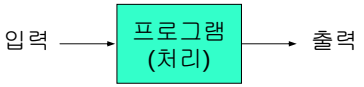
2장 C언어의 기초

- 식별자, 예약어
- 자료형
- 상수와 변수
- 수식
- 연산자: 산술연산자
- 자료형 변환

프로그램과 표준 입출력

❑ 프로그램(program)
- 자료를 입력 받아서 이를 처리하여 출력을 하는 일을 수행함



❑ 표준입출력(standard input and output: **stdio**)
- 표준입력: 키보드 입력
- 표준출력: 모니터 또는 터미널로 출력

(cf) DOS, Windows의 DOS 창/command 창, UNIX/Linux의 터미널에서 표준입출력 가능

출력 프로그램

❑ (예 2.1) 여러 개의 printf()가 있는 프로그램

```
#include <stdio.h>

int main(void)
{
    printf("ABCDE");
    printf("12345\n");
    printf("abcde\n");
    return 0;
}
```

교과서(개정판)에 한 줄이 빠졌음

→ printf("ABCDE12345\nabcde\n"); 와 같음

❑ 출력

```
ABCDE12345
abcde
```


계산을 하는 프로그램

❑ (예 2.2) 계산을 포함한 프로그램

```
#include <stdio.h>

int main(void)
{
    int x, y, z;                    ... 변수 선언: 정수 자료형

    x = 500;  y = 125;              ... 치환문: 변수값 초기화
    z = x + y;                      ... 연산: 덧셈
    printf("%d",z);                 ... 변수 출력문: 정수
    return 0;
}
```

[출력] 625

```
printf("Sum is %d.", z);
```

[출력] Sum is 625.

입력을 포함한 프로그램

- 변수 값 입력
  - scanf("%d", &a); … a = (정수 입력)
  - scanf("%f", &x); … x = (실수 입력)
- (예 2.4) 변수 값을 입력 받는 프로그램

```
#include <stdio.h>

int main(void)
{
    float r, s;     … 실수 변수 선언

    printf("Enter the radius of a circle : ");     … 입력 안내문
    scanf("%f",&r);                    … 실수 입력
    s = 3.14159 * r * r;               … 원의 넓이 계산
    printf("Area is %f \n", s);        … 변수 s 값 출력
    printf("Diameter is %f \n", 2*r);          … 수식 2*r 값 출력
    return 0;
}
```

[출력]
Enter the radius of a circle : *4*
Area is 50.265472
Diameter is 8.000000

식별자

- 식별자(identifier)
  - 변수, 함수, 상수와 같은 프로그램 구성 요소의 **이름**
- 식별자 생성 규칙
  - **규칙**: 영문자, 숫자, 밑줄문자(_)로 구성, 숫자로 시작하지 않음
  - **올바른 예**: sum, x2000, tax_rate1, tax_rate2, Table, table
  - **잘못된 예**: 4th, "x", tax-rate, id@host
- ANSI 표준은 31자까지 구별, 일부 컴파일러는 8자까지 구별
  - tax_rate1과 tax_rate2는 일부 컴파일러에서는 같은 식별자로 인식
- 식별자 권장사항
  - 의미를 알 수 있도록 부여하는 것이 바람직함 (예) tax
  - 두 단어를 결합한 식별자는 밑줄문자 또는 대문자로 구분함
    (예) tax_rate TaxRate

예약어

- 예약어(reserved word)
  - C언어에서 특별한 용도로 미리 예약된 이름
  - 키워드(keyword)라고도 함
  - 식별자로서 사용할 수 없음
- 예약어의 예
  - 자료형: char short int long unsigned float double
    struct union typedef enum void const signed
  - 제어문: if else switch case default for while do
    break continue goto return
  - 기억장소: auto register static extern volatile
  - 연산자: sizeof
- 일부 컴파일러에서 사용하는 예약어
  - ada fortran pascal asm
  - entry near far huge

자료형

- 자료형(data type)
  - 정수 (integer)
  - 실수 (floating point number): 부동소수점 수
  - 문자 (character)

  (cf) 정수 1과 실수 1의 컴퓨터 내부 표현은 다르다.
  문자는 컴퓨터 내부에서 정수로 표현된다.
- C언어의 기본 자료형
  - int 정수
  - char 문자
  - float 실수 (보통 정밀도)
  - double 실수 (2배 정밀도)

```
double  x;          … x는 2배정밀도 실수 변수
char  c;            … c는 문자 변수
```

