

Basic Computer Programming

Lecture 3

Electrical & Electronics Engineering
Chung-Ang University

Contents

- Understand the concepts of variables and constants.
- Understand the types of variables available in C.
- Integer variables and constants can be declared and used.
- Floating point variables and constants can be declared and used.
- Symbol constants can be used.
- Understand overflow and underflow.

The program that we are going to make in this chapter

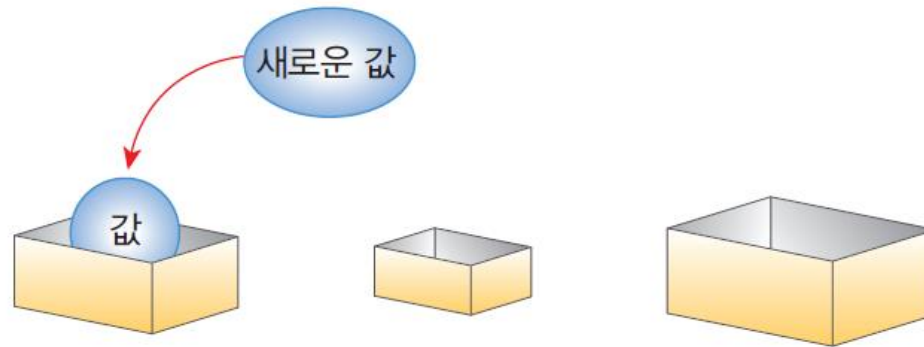
```
Microsoft Visual Studio 디버그 콘솔
달러화 금액을 입력하시오: 1000
달러화 1000달러는 1120000원입니다.
```

```
Microsoft Visual Studio 디버그 콘솔
빛의 속도는 300000.000000km/s
태양과 지구와의 거리 149600000.000000km
도달 시간은 8.311111분
```

```
Microsoft Visual Studio 디버그 콘솔
화씨 온도=100
섭씨 온도=37.777778
```

Variables

- The space in which values are stored in a program is called a variable.
- Variables must be **declared** in advance before use.



Declaration of variables.

- Variables must be **declared** in advance before use.

Syntax 3.1 변수 선언

형식 자료형 변수이름;

예

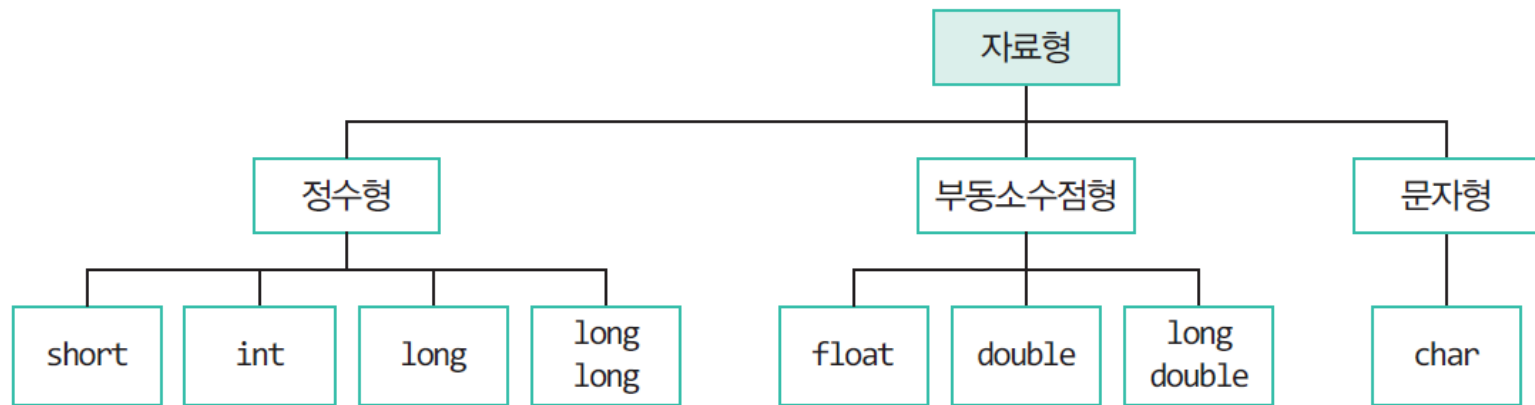
int i;



What is data type?

- Data type?
 - Integer data: 100
 - Real data: 3.141592
 - Character data: "A"

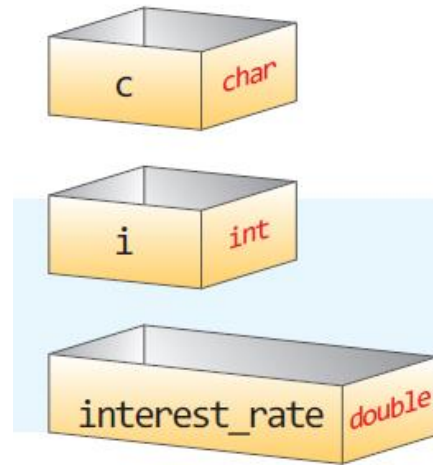
Data types



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

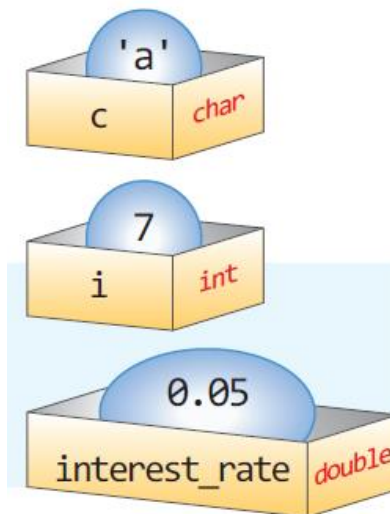
Example of declaration of variables.

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



Variable save

```
c = 'a';           // 문자 a를 저장  
i = 7;             // 정수 7을 저장  
interest_rate = 0.05; // 실수 0.05를 저장
```



When we declare the same data types,

```
int width, height;
```

When declaring multiple variables of the same data type, you may declare them in one line as follows.



참고

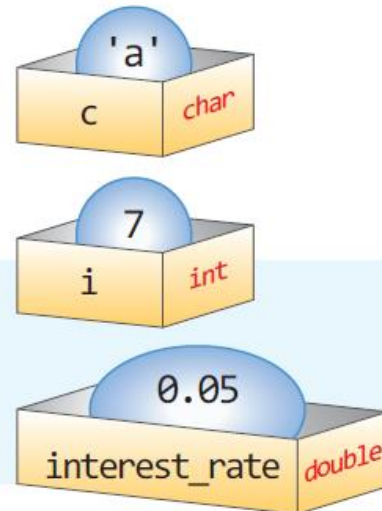
일부 프로그래밍 언어(파이썬이나 자바 스크립트)에서는 하나의 변수에 모든 종류의 값을 저장할 수도 있다. 입문자들에게는 친절하고 편리한 방법이지만 이러한 방법은 오류가 발생하기 쉽고 비효율적인 방법이기도 하다. 뭐든 장단점이 있다.

Initialization of variables.

Syntax 3.2 변수 초기화

형식 자료형 변수이름 = 초기값;

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



What if we don't initialize the variables?

If you don't initialize a variable, what value is in it?

It depends on where the variable is declared, but in general, uninitialized variables have garbage values. Garbage values mean some random values.

