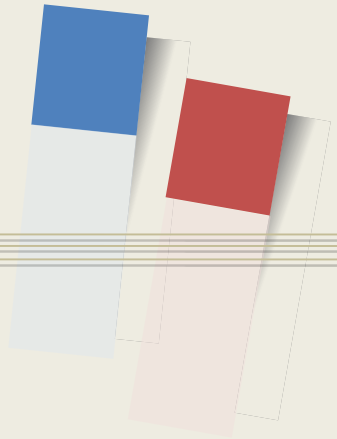


제 4 장
변수와 데이터 유형 1부

Variables & Data Types

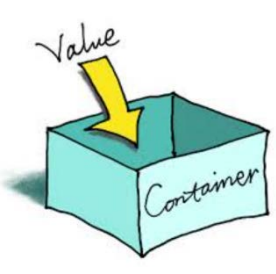
자료(데이터)

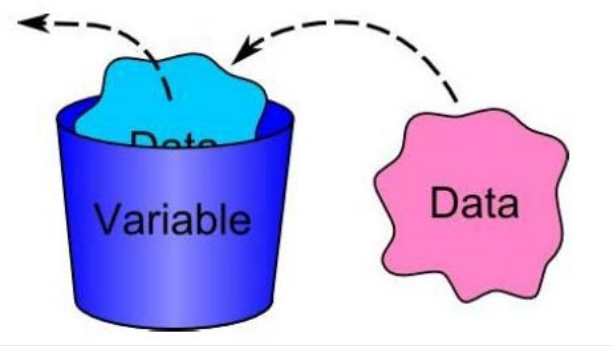
- 어떠한 일이든 작업을 수행하기 위해서는 그에 필요한 재료 및 도구와 그 재료와 도구 등을 보관할 보관함이 필요
  - 예를 들어, 음식을 만들기 위해서는 식 **재료**와 그것들을 보관할 **그릇**이 필요하고,
  - 의자, 책상과 같은 가구를 만들기 위해서는 나무와 망치, 톱 등이 필요하고, 나무 등을 놓아둘 선반과, 못, 망치 등 도구를 보관할 도구함 등이 필요한 것처럼,
- 컴퓨터에서 프로그램(작업)의 작성(수행)을 위해서도 이를 위해 필요한 프로그램 **데이터**(혹은 자료)와 이를 보관할 **저장소**(변수)가 필요하다

변수(Variable)

[정의] 프로그램이 실행되는 동안 생성되면서 저장할 필요가 있는 데이터를 임시로 저장하는 주 메모리 상의 공간에 붙인 이름으로 저장된 값은 수정 변경이 가능하다

[필요성] 입력데이터, 중간결과, 최종결과, 출력데이터 등의 저장





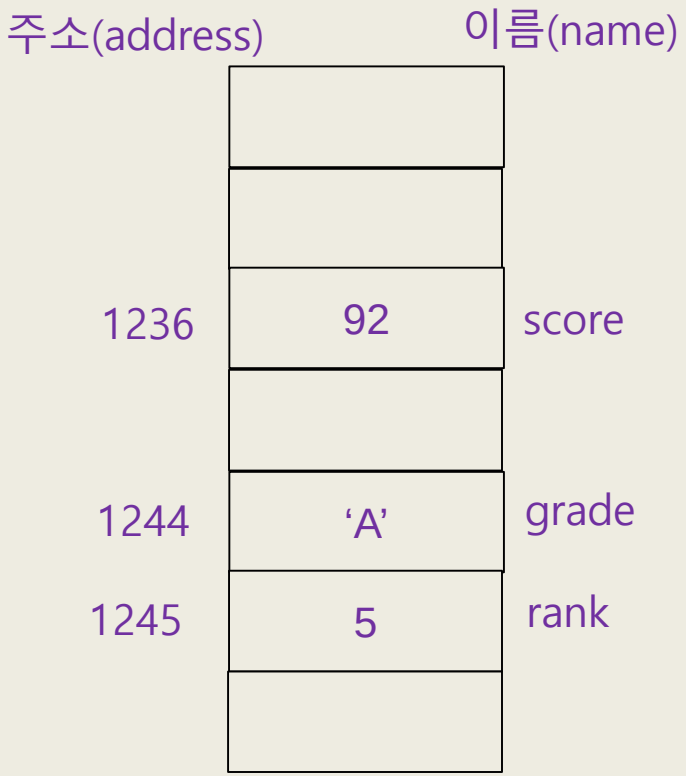
변수, 주 메모리 공간, 주소

- 주 메모리 공간은 Byte 단위마다 하나의 주소를 지정된다
- 변수는 주 메모리에 생성되며, 데이터가 저장되는 주 메모리에 생성된 특정 공간에 부여한 이름이다