자료형의 표현 범위

- int 자료형
  - 컴퓨터와 컴파일러에 따라서 16-bit 또는 32-bit로 표현됨.
  - 16-bit 정수: -32,768 (2^{15}) ~ 32,767(2^{15}-1)
  - 32-bit 정수: -2,147,483,648(2^{31}) ~ 2,147,483,647(2^{31}-1)

- float, double 자료형

	유효숫자	지수범위
float (32-bit 실수):	6자리	-38 ~ 38
double (64-bit 실수):	15자리	-308 ~ 308

유효숫자 x $10^{지수}$

- char 자료형
  - 문자는 8-bit로 표기 (cf) 한글 한 글자는 16-bit로 표기
  - char 자료형은 8-bit 정수 표기에 사용 가능
  - 8-bit 정수: -128 ~ 127

자료형 수정자

- 자료형 수정자(modifier)
  - 자료형 앞에 붙여서 자료형의 표현 범위를 변경
    (예) long, short, unsigned

long int	긴 정수(32-bit)
short int	짧은 정수(16-bit)
unsigned int	부호 없는 정수→ 양수의 표현 범위 2배 증가
unsigned long int	부호 없는 긴 정수(32-bit)
unsigned short int	부호 없는 짧은 정수(16-bit)

  - 자료형 수정자와 함께 사용하는 int는 생략 가능

```
long int a;
long a;
```
} 두 변수 선언은 같은 의미임

- unsigned 정수의 표현 범위
  - 16-bit unsigned: 0 ~ 65,535 (2^{16}-1)
  - 32-bit unsigned: 0 ~ 4,294,97,295(2^{32}-1)

상수

- 상수(constant)
  - 변하지 않는 명시적인 자료

- 정수상수: 12 056 0x4f
- 실수상수: 3.14 1.0 1.5e5
- 문자상수: 'A' '1' '\n' '@' ~~'한'~~
- 문자열상수: "Korea" "대한민국" "한"

정수 상수

- 정수 상수
  - 10진수: 1~9로 시작하여 표현 (예) 789
  - 8진수: 0으로 시작하여 표현 (예) 064
  - 16진수: 0x 또는 0X로 시작하여 표현 (예) 0x3af

  - 정수 상수는 기본적으로 int 형으로 표현됨
  - 정수 상수가 int 형 표현범위를 넘어서면
    컴파일러 판단에 의해서 unsigned 또는 long형으로 표현됨

- 정수형 지정 접미사 U, L

```
1234      … int형 정수
1234L     … long형 정수
40000U    … unsigned형 정수
40000UL   … unsigned long형 정수
```

실수 상수

- 실수 상수
  - 소수점을 포함한 숫자
    - 3.14
    - 1.0 1. … 소수부분=0 이면 소수점이하 생략 가능
    - 0.12 .12 … 정수부분=0 이면 정수부분 생략 가능
  - 과학용 표기법
    - 6.2e3 (= 6.2 x 10^3) 6.2E3
    - 4e-2 (= 4 x 10^{-2}) … 소수점이 없어도 실수 상수
  - 실수 상수는 기본적으로 double형으로 표현됨
- float형 지정 접미사 F

  6.2 … double형 실수

  6.2F … float형 실수

문자 상수

- 문자 상수
  - 영문자, 숫자, 특수문자를 작은 따옴표(' ') 로 묶어서 표시

    (예) 'A' 'b' '5' '@' '&'
- ASCII코드
  - 문자에 대응되는 정수 값을 정한 미국 표준 코드
  - ASCII코드는 8-bit 정수로 표현
  - ASCII코드의 순서는 숫자와 알파벳 순서와 같음
  - '0'과 0은 다르다.
    - '0'은 정수 48과 같음

ASCII	Char	ASCII	Char	ASCII	Char
32	Space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_	127	_

백슬래시 코드

- 백슬래시 코드 (탈출순서 표기)
  - 제어용 문자와 문자 표현에 특수한 용도로 사용되는 문자는 backslash(\)와 함께 나타냄 (예) '\n' '\t'
  - 컴퓨터 내부적으로는 한 문자로 표현됨
- 화면에 출력되지 않는 제어용 문자

  \n new line: 개행 문자 (다음 줄로)

