C Language

Lecture 3. 연산자와 명제논리

BAE SEKANG(배세강)

Contents

Operators

♣ 산술 연산자(Arithmetic Operators)

연산자	내용	예시
+	덧셈	4 + 2
-	뺄셈	10 - 5
*	곱셈	2 * 10
/	나누기	5 / 3
%	나머지	5 % 3

Operators

♣ 산술 연산자(Arithmetic Operators)

- 산술 연산자 예시 코드

```
#include <stdio.h>
int main() {
    int a = 7, b = 3;
    printf("더하기: %d₩n", a + b);
    printf("나누기(몫): %d₩n", a / b);
    printf("나머지: %d₩n", a % b);

    return 0;
}
```

Operators

♣ 증감 연산자(Increment/Decrement Operators)

- 전위 표기(++a): 1을 먼저 더한 후 그 줄의 나머지 코드를 실행
- 후위 표기(a++): 현재 값을 먼저 사용 후, 그 줄이 끝나면 1을 더함

```
#include <stdio.h>
int main() {
    int a = 10;
    int b = 10;
    printf("%d₩n", ++a);
    printf("%d₩n", b--);
    printf("%d₩n", b);
    return 0;
}
```

Operators

[Practice]

물건의 가격과 개수를 입력 받을 때, 총 지불할 금액을 출력한다.

[입력 1]
1000 5

[입력 2]
1500 2

[출력 1]
5000원

[출력 2]
3000원

Operators

♣ 비교 연산자(Comparison Operators)

연산자	내용	예시
==	같다	5 == 5
!=	다르다	3 != 1
>	크다	5 > 2
<	작다	5 < 2
>=	크거나 같다	5 >= 2
<=	작거나 같다	5 <= 2

Operators

♣ 비교 연산자(Comparison Operators)

- 1은 참, 0은 거짓을 의미

```
#include <stdio.h>

int main() {
    int a = 10, b = 20;
    printf("a == b : %d₩n", a == b);
    printf("a != b : %d₩n", a != b);
    printf("a < b  : %d₩n", a < b);
    return 0;
}
```

Operators

[Practice]

두 정수 A, B를 순서대로 입력할 때, A보다 B가 더 크면 1을, 아니면 0을 출력한다.

[입력 1]
10 -3

[입력 2]
239 239

[출력 1]
1

[출력 2]
0

Operators

♣ 논리 연산자(Logical Operators)

연산자	내용	예시
||	둘 중 하나만 참이면 참	x > 5 || x < 0
&&	둘 다 참이어야 참	x > 0 && x < 10
!	논리 부정	!(x < 3 && x < 9)

Operators

♣ 논리 연산자(Logical Operators)

- 1은 참, 0은 거짓을 의미

```
#include <stdio.h>

int main() {
    int age = 25;
    int is_twenties = (age >= 20) && (age < 30);
    printf("20대인가요? : %d₩n", is_twenties);
    printf("20대가 아닌가요? : %d₩n", !is_twenties);
    return 0;
}
```

Operators

♣ 비트와 진법

- 비트(Bit): 정보를 표현하는 최소 단위. 즉 0 또는 1

- n진법: 수를 표현할 때, n개의 문자를 활용하는 방법
예) 10진법은 0~9로 모든 수를 표현.

- 10진수(decimal): 0~9를 활용하는 수 체계
- 2진수(binary): 0과 1 두 개의 숫자만 활용하는 수 체계

Operators

♣ 자릿값 표현

- b진법일 때,

$$(d_n d_{n-1} \dots d_1 d_0)_b = d_n \cdot b^n + d_{n-1} \cdot b^{n-1} + \cdots + d_1 \cdot b^1 + d_0 \cdot b^0$$

e.g.) $(135)_{10} = 1 \times 10^2 + 3 \times 10^1 + 5 \times 10^0$

e.g.) $(101)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

Operators

♣ 비트 연산자

- 정수(int)에 대한 비트 단위 연산

연산자	이름	내용	예시
|	OR	두 비트 중 하나가 1이면 1	3 | 1
&	AND	두 비트가 1이면 1	6 & 2
~	NOT	모든 비트를 반전함	~5
^	XOR	두 비트가 다르면 1, 같으면 0	4 ^ 3
<<	Left Shift	왼쪽으로 비트를 shift함	5 << 2
>>	Right Shift	오른쪽으로 비트를 shift함	5 >> 2

Operators

♣ 비트 연산자

```
#include <stdio.h>

int main() {
    unsigned char a = 2;  // 2진수: 0000 0010
    unsigned char b = 3;  // 2진수: 0000 0011
    printf("AND : %u₩n", a & b);  // 0000 0010 (결과 2)
    printf("OR  : %u₩n", a | b);  // 0000 0011 (결과 3)
    printf("Left Shift: %u₩n", a << 1); // 0000 0100 (결과 4)
    return 0;
}
```

