

# 어휘원소, 연산자와 C 시스템

**박 종 혁 교수**

**UCS Lab**

**Tel: 970-6702**

**Email: [jhpark1@seoultech.ac.kr](mailto:jhpark1@seoultech.ac.kr)**

# 목차

- 1.1 문자와 어휘 원소
- 1.2 구문 법칙
- 1.3 주석
- 1.4 키워드 (Keyword)
- 1.5 식별자 (Identifier)
- 1.6 상수 (Integer, Constant)
- 1.7 문자 상수, 문자열 상수
- 1.8 연산자와 구두점

# 어휘 원소, 연산자, C 시스템

- **구문**

- 올바른 프로그램을 만들 수 있게 하는 규칙

- **컴파일러**

- C 프로그램이 구문에 맞는지 검사
- 오류가 있다면, 오류 메시지 출력
- 오류가 없다면, 목적 코드 생성

- **컴파일 과정**

- C 프로그램 → 토큰으로 분리 → 토큰을 목적 코드로 변환
- 토큰 종류 : 키워드, 식별자, 상수, 문자열 상수, 연산자, 구두점

# 1.1 문자와 어휘 원소

- 프로그램에서 사용할 수 있는 문자
  - 소문자 : a b c ... z
  - 대문자 : A B C ... Z
  - 숫자 : 0 1 2 3 4 5 6 7 8 9
  - 특수문자 : + - \* / = ( ) [ ] < > ' “  
! @ # \$ % & \_ | . , ; : ?
  - 여백문자 : 공백, 개행, 탭
- 컴파일러는 이러한 문자들을 구문 단위인 토큰으로 모은다.

# 어휘 분석

- **/\* Read in two integers and print their sum. \*/**
  - 주석문 : /\*부터 \*/까지는 공백으로 대치
- **#include <stdio.h>**
  - 전처리 지시자 : 전처리가 처리
- **int main(void){**
  - int a, b, sum;**
    - 키워드 : int, void
    - 식별자 : main, a, b, sum
    - 연산자 : ( )
    - 구두점 : "{", ":", ";"
  - inta, b, sum; -> (X), int absum -->absum을 하나의 식별자,

- 예제 프로그램

- ▶ `printf("Input two integer: ");`

- `scanf("%d%d", &a, &b);`

- `printf, scanf` : 식별자, (): 함수임을 알림

- "Input two integer: " : " 문자열 상수 "

- `&` : 주소연산자, `& a, & b` (O), `&a,&b` (O), `&a &b` (X),

- `a&, &b` (X)

- ▶ `sum=a + b ;`

- `=, +` : 연산자, `sum=a+b;` (O), `sum = a + b ;` (O),

- `s u m = a + b ;`(X)

## 1.2 구문 규칙

- **BNF(Backus-Naur Form)으로 기술**

예)  $digit = 0|1|2|3|4|5|6|7|8|9$

- 생산 규칙
- 의미 : 구문 카테고리  $digit$ 는 기호 0 또는 1, ..., 또는 9로 다시 쓸 수 있다.

## • 생산 규칙에 사용되는 기호들

- *italics* 구문 카테고리
- ::= "다시 쓰면"의 기호
- | 선택들을 분리
- $\{ \}_1$  괄호 안의 항목 중 하나만 선택
- $\{ \}_{0+}$  괄호 안의 항목을 영번 이상 반복
- $\{ \}_{1+}$  괄호 안의 항목을 한번 이상 반복
- $\{ \}_{opt}$  옵션인 항목



## \*\*생산 규칙 예제\*\*

- ***letter\_or\_digit***
  - $letter\_or\_digit ::= letter | digit$
  - $letter ::= lowercase\_letter | uppercase\_letter$
  - $lowercase\_letter ::= a | b | c | \dots | z$
  - $uppercase\_letter ::= A | B | C | \dots | Z$
  - $digit ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9$
- ***alphanumeric\_string***  $::= \{letter\_or\_digit\}_{o+}$ 
  - “3”, “ab777c”, “ ” ← 알파뉴메릭 문자열
- ***u\_alpha\_string***  $::= uppercase\_letter \{letter\_or\_digit\}_{o+}$
- ***conditional\_statement***  $::= if (expression) statement \{else\_statement\}_{opt}$

