

# 프로그래밍 언어 응용

chapter04

## 제어문

제공된 자료는 훈련생의 수업을 돕기 위한 것으로, 타인과 공유하시면 안됩니다.

# Contents

part.1

조건문

part.2

반복문

part.3

디버깅

# 조건문

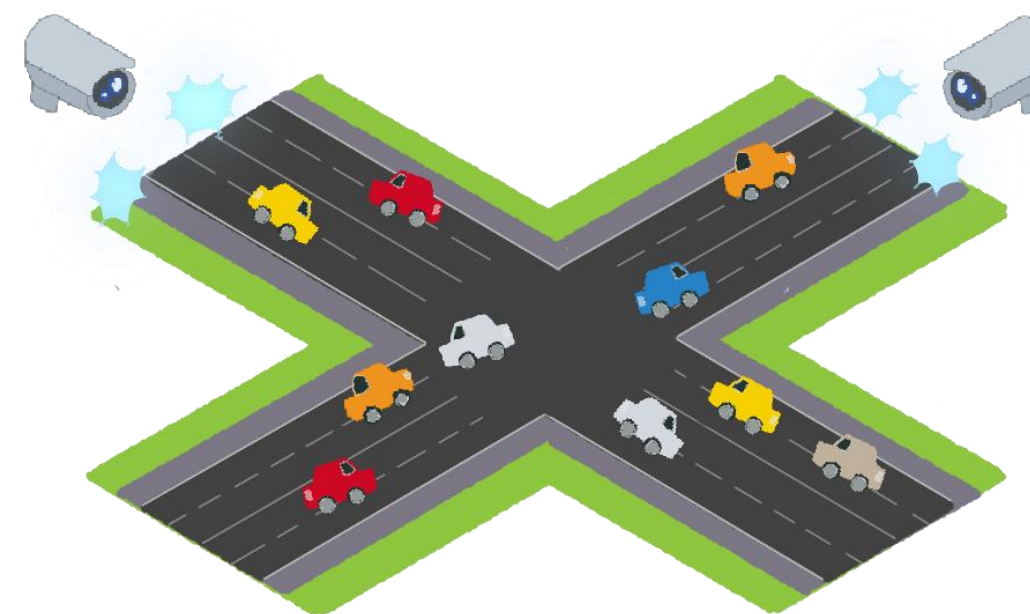
## 조건문이란 무엇일까?

### 조건문이란?

- 주어진 조건에 맞으면 어떤 작업을 수행하도록 하는 문법이다.



학생의 나이가 8살이상이면 학교에 다닌다  
(조건)

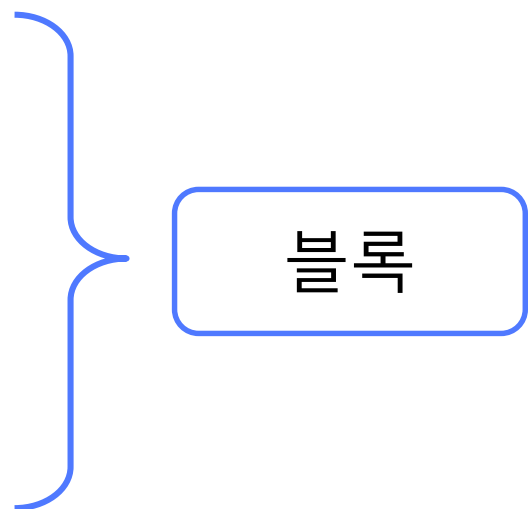


자동차가 시속이 80km 이상이면 과속카메라가 사진을 찍는다  
(조건)

### 조건문의 특징

- 조건문은 {} 중괄호 블록을 함께 사용한다.
- 블록은 조건문의 결과에 따라 수행되는 코드의 범위이다.
- 주어진 조건이 참이면 블록 안에 있는 코드를 수행하고, 아니면 수행하지 않는다.

```
if (조건식) {  
    수행문;  
}
```



블록

“나이가 8살 이상이면 학교에 다닌다”

