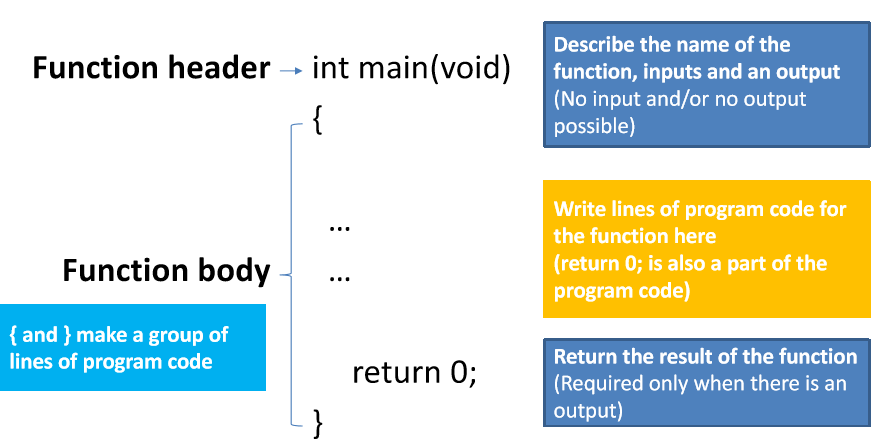
C프로그래밍 실습

**#2 Data and C**

함수의 기본구조



함수의 body의 기본구조

* 프로그램 코드는 Data(데이터)와 Instruction(명령)으로 구성 되어있다.

Data: 사용할 정보를 담고있는 숫자와, 글자

Instruction: 연산자를 사용하여 작업을 수행하는 방법을 설명하는 문장

Ex) + addition, - subtraction, \* multiplication, / division

Data variable(변수)

* 프로그램에 의해서 수정될 수 있는 값(value)

Constant(상수)

* 프로그램에 의해서 수정될 수 없고 미리 정의된 값(value)

Ex) area =

* Constant = (3.141592) , Data variables = r, area

Data types(자료형)

* 변수가 어떻게 보일 것인가? (number / alphabets)

review: 컴퓨터는 모든 것은 0또는 1로 저장한다.

따라서 메소드, 실수, 정수인지 구별하기위해서 Data type을 사용한다.

* 프로그래머는 반드시 자료형을 정의해야 한다.

정말 작은 부분이지만 반드시 양수인지, 음수인지에 대해 표기(sign)하여 숫자 자료형을

저장 해야 한다.

자료형은 보통 유효숫자 최대 7자리까지 실수를 저장한다

Data variable name (indentifier-식별자)

* 이름을 정할 때 규칙이 있다.

대소문자 구별: Hello와 hello는 다르다.

첫 문자는 알파벳이나 “\_”: hello, \_hello는 가능하지만, 1hello는 불가능

C언어에서 특별한 의미를 가진 이름은 사용할 수 없다. (Reserved Identifiers)

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

Declaration of Data Variables (변수선언)

* 변수를 적절한 이름으로 선언해주어야 한다.

특정한 data type(데이터형)과 관련 있는 identifier(식별자)로 정의해주어야 한다.

* Data type identifier; : 변수 하나만 선언

Int erns;

* Data type identifier1, identifier2, … identifierN; : 같은 데이터형의 N가지 변수를 선언

Int hogs, cows, goats;

“int”는 C에서의 정수형 숫자이며, 소수점 이하가 없고 부호가 있는 데이터 형이다.

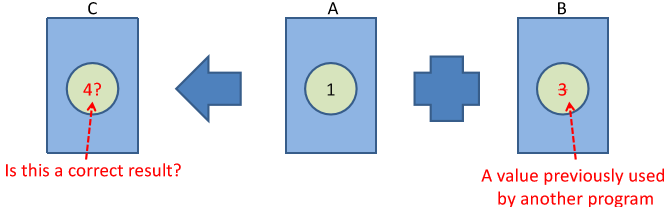
* 모든 변소는 함수의 제일 처음에 선언되어야 한다. (중간에 할당이 들어가면 안된다.)

Initialization of Data Variables (변수 초기화)

* 사용하기 전에 반드시 initialization(초기화)를 해주어야 한다.

데이터가 할당된 메모리 공간은 프로그램의 다른 부분에서 이미 사용되었을 수 있다.

만약 메모리 공간을 정리하거나, 초기화하지 않으면 잘못된 값을 사용할 수 있다.