- 변수의 주소를 알아야할 경우,
- 변수의 주소를 표현하는 법

예) score는 변수 명이고 score 변수의 주소는 변수 명 앞에 '&'를 붙여 표시한다.

- 변수 명 score 는 저장된 값 즉, 92이며
- &score 는 score 변수의 주소이므로 1236



상수(constant)

상수(constant):

- **변수 상수** : 변수와 동일하나, 저장된 값의 변경이 불가능하며 초기화하여 사용한다 (키워드 "**const**" 사용)
- **기호 상수**(symbolic constant) : 이름은 가지나 저장 공간이 할당되지 않은 상수 **("#define**"을 이용한다)
- **리터럴(literal) 상수 :** 이름을 가지지 않는 원 값 그대로의 상수

(예) 3.14, 100, 'A', "Hello World!"

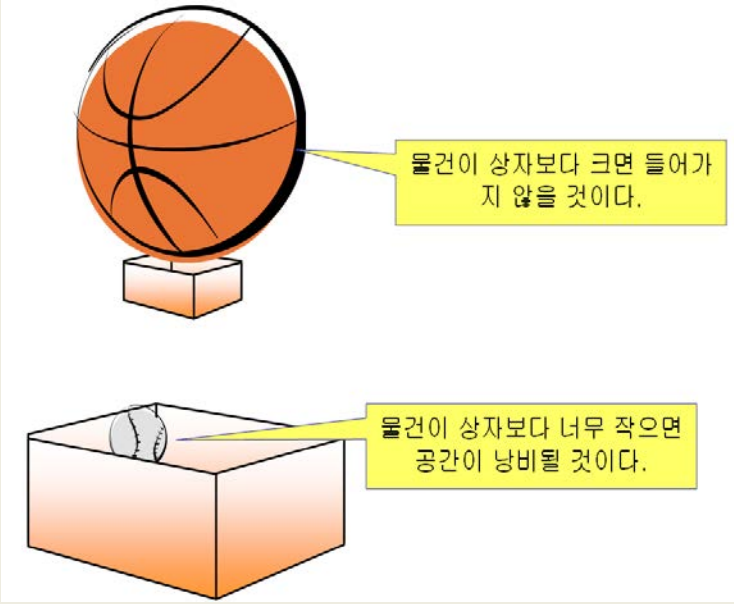
데이터 유형(Data Type)

- 데이터 유형
  - 기본형 : 문자형, 정수형, 실수형,
  - 복합형 : 배열형, 포인터형, 구조체, 유니온 등

(A) 요리를 만들 때 식재료와 그것을 보관할 그릇이 필요하며, 식재료의 **유형**에 따라, 보관할 **그릇의 크기**가 달라진다

(B) 예를 들어 고추장을 보관할 그릇과 감자를 보관할 그릇의 크기는 달라야 한다

(C) 그렇지 않고 모든 식재료에 대해 동일한 크기의 그릇을 사용한다면 매우 비효율적인 그릇의 사용으로 그릇의 낭비 혹은 식재료의 부식이 발생할 것이다



데이터 유형의 분류

자료 유형			설 명	바이트수	범 위
정수형	부호있음	short	short형 정수	2	-32768 ~ 32767
		int	정수	4	-2147483648~2147483647
		long	long형 정수	4	-2147483648~2147483647
	부호없음	unsigned short	부호없는 short형 정수	2	0 ~ 65535
		unsigned int	부호없는 정수	4	0 ~ 4294967295
		unsigned long	부호없는 long형 정수	4	0 ~ 4294967295
문자형	부호있음	char	문자 및 정수	1	-128 ~ 127
	부호없음	unsigned char	문자 및 부호없는 정수	1	0 ~ 255
부동소수점 실수형		float	단일정밀도 부동소수점	4	1.2E-38 ~ 3.4E38
		double	두배정밀도 부동소수점	8	2.2E-308 ~ 1.8E308