8 9 10 11 12 ...

```
if (age >= 8) {  
    “학교에 다닌다”  
}
```

“등수가 1등이면 금메달을 수여한다”

```
switch (rank){  
    case 1 : “금메달을 수여한다”  
}
```

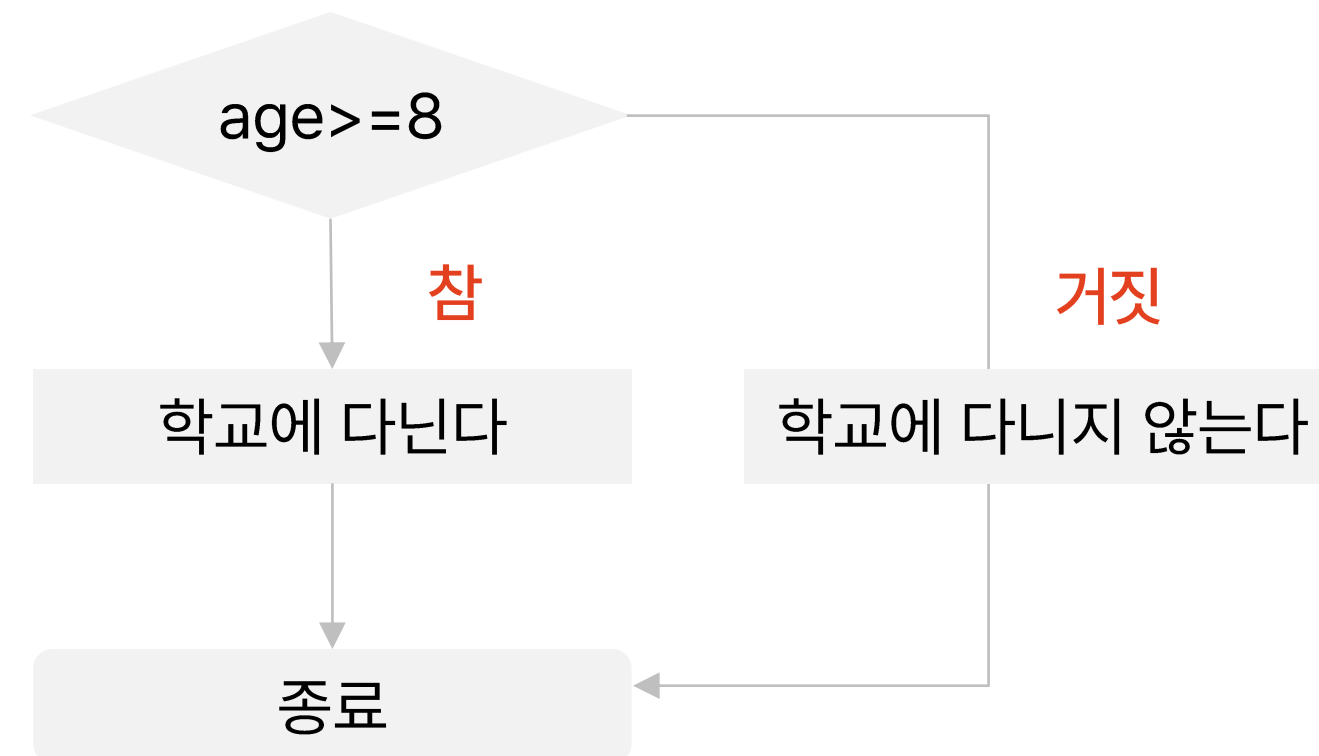
조건문	내용
if문	조건이 포괄적이거나 복잡한 경우에 사용한다.
switch문	조건이 하나의 값을 기준으로 비교할 때 사용한다.

If문은 조건식의 결과에 따라 실행된다.

주어진 조건을 만족하면 첫번째 블록을 실행하고, 만족하지 않으면 두번째 블록을 실행한다.