* Identifier = value;

Hogs = 21;

* 할당 연산자 =

왼쪽에 위치한 변수안에 오른쪽에 위치한 결과 값을 할당 및 저장한다.

Declare and Initialize a Data Variables

* Declare과 initializing을 동시에 하는 방법
* data identifier = value;

int hogs= 21;

하지만 이것을 할 때 하나씩 해야함을 주의하자!

int dogs, cats = 94 : dogs는 initialized 되지 않았다.

모두 같은 표현이다.

Integer data type

* Integer = 소수점 부분이 없는 숫자
* 숫자를 메모리 공간에 저장하는 방법! => 이진수를 활용한다.

Ex) number 7

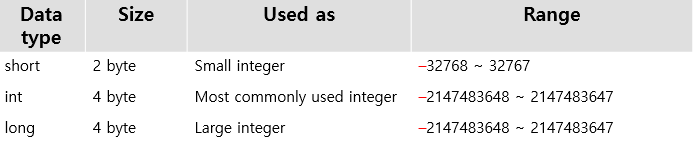


* 정수형 숫자의 범위

데이터형의 크기에 의해서 숫자의 범위는 변수의 데이터형에 따라 다양하다.

short int long 데이터형은 모두 signs, 부호가 있다.

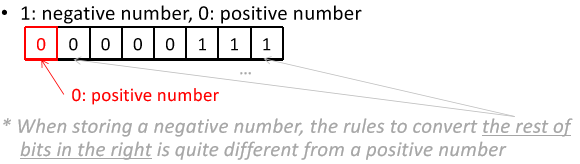




* signed vs unsigned data

signed: 양수, 음수, 0 / Unsigned 양수, 0

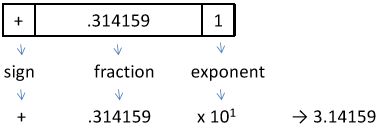
메모리공간 저장법



* sign은 언급 없이 정의 할 수 있고, unsigned는 data type 앞에 놓인다.
* int a; signed int a;
* Unsigned int a;

사용의 이유: 만약 양수인 데이터만 사용할 수 있다면 sign에서의 음수 범위만큼을 더해서 저장할 수 있다.

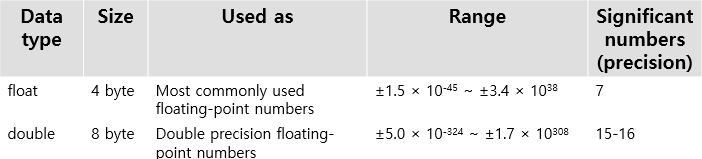
Floating-Point types

* Floating-point number: 부동 소수점수 (real number/ 실수)
* 7을 저장하는 방법

먼저 0.aaaa\*의 형태로 변환한다.

그리고 3개의 부분으로 나뉜다.

* 실수형 숫자의 범위



* [sign] [base number] [e or E] [exponent]

-1.56 \* -> -1.56e+12

* 접미사로 F나 F를 붙일 수도 있다.

-1.56e+12f

Using Characters: type char

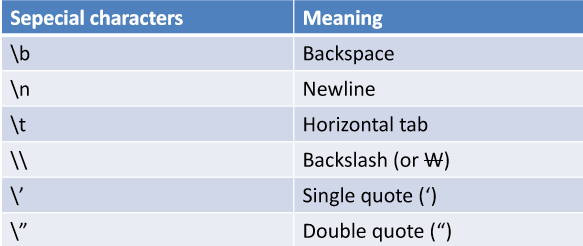
* char데이터형은 오로지 한 개의 글자만 저장할 수 있다.

Characters: letter(a, b, A, B, …), symbols (+, -, …), digits (0,1, …)

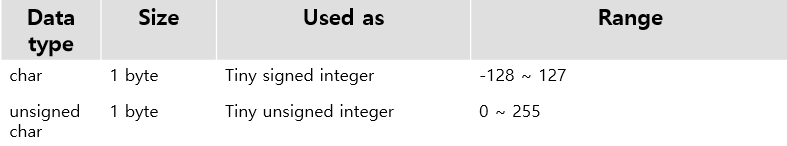
* char grade = ‘A’;

잘못된 사용: char grade = A; char grade =”A”;

* 특별한 character들

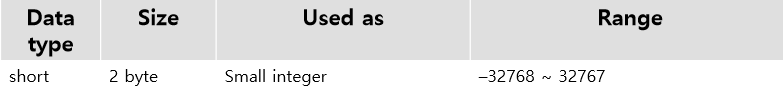


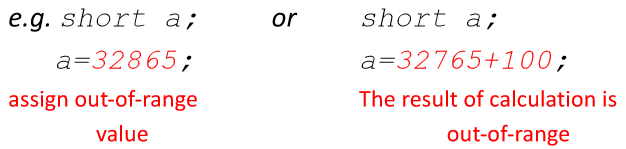
* 또한 이 데이터형은 작은 정수를 저장할 때에도 사용 될 수 있다.

