3장 C 프로그램을 이루는 구성 요소

김명호

내용

- 주석문
- 토큰
- 키워드
- 식별자
- 상수
- 문자열 상수
- 구두자

구문

- Syntax
- 올바른 프로그램을 만들 수 있게 하는 규칙
- 컴파일러
  - C 프로그램이 구문에 맞는지 검사
  - 오류가 있다면, 오류 메시지 출력
  - 오류가 없다면, 목적 코드 생성
- 전처리기
  - 컴파일러 이전에 호출

컴파일러

- **컴파일 과정**
  - C 프로그램 → 토큰으로 분리 → 토큰을 목적 코드로 변환
  - 토큰 종류 : 키워드, 식별자, 상수, 문자열 상수, 구두자

주석문

- 컴파일러는 주석문을 공백으로 처리
- 주석문
  - 여러 줄 주석
  - 줄 단위 주석 (C99)
- 주석은 문서화 도구로 사용됨
  - 프로그램 설명이나 알고리즘 등 기술

주석문

- **여러 줄 주석**
  - /*과 */ 사이에 있는 임의의 문자열
  - 주석은 토큰이 아님
  - 컴파일러는 주석을 하나의 공백 문자로 대치
  - 문서화 도구로 사용함(프로그램 설명, 정확성 증명 등)
- **줄 단위 주석**
  - // 으로 시작 : 그 다음부터 그 행 끝까지가 주석임
  - C99에 추가

주석 예제

프로그램 3.1

```
/*************************************
 * 이 프로그램은 예제 프로그램입니다.      *
 *          2431년 9월 1일               *
 *************************************/
#include <stdio.h>
int main(void)
{
   printf("나의 ");
   printf("첫 번째 프로그램");
   return 0;
}
```

주석 예제

```
/* 첫 번째 주석 예제 */

/*** 두 번째 주석 예제 ***/

/*
 * 세 번째
 * 주석 예제
 */

/**************************
 * 네 번째                *
 * 주석 예제입니다.       *
 **************************/
```

주석 예제

프로그램 3.2

```
/* 반지름을 입력받아 원의 넓이를 구하는 프로그램
 *  2014년 3월 20일
 *  김 진혁 작성
 */
#include <stdio.h>
int main(void)
{
   float a;                  // 반지름 변수
   printf("반지름 : ");
   scanf("%f", &a);
   // 원의 넓이 = 반지름 * 반지름 * 원주율(3.14)
   printf("반지름이 %f인 원의 넓이 : %f\n", a, a * a * 3.14);
   return 0;
}
```

키워드

- 키워드
  - C 언어에서 고유한 의미를 가지는 토큰
  - 예약된 단어
  - 프로그래머가 다른 의미로 재정의할 수 없음
    - 변수나 함수 이름으로 사용할 수 없음

프로그램

```c
/* 반지름을 입력받아 원의 넓이를 구하는 프로그램
 *  2014년 3월 20일
 *  김 진혁 작성
 */
#include <stdio.h>
int main(void)
{
    float a;          // 반지름 변수
    printf("반지름 : ");
    scanf("%f", &a);
    // 원의 넓이 = 반지름 * 반지름 * 원주율(3.14)
    printf("반지름이 %f인 원의 넓이 : %f\n", a, a * a * 3.14);
    return 0;
}
```

프로그램

```c
/* 반지름을 입력받아 원의 넓이를 구하는 프로그램
 *  2014년 3월 20일
 *  김 진혁 작성
 */
#include <stdio.h>
int main(void)
{
    float a;          // 반지름 변수
    printf("반지름 : ");
    scanf("%f", &a);
    // 원의 넓이 = 반지름 * 반지름 * 원주율(3.14)
    printf("반지름이 %f인 원의 넓이 : %f\n", a, a * a * 3.14);
    return 0;
}
```