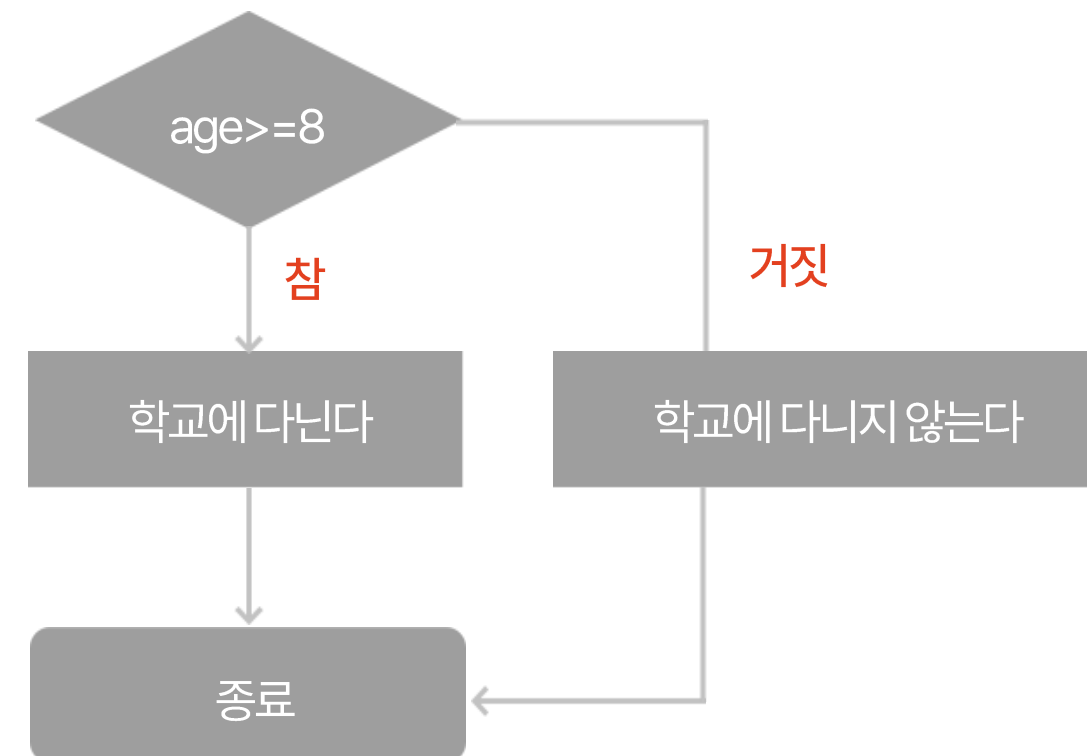
\*조건: 학생의 나이가 8살 이상이면

```
if (age >= 8) {  
    "학교에 다닌다"  
} else {  
    "학교에 다니지 않는다"  
}
```



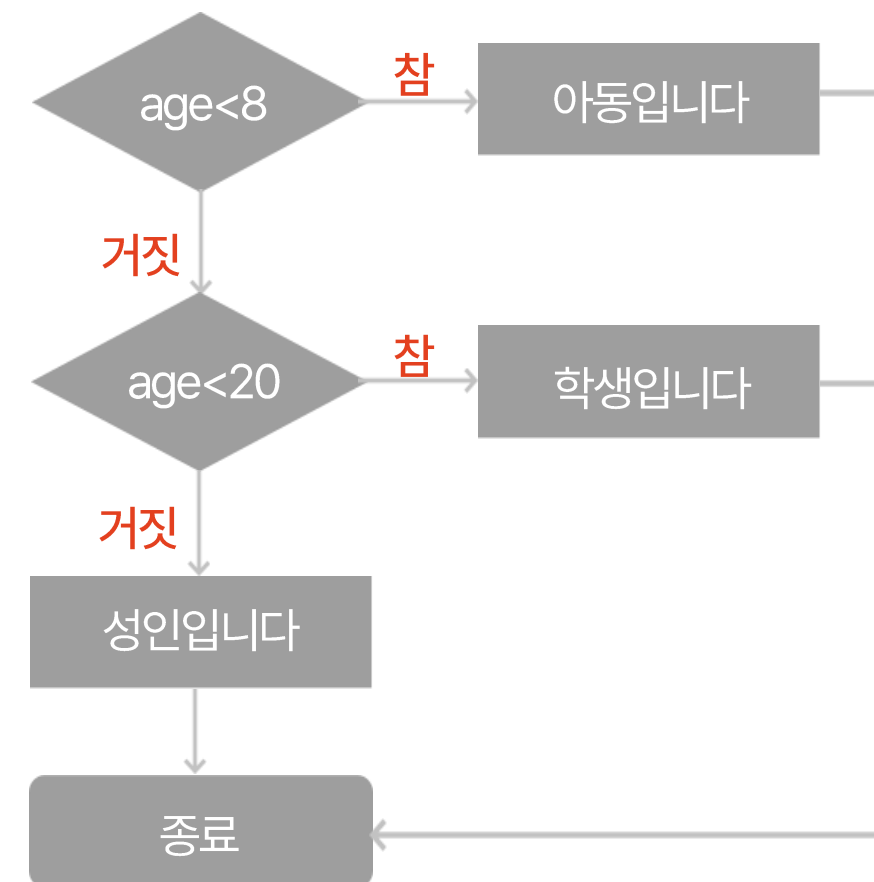
조건식을 만족하는 경우와 만족하지 않는 경우를 모두 나타낼 때는 if~else문을 사용한다.  
주어진 조건을 만족하면 if블록 안에 코드를 수행하고, 만족하지 않으면 else블록 안에 있는 코드를 수행한다.  
예시에서 만약 age 값이 7이라면 "학교에 다니지 않는다"라는 문장이 출력된다.

```
if (age >= 8) {  
    print(학교에 다닌다)  
} else {  
    print(학교에 다니지 않는다)  
}
```