변수의 이름(identifier) 만드는 규칙

- 알파벳 문자와 숫자, 밑줄 문자(underscore) _로 구성
- 시작 문자는 반드시 알파벳 또는 밑줄 문자 _
- 대문자와 소문자 구별
- 키워드(혹은 예약어)와 동일한 변수 명은 불허용

(Q) 올바른 identifier는?

sum	O	
_count	O	
king3	O	
n_pictures	O	
2nd_try	X	// 숫자로 시작
dollor$	X	// $기호
double	X	// 키워드

키워드(예약어)

- 키워드(keyword): 언어에서 특정 의미를 가지고 사용하고 있는 특별한 단어, **예약어**(reserved words) 라고도 하며, 프로그램 작성시 일반 변수의 이름으로 사용하지 못한다

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

변수의 선언(variable declaration)

- 프로그램 작성 시에 변수를 사용하려면 사용하고자 하는 변수의 **이름**과 **유형**(type)을 컴파일러에게 지정해 주어야 하는데, 이를 "**변수의 선언**" 이라고 하며 다음과 같이 선언한다

데이터유형 변수이름 ;

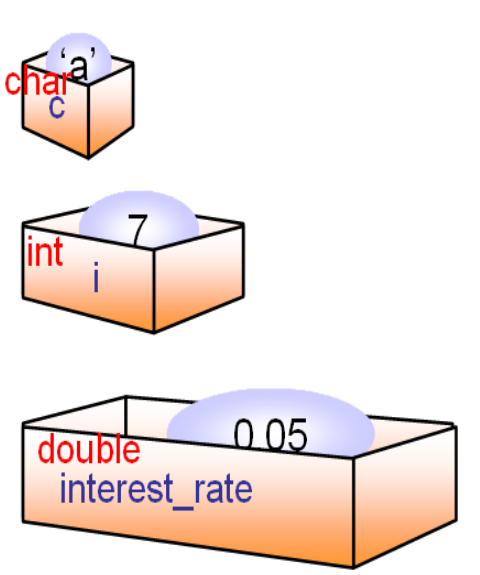
- 예

  문자형 --- **char** c;

  정수형 --- **int** I;

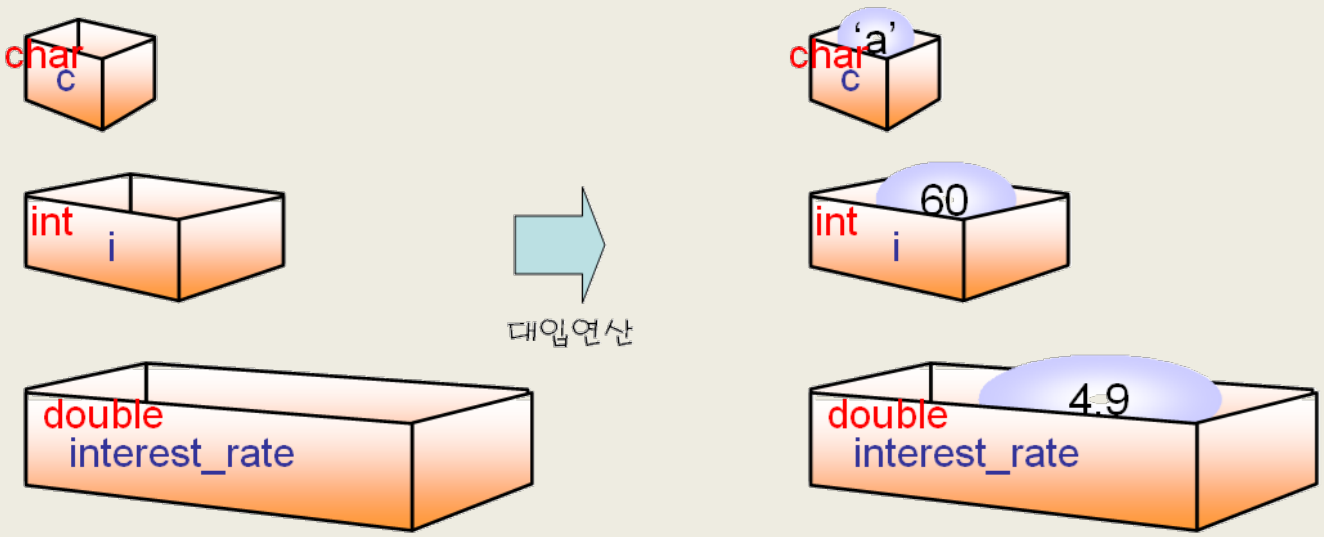
  배정도 실수형 --- **double** interest_rate;

