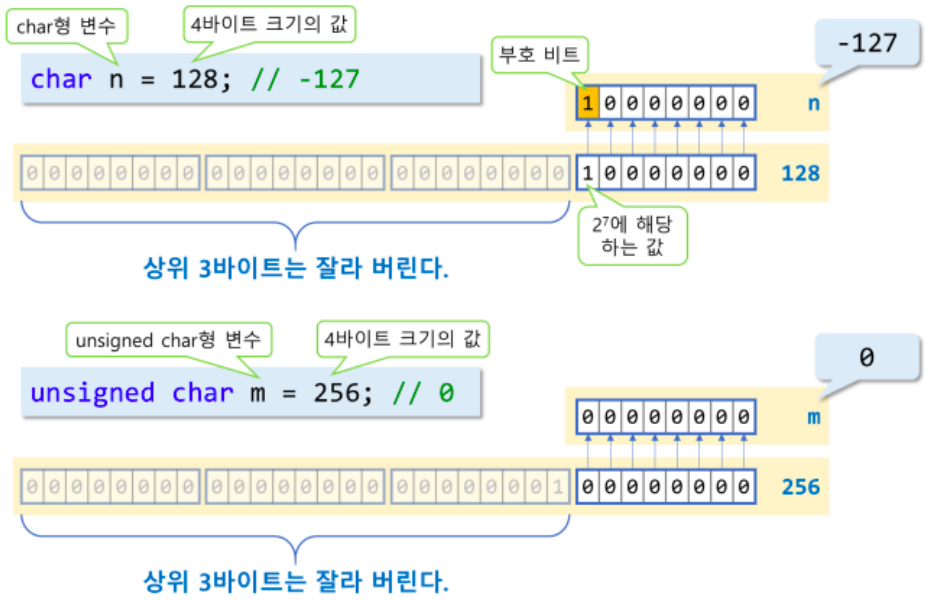
▼ 출력 형식 지정자들

서식문자	출력대상 (자료형)	출력형태
%d	char,short,int	부호 있는 10진수 정수
%ld	long	부호 있는 10진수 정수
%lld	long long	부호 있는 10진수 정수
%u	unsigned int	부호 없는 10진수 정수
%o	unsigned int	부호 없는 8진수 정수
%x, %X	unsigned int	부호 없는 16진수 정수
%f	float, double	10진수 방식의 부동소수점 실수
%Lf	long double	10진수 방식의 부동소수점 실수
%e, %E	float, double	e 또는 E 방식의 부동소수점 실수
%g, %G	float, double	값에 따라 %d와 %e 사이에서 선택
%c	char, short, int	값에 대응하는 문자
%s	char *	문자열
%p	void *	포인터 주소 값

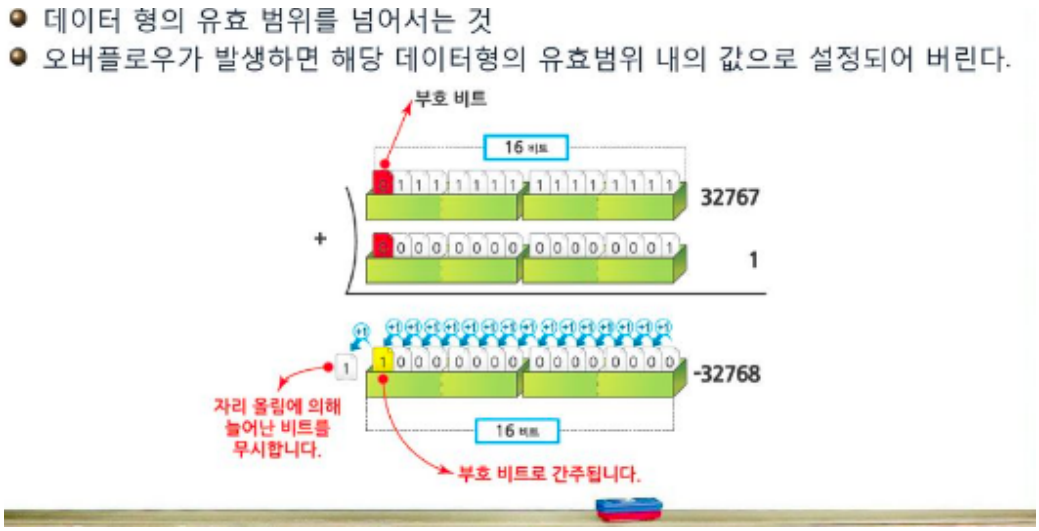
- 기호 상수(매크로 상수와 const 변수)는 하드코딩을 방지하는데 효율적이다.