하나의 상황에 조건문이 여러 개인 경우에는 elseif문을 사용한다.  
어떤 조건을 만족하면 블록을 수행하고, 다른 조건은 비교하지 않고 조건식을 종료한다.  
모든 조건을 만족하지 못하면 else블록을 실행한다.  
elseif문을 사용하면 나이에 따라 다른 문장을 출력할 수 있다.

```
if (age < 8) {  
    print(미취학 아동입니다)  
} else if (age < 20) {  
    print(학생입니다)  
} else {  
    print(성인입니다)  
}
```





**switch문이란?**

switch문은 변수의 값이 정확히 일치하는지 비교한 후 실행된다.

변수와 값이 일치하는 case문으로 이동하여 코드를 수행하고, break문을 만나면 빠져나온다.

```
switch (rank){
```

등수

```
case 1 :
```

```
    "금메달 입니다"
```

```
case 2 :
```

```
    "은메달 입니다"
```

```
case 3 :
```

```
    "동메달 입니다"
```

```
default :
```

```
    "메달이 없습니다"
```



**switch문의 특징**

- case : ~break 까지가 조건에 해당하는 하나의 블록이다.
- switch문은 블록 {}을 생략한다.
- 변수와 값이 일치하는 case문으로 이동하여 코드를 수행하고, break문을 만나면 빠져나온다.
- 변수의 값과 일치하는 case문이 없다면 default문이 수행된다.

```
switch (변수) {  
  case 값1:  
    ...  
    break;  
  case 값2:  
    ...  
    break;  
  default :  
    ...  
}
```



case 끝에 break가 있는 이유는 switch문을 빠져나가기 위함이다.

조건에 맞는 수행문을 수행한 후에 switch문을 빠져나가려면 break문을 꼭 작성해야 한다.

```
int time = 9;
switch (time){
case 8 :
    System.out.println("출근합니다.");
    break;
case 9 :
    System.out.println("회의를 합니다.")
    break;
case 10 :
    System.out.println("업무를 봅니다.")
    break;
default :
    System.out.println("외근을 나갑니다.")
}
```

두 번째 case문을 실행하고 switch문을 빠져나간다

[ 실행결과 ]  
회의를 합니다.

### 만약 break가 없다면?

break가 없으면 switch문을 빠져나오지 못하고 다음 case를 연달아 실행한다.

```
int time = 9;
switch (time){
case 8 :
    System.out.println("출근합니다.");
case 9 :
    System.out.println("회의를 합니다.")
case 10 :
    System.out.println("업무를 봅니다.")
default :
    System.out.println("외근을 나갑니다.")
}
```

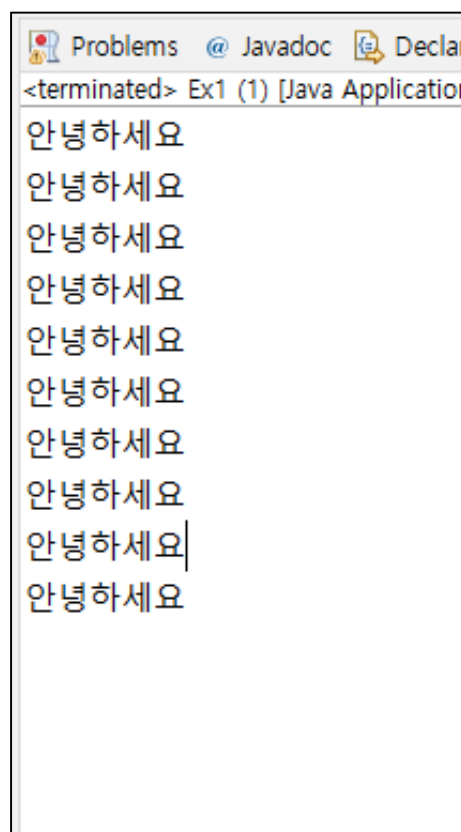
두번째 case문을 실행 한 후,  
break문을 만나지 못하고 남은 코드를 연달아 실행한다

[ 실행결과 ]  
회의를 합니다.  
업무를 봅니다.  
외근을 나갑니다.