Operators

♣ 비트 연산자

- Left shift 연산: x << k는 x의 비트를 왼쪽으로 k칸 밀고, 오른쪽은 0으로 채운다.
  e.g.) 15 << 1의 결과는 30, -3 << 2의 결과는 -12

- Right shift 연산: x >> k는 x의 비트를 오른쪽으로 k칸 밀고, 왼쪽은 부호 비트로 채운다.
  e.g.) 15 >> 1의 결과는 7, -3 >> 2의 결과는 -1

Operators

♣ 2의 보수(2's complement)

- 어떤 숫자의 2의 보수를 구하는 방법

STEP1. 해당 수를 이진수로 나타낸다.
STEP2. 비트를 반전한다.
STEP3. 1을 더한다.

STEP1 ~ STEP3을 거쳐 나온 비트가 처음 수의 2의 보수에 해당한다.

e.g.) 6 -> 0110 -> 1001 -> 1010

Operators

♣ 2의 보수(2's complement)

- 2의 보수로 음수를 표현할 수 있다.
- 6을 2진수로 나타내면 0110이고, 2의 보수는 1010이다. 즉, 1010은 -6을 나타내는 이진수 표현이다.
- 6 + (-6)은 음수인 것처럼, 0110과 1010을 더하면 0000을 얻을 수 있다.

Operators

♣ 대입 연산자

연산자	예시
=	x=5이면, x에 5를 대입한다.
+=	x +=5이면, x+5의 결과를 x에 대입한다.
-=	x -=5이면, x-5의 결과를 x에 대입한다.
…	+=, -=외에도 /=, %=, &=, >>= 등 존재한다.

Operator Precedence

♣ 연산자 우선순위

- 연산자가 여러 개 있을 땐, 우선 순위에 따라 순서대로 계산된다.

```
#include <stdio.h>
int main() {
    int res1 = 10 + 2 * 3;     // 16 (곱하기 먼저)
    int res2 = (10 + 2) * 3;   // 36 (괄호 먼저)
    printf("%d, %d₩n", res1, res2);
    int check = 10 + 5 > 20; // 더하기 먼저 후 비교
    printf("결과: %d₩n", check);
    return 0;
}
```

Operator Precedence

♣ 연산자 우선순위

- 우선순위는 오른쪽 표와 같으며, 위쪽에 있을수록 연산의 우선순위가 높다.

- 오른쪽 표는 C언어의 모든 연산자의 우선순위를 나타내지 않았으므로, 아래 공식 문서를 참고한다.

https://en.cppreference.com/w/c/language/operator_precedence.html

연산자
++, --, (), []
!, &
*, /, %
+, -
<<, >>
<, >
==, !=
&
...

Propositional Logic

♣ 명제와 논리연산

- 명제: 참과 거짓이 확실히 결정되는 문장

  예) 한국의 수도는 서울이다 – True, 2는 홀수이다 – False

- 명제 p, q에 대하여 아래와 같은 논리 연산을 수행할 수 있다.

  부정(NOT): $\neg p$
  논리합(OR): $p \vee q$
  논리곱(AND): $p \wedge q$

Propositional Logic

♣ 진리표(Truth Table)

- 논리식의 참과 거짓을 판정하기 위한 표

예)

p	q	p ∧ q
T	T	T
T	F	F
F	T	F
F	F	F

Propositional Logic

♣ 동치(Equivalence)

- 두 논리식 A, B가 같은 진리값을 가질 때, A와 B는 동치이다.
- $A \equiv B$로 표기한다.

예) $\neg(p \wedge q) \equiv (\neg p \vee \neg q)$ … De Morgan's Law

Propositional Logic

♣ 조건문

- 조건문 $p \rightarrow q$: p이면 q이다.

- $p \rightarrow q$의 진리표

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Propositional Logic

♣ 쌍조건문

- 조건문 $p \leftrightarrow q$: p이면 q이고, q이면 p이다. (필요충분조건)

- $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$와 같다.

p	q	$p \rightarrow q$	$q \rightarrow p$	$p \leftrightarrow q$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

QUIZ

Problem 1.

*와 >> 중 어느 연산자의 우선 순위가 더 높은지 판단한다.

Problem 2.

정수를 입력 받았을 때, 홀수이면 1, 짝수이면 0을 출력하는 프로그램을 작성한다.

Problem 3.

$p \oplus q$ (xor 연산)에 대한 진리표를 작성한다.

THANK YOU!

Any Question?