  단정도 실수형 --- **float** height, width;



변수로 데이터 저장 (“=“ 사용)

```
char c;                          // 문자형 변수 c 선언
int i;                           // 정수형 변수 i 선언
double interest_rate;            // 실수형 변수 interest_rate 선언


c = 'a';                         // 문자형 변수 c에 문자 'a'를 대입
i = 60;                          // 정수형 변수 i에 60을 대입
interest_rate = 4.9;     // 실수형 변수 interest_rate에 82.9를 대입
```

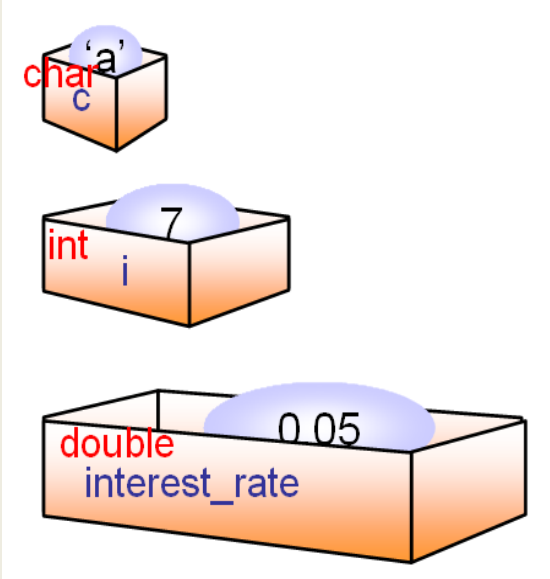


변수의 초기화

자료형 변수이름 = 초기값;

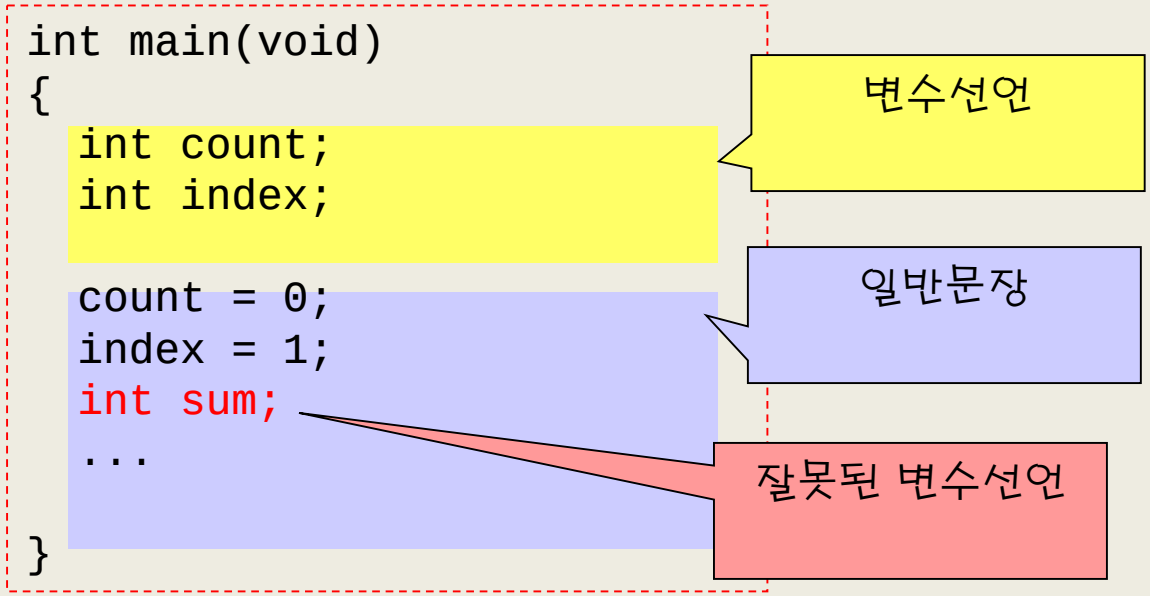
- 변수의 선언과 동시에 데이터를 저장하는 경우
- 초기화를 반드시 수행해야 하는 경우가 종종 존재, 그러므로 가능하면 초기화하여 변수를 선언하는 습관을 키우는게 좋다

```
char c = 'a';
int i = 7;
double interest_rate = 0.05;
```



