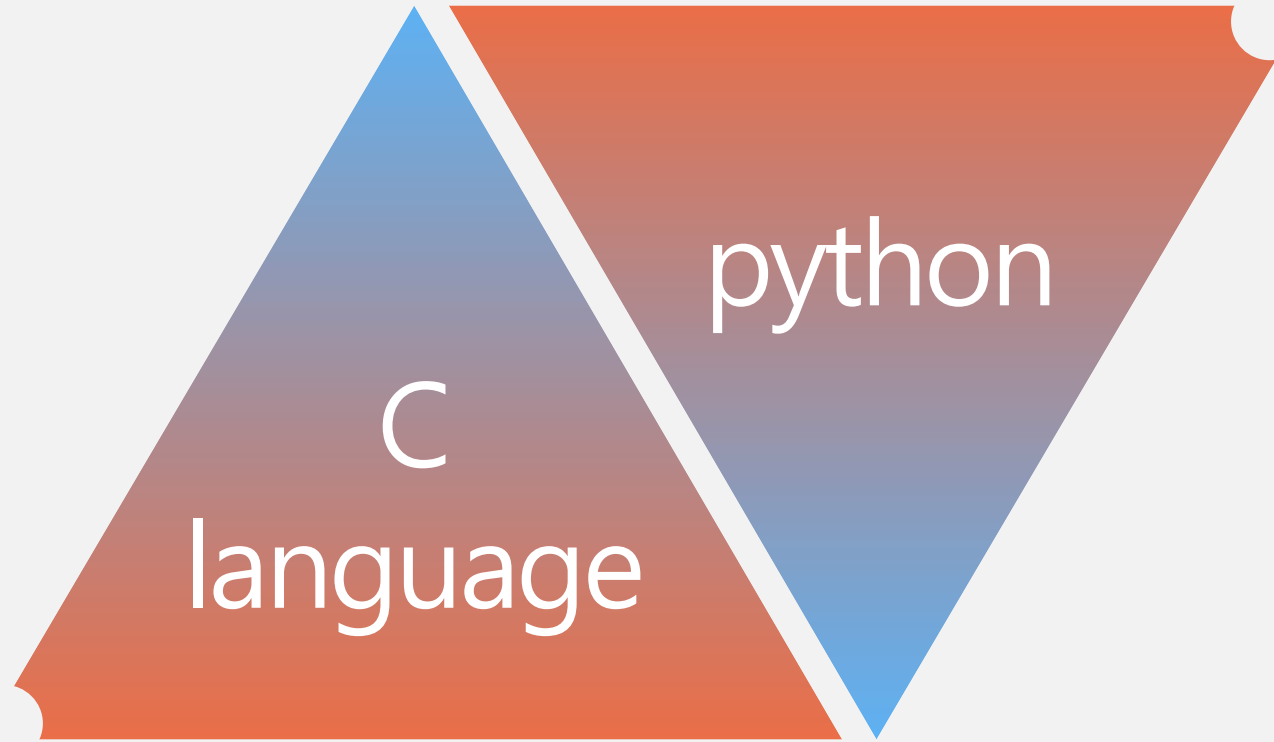




C 언어  
멘토링 수업 ●  
(3주차)