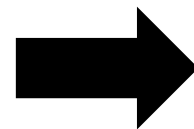
### 반복문이란?

- 특정 코드를 반복적으로 실행하는 문법이다.

만약 안녕하세요를 10번 출력하려면 어떻게 해야할까?



```
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");  
print ("안녕하세요");
```



```
while (num <= 10) {  
    print ("안녕하세요");  
}
```

# 반복문

## 반복문의 종류

반복문의 종류에는 while문과 for문이 있다.

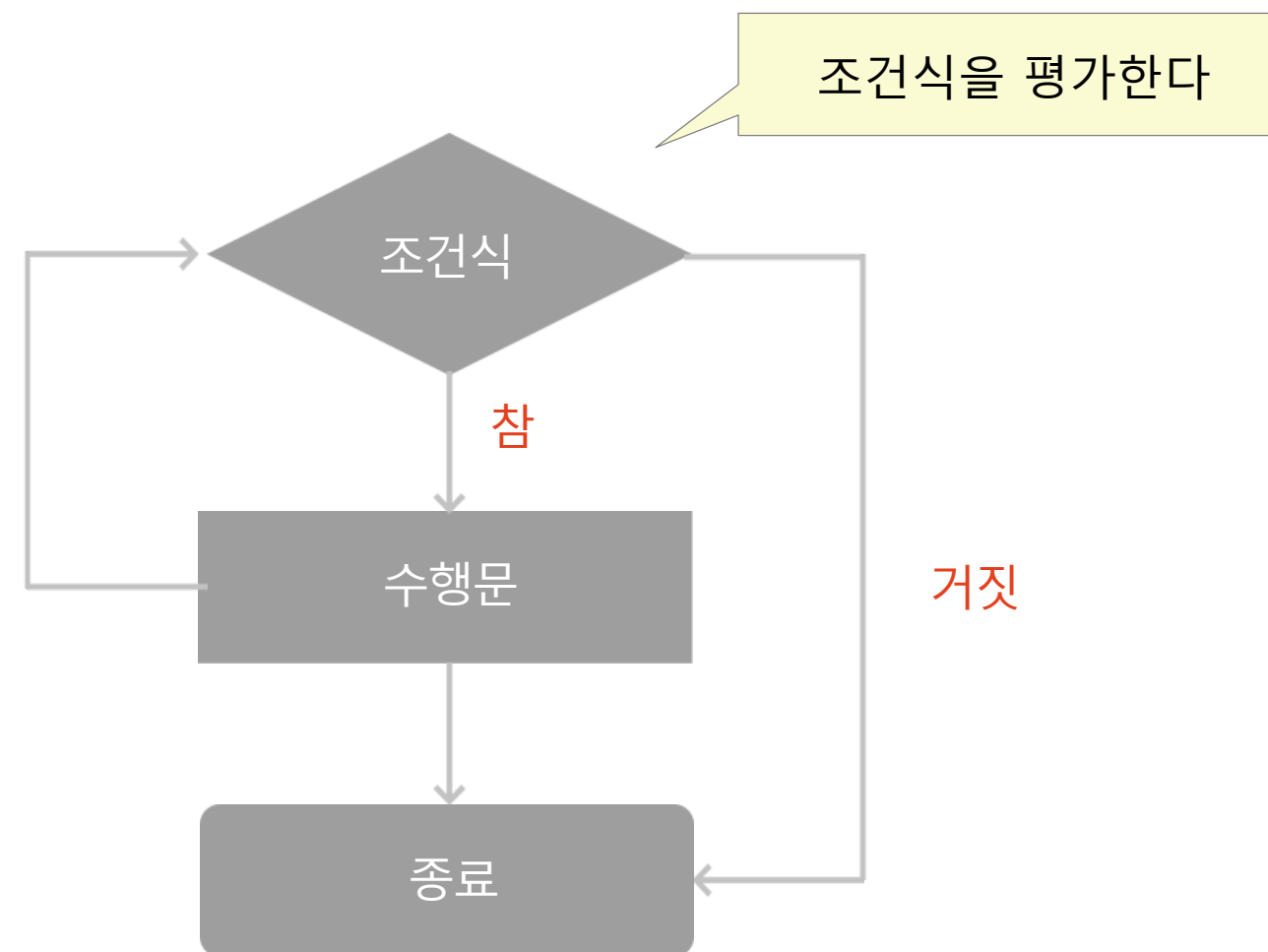
모두 조건을 만족하는 동안 반복 수행을 한다는 것은 동일하지만, 사용방법에 차이가 있다.