키워드

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

* inline, restrict, _Bool, _Complex, _Imaginary는 C99에서 추가

키워드

프로그램 3.3

```
#include <stdio.h>
int main(void)
{
    typedef int double;  // 오류 : double을 int로 재정의
    int float;           // 오류 : float을 int 형의 변수로 선언
    return 0;
}
```

식별자

- 변수, 함수, 사용자 정의형에 이름을 부여하기 위함

프로그램

```c
/* 반지름을 입력받아 원의 넓이를 구하는 프로그램
 *  2014년 3월 20일
 *  김 진혁 작성
 */
#include <stdio.h>
int main(void)
{
    float a;          // 반지름 변수
    printf("반지름 : ");
    scanf("%f", &a);
    // 원의 넓이 = 반지름 * 반지름 * 원주율(3.14)
    printf("반지름이 %f인 원의 넓이 : %f\n", a, a * a * 3.14);
    return 0;
}
```

식별자

- 식별자는 문자, 숫자, 그리고 특수문자인 밑줄문자(_)로 구성된 토큰으로, 문자 또는 밑줄문자로 시작해야 함
- C는 소문자와 대문자를 구별함
- 식별자는 그 의미를 생각하고 선택해야 함
- 키워드는 사용자 식별자로 사용할 수 없음
- 라이브러리 함수명, main, 미리정의된 문자상수 등은 사용자 식별자로 사용하지 않는 것이 좋음

식별자

- **올바른 식별자**

  k

  _id

  iamanidentifier2

  so_am_i

- **잘못된 식별자**

  not#me

  101_south

  -plus

예제 프로그램

프로그램 3.4

```
#include <stdio.h>
int main(void)
{
    float a, b;

    scanf("%f", &a);
    b = a * 2.54;
    printf("%f => %f\n", a, b);
    return 0;
}
```

예제 프로그램

프로그램 3.5

```
#include <stdio.h>

#define INCH2CM 2.54
int main(void)
{
    float length_inch, length_cm;
    scanf("%f", &length_inch);
    length_cm = length_inch * INCH2CM;
    printf("%f => %f\n", length_inch, length_cm);
    return 0;
}
```

상수

- 정수 상수
- 부동형 상수
- 문자 상수
- 열거 상수
  - enum에 의해 선언된 상수

- 상수는 값과 형을 가짐

정수 상수

- 정수 상수(Integer Constant)
  - 10진 정수
  - 16진 정수 : 0x로 시작
  - 8진 정수 : 0으로 시작

정수 상수

- 10진 정수 상수
  - 0, 17, 234
- 16진 정수 상수
  - 0~9, a(A)~f(F)
  - 0x17, 0x234, 0xFFFF
  - 변환명세 : %x, %#x
- 8진 정수 상수
  - 0~7
  - 017, 0234
  - 변환명세 : %o, %#o

예제 프로그램

프로그램 3.6

```
#include <stdio.h>
int main(void)
{
    printf("%d %x %o\n", 19, 19, 19);
    printf("%d %x %o\n", 0x1c, 0x1c, 0x1c);
    printf("%d %#x %#o\n", 017, 017, 017);
    return 0;
}
```

프로그램 결과

```
19 13 23
28 1c 34
15 0xf 017
```

부동형 상수

- **부동형 상수 (Floating Constant)**
  - 0.17, 0.234
  - 정수 상수와 구분해서 사용해야 함
- **10진 부동형 상수**
  - 1.7e10 (= 1.7×10^{10})
- **16진 부동형 상수**
  - 0x1.7p10 (= $(0x1.7) \times 2^{10}$)

예제 프로그램

프로그램 3.7

```
#include <stdio.h>
int main(void)
{
    printf("%.2f\n", 1 / 3);
    printf("%.2f\n", 1 / 3.0);
    return 0;
}
```

프로그램 결과

```
0.00
0.33
```

문자 상수

- 문자 상수(Character Constant)
  - 작은 따옴표로 둘러싸인 문자
  - 'a', 'b', '+', ...
- 와이드 문자 상수
  - 일반 문자 상수에 L 접두사 붙여 표현
  - L'a', L'b', ...
- 탈출 문자
  - 키보드로 표현할 수 없는 문자를 나타내기 위해
  - '\n', '\a', '\t', ...