- 1 지난 주 수업 복습
  - 2 printf 와 scanf
  - 3 C Hello World!
  - 4 리눅스
-

# 1. 지난주 수업 복습



# 이주호



3. 숫자로 표현할 수 있다면 우



3L. Index.pdf

칸, list,  
len(), range(),  
adress

if문,  
while문,  
for문

# 임을규



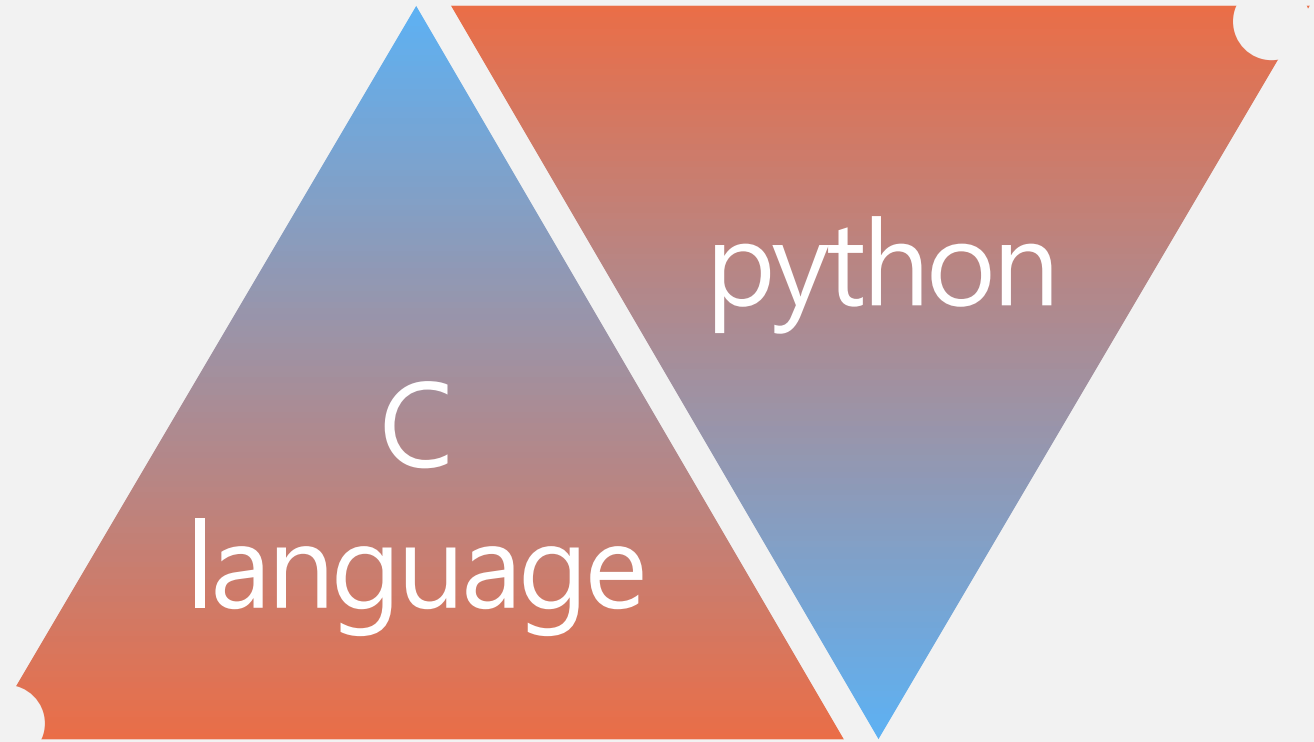
03. Jokes.pdf



소입설3-2

모듈

## 2. printf 와 scanf



## 변수 선언 & 값 저장 동시에도 가능

```
#include <stdio.h>

int main(void) {
    int num;
    num=3;
    int variable=10;

    return 0;
}
```

**printf는 문자열을 출력하는 함수이다.**

**printf는 특수 문자 출력이 가능하다.**

특수 문자	의 미
\a	경고음 소리 발생
\b	백스페이스(backspace)
\f	폼 피드(form feed)
\n	개행
\r	캐리지 리턴(carriage return)
\t	수평 탭
\v	수직 탭
\\	백슬래시(\)
\'	작은 따옴표
\"	큰 따옴표

## 특수 문자가 필요한 이유

### 잘못된 문자열 출력

```
#include <stdio.h>

int main(void)
{
    printf("앞집 강아지가 말했다. "멍! 멍!" 정말 귀엽다.");
    return 0;
}
```

printf( "앞집 강아지가 말했다. " 멍! 멍! " 정말 귀엽다.");

문자열 1      정체불명      문자열 2



## printf 함수는 서식 지정이 가능하다.

printf의 f는 "formatted"를 의미한다.

서식 지정 : 출력의 형태를 지정한다는 의미  
(ex : 문자열 안에 숫자 삽입)

서식 지정의 예

```
#include <stdio.h>

int main(void)
{
    int age=12;
    printf("10진수로 %d살이고 16진수로 %x살 입니다.", age, age);
    return 0;
}
```

## 서식 문자의 종류와 그 의미

서식 문자	출력 형태
%c	단일 문자
%d	부호 있는 10진 정수
%i	부호 있는 10진 정수, %d와 같음
%f	부호 있는 10진 실수
%s	문자열
%o	부호 없는 8진 정수
%u	부호 없는 10진 정수
%x	부호 없는 16진 정수, 소문자 사용
%X	부호 없는 16진 정수, 대문자 사용
%e	e 표기법에 의한 실수
%E	E 표기법에 의한 실수
%g	값에 따라서 %f, %e 둘 중 하나를 선택
%G	값에 따라서 %f, %G 둘 중 하나를 선택
%%	% 기호 출력