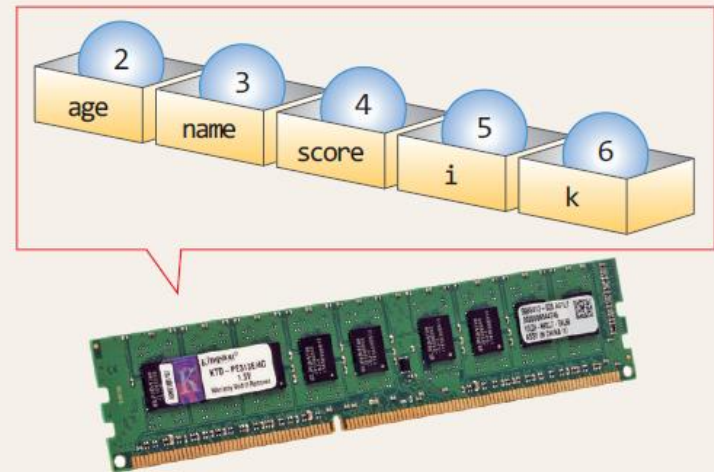


Where are the variables made?



참고: 변수는 어디에 만들어 질까?

변수는 물리적으로 컴퓨터의 어디에 만들어지는 것일까? 변수는 메모리(memory)에 만들어 진다. 우리는 프로그램 안에서 변수를 만들고 변수에 이름을 부여한 다음, 변수 이름을 사용하여 메모리 공간을 사용하게 된다. 만약 변수를 이용하지 않으면 메모리의 주소를 가지고 데이터를 저장하여야 할 것이다. "300번지에 정수 20을 저장하라"와 같이 주소를 이용하여 메모리를 사용하는 것은 가능한 방법이지만 인간에게는 상당히 불편한 방법이다. 변수라는 개념을 사용할 수 있기 때문에 특별한 경우를 제외하고는 우리는 메모리를 주소로 접근할 필요가 없다.



The name of the variable.

- The name of the variable can be given to the programmer at will, but several rules must be observed. Just as names such as "Hong Gil-dong" and "Kim Young-hee" identify people, names of variables play a role in identifying variable



그림 3.1 변수의 이름은 변수를 식별한다.

The name of the variable.

- It consists of alphabetic characters, numbers, and underscores _.
- There should be no spaces in the middle of the name.
- The first letter must be alphabetic or underscore _.
- Therefore, the name cannot start with a number.
- They are treated as different ones by distinguishing upper and lower case letters. Therefore, variables index, Index, and INDEX are all different variables.
- The same name as the keyword used in the C language is not allowed.

Keyword

- Keyword is a special word that has a unique meaning in C language.
- Keywords are also referred to as reserved words.
- Keywords are prohibited from being redefined or used by users. Therefore, keywords cannot be used as identifiers.

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				

The name of the variables.

- *sum* *// 영문 알파벳 문자로 시작*
- *_count* *// 밑줄 문자로 시작할 수 있다.*
- *number_of_pictures* *// 중간에 밑줄 문자를 넣을 수 있다.*
- *King3* *// 맨 처음이 아니라면 숫자도 넣을 수 있다.*

- *2nd_base(X)* *// 숫자로 시작할 수 없다.*
- *money#* *// #과 같은 기호는 사용할 수 없다.*
- *double* *// double은 C 언어의 키워드이다.*

Some tips



Tip: 좋은 변수 이름

변수의 이름을 짓는 것은 상당히 중요한 작업 중의 하나이므로 신중하고 시간을 투자해야 한다. 변수의 이름을 지을 때는 변수의 역할을 가장 잘 설명하는 이름을 지어야 한다. 좋은 변수 이름은 전체 프로그램을 읽기 쉽게 만든다. 하지만 반대로 즉흥적으로 지은 이름을 사용하게 되면 나중에 프로그램을 읽기가 아주 힘들어진다. 예를 들면 연도와 달, 일을 나타내는데 `i`, `j`, `k`라고 이름을 짓는 것보다 `year`, `month`, `date`라고 하면 프로그램이 읽기 쉬워질 것이다. 영어 단어만을 사용해야 하므로 한영 사전을 이용하여 한글을 영문으로 바꾸는 것도 좋은 아이디어이다.



Tip: 여러 단어로 된 변수 이름

여러 단어로 되어 있는 변수 이름을 나타내는 데 몇 가지의 방식이 존재한다. 먼저 가장 전통적인 방법은 `bank_account`처럼 중간에 밑줄 문자를 사용하는 것이다. 두 번째 방법은 `BankAccount`처럼 단어의 첫 번째 글자를 대문자로 하는 것이다. 어떤 방식도 사용해도 상관없고 다만 일관성있게 사용하면 된다. 이 책에서는 전통적인 C의 방법을 따라서 밑줄 문자를 사용하여 단어들을 분리하였다.

Types of data types

- Data type is a type of data. The data types in the visual studio are as follows. If the compiler is changed, the size of the data type can be changed.

자료형		바이트수	범위
정수형	short	2	-32768 ~ 32767
	int	4	-2147483648 ~ 2147483647
	long	4	-2147483648 ~ 2147483647
	long long	8	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807
문자형	char	1	-128 ~ 127
부동소수점형	float	4	1.2E-38 ~ 3.4E38
	double	8	2.2E-308 ~ 1.8E308
	long double	8	2.2E-308 ~ 1.8E308

Why do we use various data types?

물건이 상자보다
크면 들어가지 않을
것이다.



물건이 상자보다
너무 작으면 공간이
낭비될 것이다.



sizeof operator



참고

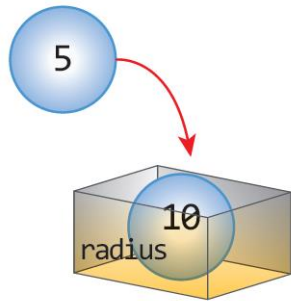
자료형의 크기를 알아보려면 sizeof 연산자를 사용하면 된다. sizeof는 변수나 자료형의 크기를 바이트 단위로 반환한다.

```
int x;  
printf("변수 x의 크기: %d\n", sizeof(x));           // 변수 x의 크기: 4  
printf("char형의 크기: %d\n", sizeof(char));        // char형의 크기: 1  
printf("int형의 크기: %d\n", sizeof(int));          // int형의 크기: 4
```

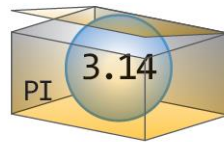
Constant

- $area = 3.14 * radius * radius;$

Constant



변수

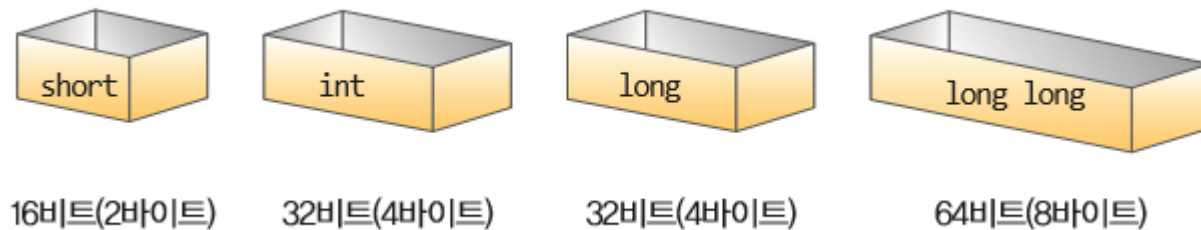


상수



그림 3.3 변수와 상수

Integers



- The most basic thing is int.
- The size varies depending on the CPU.
- 16 bits, 32 bits, 64 bits.

(Q) Why do we need multiple integer types?