  \r carriage return: 줄 처음으로

  \t tab \b backspace

  \0 **null 문자** (정수 0에 대응되는 문자)
- C언어에서 특수한 용도로 사용되는 문자

  \" 큰 따옴표 " \' 작은 따옴표 '

  \\ backslash \
- 8, 16진수 코드값 표기

  \101 8진수 코드로 표현된 문자 (10진수 65, 'A')

  \x42 16진수 코드로 표현된 문자 (10진수 66, 'B')

문자열 상수

- 문자열 상수
  - 연속된 문자들은 큰 따옴표(" ")로 묶어서 표시

    (예) "Korea"
- 문자열의 표현
  - 문자열은 컴퓨터 내부에서 null 문자로 끝나는 연속적인 문자들로 표현

    (예) "Korea" ➔ 'K' 'o' 'r' 'e' 'a' '\0'

K	o	r	e	a	\0

- 주의사항
  - "a"는 'a', '\0'로 구성되므로 'a'와 같지 않음
  - **null 문자열** ""은 '\0'만으로 구성되는 문자열임

기호 상수

- 기호상수
  - 상수에 부여한 이름
  - **#define**을 사용하여 기호상수 정의
  - 여러 번 사용하는 상수나 값이 바뀔 수 있는 상수에 대해서 유용함
- (예 2.6) 기호상수를 사용한 프로그램

```
#include <stdio.h>
#define PI  3.14159              ... 실수상수 대신에 PI를 사용

int main(void)
{
   float r, s;                   ... 변수 선언

   printf("Enter the radius of a circle : ");    ... 입력 안내문
   scanf("%f",&r);               ... 실수 입력
   s = PI * r * r;               ... 원의 넓이 계산
   printf("Area is %f \n", s);   ... 변수 s 값 출력
   return 0;
}
```


변수

- 변수(variable)
  - 값을 저장할 수 있는, 이름이 부여된 기억장소
- 변수의 선언
  - 변수를 사용하기 전에 반드시 변수의 자료형을 선언해야 함.

```
int  p;
float a, b, c;        ... 여러 개의 같은 자료형의 변수 선언
```

  - 변수를 선언하면 변수의 메모리 공간이 확보됨
  - 변수 선언 위치: 함수의 시작 부분 (다른 위치는 나중에 다룸)

```
int main(void)
{
   변수 선언
   프로그램 실행문
}
```

(cf) C++ 언어에서는 변수를 사용하기 전에만 변수선언을 하면 됨

변수의 초기화

- 변수는 초기값을 지정한 다음에 사용해야 함

```
int  a;
  …
b = a;   (X)          … a의 값이 정의되지 않았으므로 잘못
```

- 실행문에서 초기화

```
int  a;
  …
a = 25;
```

- 변수의 초기화 선언

```
int  a = 25;
float  data, sum = 0.0, average;     … sum만 초기화 (권장하지 않음)
```


주석

- 주석(comment)

```
/* This is a comment */          … /*  */로 둘러싸인 부분
```

  - 설명문으로서 컴파일러에 의해서 무시됨
  - 프로그램 저작권, 수정 이력, 동작 설명 등의 문서화 용도로 사용
- 예

```
/*  hello.c
 *  내용: 자료를 순서대로 나열한다.
 *  ver 0.0: 03/6/10  작성자: 홍길동
 */
int sum;    /* 자료들의 합 */
int max;    /* 최대값 */
```

여러 줄에 걸쳐서 작성 가능

- C++ 주석

```
//  C++ comment                  ... // 부터 시작하여 줄 끝까지가 주석
(예)   int sum;     // 자료들의 합
```

수식

- 수식(expression)
  - 상수, 변수, 함수 호출 또는 이들과 연산자와의 조합

    10 … 상수
    x … 변수
    sin(x) … 함수호출
    x + 10*y … 이들과 연산자와의 조합
- 수식의 값
  - 대부분의 수식은 **값**을 가짐
    (예외) 반환 값이 없는 함수 호출
  - 수식의 연산들은 정해진 순서대로 수행하며
    수식의 최종 연산 결과가 수식의 값이 됨.