**%c, %d, %f, %s**

가장 많이 쓰이는 서식 문자들

**%o, %u, %x, %X**

부호 없는 정수형 출력

**%e, %E**

'부동소수점 표현 방식'에 의한 출력

$3.1245e+2 \rightarrow 3.1245 \times 10^{+2}$

$2.45e-4 \rightarrow 2.45 \times 10^{-4}$

---

## 필드 폭을 지정하여 멋진 출력을!

서식 문자를 이용해서 출력의 폭 지정 가능

서식 문자	출력의 형태
%8d	필드 폭을 8칸 확보하고 오른쪽 정렬해서 출력하라.
%-8d	필드 폭을 8칸 확보하고 왼쪽 정렬해서 출력하라.
%+8d	필드 폭을 8칸 확보하고 오른쪽 정렬한 상태에서 양수는 +, 음수는 -를 붙여서 출력하라.

## scanf 함수의 입력 형태 정의

데이터를 입력받는 형태를 지정할 수 있다.

즉 입력 서식을 지정하는 것이다.

예 : "%d %o %x"

## 실수 입력에 있어서 주의사항

정밀도 생각!

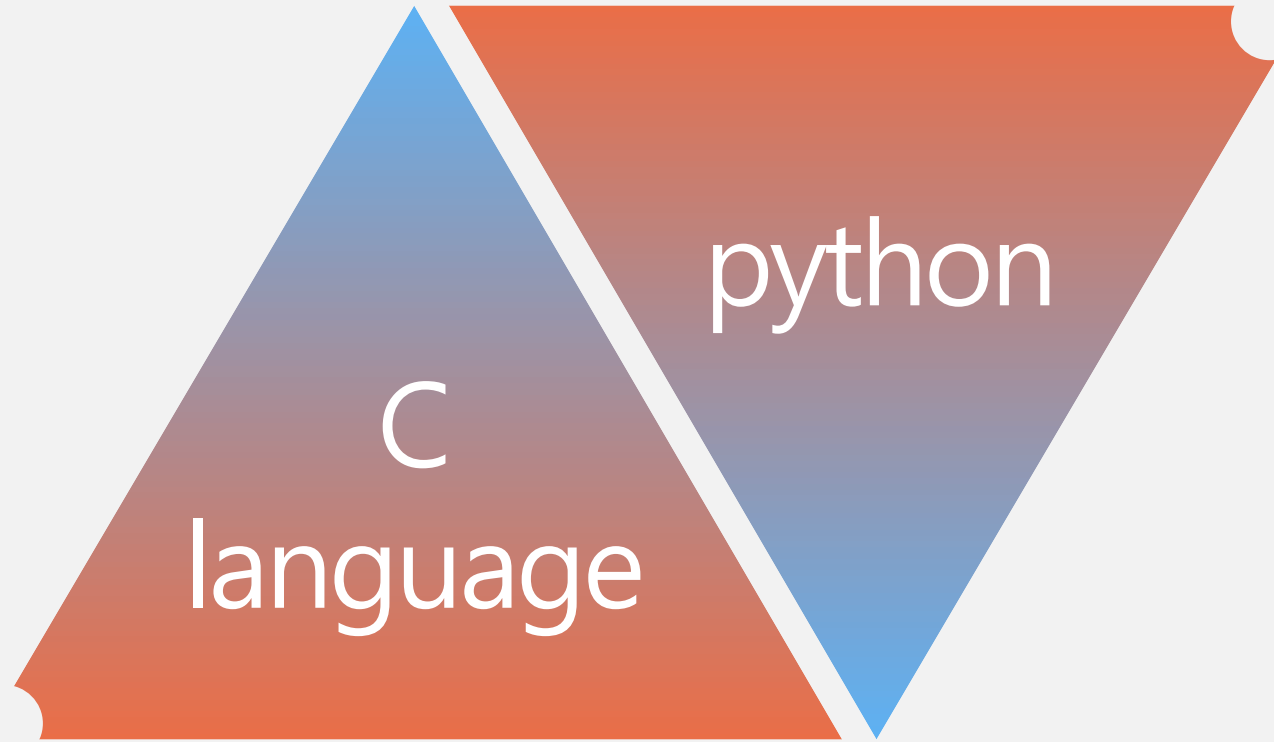
소수 6자리 이하의 실수 입력 시 %f 사용

소수 6자리를 넘는 실수 입력 시 %e 사용

단! double형 변수를 사용하는 경우에는 서식 문자 %le를 사용

---

### 3. 연산자, 제어문, 함수



## 연산자, 제어문, 함수

C언어에서의 연산자, 제어문, 함수에 대해 배워보자.

연산자

제어문

함수

## 연산자란 무엇인가?

연산을 요구할 때 사용되는 기호

ex : +, -, \*, /

```