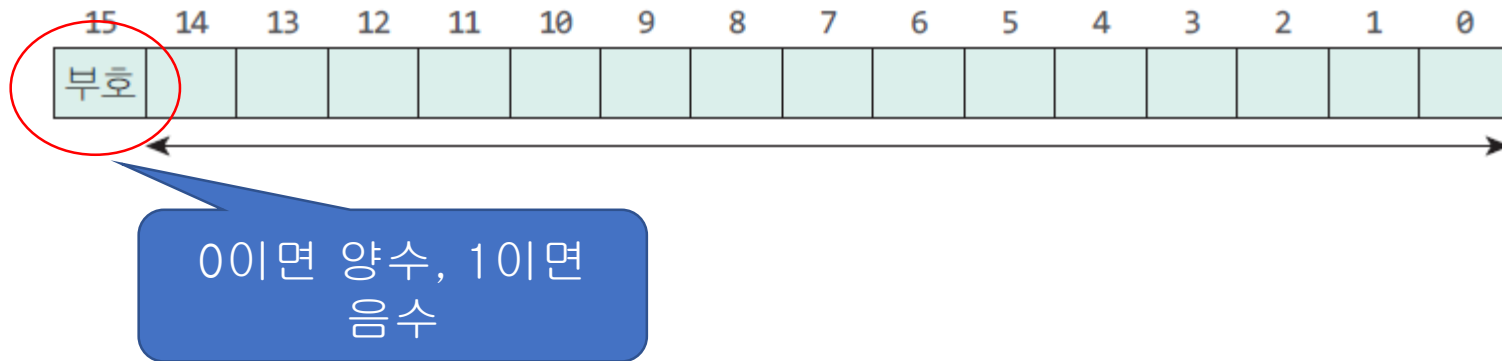
(A) To allow programmers to select and use according to their applications

Example of declaring an integer variable.

<code>short grade;</code>	<code>// short형의 변수를 생성한다.</code>
<code>int count;</code>	<code>// int형의 변수를 생성한다.</code>
<code>long price;</code>	<code>// long형의 변수를 생성한다.</code>
<code>long long distance;</code>	<code>// long long형의 변수를 생성한다.</code>

The range of integers represented by the integer type.

- -32768에서 +32767까지의 정수



참고

현재 자기가 사용하는 자료형이 나타낼 수 있는 범위가 얼마인지를 알고 싶으면 `limits.h` 헤더 파일을 참고하면 된다. 여기에는 정수형들의 최대값과 최소값을 기호 상수로 정의해 놓았다. 예를 들어서 `int`형의 최대값은 `INT_MAX`로, 최소값은 `INT_MIN`으로 알 수 있다.

unsigned

- Unsigned means that variables represent only non-negative values.

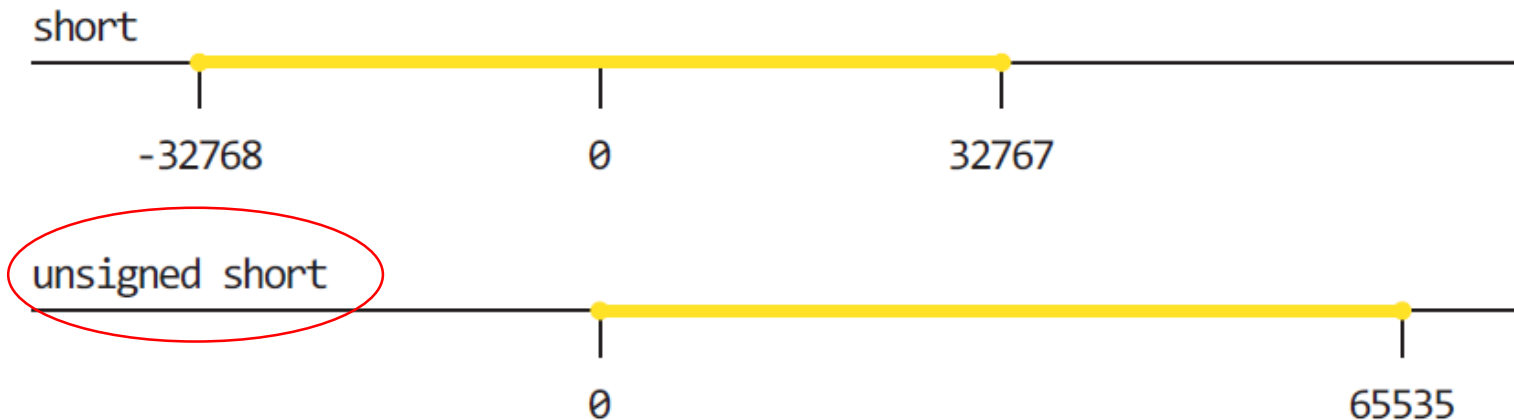


그림 3.4 unsigned를 붙이면 양수만 표현할 수 있다.

unsigned modifier

```
unsigned int speed;           // 부호없는 int형 변수 speed
```

```
unsigned speed;               // 이렇게 해도 된다
```



참고

unsigned고 쓰면 unsigned int형을 의미한다. 또한 short 형은 short int라고 쓸 수도 있다. 마찬가지로 long 형은 long int라고 쓸 수도 있다.

Overflow

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

```
int main(void)
```

```
{
```

```
    short s_money = 32767; // 최대값으로 초기화한다.
```

```
    s_money = s_money + 1;
```

```
    printf("s_money = %d\n", s_money);
```

```
    return 0;
```

```
}
```



Microsoft Visual Studio 디버그 콘솔

s_money = -32768

Overflow

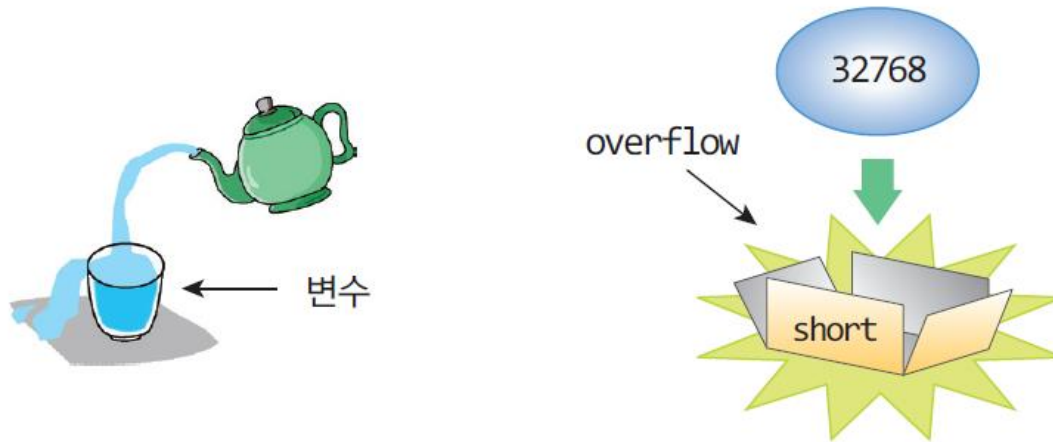


그림 3.5 오버플로우는 변수가 저장할 수 있는 범위를 넘어서는 수를 저장했을 경우에 발생한다.

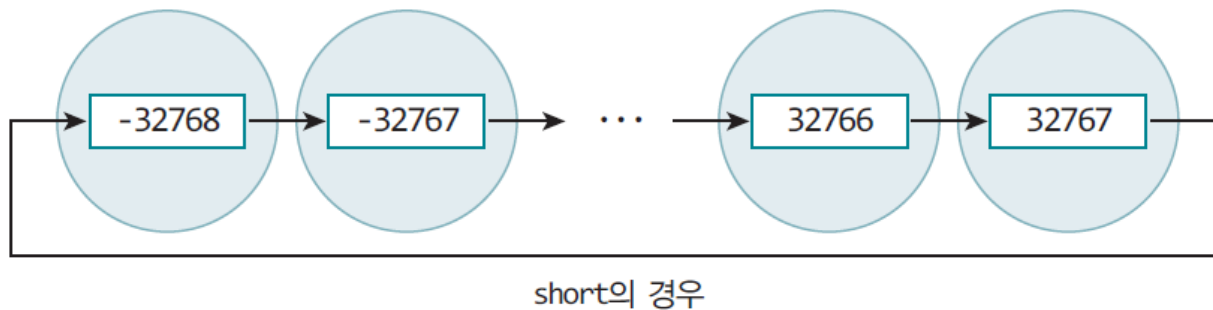


그림 3.6 오버플로우가 발생하면 수도계량기나 자동차의 주행거리계처럼 처음으로 되돌아간다. short형의 경우, 음수부터 시작하므로 음수로 되돌아간다.