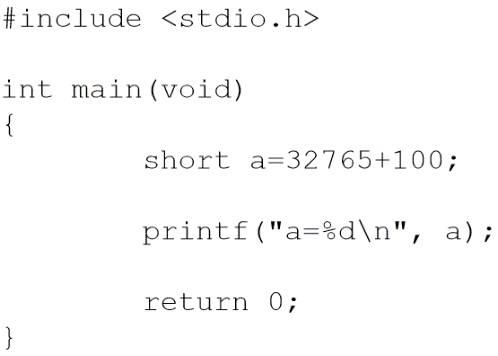


Integer Overflow

* 값은 어떻게 될까?







>>> a = -32671

무엇이 잘못되었을까?

결과값은 32865이다. 이것은 short의형의 범위 밖에 있다. (-32768~32767)

따라서 overflowed가 되었다.

<주의>: 컴파일러는 당신의 프로그램이 갖고있는 overflow문제를 고쳐주지 않는다.

따라서 overflow를 피하기 위해서는 좀더 넓은 범위의 데이터형을 사용하면 된다.

Floating-point overflow and round-off errors

* Overflow: 너무 큰 숫자일 때

Float toobig = 3.4E38 \* 100.0f;

수학적으로는 3.4E40이지만 컴퓨터에서는 infinite number로 인식한다.

* Round-off errors

float a,b;

b =2.0e20 + 1.0;

a = b – 2.0e20;

>>> result



수학에서 a=1.0, 하지만 결과 값으로 a= 4008175468544.00000을 얻는다.

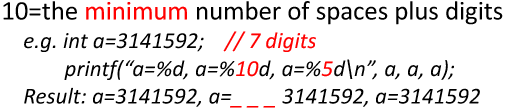
이 결과는 예상하지 못한 것으로 이외에도 등의 값이 나올 수 있다.

실수형 데이터의 유효숫자는 오로지 7이다. 그러나 위 예시의 유효숫자가 20이기 때문이다. 이것을 round-off errros라고 부른다.

How to use printf() in depth

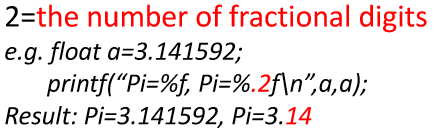
* %d: int data type. (좀더 작은 범위인 short, char의 경우에도 이것을 사용한다.)

%10d : 스페이스의 최소 숫자를 의미한다.

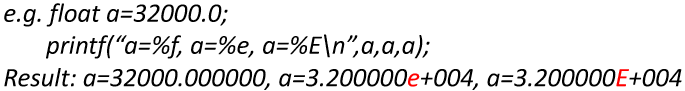


* %u: unsigned int data type (unsigned short, unsigned char)
* %ld: long data type
* %lu: unsinged long data type
* %f: float data type

%.2f: 유효숫자를 의미한다.

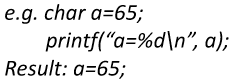
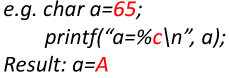


* %lf: double data type
* %e, %E: floating-point number의 표현!



* %c: char data type

Char data type이 작은 integer로 사용될 때!

컴파일러는 65를 A로 해석한다. Char ‘A’는 65와 같은 비트의 순서로 저장되어서

* printf()의 형식
* printf(“ %d %d …”, [data name], [data name], …);

이때, data name대신에 [a statement with arithmetic opterators]로 대체 할 수 있다.

Arithmetic operators: + , - , \* , /

scanf()

* 같은 프로그램은 항상 같은 결과를 낸다.

만약 input에 따라서 일이 수행되도록 발전시키고 싶다면 scanf가 필요하다.

* cmd창에 키보드 input을 제공한다. (pirntf()의 반대)
* scanf(“%d”, &[int data name]);
* scanf(“%f”, &[float data name]);