반복문	내용
while문	반복 횟수를 정확히 모를 때 사용한다.
do-while문	최소한 한번 이상 수행해야할 때 사용한다.
for문	반복 횟수를 정확히 알 때 사용한다.

while문은 조건식이 참인 동안 코드블록을 반복해서 실행한다.  
조건식이 거짓이 되면 while문을 빠져나간다.

```
while (조건식) {  
    수행문  
}
```

조건이 참인 동안  
반복 수행



다음은 while문을 사용하여 1부터 10까지 출력하는 과정이다.

1) 프로그램에 필요한 변수를 만든다.

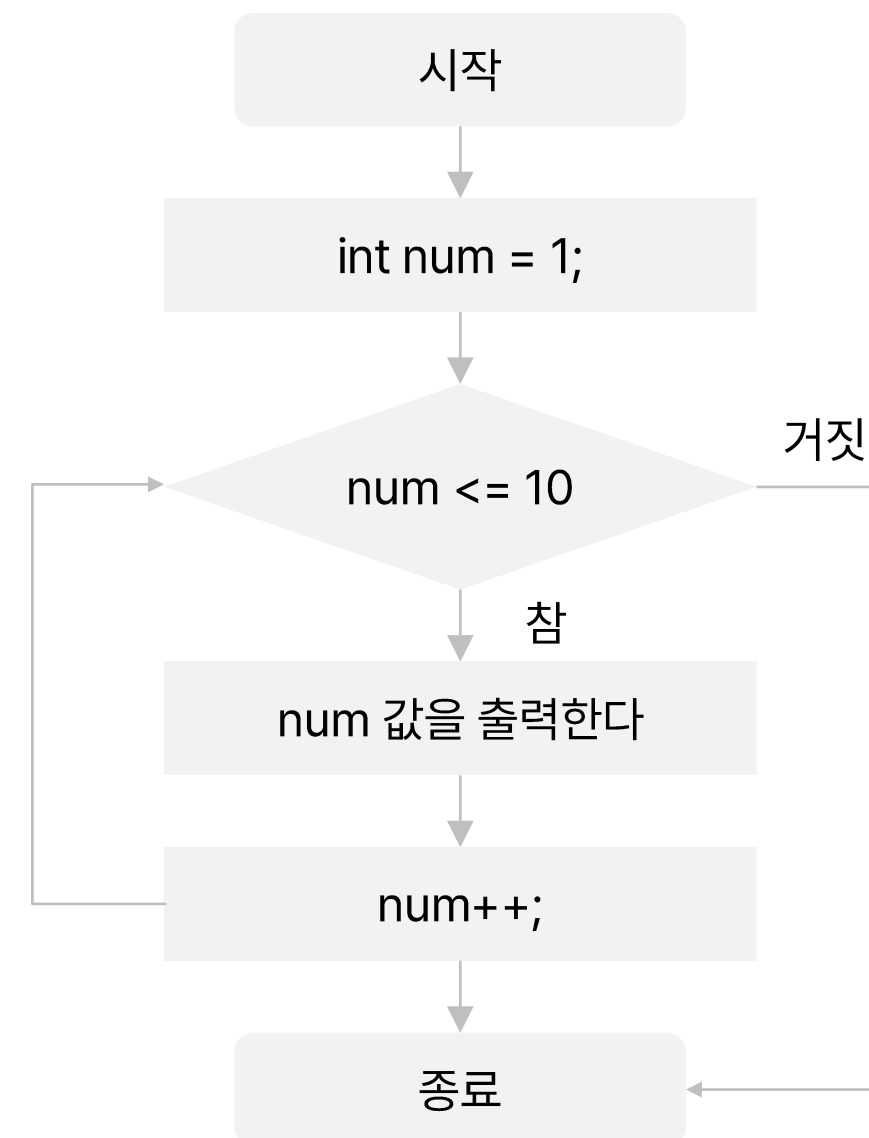
→ num (출력할 수)

2) 조건식을 작성한다.

