프로그래밍	**6. 연산자 우선순위, while 반복문과 디버깅 방법**	일자: 2024. . .
학습 목표	☐ 연산자 우선순위와 결합방향 이해 ☐ while 반복문 ☐ 디버깅	하이디:

1. 연산자

종류	기호	의미	문법
복합 대입 연산자	+=	좌항 변수에 우항을 더해 대입한다.	n += 10
	-=	좌항 변수에 우항을 빼 대입한다.	n -= 1
	*=	좌항 변수에 우항을 곱해 대입한다.	n *= 5
	/=	좌항 변수에 우항을 나누어 대입한다.	n /= 2
	%=	좌항 변수에 우항을 나눈 나머지를 대입한다.	n %= m
증감 연산자	++	(전위) 가장 먼저 피연산자의 값을 1 증가시킨다.	++x
	--	(전위) 가장 먼저 피연산자의 값을 1 감소시킨다.	--x
	++	(후위) 가장 나중에 피연산자의 값을 1 증가시킨다.	x++
	--	(후위) 가장 나중에 피연산자의 값을 1 감소시킨다.	x--
기타 연산자	,	두 연산식을 하나의 연산식으로 나타낸다. 둘 이상의 인수를 함수로 전달한다.	int m = 10, n = 2; printf("%d %d", m, n);

2. 연산자의 우선순위와 결합방향

우선순위	연산자	설명	그룹
1	::	scope	Left-to-right
2	() [] . -> ++ -- dynamic_cast static_cast reinterpret_cast	postfix	Left-to-right
3	++ -- ~ ! sizeof new delete	unary (prefix)	Right-to-left
	* &	indirection and reference (pointers)	
	+ -	unary sign operator	
4	(type)	type casting	Right-to-left
5	.* ->*	pointer-to-member	Left-to-right
6	* / %	multiplicative	Left-to-right
7	+ -	additive	Left-to-right
8	<< >>	shift	Left-to-right
9	< > <= >=	relational	Left-to-right
10	== !=	equality	Left-to-right
11	&	bitwise AND	Left-to-right
12	^	bitwise XOR	Left-to-right
13	|	bitwise OR	Left-to-right
14	&&	logical AND	Left-to-right
15	||	logical OR	Left-to-right
16	?:	conditional	Right-to-left
17	= *= /= %= += -= >>= <<= &= ^= |=	assignment	Right-to-left
18	,	comma	Left-to-right

출처 : http://www.cplusplus.com/doc/tutorial/operators/

3. while 반복문

1) while

① while문은 조건문이 참일 때 하단의 실행문을 반복(loop)하는 반복문이다.

2) 조건식

① 조건식은 실행 전 항상 체크되어 조건식이 참인 경우에는 하단 실행문 실행, 거짓인 경우 실행 없이 while문을 탈출한다.

② 조건식에는 일반적으로 비교연산자가 들어가나, 변수, 상수, 함수 등도 들어갈 수 있다.

3) break;

① break 키워드는 while문을 중단하고 빠져나가게 하는 역할로 기능한다.

4) continue;

① continue 키워드는 현재 실행을 중단하고 다음 반복으로 넘어가게 하는 역할로 기능한다.

5) 코드 예시

① 계수기 제어 반복문

```
카운터 초기화;
while (카운터에 관한 조건식) {
    명령문;
    카운터++; // 꼭 1씩 증가할 필요는 없다.
}
```

② 감시값 제어 반복문

```
while (조건식) {
      if (감시값 확인 조건문)
          break;
      명령문;
      감시값 변경문;
}
```

- **<u>무한 반복을 주의하라!</u>**
- while문 내부에 조건식의 결과를 변경하는 명령문이 존재하지 않을 때는 프로그램이 영원히 반복되게 된다. 이렇게 발생되는 무한 반복(infinite loop)은 컴퓨터에 부하를 줄 수 있으니 주의할 필요가 있다.

4. 디버깅

1) 디버깅 실행

- '디버그' > '디버깅 시작(F5)'
- '디버그' > '한 단계씩 코드 실행(F11)' || '프로시저 단위 실행(F10)'

2) 중단점 생성

- '디버그' > '중단점 설정/해제(F9)'
- 코드 옆 세로줄 클릭

3) 디버깅

① 자동

- 현재 실행에 영향을 미치는, 사용되고 있는 변수만의 값을 자동으로 나타내는 창.

