

3. 표준 입력 함수 scanf()의 이해

3. 표준 입력 함수 scanf()의 이해

01 형식

```
int scanf ( const char *format [, argument]... );
```

02 기능

표준 입력 장치인 키보드로 입력된 데이터를 저장

3. 표준 입력 함수 scanf()의 이해

03 헤더파일

stdio.h

3. 표준 입력 함수 scanf()의 이해

04 사용 형식

정수를 입력 받는 경우	문자를 입력 받을 경우
<pre>int data; scanf("%d", &data);</pre>	<pre>char data; scanf("%c", &data);</pre>
실수를 입력 받는 경우	문자열을 입력 받는 경우
<pre>float data1; double data2; scanf("%f", &data1); scanf("%lf", &data2);</pre>	<pre>char data[10]; scanf("%s", data);</pre>

3. 표준 입력 함수 scanf()의 이해

05 예 : scanf()의 이용 - 정수 데이터의 입력

```
#include <stdio.h>

void main()
{
    int n1, n2, sum=0;

    printf("정수 2개를 입력 :");
    scanf("%d%d", &n1, &n2);

    sum = n1 + n2;
    printf("두 수의 합은 %d 입니다. \n", sum);
}
```

3. 표준 입력 함수 scanf()의 이해

06 예 : scanf()의 이용 - 문자열 데이터의 입력

```
#include <stdio.h>

void main()
{
    char name1[30], name2[30];

    printf("이름을 입력 : "); scanf("%s", name1);
    printf("이름을 입력 : "); scanf("%s", name2);

    printf("나의 이름은 %s 입니다. \n", name1);
    printf("나의 이름은 %s 입니다. \n", name2);
}
```

3. 표준 입력 함수 scanf()의 이해

07 예 : scanf()의 이용 - 잘못된 이용의 예

```
#include <stdio.h>

void main()
{
    int su1, su2, sum = 0;

    printf("정수 두 개 입력 : ");
    scanf("%d%d\\n", &su1, &su2);

    sum = su1 + su2;

    printf("두 정수의 합은 %d 입니다. \\n", sum);
}
```

3. 표준 입력 함수 scanf()의 이해

07 예 : scanf()의 이용 - 잘못된 이용의 예

```
#include <stdio.h>

void main()
{
    int su1, su2, sum = 0;

    printf("정수 두 개 입력 : ");
    scanf("%d%d ", &su1, &su2);

    sum = su1 + su2;

    printf("두 정수의 합은 %d 입니다. \n", sum);
}
```


**3. 표준 입력 함수 scanf()의 이해에 대한
강의가 끝났습니다.**

C언어

LESSON 03.

C언어의 연산자

1. 연산자의 이해

1. 연산자의 이해

01 산술연산자

두 개 피연산자 간의 산술 연산을 처리

산술 연산자	의 미	사용 예
+	두수의 합	$a + b$
-	두수의 차	$a - b$
*	두수의 곱	$a * b$
/	나누기 몫	a / b
%	나누기 나머지	$a \% b$

1. 연산자의 이해

01 산술연산자

I 모드(%) 연산자의 사용

짝, 홀수 및 배수의 구분, 범위 설정 등

1. 연산자의 이해

02 예 : 산술연산자

```
#include <stdio.h>

void main()
{
    int s1 = 20, s2 = 3;

    printf("%d + %d = %d \n", s1, s2, s1 + s2);
    printf("%d - %d = %d \n", s1, s2, s1 - s2);
    printf("%d * %d = %d \n", s1, s2, s1 * s2);
    printf("%d / %d = %d \n", s1, s2, s1 / s2);
    printf("%d %% %d = %d \n", s1, s2, s1 % s2);
}
```

1. 연산자의 이해

03 관계연산자

두 개 피연산자 간의 대소 관계의 비교를 처리

두 개 피연산자 간의 산술 연산을 처리		
산술 연산자	의 미	사용 예
>	.. 보다 크다	$a > 10$
<	.. 보다 작다	$a < 10$
<=	.. 보다 작거나 같다.	$a <= 10$
>=	.. 보다 크거나 같다.	$a >= 10$
==	.. 의 값과 같다.	$a == 10$
!=	.. 의 값과 같지 않다.	$a != 10$

1. 연산자의 이해

04 예 : 관계연산자

```
#include <stdio.h>

void main()
{
    int s1 = 20, s2 = 3;

    printf("#.1 결과 : %d \Wn", s1 <= s2);
    printf("#.2 결과 : %d \Wn", s1 >= s2);
    printf("#.3 결과 : %d \Wn", s1 == s2);
    printf("#.4 결과 : %d \Wn", s1 != s2);
}
```


1. 연산자의 이해

05 대입연산자

우측에 수행한 결과의 값을 좌측 지정된 변수에 저장

복합대입연산자 : 대입연산자와 다른 연산자의 복합 사용

구 분	연산자	의 미	사 용 예
대입연산자	=	b의 값을 변수 a에 저장	a = b
복합대입 연산자	+=	a = a + b	a += b
	-=	a = a - b	a -= b
	*=	a = a * b	a *= b
	/=	a = a / b	a /= b
	%=	a = a % b	a %= b

1. 연산자의 이해

06 예 : 대입연산자

```
#include <stdio.h>

void main()
{
    int s1, s2;
    s1 = s2 = 5;

    printf("#.1 결과 : %d \n", s1 += 1);
    printf("#.2 결과 : %d \n", s1 -= 1);
    printf("#.3 결과 : %d \n", s1 *= s2);
    printf("#.4 결과 : %d \n", s1 /= s2);
    printf("#.5 결과 : %d \n", s1 %= s2);
}
```

1. 연산자의 이해

07 논리연산자

참과 거짓을 판별		
논리 연산자	의 미	사용 예
	논리합(OR)	if(a>5 a < -5) ..
&&	논리곱(AND)	if(a>5 && a < -5) ..
!	부정(NOT)	if(!(a%2 == 1)) ..