# 1.3 주석

- **주석**

- /\*과 \*/ 사이에 있는 임의의 문자열
- 주석은 토큰이 아님
- 컴파일러는 주석을 하나의 공백 문자로 대치
- 문서화 도구로 사용함(프로그램 설명, 정확성 증명 등)

- **C++ 주석**

- 줄 단위 주석
  - // 다음부터 그 행 끝까지가 주석임
- C 스타일의 주석도 사용

## \*\*주석 예제\*\*

- **C 스타일 주석**

```
/* a comment */
```

```
/*  
 * A comment can be written in this fashion  
 * to set it off from the surrounding code.  
 */
```

```
/******  
 * If you wish, you can *  
 * put comments in a box. *  
 *****/
```

- **C++ 스타일 주석**

```
// This is a comment in C++.
```

# 1.4 키워드

- 키워드

- C 언어에서 고유한 의미를 가지는 토큰
- 예약된 단어

- C 키워드

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

## 1.5 식별자

- 식별자는 문자, 숫자, 그리고 특수문자인 밑줄문자(\_)로 구성된 토큰으로, 문자 또는 밑줄문자로 시작해야 함.
- C 시스템은 소문자와 대문자를 구별함.
- 식별자의 선택은 의미를 생각하여 함.

- 식별자 생성 규칙

*identifier ::=*

$\{letter|underscore\}_1\{letter|underscore|digit\}_{0+}$

*underscore ::=* \_

- 올바른 예제

K, \_id, iamanidentifier2, so\_am\_I

- 식별자의 틀린 예

not#me /\* special character # not allowed \*/

101\_south /\* must not start with a digit \*/

-plus /\* do not mistake - for \_ \*/

- C 표준 라이브러리에 정의된 식별자 :  
**printf, scanf, ...**

- 의미가 쉽게 연상되는 식별자(변수) 사용할것.
- 밑줄문자(\_)로 시작되는 식별자는 가급적 사용하지 말 것.

# 1.6 상수

- 정수 상수
  - 0, 17, 234, 0x17
- 실수 상수
  - 1.0, 3.141592, 23E2
- 문자 상수
  - 'a', 'b', '+', '\n'
- 문자열 상수
  - "hello", "very good"
- 열거 상수
  - enum에 의해 선언된 상수

**(주의) -49 : 상수 수식 // 음수정수상수는 상수 수식으로 간주함**



## 1.7 문자열 상수

- 문자열 상수 - 큰따옴표에 의해 묶인 일련의 문자들

- 올바른 예제

```
"a string of text"  
""
```

```
"  "
```

```
"  a = b + c; "
```

```
" /* this is not a comment */ "
```

```
" a string with double quotes \" within"
```

- 잘못된 예제

```
/* "this is not a string" */
```

```
"and
```

```
neither is this"
```

# 1.8 연산자와 구두점

- 연산자

+ , - , \* , / , %

- 구두점

- 괄호, 중괄호, 콤마, 세미콜론 등

- 연산자와 구두점은 문맥에 따라 결정됨

- %

```
printf("%d", a); //형식 제어 문자
```

```
a = b % 7; //나머지 연산자
```

- ( )

```
printf("hello"); //연산자
```

```
a = (23 + 2) * 2 //구두점
```

# 예제) 자판기 잔돈 계산 프로그램

```
#include <stdio.h>
int main()
{
    int input, change;                // input은 투입액, change는 잔돈을 저장할 변수
    int w500, w100, w50, w10;       // 각 동전의 개수를 저장할 변수
    printf("돈을 입력하세요 : ");
    scanf("%d", &input);
    change = input - 150;            // 커피값이 150원일 때 잔돈 계산
    w500 = change / 500;            // 500원 동전의 개수
    change = change % 500;          // 아직 지급하지 못한 남은 잔돈
    w100 = change / 100;
    change = change % 100;
    w50 = change / 50;
    change = change % 50;
    w10 = change / 10;

    printf("500원짜리 동전 %d개\n", w500);
    printf("100원짜리 동전 %d개\n", w100);
    printf("50원짜리 동전 %d개\n", w50);
    printf("10원짜리 동전 %d개\n", w10);
    return 0;
}
```