I/O format designator

자료형	형식 지정자	설명
short	%hi	입력할 때는 %hi를 사용하는 것이 좋다. 출력 시에는 %d도 가능하다.
int	%d	
long	%ld	
long long	%lld	특히 입력할 때는 반드시 %lld를 사용하여야 한다. 출력 시에도 %lld를 사용하여야 한다.

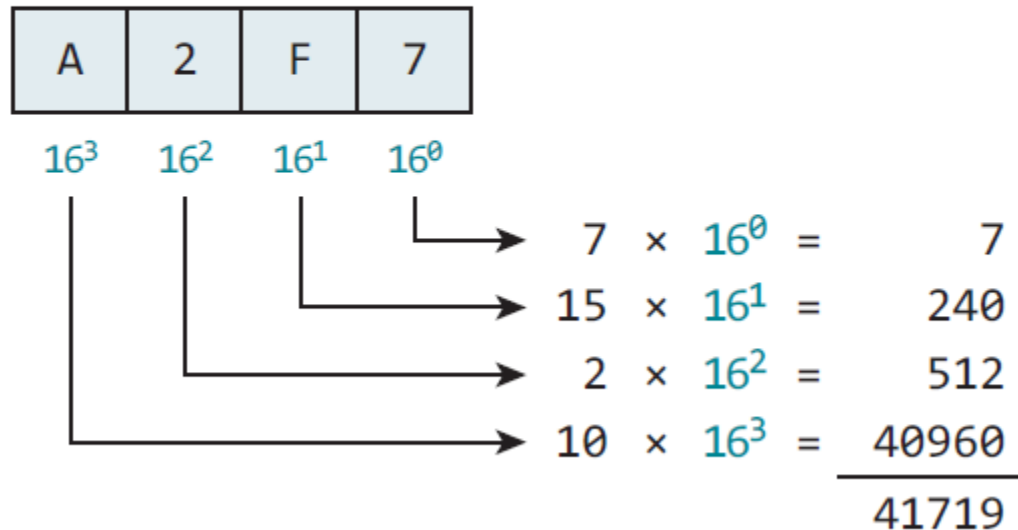
Integer constant

표 3.2 정수 상수

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

Hexadecimal

$$0xA2F7_{16} = 10 \times 16^3 + 2 \times 16^2 + 15 \times 16^1 + 7 \times 16^0 = 41719_{10}$$



Hexadecimal

```
#include <stdio.h>

int main(void)
{
    printf("%d %#x %#o \n", 128, 128, 128);
    return 0;
}
```

Microsoft Visual Studio 디버그 콘솔

128 0x80 0200

Symbolic constant

- A symbol constant is a name for a constant.

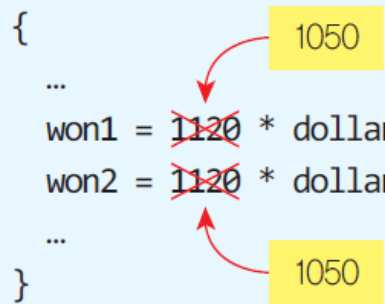
```
#define EXCHANGE_RATE 1120
```

- Advantages of the symbol constant.
 - It becomes easier to read programs.
 - You can change it easily.

Symbolic constant

```
#include <stdio.h>

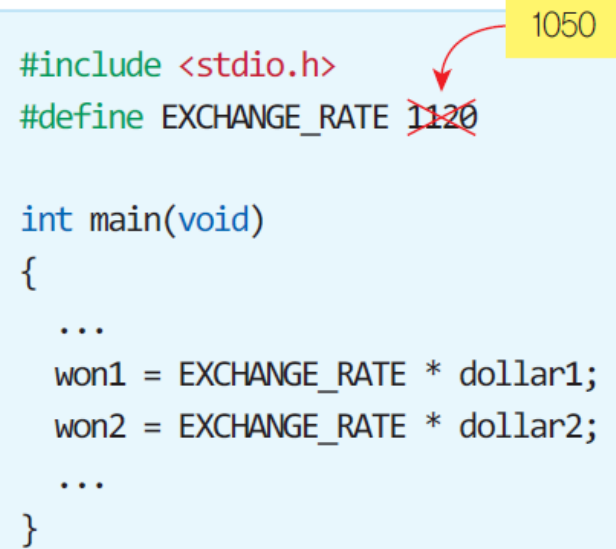
int main(void)
{
    ...
    won1 = 1120 * dollar1;
    won2 = 1120 * dollar2;
    ...
}
```



리터럴 상수를 사용하는 경우:
등장하는 모든 곳을 수정하여야 한다.

```
#include <stdio.h>
#define EXCHANGE_RATE 1120

int main(void)
{
    ...
    won1 = EXCHANGE_RATE * dollar1;
    won2 = EXCHANGE_RATE * dollar2;
    ...
}
```



기호 상수를 사용하는 경우:
기호 상수가 정의된 곳만 수정하면 된다.

그림 3.7 리터럴 상수와 기호 상수의 비교

Interim check



중간점검

1. 정수형에 속하는 자료형을 모두 열거하라.
2. 숫자 값을 직접 사용하는 것보다 기호 상수를 사용하는 것의 이점은 무엇인가?
3. 왜 정수를 하나의 타입으로 하지 않고 `short`, `int`, `long` 등의 여러 가지 타입으로 복잡하게 분류하여 사용하는가?
4. 변수가 저장할 수 있는 한계를 넘어서는 값을 저장하면 어떻게 되는가? 구체적인 예로 `short`형의 변수에 32768을 저장하면 어떻게 되는가?

Lab: Calculating Dollar in Korean Won

- Let's write a program that calculates dollars in Korean won as an example of declaring and using variables.



```
Microsoft Visual Studio 디버그 콘솔
달러화 금액을 입력하시오: 1000
달러화 1000달러는 1120000원입니다.
```

Lab: Calculating Dollar in Korean Won

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#define EXCHANGE_RATE    1120    //기호 상수 정의

int main(void)
{
    int usd;        // 달러화
    int krw;        // 원화
    printf("달러화 금액을 입력하시오: ");        // 입력 안내 메시지
    scanf("%d", &usd);        // 달러화 금액 입력
    krw = EXCHANGE_RATE * usd;        // 원화로 환산
    printf("달러화 %d달러는 %d원입니다.\n", usd, krw);        // 계산 결과 출력
    return 0;        // 함수 결과값 반환
}
```

Lab: Calculating cumulative money

- How much will it be if I save 5 million won a month for 30 years?



The screenshot shows a debug console window from Microsoft Visual Studio. The title bar is orange and contains the text 'Microsoft Visual Studio 디버그 콘솔' along with standard window control buttons. The console area has a white background and contains two lines of Korean text: '매달 저축 금액을 입력하시오: 5000000' and '30년 후의 재산 = 18000000000원'. A vertical scrollbar is visible on the right side of the console.