→ num이 1부터 10이 될 때 까지

3) 반복할 작업을 작성한다.

→ num을 1씩 증가시키면서 출력한다



(10번 반복)  
1,2,3,4,5,6,7,8,9,10



while문은 조건식에 맞지 않으면 반복 수행이 한번도 일어나지 않는다.  
하지만 do while문은 조건식이 거짓이더라도 무조건 한번 이상 블록을 실행한다.  
do while문은 먼저 블록을 실행하고, 마지막에 조건식을 검사한다.

**1** do {

수행문

**2** } while (조건식)

```
int num = 11;  
do{  
    system.out.println( num );  
    num++;  
} while ( num <= 10 )
```

[ 실행결과 ]

11

for문은 조건문을 만들기 위한 여러 요소(초기화식, 조건식, 증감식)를 함께 작성해야 한다.

구조는 복잡하지만, 반복 수행할 조건을 한눈에 볼 수 있어서 편리하다.

초기화 식은 for문이 시작할 때 한번만 실행하며, 사용할 변수를 초기화한다.

```
    1           2           4  
for ( 초기화식; 조건식; 증감식 ) {  
    3 수행문  
}
```

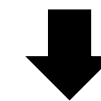
다음은 for문을 사용하여 1부터 10까지 출력하는 과정이다.

```
for (int i = 1; i <= 10; i++){  
    print( i );  
}
```

<변수 i의 용도>

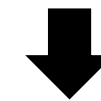
1. 조건식을 작성할 때 사용한다.
2. 반복 작업에 사용한다.

1) 변수 i를 1로 초기화한다.



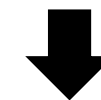
2) i가 10보다 작은지 검사한다.  
3) 조건이 참이기 때문에 i를 출력한다.  
4) 증감식 i++을 수행하여 i의 값이 2가 된다.

"1"



2) i가 10보다 작은지 검사한다.  
3) 조건이 참이기 때문에 i를 출력한다.  
4) 증감식 i++을 수행하여 i의 값이 3가 된다.

"2"



조건식이 거짓이 될 때 까지 반복..

...

**반복문 안에서 continue문을 만나면?**

- 다음 코드를 실행하지 않고 증감식으로 돌아간다.
- 반복문은 계속 실행하지만 특정 조건에서 건너뛰어야 할 때 사용한다.

**반복문 안에서 break문을 만나면?**

- 반복문이 종료된다.
- 특정조건에서 반복문을 빠져나와야 할 때 사용한다.

```
for (초기문; 조건식; 증감식){  
    ...  
    continue;  
    ...  
}
```

Skip

```
for (초기문; 조건식; 증감식){  
    ...  
    break;  
    ...  
}
```

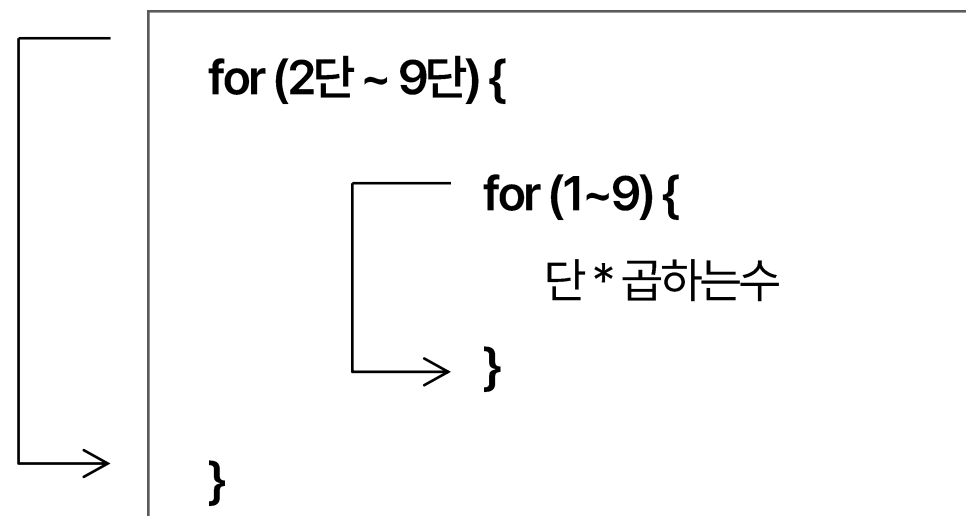
Stop

### 중첩반복문이란?

- for문 안에서 또 다른 for문을 사용하는 경우를 말한다.
- 대표적인 예시로 구구단이 있다.