변수 선언 위치

- 변수는 선언 자체와 선언되는 위치 그리고 부가되는 접근지정자(modifier)에 따라, 내포하고 있는 의미가 달라진다
- 변수는 블록의 내부 혹은 외부에 선언할 수 있다
  - 블록 내부의 시작 부분 : 지역변수(local variable)
  - 함수 블록의 외부 : 전역 변수(global variable)



- 변수 선언 자체로부터 알아낼 수 있는 변수에 관한 정보
  - 변수의 이름과 유형
  - 주 메모리 상 변수가 차지하는 공간 크기 → **필요한 byte 수**
  - 변수로 **표현 가능한 값의 범위**

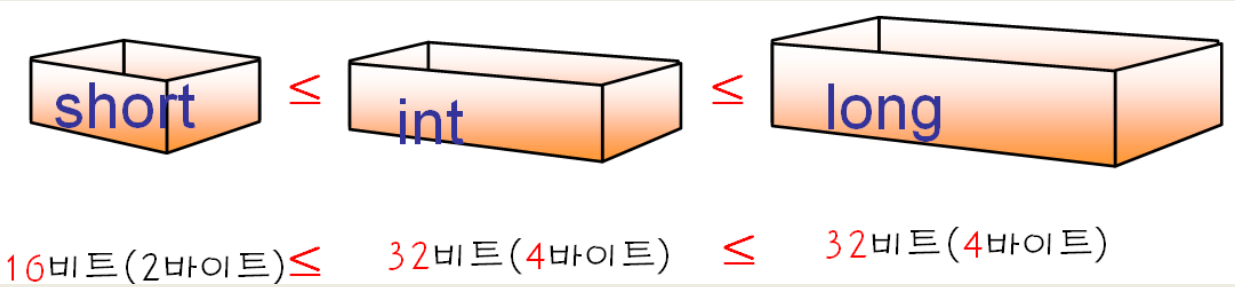
- 변수의 선언 위치(블록의 내부/외부)에 따라
  - 지역변수/전역변수로 구분되며, 그에 따라
    - 변수의 **유효 영역(scope)**
    - 변수의 **수명(life-time)** **을 알 수 있다**

- 변수에 부가되는 접근 지정자(modifier)에 따라서
  - **static** 지정자 : 지역변수에 부가하는 경우와, 전역변수에 부가하는 경우가 전혀 다른 의미를 나타낸다
  - **extern** 지정자 : 그 변수를 사용하는 해당 파일이 아닌 다른 파일에 사용하는 변수가 선언되어 있음을 제보

정수형

int 변수명;

- 변수에 할당되는 주 메모리 공간 크기에 따른 3가지 유형 :
  - **short, int, long**
- 음수/양수의 표현을 지정하는 2가지 지정자 :
  - **unsigned, signed**



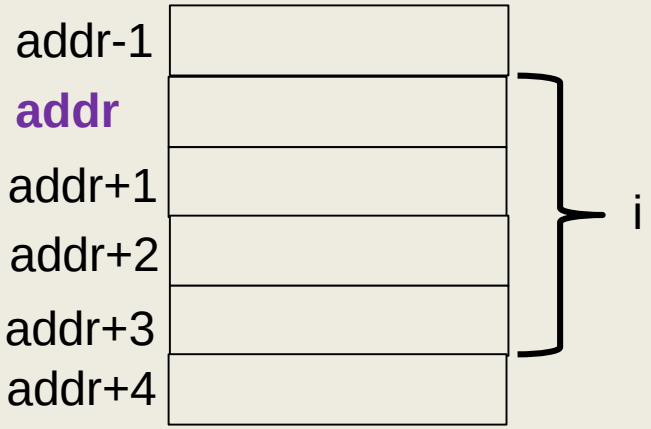
- 기본 유형은 **int**

  CPU의 성능과 운영체제에 따라 그 크기가 달라진다

  16비트(2bytes), 32비트(4 bytes), 64비트(8bytes)