1. 연산자의 이해

08 예 : 논리연산자

```
#include <stdio.h>

void main()
{
    int num;

    printf("국어 점수 입력(0~100점) : ");
    scanf("%d", &num);

    if( !( (num)>=0) && (num<=100) ) )
        printf("잘못된 입력입니다. \n");
    else
        printf("입력된 점수 : %d \n", num);
}
```

1. 연산자의 이해

09 증감연산자

피연산자의 값을 1 증가 혹은 감소하여 다시 저장		
논리 연산자	의 미	사용 예
++	$a = a + 1$	$++a / a++$
--	$a = a - 1$	$--a / a--$

1. 연산자의 이해

09 증감연산자

I 전치와 후치에 따른 연산자 비교

전치

$++a$ 또는 $--a$ 로 표기하며
지정된 기능을 처리

후치

$a++$ 또는 $a--$ 로 표기하며
지정된 기능을 처리

전치와 후치는 기본적인 기능이 동일하며,
다른 명령문과 병행되는 경우, 우선순위가 달라짐

1. 연산자의 이해

10 예 : 증감연산자 - 독립적인 연산 기능

```
#include <stdio.h>

void main()
{
    int s1, s2;

    s1 = 5;          ++s1;
    s2 = 5;          s2++;
    printf("#.1 결과 : s1 = %d, s2 = %d ", s1, s2);

    s1 = 5;          --s1;
    s2 = 5;          s2--;
    printf("#.2 결과 : s1 = %d, s2 = %d ", s1, s2);
}
```

1. 연산자의 이해

11 예 : 증감연산자 - 다른 명령문과의 병행

```
#include <stdio.h>

void main()
{
    int s1, s2, s3, s4;

    s1 = 5;          s2= ++s1;
    printf("#.1 결과 : s1 = %d, s2 = %d ", s1, s2);

    s3 = 5;          s4 = s3++;
    printf("#.2 결과 : s3 = %d, s4 = %d ", s3, s4);
}
```


1. 연산자의 이해

12 조건연산자

조건식의 결과에 따라 명령 중 한 개를 선택하여 실행

- ✓ 결과가 참인 경우 : 콜론의 앞 명령을 실행
- ✓ 결과가 거짓인 경우 : 콜론의 뒤 명령을 실행

형식

조건식 ? 참일때의 명령문 : 거짓일 때의 명령문

1. 연산자의 이해

13 예 : 조건연산자

```
#include <stdio.h>

void main()
{
    int s;

    printf("정수 입력 : ");
    scanf("%d", &s);

    s % 2 == 0 ?
        printf("%d는 짝수입니다. \n", s)
        : printf("%d는 홀수입니다. \n", s);
}
```

1. 연산자의 이해

14 비트연산자

2진수 데이터에 대한 각 비트의 논리/이동 연산을 처리	
비트 연산자	의 미
	비트 단위 논리합(OR)
&	비트 단위 논리곱(AND)
^	비트 단위 배타적 논리합(XOR)
~	비트 부정(NOT)
<<	비트 좌측 이동(Left Shift)
>>	비트 우측 이동(Right Shift)

1. 연산자의 이해

15 예 : 비트연산자 - 비트 논리합

```
#include <stdio.h>

void main()
{
    int s1 = 12, s2 = 7, res;

    res = s1 | s2;
    printf("s1과 s2의 비트 논리합 결과 : %d \n", res);
}
```

	1	1	0	0
	0	1	1	1
	1	1	1	1

1. 연산자의 이해

16 예 : 비트연산자 - 비트 부정

```
#include <stdio.h>

void main()
{
    int s1 = 15, res;
    res = ~s1;
    printf("s1의 비트 부정 결과 : %d \n", res);
}
```

~	0000 0000	0000 0000	0000 0000	0000 1111
	1111 1111	1111 1111	1111 1111	1111 0000
	1000 0000	0000 0000	0000 0000	0000 1111
	1000 0000	0000 0000	0000 0000	0001 0000

1. 연산자의 이해

17 기타연산자

1 sizeof() 연산자

- ✓ 피연산자의 크기를 바이트 단위로 표기

2 콤마 연산자

- ✓ 명령 단위의 기준이 됨
- ✓ 연산식이 여러 개인 경우 좌측부터 실행한 후,
최종 연산 결과는 우측식으로 인식함

1. 연산자의 이해

18 예 : sizeof() 연산자


```
#include <stdio.h>

void main()
{
    int a= 0, b=0, c=0;
    printf("변수 a의 공간 크기 : %d \n", sizeof(a) );
    printf("정수의 공간 크기 : %d \n", sizeof(10) );
    printf("int 자료형 공간 크기 : %d \n", sizeof(int) );

    a = (b=3, c=5, b+c);
    printf("a= %d, b= %d, c= %d \n", a, b, c);
}
```

1. 연산자의 이해

19 연산자 우선순위

연산자	연산순서	우선순위	비고
() , [] , -> , . (점)	좌에서 우		
sizeof, (type), &, *, -(단항), +(단항), --, ++, ~, !	좌에서 우		단항
*(곱셈), / , %, +, /	좌에서 우		산술
<<, >>	좌에서 우		비트
<, <=, >, >=, ==, !=	좌에서 우		비교
&, ^,	좌에서 우		비트
&&,	좌에서 우		논리
? :	우에서 좌		상항
%=, /=, *=, -=, +=, =	좌에서 우		대입
,	좌에서 우		coma

**1. 연산자의 이해에 대한
강의가 끝났습니다.**