```
Microsoft Visual Studio 디버그 콘솔
매달 저축 금액을 입력하시오: 5000000
30년 후의 재산 = 18000000000원
```

Lab: Calculating cumulative money

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

int main(void)
{
    int property;
    int saving;

    printf("매달 저축 금액을 입력하시오: ");
    scanf("%d", &saving);

    property = saving * 12 * 30;
    printf("30년 후의 재산 = %d원 \n", property);
    return 0;
}
```


Lab: Calculating cumulative money

- But what happens if you increase your savings to 10 million won?

```
Microsoft Visual Studio 디버그 콘솔
매달 저축 금액을 입력하시오: 10000000
30년 후의 재산 = -694967296원
```



Sol: Calculating cumulative money

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
int main(void)
{
    long long int property;
    long long int saving;

    printf("매달 저축 금액을 입력하시오: ");
    scanf("%lld", &saving);
    property = saving * 12 * 30;
    printf("30년 후의 재산 = %lld원 \n", property);
    return 0;
}
```



Microsoft Visual Studio 디버그 콘솔

매달 저축 금액을 입력하시오: 10000000
30년 후의 재산 = 3600000000원

Lab: Exchange values of variables

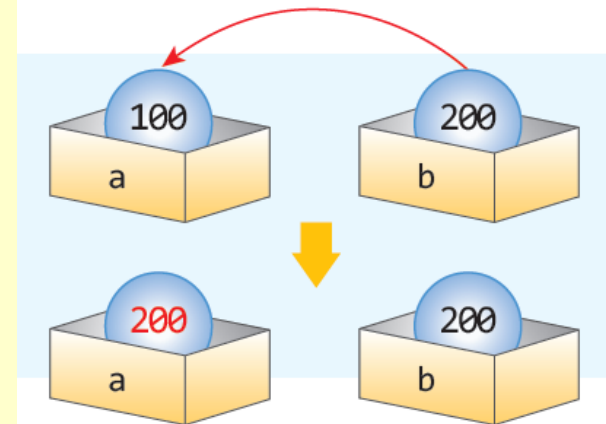
- Let's write a code that changes the values of the variable x and the variable y.

```
int a = 100;
```

```
int b = 200;
```

```
a = b;
```

```
b = a;
```



Sol: Exchange values of variables

- Let's write a code that changes the values of the variable x and the variable y.

```
int a = 100;
```

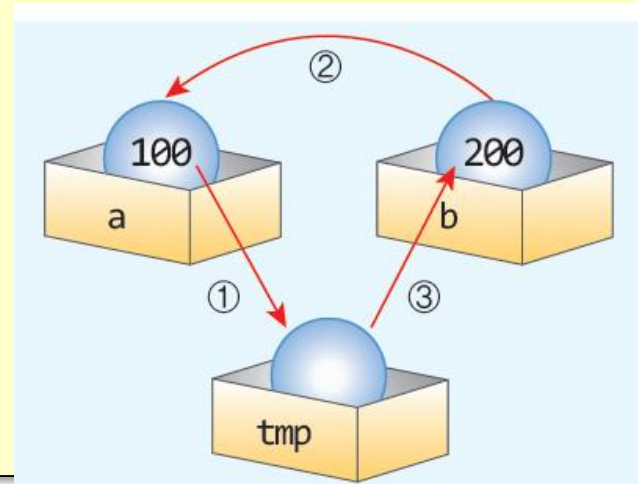
```
int b = 200;
```

```
int tmp;
```

```
tmp = a;           // ①
```

```
a = b;            // ②
```

```
b = tmp;          // ③
```



Floating point

- On a computer, real numbers are expressed in floating point form.
 - It means that the decimal point floats and moves.
 - Similar to scientific notation that scientists use

The diagram illustrates the conversion of a decimal number to scientific notation. On the left, the number 12345 is enclosed in a dashed green box and labeled "10진수 표현" (Decimal representation) above it. This is followed by an equals sign. To the right of the equals sign, the expression 1.2345×10^4 is shown, also enclosed in a dashed green box and labeled "부동소수점 표현" (Floating-point representation) above it. Three green arrows point to the components of the scientific notation: one points to the "1" in "1.2345" and is labeled "가수" (Mantissa) below it; another points to the "2345" in "1.2345" and is labeled "밀수" (Fractional part) below it; and a third points to the "4" in "10^4" and is labeled "지수" (Exponent) to its right.

$$\text{10진수 표현} \quad 12345 = \text{부동소수점 표현} \quad 1.2345 \times 10^4$$

Labels for the scientific notation components:

- 가수 (Mantissa) points to 1
- 밀수 (Fractional part) points to .2345
- 지수 (Exponent) points to 4

Floating point

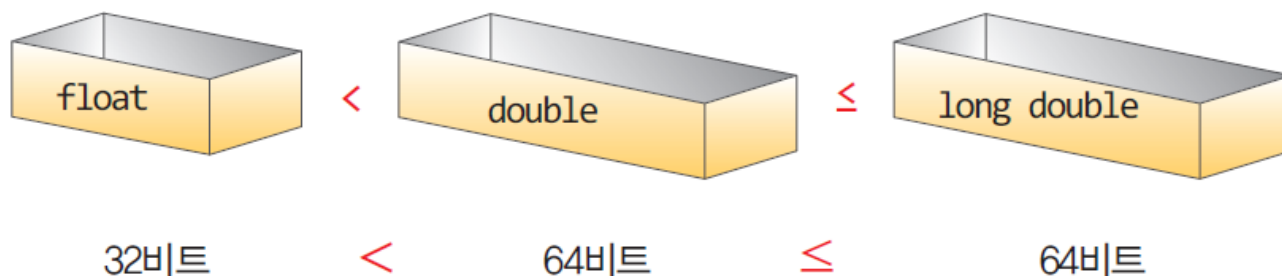
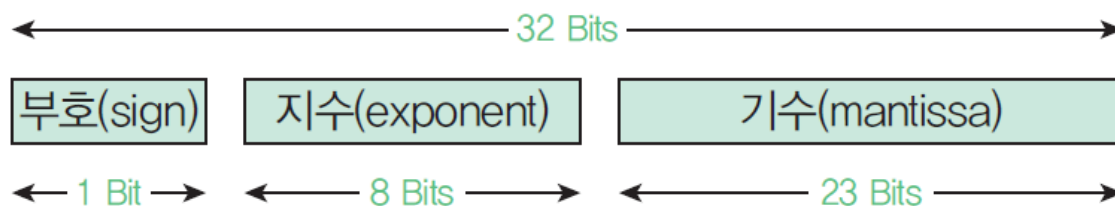


그림 3.8 C에서는 float, double, long double형과 같은 부동소수점형을 이용하여 실수를 표현한다.



Floating point datatypes

표 2.6 부동소수점 자료형

자료형	명칭	크기	유효 숫자	범위
float	단일 정밀도 (single-precision)	32비트	6자리	$\pm 1.17549 \times 10^{-38} \sim \pm 3.40282 \times 10^{+38}$
double long double	두 배 정밀도 (double-precision)	64비트	16자리	$\pm 2.22507 \times 10^{-308} \sim \pm 1.79769 \times 10^{+308}$



참고

유효 숫자란 믿을 수 있는 의미 있는 숫자를 말한다. 예를 들어서 2,696을 십의 자리에서 반올림하면 2,700이다. 여기서 2와 7은 의미있는 숫자이다. 반면에 뒤에 붙은 00은 단순히 자릿수를 나타내는데 사용된다. 유효 숫자의 개수는 소수점의 위치와는 상관이 없다. 2,700을 2.7e3이라고 표현해도 유효 숫자는 역시 2개이다.

Floating point constants

표 3.3 부동소수점 상수의 표기법 비교

소수점 표기법	지수 표기법	의미
123.45	1.2345e2	1.2345×10^2
12345.6	1.23456e4	1.23456×10^4
0.000023	2.3e-5	2.3×10^{-5}
2000000000	2.0e9	2.0×10^9

3.141592F

```
.           // 소수점만 붙여도 된다.
.28        // 정수부가 없어도 된다.
9.26E3     //  $9.26 \times 10^3$ 
0.67e-7    //  $0.67 \times 10^{-7}$ 
```


Designator

- To output or input a float type value, use "%f" as the type designator.
- "%lf" is used to input and output double-type values.