② 로컬

- 현재 실행되고 있는 코드에서 선언된 변수들의 값을 모두 나타내는 창.

③ 조사식

- 수동으로 변수, 수식 등을 추가하여 그 결과값을 볼 수 있는 창.

5. 개념 확인 문제

1) 아래 코드들의 출력을 예상하여 주석으로 적어보고, 디버깅하며 답을 확인하라.

```c
#include <stdio.h>

int main(void) {
    int x = 10, y = 20;
    int result1, result2, result3;

    printf("두 수의 덧셈 : %d \n", x + y);

    x += 3;
    y *= 4;

    printf("result : %d %d\n", x, y);

    x = -x; // 단항연산자

    printf("x=-x 연산 이후 x는 %d\n", x);

    x = 10, y = 10; // 콤마연산자

    // 전위/후위 증감연산자의 실행 순서
    printf("선 연산 후 증가 : %d \n", x++);
    printf("다시 한번 출력 : %d \n\n", x);
    printf("선 증가 후 연산 : %d \n", ++y);
    printf("다시 한번 출력 : %d \n", y);

    x = 10;
    y = (x--) + 2;

    printf("x : %d \n", x);
    printf("y : %d \n", y);

    // 관계연산자
    x = 10, y = 10;
    result1 = (x == y);
    result2 = (x <= y);
    result3 = (x > y);

    printf("result1 : %d \n", result1);
    printf("result2 : %d \n", result2);
    printf("result3 : %d \n", result3);

    // 논리연산자
    x = 10, y = 20;
    result1 = (x == 10 && y == 10);
    result2 = (x == 1 || y == 10);
    result3 = !(x == 10);

    printf("result1 : %d \n", result1);
    printf("result2 : %d \n", result2);
    printf("result3 : %d \n", result3);

    return 0;
}
```

- 한 줄짜리 주석은 "//"를 사용하여 적는다.
- 두 줄 이상의 주석을 한 번에 묶으려면, "/* */" 형태로 적어야 한다.

6. 실습

1) LAB6_1 (while, 복합대입연산자)

n을 입력받아 1부터 n까지의 합을 출력하여라.

```
Enter a number: 10
55
```

```
Enter a number: 99
4950
```

2) LAB6_2 (while문, 비교연산자)

n명의 학생의 점수를 받아서 가장 큰 점수를 찾아 출력하라.

```
Enter a student number: 3
Enter a score: 60
Enter a score: 85
Enter a score: 10
The Biggest score is 85
```

```
Enter a student number: 2
Enter a score: 50
Enter a score: 50
The Biggest score is 50
```

3) LAB6_3 (while문)

n명의 학생의 점수를 받아서 총점과 평균을 계산하여 출력하여라. 각 성적과 총점은 int형, 평균은 double형 변수를 사용한다. 출력결과는 소수점 1자리까지 출력하여라.

```
Enter a student number: 4
Enter a score: 30
Enter a score: 50
Enter a score: 100
Enter a score: 50
The total is 230
The average is 57.5
```

```
Enter a student number: 0
The total is 0
The average is 0.0
```

- Runtime Error가 발생한다면 어떤 수를 0으로 나누었는지 확인하라. 'n / 0'은 수학적으로 성립하지 않으므로 컴퓨터도 이를 계산할 수 없다.

4) LAB6_4 (while문 감시값 반복)

정수를 입력받아 다음과 같이 거꾸로 출력하라.

힌트는 아래와 같다.

- 10진수의 일의 자릿수는 10으로 나눈 나머지와 같다.
- 10진수의 일의 자릿수를 제외한 수는 10으로 나눈 몫이다.

```
Enter a number: 1000
0001
```

```
Enter a number: 12345
54321
```

5) LAB6_5 (while문 감시값 반복)

LAB6_4를 수정하여 이진수를 거꾸로 출력하는 것으로 수정하여라.

```
Enter a number: 8
0001
```

```
Enter a number: 18
01001
```

6) LAB6_6 (while문)

입력받은 수의 층만큼 1로 이루어진 수를 출력하라. 다만, **중첩 반복문은 사용하지 않는다**.

```
Enter a number: 4
1
11
111
1111
```

```
Enter a number: 1
1
```