정수형 변수의 주 메모리 할당

- 하나의 int 형 변수는 보통 4 bytes의 크기를 가지고, 주 메모리 주소는 byte당 배정되므로, 하나의 int 형 변수는 4개의 주소를 점유
- 예를 들어, 어떤 int 형 변수 i에 addr 번지가 배정된다는 것은 변수 i가 addr 번지 부터 (addr+3)번지까지의 주소를 점유한다는 것을 의미



정수형 변수의 표현 범위

- **int 형**

$$-2^{31},\ldots,-2,\ -1,\ 0,\ 1,2,\ldots,\ 2^{31}-1$$

$$-2147483648 \leqq n \leqq +2147483647$$

- **short 형**

$$-2^{15},\ldots,-2,\ -1,\ 0,\ 1,2,\ldots,\ 2^{15}-1$$

$$-32768 \leqq n \leqq +32767$$

- **long 형**

  - **C언어는 보통 int형과 같음**

예제

```c
/* 정수형 자료형의 크기를 계산하는 프로그램*/
#include <stdio.h>

int main(void)
{
    short year = 0;          // 0으로 초기화한다.
    int sale = 0;            // 0으로 초기화한다.
    long total_sale = 0;     // 0으로 초기화한다.

    year = 10;               // 약 3만2천을 넘지 않도록 주의
    sale = 200000000;        // 약 21억을 넘지 않도록 주의
    total_sale = year * sale; // 약 21억을 넘지 않도록 주의

    printf("total_sale = %d \n", total_sale);

    printf("short형의 크기: %d바이트\n", sizeof(short));
    printf("int형의 크기: %d바이트\n", sizeof(int));
    printf("long형의 크기: %d바이트\n", sizeof(long));

    return 0;
}
```

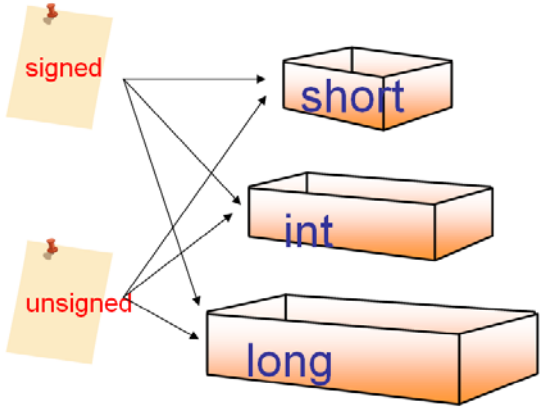
short형의 크기: 2바이트
int형의 크기: 4바이트
long형의 크기: 4바이트

signed, unsigned 수식자

- unsigned
  - 0 이상의 양수만 표현하고자 할 경우 사용
  - unsigned int

$$0,\ 1,\ 2,\ldots,\ 2^{32}-1$$
$$(0\ \sim\ +4294967295)$$

- Signed
  - 음수까지 표현하고자 할 경우 사용
  - 보통 생략하면 signed를 의미



signed short
unsigned short

signed int
unsigned int

signed long
unsigned long

오버플로우(Overflow)/언더플로우(Underflow)