int main(void)
{
    3+4;    // 덧셈 결과를 저장할 필요가 있다.
    return 0;
}
```

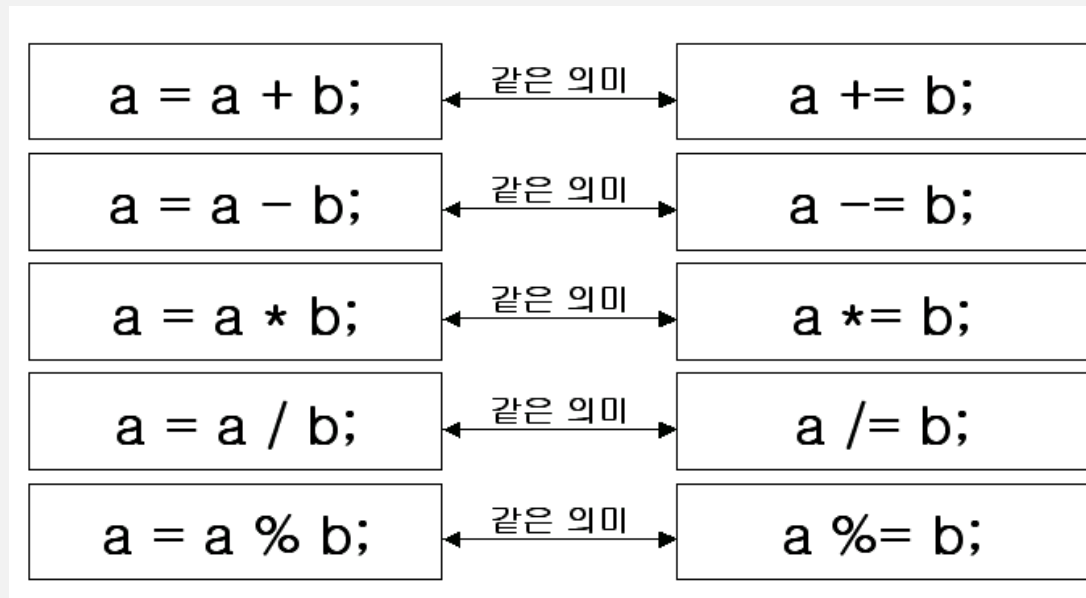


## 대입 연산자와 산술 연산자

연산자	연산의 예	의미	결합성
=	$a=20$	대입	←
+	$a=4+3$	덧셈	→
-	$a=4-3$	뺄셈	→
*	$a=4*3$	곱셈	→
/	$a=4/3$	나눗셈	→
%	$a=4\%3$	나머지	→

## 기타 대입 연산자

대입 연산자와 산술 연산자가 합해져서 다양한 형태의 대입 연산자 정의



## 부호 연산으로서 +, - 연산자

단항 연산자로서 +, -

## 증가 감소 연산자

연산자	연산의 예	의미	결합성
++a	printf("%d", ++a)	선 증가, 후 연산	←
a++	printf("%d", a++)	선 연산, 후 증가	←
--b	printf("%d", --a)	선 감소, 후 연산	←
b--	printf("%d", a--)	선 연산, 후 감소	←

## 관계 연산자(비교 연산자)

두 피연산자의 관계(크다, 작다 혹은 같다)를 따지는 연산자  
true(논리적 참, 1), false(논리적 거짓, 0) 반환

연산자	연산의 예	의미	결합성
<	$a < b$	a가 b보다 작은가	→
>	$a > b$	a가 b보다 큰가	→
==	$a == b$	a와 b가 같은가	→
!=	$a != b$	a와 b가 같지 않은가	→
<=	$a \leq b$	a가 b보다 작거나 같은가	→
>=	$a \geq b$	a가 b보다 크거나 같은가	→