치환문

- 치환문

  c = a + 10; … 오른쪽 수식의 값을 왼쪽 변수에 저장
- 치환 연산자(assignment operator) =
  - C언어에서는 = 도 연산자로 취급함
  - = 을 치환연산자 또는 배정 연산자라고 함
  - = 은 수학적 등호가 아님

```
k = 5;
k = k + 1;          … k에 6이 저장됨
```

- 치환 수식
  - 치환 수식: 치환 연산자를 포함한 수식
  - 왼쪽 변수에 저장되는 값이 치환 수식의 값이다.

```
k = 10 + 5;         … 치환 수식 값 = 15
```


연산자

- 연산자(operator)
  - **변수, 상수, 함수 호출 값**에 대해서 연산을 수행하여 결과를 제공
    → 피연산자(operand): 연산의 대상
  - 연산은 기본적으로 같은 자료형에 대해서 수행함
    - 정수들 간의 연산 결과 → 정수
    - 실수들 간의 연산 결과 → 실수
- 산술연산자

  이항연산자 단항연산자

  \+ 덧셈
  – 뺄셈, 부호반대 11 – 4 = 7 (뺄셈), – 4 (부호반대)
  * 곱셈
  / 나눗셈 11 / 4 = 2 (몫), 11.0 / 4.0 = 2.75
  % 나머지 11 % 4 = 3
  - 나머지 연산은 정수에 대해서만 사용 가능

연산 우선순위와 결합성

- 연산 우선 순위(precedence)
  - 수식에 두 종류 이상의 연산자가 포함되어 있을 때에
    연산을 평가(evaluation)하는 순서
    (예) a + b * c … 곱셈(*) 우선
- 결합성(associativity)
  - 같은 우선 순위의 연산자에 대한 연산 실행 순서
  - 좌결합성: 앞(왼쪽)에서부터 (예) a + b – c → (a+b) – c
  - 우결합성: 뒤(오른쪽)에서부터 (예) a = b = c → a = (b=c)
- 주요 연산자의 우선순위와 결합성

우선순위	연산자	결합성
1	괄호 ( )	좌
2	단항 –	우
3	* / %	좌
4	+ –	좌
5	=	우

여러 가지 수식과 다중 치환문

- (예) 수학식에 대한 C언어 수식

$b^2 - 4ac$ → b*b – 4.0*a*c

$\frac{a+b}{c-d}$ → (a+b) / (c-d)

$2x + \frac{1}{2}y - \frac{1}{z^2}$ → 2.0*x + y/2.0 – 1.0/(z*z)

괄호 사용에 유의

- 다중 치환문

x = y = 5; → x = (y = 5); → 우결합성 →

```
y = 5;
x = y;
```

… x와 y에 5 저장

```
c = 10;
a = b = c + 5;
```

… a와 b에 15 저장

자료형 변환

- 자료형 자동 변환

1.0 + 2 → 1.0 + 2.0 = 3.0 … 자료형을 같도록 한 후에 연산 수행

- 수식에서의 자료형 변환
  - 먼저 작은 정수는 보통 정수로 형 변환

    char, short → int

    unsigned char, unsigned short → unsigned
  - 이항 연산의 피연산자는 둘 중 큰 자료형으로 변환됨

    int < unsigned < long < unsigned long < float < double

(예) s / i + f / d → i / i + f / d = i + f / d → i + d / d = i + d → d + d = d

(s: short, i: int, f: float, d: double)

자료형 변환

- 치환문에서의 자료형 변환
  - 왼쪽 변수의 자료형으로 변환됨.

```
#include <stdio.h>                  /* 예 2.9 */

int main(void)
{
    double    d;                    /* 64-bit 실수 */
    float     f;                    /* 32-bit 실수 */
    int       a;                    /* 32-bit 정수 */
    short     b;                    /* 16-bit 정수 */

    d = 123456.789012;
    f = d;                          /* f = 123456.789063 : 정밀도 손실 */
    a = f;                          /* a = 123456 : 소수부분 손실  */
    b = a;                          /* b = -7616 : 의미없는 값 */
    printf("%f  %f  %d  %d", d, f, a, b);
    return 0;
}
```

[**출력**] 123456.789012 123456.789063 123456 -7616

형 변환 연산자

- 형 변환 연산자
  - 수식의 값의 자료형을 명시적으로 변환하는 연산자

    (double) num

    (double) (a + b*c)

  - 형 변환 연산자 (*type*) 은 단항연산자 우선순위

    (double) a / 4 → ((double)a) / 4 … 형 변환 후 나눗셈

    (double) (a / 4) … 나눗셈 후 형 변환