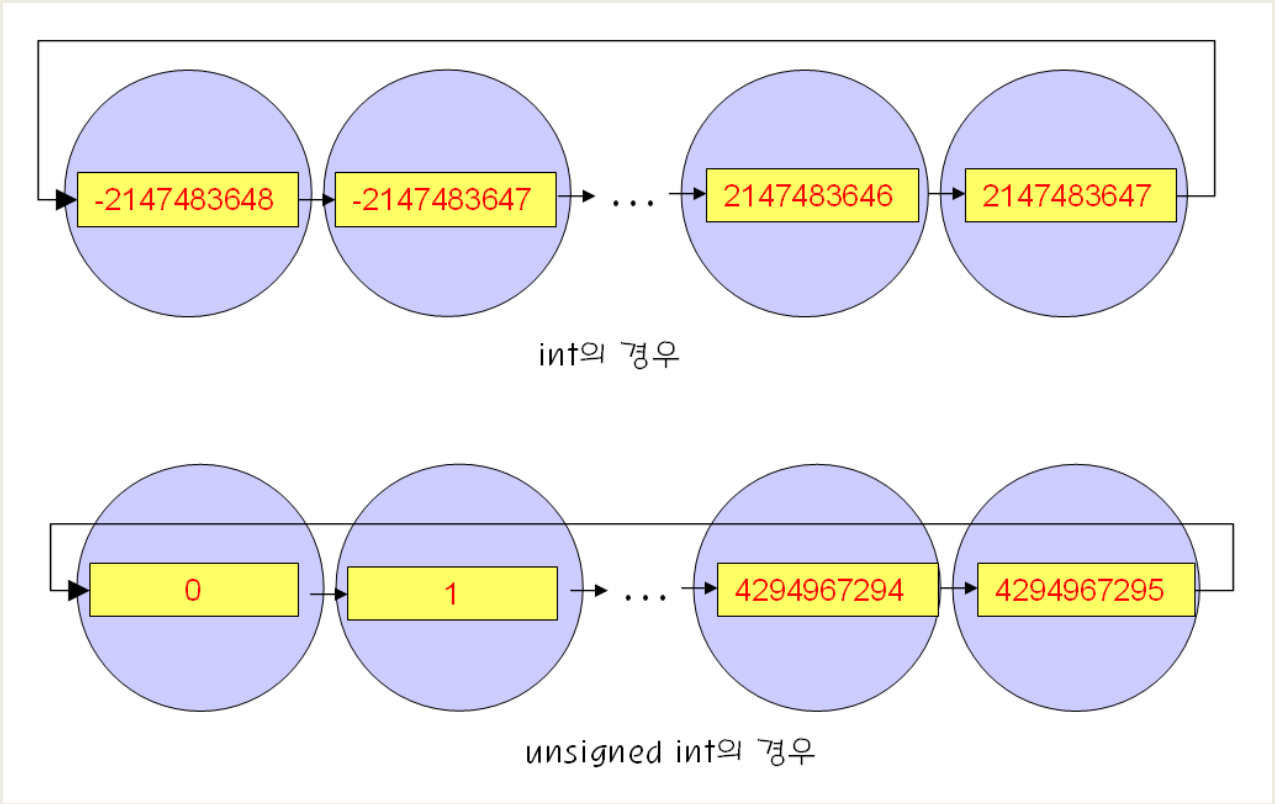
- 변수 유형이 정해지면 해당 변수가 표현할 수 있는 값의 범위가 정해진다
- 변수가 표현할 수 있는 범위를 넘어서는 값을 변수에 저장하고자 할 때 발생하는 문제
- 최대 범위를 벗어나는 경우 : overflow
- 최소 범위를 벗어나는 경우 : underflow

```
x = 2147483647
x+1 = -2147483648
x+2 = -2147483647
x+3 = -2147483646
y = 4294967295
y+1 = 0
y+2 = 1
y+3 = 2
```

```c
#include <stdio.h>
int main(void)
{
	int  x;
	unsigned  int  y;

	x = 2147483647;
	printf("x = %d\n",x);
	printf("x+1 = %d\n",x+1);
	printf("x+2 = %d\n",x+2);
	printf("x+3 = %d\n",x+3);


	y = 4294967295;
	printf("y = %u\n",y);		// unsigned를 출력하는 형식 지정자는 %u
	printf("y+1 = %u\n",y+1);
	printf("y+2 = %u\n",y+2);
	printf("y+3 = %u\n",y+3);
}
```

오버플로우 발생!!