## 논리 연산자

and, or, not을 표현하는 연산자

true(1), false(0) 반환

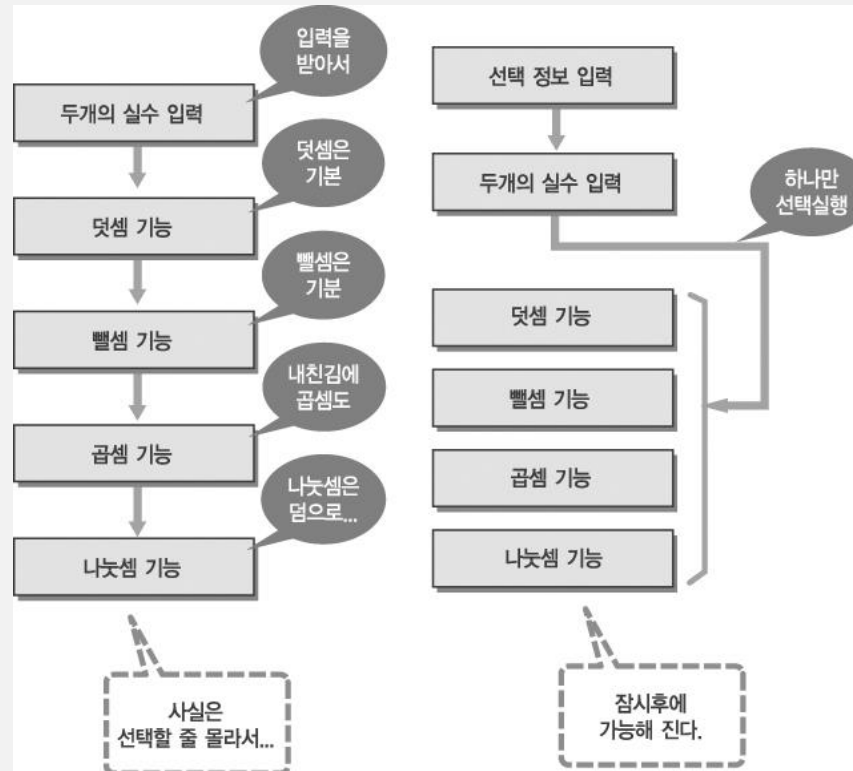
연산자	연산의 예	의미	결합성
&&	a&&b	true면 true 리턴	→
	a  b	하나라도 true면 true 리턴	→
!	!a	true면 false를, false면 true 리턴	→

## 연산자, 제어문, 함수

C언어에서의 연산자, 제어문, 함수에 대해 배워보자.