탈출 기법

- 인쇄할 수 없는 문자는 탈출 기법을 사용하여 표현함
  - 예를 들어, 수평 탭 문자는 문자 상수와 문자열에서 \t로 표현됨
  - \t가 \와 t 두 문자로 기술되지만, 이것은 한 문자임
- 프로그램 내에서 특별한 의미를 갖는 문자들이 본래의 의미를 갖기 위해서도 탈출 기법을 사용해야 함
  - 큰따옴표를 포함하는 "abc"라는 문자열은 "\"abc\""로 표기함
  - 작은따옴표 문자 상수 '는 '\''로 표기함

탈출 문자

탈출 문자	의미
\a	경고
\\	역슬래시
\b	백스페이스
\r	캐리지 리턴
\"	큰따옴표
\f	폼피드
\t	수평 탭
\n	개행
\0	널 문자
\'	작은따옴표
\v	수직 탭

예제 프로그램

프로그램 3.8

```
#include <stdio.h>
int main(void)
{
    printf("\a");
    printf("I like\b\b\bove C.\n");
    return 0;
}
```

프로그램 결과

```
<삑>
I love C.
```

열거 상수

- 열거 상수(Enumeration Constant)
  - enum으로 정의된 식별자

문자열 상수

- 문자열 상수(String Constant)
  - 큰따옴표에 의해 묶인 일련의 문자들
- 문자열 내에 “를 넣기 위해서는 앞에 \ 붙임
- 공백에 의해 분리된 두 문자열 상수는 하나의 문자열로 결합
- 문자열 상수 중간에 \를 삽입하면 다음 줄로 문자열이 연결됨

예제 프로그램

프로그램 3.9

```
#include <stdio.h>
int main(void)
{
    printf("%s\n", "재""미" "있는 " "\"C\"!");
    printf("%s\n", "\\n는 \
개행문자.\n");
    return 0;
}
```

프로그램 결과

```
재미있는 "C"!
\n는 개행문자.
```

구두자

- **구두자(Punctuator)**

```
[ ] ( ) { } . ->
++ -- & * + - ~ !
/ % << >> < > <= >= == != ^ | && ||
? : ; ...
= *= /= %= += -= <<= >>= &= ^= |=
, # ##
<: :> <% %> %: %:%:   // 다이그래프(digraph)
                      // [, ], {, }, #, ##
```

구두자

- 어떤 구두자는 사용된 위치에 따라 의미가 달라질 수 있음
  - ( )

```
printf("%d", a * (b + a));
```

- 어떤 구두자는 다른 문자와 같이 사용되면 다른 의미를 가짐
  - +

```
+, ++, +=
```

산술 연산자

프로그램 3.10

```
#include <stdio.h>
int main(void){
    printf("9 + 5 = %d\n", 9 + 5);
    printf("9 - 5 = %d\n", 9 - 5);
    printf("9 * 5 = %d\n", 9 * 5);
    printf("9 / 5 = %d\n", 9 / 5);
    printf("-9 = %d\n", -9);
    printf("+9 = %d\n", +9);
    printf("9 %% 5 = %d\n", 9 % 5);
    printf("9.0 + 5.0 = %f\n", 9.0 + 5.0);
    printf("9.0 - 5.0 = %f\n", 9.0 - 5.0);
    printf("9.0 * 5.0 = %f\n", 9.0 * 5.0);
    printf("9.0 / 5.0 = %f\n", 9.0 / 5.0);
    printf("-9.0 = %f\n", -9.0);
    printf("+9.0 = %f\n", +9.0);
    return 0;
}
```