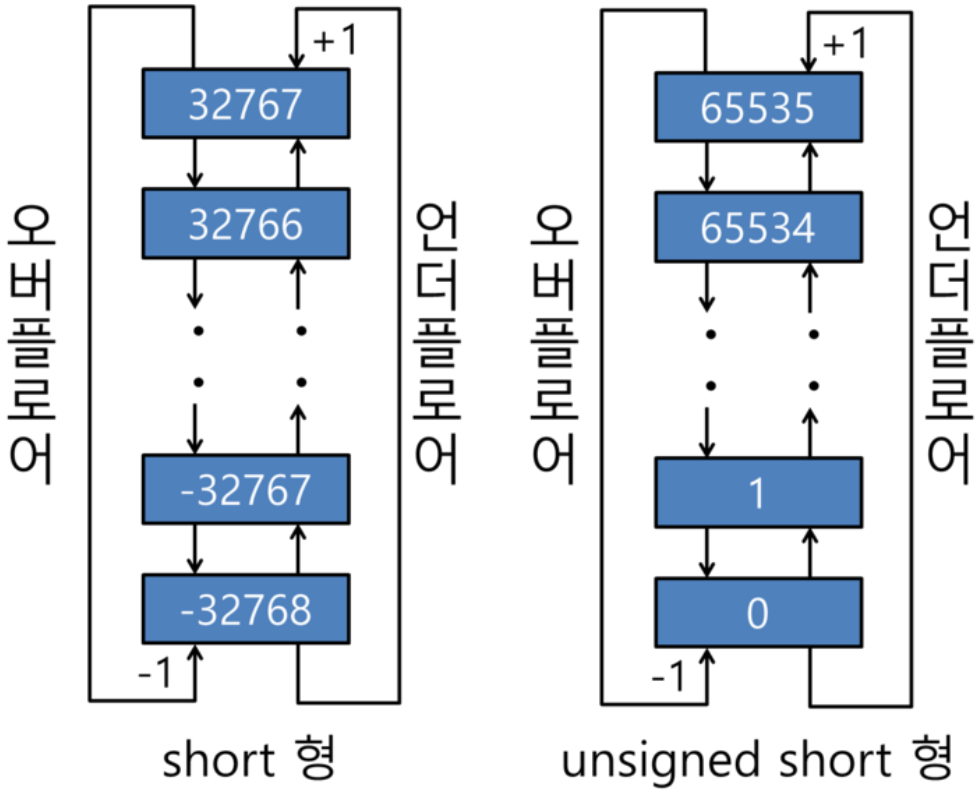
▼ 유효범위들

분류	데이터형	바이트 크기	유효 범위
문자형	char	1	$-128(-2^7) \sim 127(2^7-1)$
	unsigned char	1	$0 \sim 255(2^8-1)$
정수형	short	2	$-32768(-2^{15}) \sim 32767(2^{15}-1)$
	unsigned short	2	$0 \sim 65535(2^{16}-1)$
	int	4	$-2147483648(-2^{31}) \sim 2147483647(2^{31}-1)$
	unsigned int	4	$0 \sim 4294967295(2^{32}-1)$
	long	4	$-2147483648(-2^{31}) \sim 2147483647(2^{31}-1)$
	unsigned long	4	$0 \sim 4294967295(2^{32}-1)$
실수형	float	4	$\pm 1.17549 \times 10^{-38} \sim \pm 3.40282 \times 10^{38}$
	double	8	$\pm 2.22507 \times 10^{-308} \sim \pm 1.79769 \times 10^{308}$
	long double	8	$\pm 2.22507 \times 10^{-308} \sim \pm 1.79769 \times 10^{308}$



<개념을 콕콕 잡아주는 프로그래밍 C>





2. 실습

- ex02_01.c : 정수부, 실수부 분리 부터 12번까지