## 상황에 따른 프로그램의 유연성 부여



## if문에 의한 조건적 실행

조건이 만족되는 경우에 한해서 실행

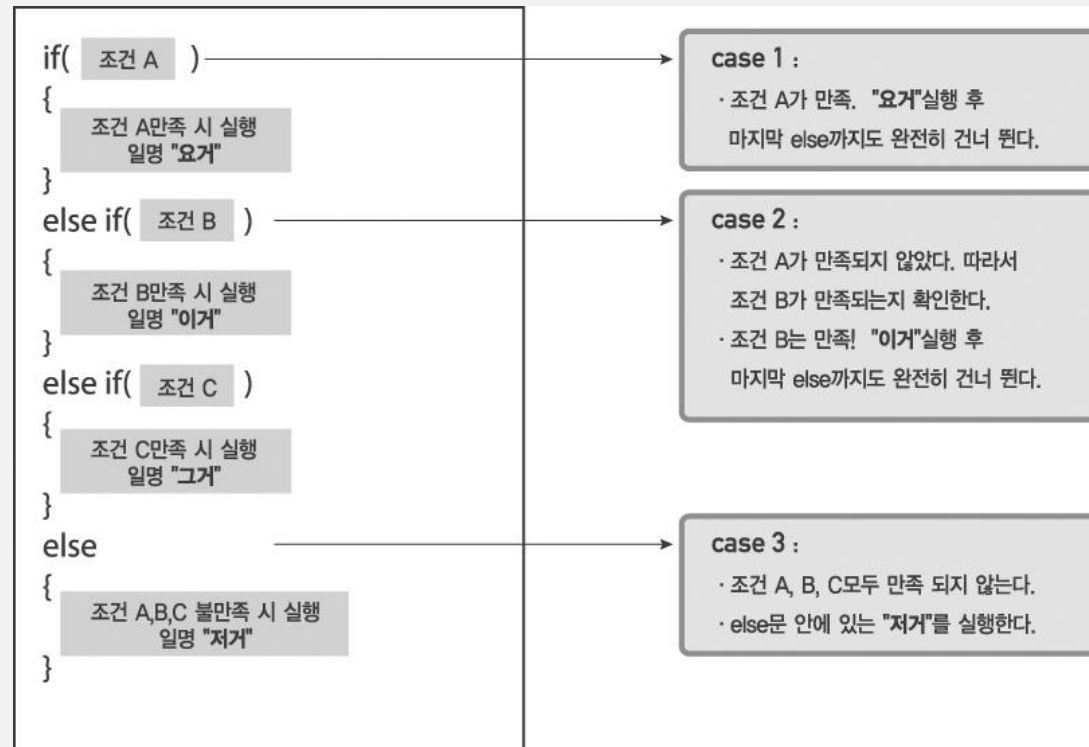




## if~else 에 대해서



## if, else if, else에 대해서



## 조건 연산자(삼항 연산자)

if~else문을 간결히 표현하는데 사용될 수 있다.

조건 ? A : B

조건이 true인 경우 A를 반환  
조건이 false인 경우 B를 반환

$X = (y < 0) ? 10 : 20;$

"y<0"이 true면 10이 반환되어 x에 대입  
"y<0" 이 false면 20이 반환되어 x에 대입

$X = (y > 0) ? a * b : a / b;$

"y>0"이 true면 a\*b이 연산결과 x에 대입  
"y>0" 이 false면 a/b이 연산결과 x에 대입

## 반복문의 기능

특정 영역을 특정 조건이 만족하는 동안에  
반복 실행하기 위한 문장

## 세 가지 형태의 반복문

while문에 의한 반복

do ~ while문에 의한 반복 ( 나중에 )

for문에 의한 반복 ( 나중에 )

---

## while문의 기본 원리와 의미

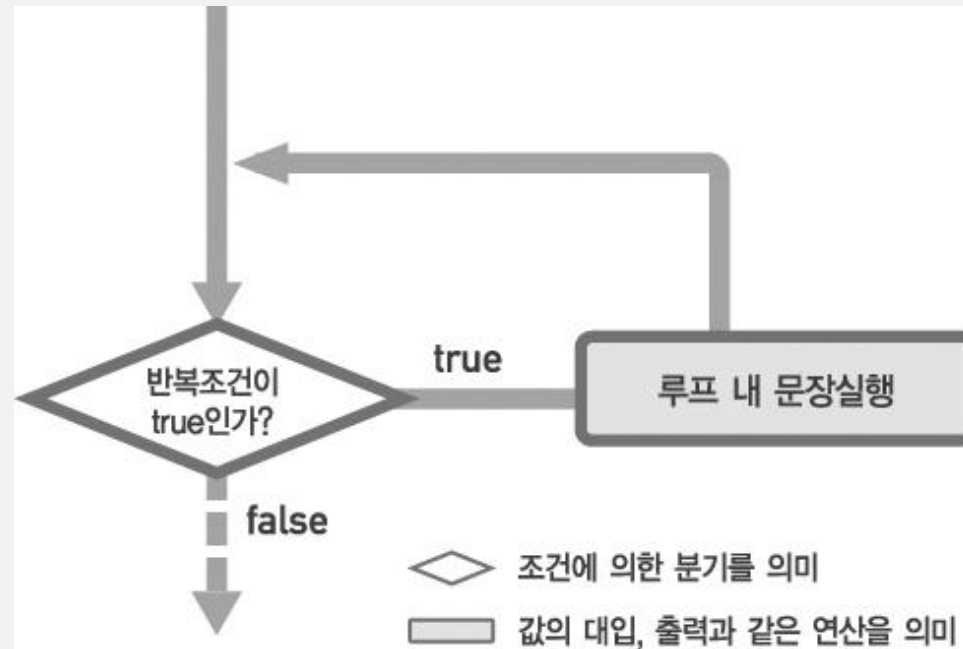
```
while( 반복 조건 )  
{  
    반복 내용  
}
```

“반복의 조건” 이 만족되는 동안  
“반복 내용” 을 반복 실행하라.

```
while( i<10 )  
{  
    printf("Hello World! \n");  
    i++;  
}
```

“i<10” 이 만족되는 동안  
“printf()와 i++” 을 반복 실행하라.