프로그램 결과

```
9 + 5 = 14
9 - 5 = 4
9 * 5 = 45
9 / 5 = 1
-9 = -9
+9 = 9
9 % 5 = 4
9.0 + 5.0 = 14.000000
9.0 - 5.0 = 4.000000
9.0 * 5.0 = 45.000000
9.0 / 5.0 = 1.800000
-9.0 = -9.000000
+9.0 = 9.000000
```

우선순위와 결합순위

- 하나의 수식에 여러 개의 연산자가 올 수 있음
- 우선순위와 결합순위는 수식의 평가 순서를 결정함

연산자	결합 순위
( ) ++ (후위) --(후위)	좌에서 우로
+ (단항) - (단항) ++ (전위) -- (전위)	우에서 좌로
* / %	좌에서 우로
+ -	좌에서 우로
= += -= *= /= %= >>= <<= &= ^= |=	우에서 좌로

우선순위와 결합순위

- 예
  - 우선순위

    1 + 2 * 3 ➔ 1 + (2 * 3)

    -a – b ➔ (-a) - b
  - 결합순위

    1 + 2 - 3 + 4 – 5 ➔ (((1 + 2) - 3) + 4) - 5
  - 괄호로 우선순위 변경 가능

    (1 + 2) * 3

배정 연산자

- **배정 수식 형식**

  변수 = 수식

  // = 기호 오른쪽 수식의 값이 변수에 배정됨

- **예**

```
a = b + c
x = x + 1
```

- **배정문**

```
b = 2;
```

배정 연산자

- 다른 언어와는 달리 C는 =를 연산자로 다룸

```
b = 2;
c = 3;
a = b + c;
```

→ `a = ( b = 2 ) + ( c = 3 );`

```
a = b = c = 0;
```

→ `a = (b = (c = 0));`

증가와 감소 연산자

- **전위 증가 연산자**

  ++i → i = i + 1

- **전위 감소 연산자**

  --i → i = i – 1

- **후위 증가 연산자**

  i++ → i = i + 1

- **후위 감소 연산자**

  i-- → i = i - 1

증가와 감소 연산자

- 전위와 후위 연산자는 같은 기능을 하지만 수식의 값에 차이가 있음

  ++i 수식의 값 : 새로운 i의 값

  --i 수식의 값 : 새로운 i의 값

  i++ 수식의 값 : 이전 i의 값

  i-- 수식의 값 : 이전 i의 값

증가와 감소 연산자

- i가 3일 때

```
++i 수식의 값 : 4
--i 수식의 값 : 2
i++ 수식의 값 : 3
i-- 수식의 값 : 3
```

예제 프로그램

프로그램 3.11

```
#include <stdio.h>
int main(void)
{
    int a, b, c, d;
    c = d = 0;
    a = ++c;        // 이전 값이 0인 c에 전위 ++적용
    b = d++;        // 이전 값이 0인 d에 후위 ++적용
    printf("a = %d, b = %d, c = %d, d = %d\n", a, b, c, d);
    return 0;
}
```

프로그램 결과

```
a = 1, b = 0, c = 1, d = 1
```

복합 배정 연산자

- 다른 연산자와 배정 연산자가 연결된 형태
- 변수의 새로운 값이 이전 값과 연관될 때 유용
  - k = k + 2 형태의 수식을 축약해서 표현
- 복합 배정 연산자

  +=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=

  - k = k + 2; ➔ k += 2;
  - (주의) j *= k + 3; ➔? j = j * k + 3;

    ➔? j = j * (k + 3);

예제 프로그램

프로그램 3.12

```
#include <stdio.h>
int main(void)
{
    int i = 1, j = 2, k = 3, m = 4;
    printf("i += j + k : %d\n", i += j + k);
    printf("j *= k = m + 5 : %d\n", j *= k = m + 5);
    return 0;
}
```

프로그램 결과

```
i += j + k : 6
j *= k = m + 5 : 18
```