```
double radius;  
printf("반지름 값을 입력하시오: ");  
scanf("%lf", &radius);           // 반드시 "%lf"을 사용하여야 한다.
```

Example: Find out significant numbers

```
#include <stdio.h>

int main(void)
{
    float fvalue = 1234567890.12345678901234567890;
    double dvalue = 1234567890.12345678901234567890;

    printf("float형 변수=%30.25f\n", fvalue);
    printf("double형 변수=%30.25lf\n", dvalue);
    return 0;
}
```

Microsoft Visual Studio 디버그 콘솔

```
float형 변수=1234567936.000000000000000000000000  
double형 변수=1234567890.1234567165374755859375000
```

Notice

```
float f = 12.7;
```

```
int a = f;
```



Q float같은 실수를 int같은 정수에 넣을 경우, 어떤 일이 발생하는가?

A 컴파일러는 이러한 경우에 경고를 한다. 실수 중에서 소수점 이하는 없어지고 정수 부분만 정수변수에 대입된다. 12.7이라는 실수를 정수 변수에 대입하면 12만 남는다.

Overflow and underflow

```
#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;
}
```

```
1>d:\wcprogram\mainarg\mainarg\mainarg.c(5): warning C4056: 부동 소수점 상수 산술 연산에서 오버플로가 발생했습니다.
1>d:\wcprogram\mainarg\mainarg\mainarg.c(6): warning C4305: '초기화 중': 'double'에서 'float'(으)로 잘립니다.
1>d:\wcprogram\mainarg\mainarg\mainarg.c(5): warning C4756: 상수 산술 연산에서 오버플로가 발생했습니다.
1> mainarg.vcxproj -> D:\wcprogram\mainarg\Debug\mainarg.exe
1> mainarg.vcxproj -> D:\wcprogram\mainarg\Debug\mainarg.pdb (Full PDB)
===== 빌드: 성공 1, 실패 0, 최신 0, 생략 0 =====
```

Microsoft Visual Studio 디버그 콘솔

```
x=inf
y=0.000000e+00
```

Floating point types may be inaccurate!

```
#include <stdio.h>
int main(void)
{
    float value = 0.1;
    printf("%.20f \n", value); // %.20f는 소수점 이하를 20자리로 출력하라는
    의미이다.
    return 0;
}
```



The screenshot shows a window titled "Microsoft Visual Studio 디버그 콘솔" (Microsoft Visual Studio Debug Console). The output text is "0.10000000149011611938". The window has standard Windows window controls (minimize, maximize, close) in the top right corner and a vertical scrollbar on the right side.

```
Microsoft Visual Studio 디버그 콘솔
0.10000000149011611938
```

Why?

- The value of 0.1 is not accurately output. What's the reason? This is because there are values that cannot be accurately represented by binary. 0.1 is one of them.
- For example, In decimal, $1/3$ cannot be accurately represented (0.3333...This is repeated infinitely). In binary, 0.1 is the case. In decimal, it is accurately expressed as 0.1, but in binary, it is impossible to accurately express 0.1. It starts with "0.000110011.." and continues indefinitely.
- Of course, you can use it practically if you round it.

Interim check

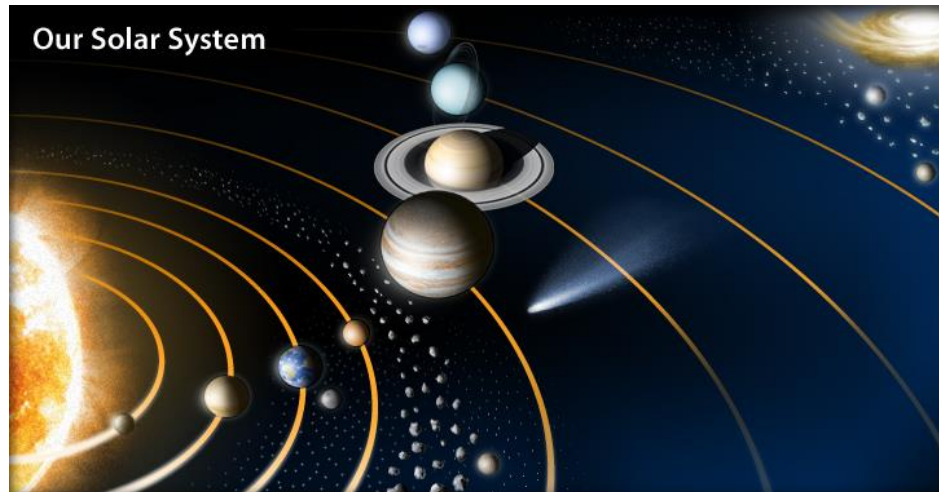


중간점검

1. 부동 소수점형에 속하는 자료형을 모두 열거하라.
2. float형 대신에 double형을 사용하는 이유는 무엇인가?
3. 12.345처럼 소수점이 있는 실수를 int형의 변수에 넣을 경우, 어떤 일이 발생하는가?

Lab: Time for sunlight to reach

- We would like to calculate on a computer how many minutes the light from the sun arrives on Earth.
- The speed of light travels 300,000 km per second.
- The distance between the sun and Earth is about 149.6 million km.



Execution result

```
Microsoft Visual Studio 디버그 콘솔
빛의 속도는 300000.000000km/s
태양과 지구와의 거리 149600000.000000km
도달 시간은 8.311111분
```



← 빛은 8분 정도 걸린다.



Hints

- In order to solve the problem, necessary variables must be created first. Here, variables representing the speed of light, the distance between the sun and the earth, and the time of arrival are needed.
- All data types of variables must be floating point types. Because they are large numbers.
- The time the light reaches can be calculated as (the arrival time = distance/(the speed of light)).
- When outputting the floating point type as printf(), %f or %lf is used.

Solution

```
#include <stdio.h>
int main(void)
{
    double light_speed = 300000;           // 빛의 속도 저장하는 변수
    double distance = 149600000;          // 태양과 지구 사이 거리 저장하는 변수
                                           // 149600000km로 초기화한다.
    double time;                           // 시간을 나타내는 변수

    time = distance / light_speed;         // 거리를 빛의 속도로 나눈다.
    time = time / 60.0;                    // 초를 분으로 변환한다.

    printf("빛의 속도는 %lfkm/s \n", light_speed);
    printf("태양과 지구와의 거리 %lfkm \n", distance);
    printf("도달 시간은 %lf초\n", time); // 시간을 출력한다.

    return 0;
}
```

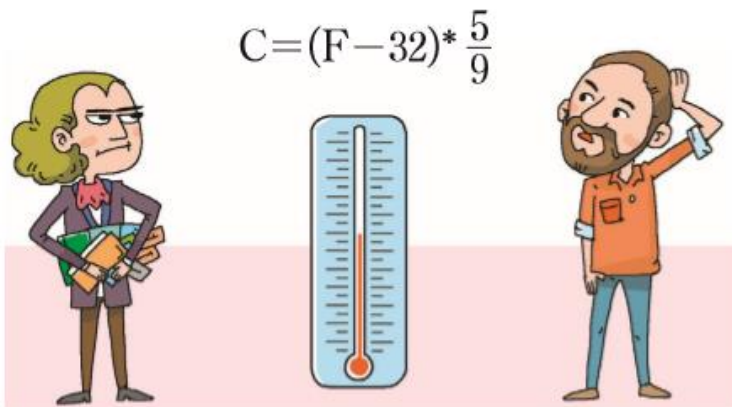


빛의 속도는 300000.000000km/s
태양과 지구와의 거리 149600000.000000km
도달 시간은 8.311111초

Lab: Converting temperature

- Let's write a program that receives the Fahrenheit temperature from the user and converts it to Celsius.