## while문의 순서도



이제 그만 **break!**(탈출)

반복문을 빠져 나올 때 사용

다음으로 넘어가자 **continue!**(생략)

다음 번 반복으로 넘어갈 때 사용

break → while, switch  
continue → while

```
while( 1 )  
{  
    ...  
    if(x<0)  
        continue;  
    ...  
}
```

나머지 루프 건너 뛴  
다시 루프 반복

printf("end of while");

```
while( 1 )  
{  
    ...  
    if(x<0)  
        break;  
    ...  
}
```

반복문 탈출

printf("end of while");

**do while**

**for**

**switch**

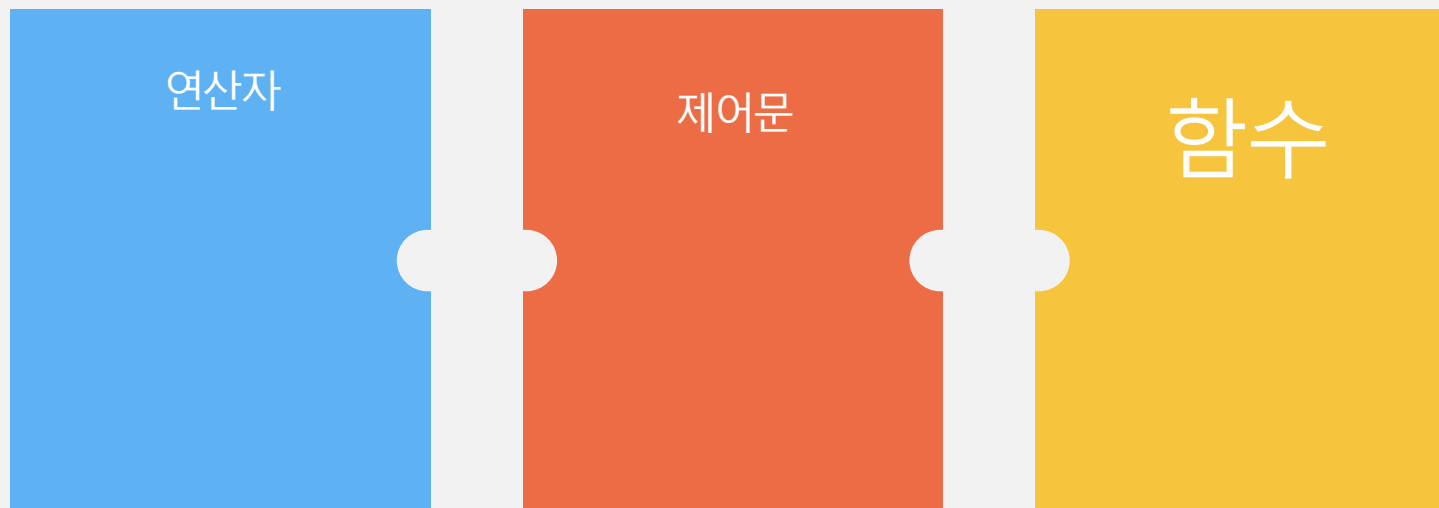
**goto**

---



## 연산자, 제어문, 함수

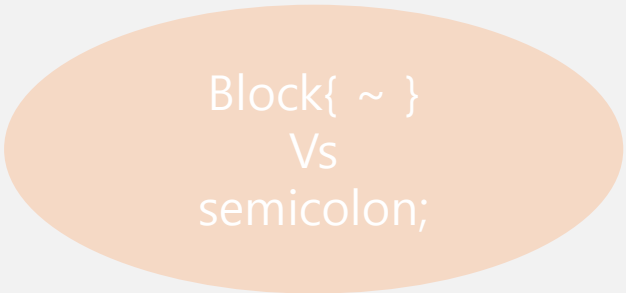
C언어에서의 연산자, 제어문, 함수에 대해 배워보자.