### 구구단 만들기

1. 외부for문에서 단이 2~9까지 반복한다.
2. 내부for문에서 곱하는 수가 1~9까지 반복한다.
3. 내부for문에서 구구단을 출력하는 코드를 실행한다.



2 × 1 = 2  
2 × 2 = 4  
2 × 3 = 6  
2 × 4 = 8  
2 × 5 = 10  
2 × 6 = 12  
2 × 7 = 14  
2 × 8 = 16  
2 × 9 = 18

단

곱하는수

제어문	문법
IF (조건문)	if, if ~ else
SWITCH (선택문)	switch ~ case
LOOP (반복문)	for, while
CONTROL (제어문)	break, continue

## 디버깅이란?

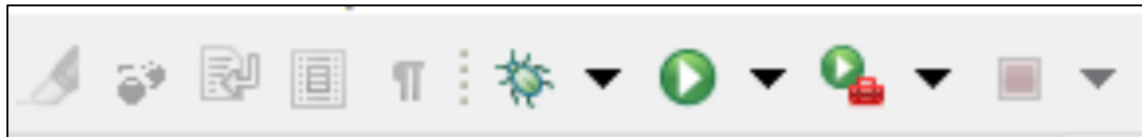
- 프로그래밍 과정을 추적하는 기능이다.

1

```
10      int sum = 0;  
11  
12      // 변수에 특별한 의미가 없으면 보통 i를 사용한다.  
13      for(int i=1; i<=10; i++){  
14  
15  
16  
17      System.out.println("1부터 10까지의 합은 " + sum + "입니다.");
```

더블클릭하여 브레이크 포인트를 건다

2



상단의 벌레모양 버튼을 눌러서 디버깅 모드를 실행한다

## 단축키

F6: 다음 라인으로 이동

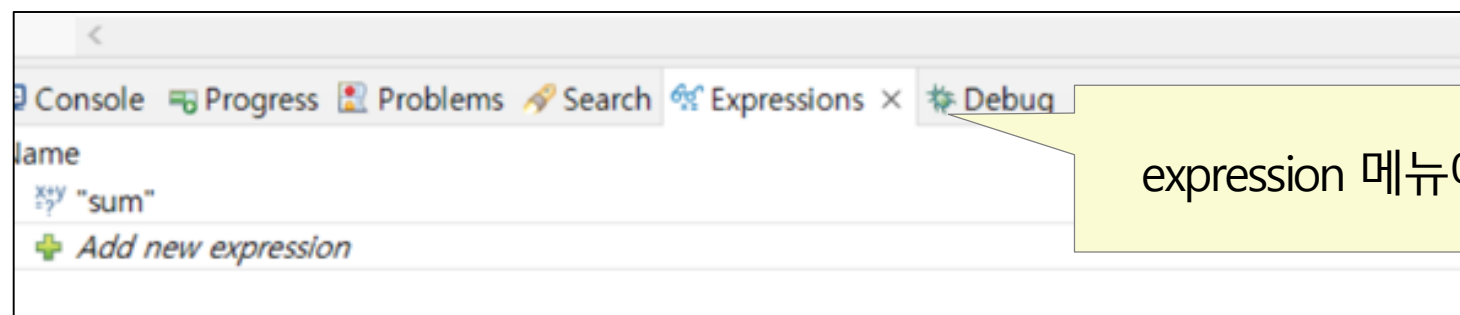
F8: 다음 브레이크 포인트로 이동

3

```
8 public static void main(String[] args) {  
9  
10     int sum = 0;  
11  
12     // 변수에 특별한 의미가 없으면 보통 i를 사용한다.  
13     for(int i=1; i<=10; i++){  
14         sum = sum + (i+1);  
15     }  
16  
17     System.out.println("1부터 10까지의 합은 " + sum + "입니다.")  
18 }  
19 }  
20 }
```

코드를 한줄씩 실행한다

4



expression 메뉴에서 표현식을 입력하고 값을 확인한다