- Overflow의 규칙성
  - 계량기나 주행거리계와 유사하게 동작



정수 상수

- 상수를 표현하면 컴파일러가 int 유형으로 묵시적 선택
- 상수의 유형을 명시적으로 선택하려면 다음과 같이 지정

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

- 10진법 이외의 진법으로도 표기 가능

```
int x = 10;   // 10은 10진수이고 int형이고 값은 십진수로 10이다.
int y = 010;  // 010은 8진수이고 int형이고 값은 십진수로 8이다.
int z = 0x10; // 010은 16진수이고 int형이고 값은 십진수로 16이다.
```

예제

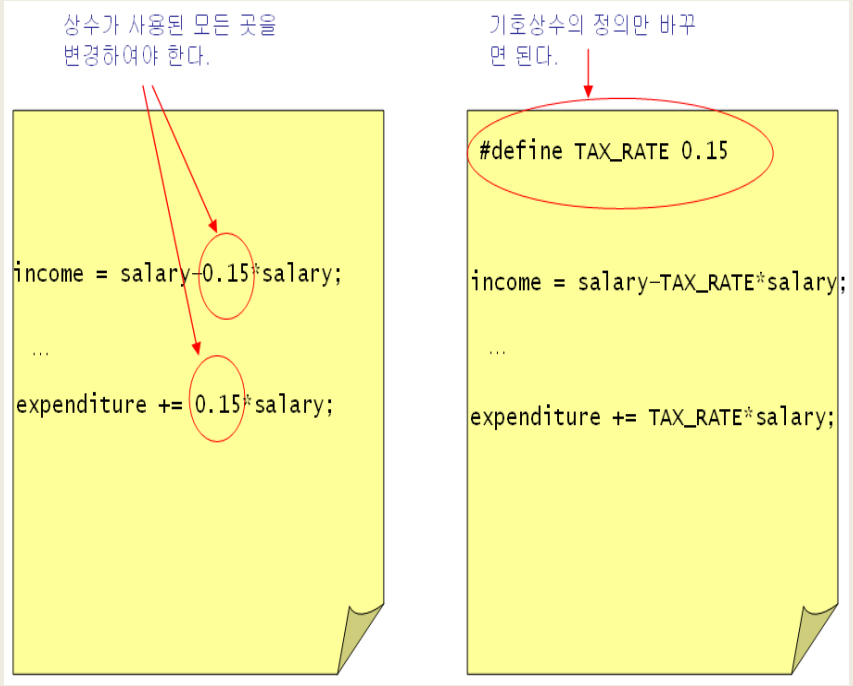
```c
/* 정수 상수 프로그램*/
#include <stdio.h>

int main(void)
{
	int x = 10;   // 10은 10진수이고 int형이고 값은 십진수로 10이다.
	int y = 010;  // 010은 8진수이고 int형이고 값은 십진수로 8이다.
	int z = 0x10; // 010은 16진수이고 int형이고 값은 십진수로 16이다.

	printf("sizeof(10L) = %d\n", sizeof(10L));
	printf("x = %d y = %d z = %d\n", x, y, z);
	printf("x = %d x = %#o x = %#x\n", x, x, x);

	return 0;
}
```

sizeof(10L) = 4
x = 10 y = 8 z = 16
x = 10 x = 012 x = 0xa

기호 상수(Symbolic Constant)

- 기호를 이용하여 상수를 표현
- (예)
  - area = 3.141592 * radius * radius;
  - area = PI * radius * radius;
  - income = salary - 0.15 * salary;
  - income = salary - TAX_RATE * salary;

- 기호 상수의 장점
  - 가독성(readability)을 높인다
  - 값의 수정이 용이



기호 상수의 선언

① #define 이름 값

```
/* 기호 상수 프로그램*/
#include <stdio.h>
#define PI 3.141592

int main(void)
{
        float radius, area, circumference;      // 변수 선언

        printf("반지름을 입력하시요:");          // 입력 안내문
        scanf("%f", &radius);                   // 사용자로부터 반지름 입력

        area = PI * radius * radius;            // 면적 계산
        circumference = 2.0 * PI * radius;      // 둘레 계산

        printf("반지름은 %f입니다.\n", radius);   // 반지름 출력
        printf("원의 면적은 %f이고 둘레는 %f입니다.\n", area, circumference);

        return 0;
}
```

② const 키워드 이용

```
/* 기호상수 프로그램*/
#include <stdio.h>

int main(void)
{
	const double TAX_RATE = 0.15;	// 기호 상수 선언
	double income, salary;		// 변수 선언


	printf("월급을 입력하시요:");		// 입력 안내문
	scanf("%lf", &salary);		// double형은 %lf로 지정하여야함


	income = salary - TAX_RATE * salary;	// 순수입 계산
	printf("순수입은 %lf입니다.\n", income);	// 순수입 출력


	return 0;
}
```