## 함수의 형태



## 함수의 형태 (정의)



Block{ ~ }  
Vs  
semicolon;

```
[data type of return value] [name] ([argument,parameter]) {  
  
    ....  
  
    ....  
  
    return [return value];  
  
}
```

## 함수(정의, 선언, 호출)

### 정의 (함수를 직접 만들 때)

```
int FirstFunction(int num)
{
    printf("hello world.");
    return num+3;
}
```

## 함수(정의, 선언, 호출)

선언 (함수가 메인 함수 아래에 정의되어 있을 때)

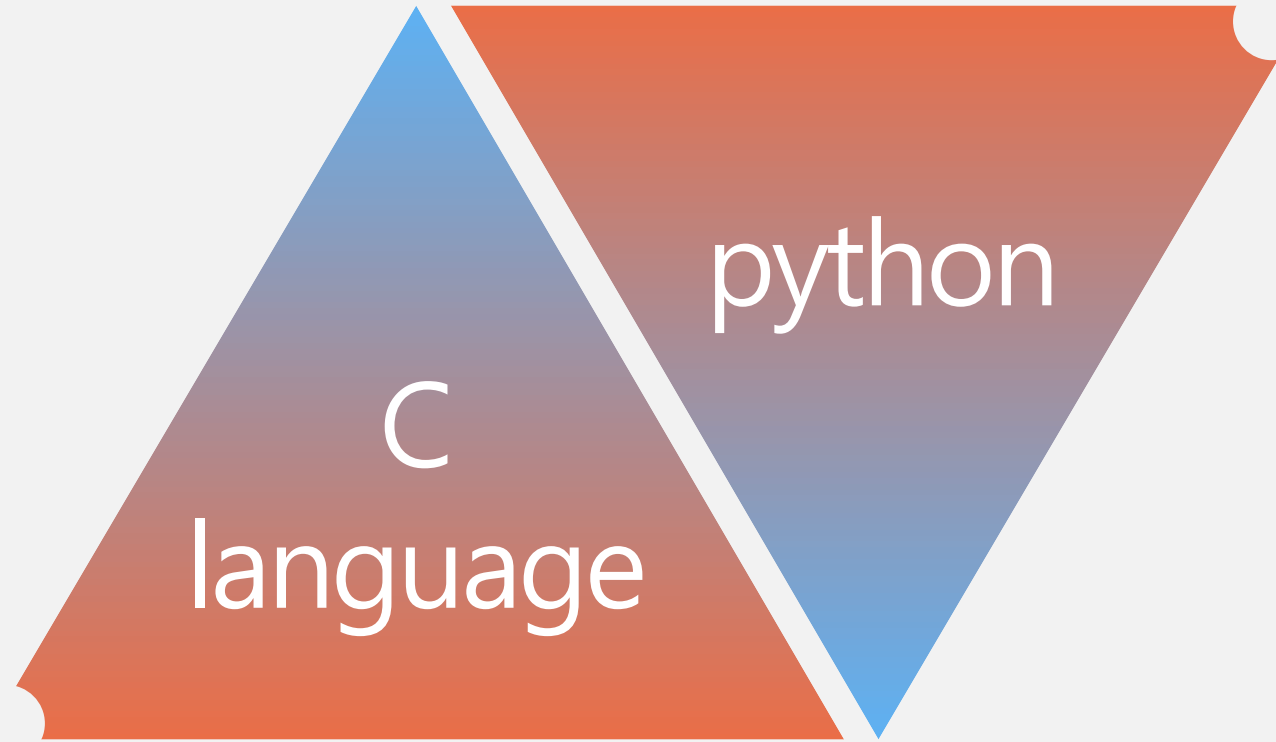
```
int FirstFunction(int num);  
int main(void) {  
    ...  
    ...  
}  
int FirstFunction(int num){  
    ...  
}
```

## 함수(정의, 선언, 호출)

### 호출 (함수를 실제 사용할 때)

```
int FirstFunction(int num);  
int main(void) {  
    int k=3;  
    k=FirstFunction(30);  
}  
int FirstFunction(int num){  
    ...  
}
```

## 4. 리눅스





소입설1-1.



소입설1-2.



---

복습

---