```
Microsoft Visual Studio 디버그 콘솔
화씨 온도=100
섭씨 온도=37.777778
```



Sol: Converting temperature

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

int main(void)
{
    double celsius, fahrenheit; // 변수 선언

    printf("화씨 온도=");
    scanf("%lf", &fahrenheit); // 부동소수점형으로 입력받는다.

    celsius = (fahrenheit - 32.0) * 5.0 / 9.0;
    printf("섭씨 온도=%lf \n", celsius);
    return 0;
}
```

Lab: Calculating the area of a circle

- Let's write a program that calculates the area of the circle by receiving the radius of the circle from the user.



The screenshot shows a window titled "Microsoft Visual Studio 디버그 콘솔" (Microsoft Visual Studio Debug Console). The window has a red title bar with standard Windows window controls (minimize, maximize, close). The console content is as follows:

```
원의 반지름을 입력하세요:10.0  
원의 면적: 314.159200
```

The text is in Korean. The first line is a prompt asking for the radius, and the second line shows the calculated area. The console has a vertical scrollbar on the right side.

Sol: Calculating the area of a circle

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#define PI          3.141592

int main(void)
{
    double radius;           // 원의 반지름
    double area;             // 원의 면적

    printf("원의 반지름을 입력하세요:");
    scanf("%lf", &radius);

    area = PI * radius * radius;
    printf("원의 면적: %f \n", area);
    return 0;
}
```

Character

- Texting is more important to humans than computers.
- Characters are represented by numbers in C language.

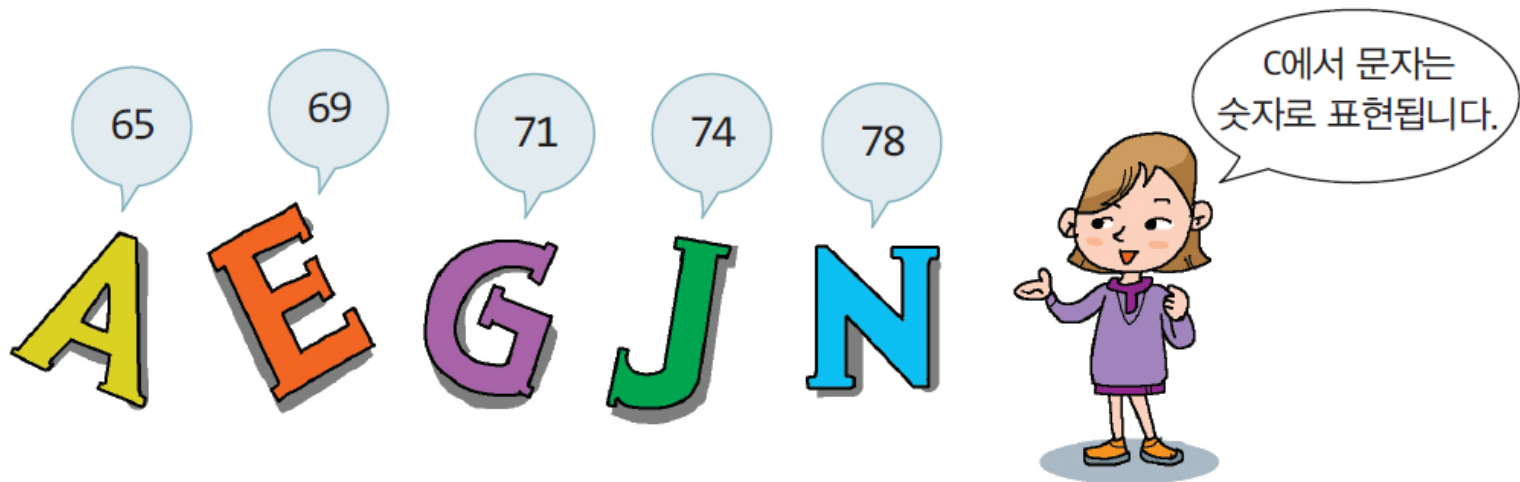


그림 3.9 C에서 문자는 숫자로 표현된다.

Character

- Common standards are needed.
- ASCII (American Standard Code for Information Interchange)
 - Using 8 bits to express English alphabets.
 - (e.g.) ' ': 32, 'A': 65, 'B': 66, 'a': 97, 'b': 98

The diagram illustrates the mapping of three characters to their ASCII binary representations. Each character is shown in blue on the left, followed by a wavy orange arrow pointing to its binary value in red on the right. The characters and their corresponding binary values are: 'B' (01000010), 'L' (01101100), and 'e' (01100101). The binary value for 'B' also includes the hexadecimal equivalent (0x42) in parentheses.

Character	ASCII Binary	Hexadecimal
B	01000010	0x42
L	01101100	
e	01100101	

ASCII code

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS, BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCLE]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

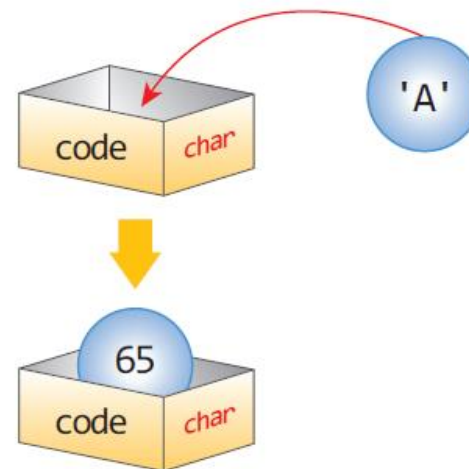
Character variables and character constants.

- Since characters are expressed as integers, data types that can store integers can also store characters. Since the ASCII code uses numbers from 0 to 127, it can be expressed in 8 bits.

char code;

- How can we store the letter A in the variable code of this char type?

code = 'A';



Example

```
#include <stdio.h>
int main(void)
{
    char c;           // 변수 선언

    c = 'A';          // 변수 c에 문자 'A'를 저장한다.
    printf("A의 아스키 코드= %d\n", c); // 문자와 아스키코드를 출력

    printf("문자를 입력하시오: ");      // 입력 안내문
    c = getchar();                       // scanf("%c", &c)하여도 된다.

    printf("%c의 아스키 코드= %d\n", c, c);
    return 0;
}
```

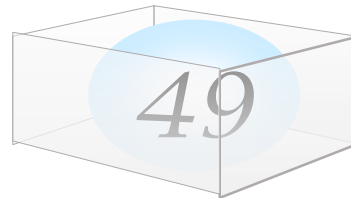
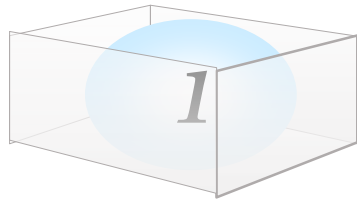
Microsoft Visual Studio 디버그 콘솔

A의 아스키 코드= 65
문자를 입력하시오: B
B의 아스키 코드= 66

Quiz

(Q) What's the difference between 1 and "1"?

(A) 1 is an integer and '1' is an ASCII code representing the letter '1'.



Check



Q 문자가 정수로 표현된다면 char 형의 변수가 문자를 저장하는지, 정수를 저장하는지를 어떻게 구별하나요.

A char 형의 변수에 저장된 값을 문자로 해석하면 문자라고 간주된다. 예를 들어서 화면에 출력할 때 %c를 사용하여서 출력하면 변수에 들어 있는 값을 아스키 코드로 해석한다. 반면에 %d를 사용하면 문자가 아니고 정수로 해석한다.

Interim check



중간점검

1. 컴퓨터에서는 문자를 어떻게 나타내는가?
2. C에서 문자를 가장 잘 표현할 수 있는 자료형은 무엇인가?
3. 컴파일러가 'A'를 만나면 어떻게 처리하는가?

Control text

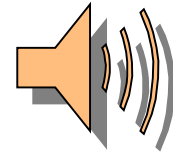
- Characters used for control, not for printing purposes.
 - (e.g.) Line changing text, tap text, ringtone text, backspace text.



How to express control characters.

- Use ASCII code

```
char beep = 7;  
printf("%c", beep);
```



- Use the escape sequence.

```
char beep = '\a';  
printf("%c", beep);
```



Control characters

제어 문자 이름	제어 문자 표기	값	의미
널문자	\0	0	
경고(bell)	\a	7	"삐"하는 경고 벨소리 발생
백스페이스 (backspace)	\b	8	커서를 현재의 위치에서 한 글자 뒤로 옮긴다.
수평 탭 (horizontal tab)	\t	9	커서의 위치를 현재 라인에서 설정된 다음 탭 위치로 옮긴다.
줄바꿈(newline)	\n	10	커서를 다음 라인의 시작 위치로 옮긴다.
수직탭(vertical tab)	\v	11	설정되어 있는 다음 수직 탭 위치로 커서를 이동
폼피드(form feed)	\f	12	주로 프린터에서 강제로 다음 페이지로 넘길 때 사용된다.
캐리지 리턴 (carriage return)	\r	13	커서를 현재 라인의 시작 위치로 옮긴다.
큰따옴표	\"	34	원래의 큰따옴표 자체
작은따옴표	\'	39	원래의 작은따옴표 자체
역슬래시(back slash)	\\	92	원래의 역슬래시 자체

Example

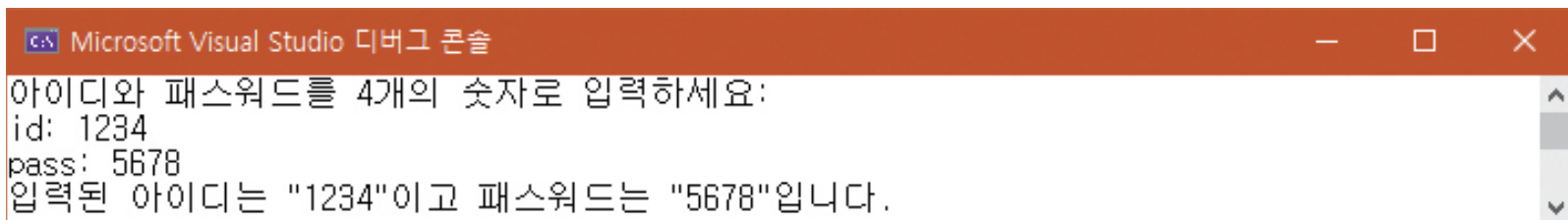
```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    int id, pass;

    printf("아이디와 패스워드를 4개의 숫자로 입력하세요:\n");
    printf("id: ____\b\b\b\b");
    scanf("%d", &id);

    printf("pass: ____\b\b\b\b");
    scanf("%d", &pass);

    printf("\a입력된 아이디는 \"%d\"이고 패스워드는 \"%d\"입니다.\n", id, pass);
    return 0;
}
```

A screenshot of the Microsoft Visual Studio Debug Console window. The title bar is orange and reads "Microsoft Visual Studio 디버그 콘솔". The console output shows the program's execution: it prompts for an ID and password, reads the input "1234" and "5678", and then prints a confirmation message with a bell character at the start. The output is: "아이디와 패스워드를 4개의 숫자로 입력하세요:", "id: 1234", "pass: 5678", and "입력된 아이디는 \"1234\"이고 패스워드는 \"5678\"입니다.". There are standard window control buttons (minimize, maximize, close) in the top right corner.

```
Microsoft Visual Studio 디버그 콘솔
아이디와 패스워드를 4개의 숫자로 입력하세요:
id: 1234
pass: 5678
입력된 아이디는 "1234"이고 패스워드는 "5678"입니다.
```

Check



참고: 정수를 입력받은 후에 문자를 입력받을 때

정수를 입력받은 후에 문자나 문자열을 입력받으면 이상하게 동작하는 것처럼 보인다.

```
scanf("%d", &i); // 정수 입력
```

```
c = getchar(); // 문자 입력
```

위의 코드를 실행하면 우리가 문자를 입력하기도 전에 실행이 끝나버린다. 그 이유는 정수를 입력할 때 엔터를 눌러야 하는데 이것이 줄바꿈 문자로 바뀌어서 입력 버퍼에 있어서이다.

'1'	'0'	'\n'	scanf("%d", &i)
		'\n'	getchar()

예를 들어서 우리가 키보드로 "10^{Enter}"를 치면 입력 버퍼에는 위와 같이 저장된다. "10"은 정수 10으로 변환이 되어서 입력 버퍼에서 없어지지만 '\n'은 남아있게 되고, 이어서 `getchar()`를 호출하면 이것이 우리에게 반환이 된다. 남아있는 줄바꿈 문자를 없애려면 `getchar()`를 한번 불러주면 된다.

```
scanf("%d", &i); // 정수 입력
```

```
getchar(); // 줄바꿈 문자를 없앤다.
```

```
c = getchar(); // 문자 입력
```

Interim check



중간점검

1. C에서 문자를 나타내기 위해 사용하는 코드는?
2. 경고음을 출력하는 제어 문자는 무엇인가?
3. 화면에 '\n'을 출력하려면 어떻게 하는가?

Mini Project

- In this chapter, we learned about the data types provided in C. Let's create a program that receives values from the user, stores them in appropriate datatype variables, and outputs them again.
 - Your age.
 - Company employee ID (total number of employees is less than 1000)
 - Annual product production (1 million units on average)
 - The number of stars in the universe.
 - One letter from A to z.

Mini Project



The screenshot shows the 'Microsoft Visual Studio 디버그 콘솔' (Debug Console) window. It contains a series of prompts and responses in Korean. The prompts are: '나이를 입력하시오:' (Enter age:), '직원 아이디를 입력하시오:' (Enter employee ID:), '생산량을 입력하시오:' (Enter production volume:), '별의 개수를 입력하시오:' (Enter the number of stars:), and '문자를 입력하시오:' (Enter a character:). The corresponding outputs are: '나이: 20', '아이디: 123', '생산량: 1000000', '별의 개수: 7.000000e+22', and '문자: a'. The window has a standard Windows title bar with minimize, maximize, and close buttons.

```
Microsoft Visual Studio 디버그 콘솔
나이를 입력하시오: 20
나이: 20
직원 아이디를 입력하시오: 123
아이디: 123
생산량을 입력하시오: 1000000
생산량: 1000000
별의 개수를 입력하시오: 7e22
별의 개수: 7.000000e+22
문자를 입력하시오: a
문자: a
```

When you receive a text, be careful of the remaining text in the input buffer.

```
#define _CRT_SECURE_NO_WARNINGS
...
...
scanf("%lf", &stars);
printf("별의 개수: %e \n", stars);

getchar();      // 줄바꿈 문자를 없앤다.
...
```


Q & A

