chapter 03.

주석문과 제어문

❑ 1) 주석문 (94p)

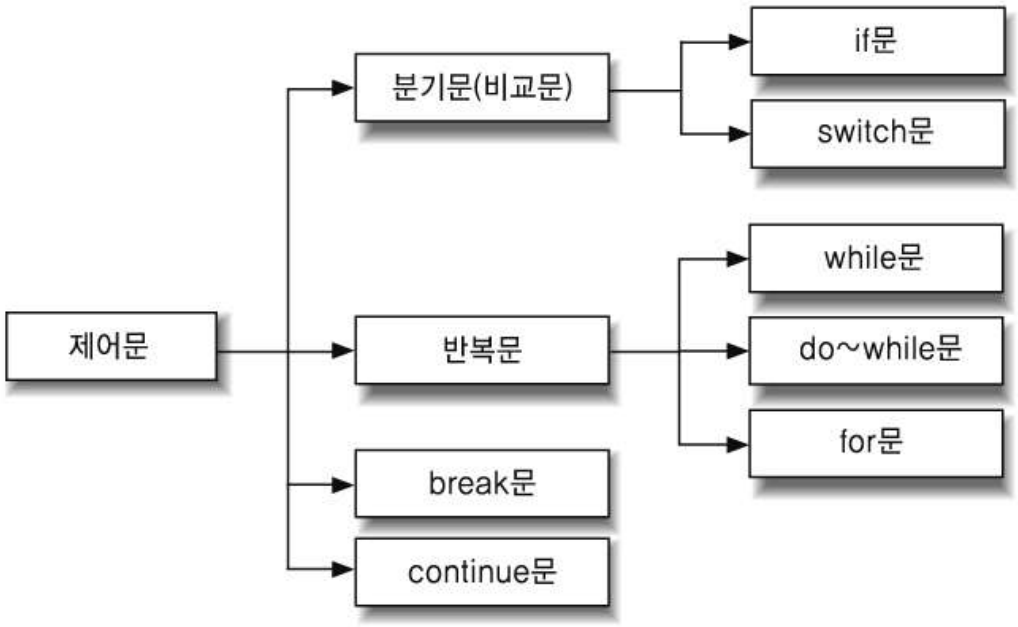
- 주석문은 실제 프로그램에 영향을 주지 않으며 단지 소스코드의 기능이나 동작을 설명하기 위해 사용되는 것이다.

[표 3-1] 주석문의 종류

주석 종류	의미	설명
// 주석문	단행 주석처리	현재 행에서 //의 뒷문장부터 주석으로 처리된다.
/* 주석문 */	다행 주석처리	/*에서 */ 사이의 문장이 주석으로 처리된다.
/** 주석문 */	HTML 문서화 주석처리	/**에서 */ 사이의 문장이 주석으로 처리된다. 장점은 HTML 문서화로 주석이 처리되므로 API와 같은 도움말 페이지를 만들 수 있다.

:: 본 교재 94p의 예문과 예제를 참조

❍ 제어문의 구성



[그림 3-29] 제어문의 구성

❑ 2) 제어문1 (97p)

프로그램의 흐름에 영향을 주고 때에 따라 제어가 가능하도록 하는 것이 바로 '제어문'이다.

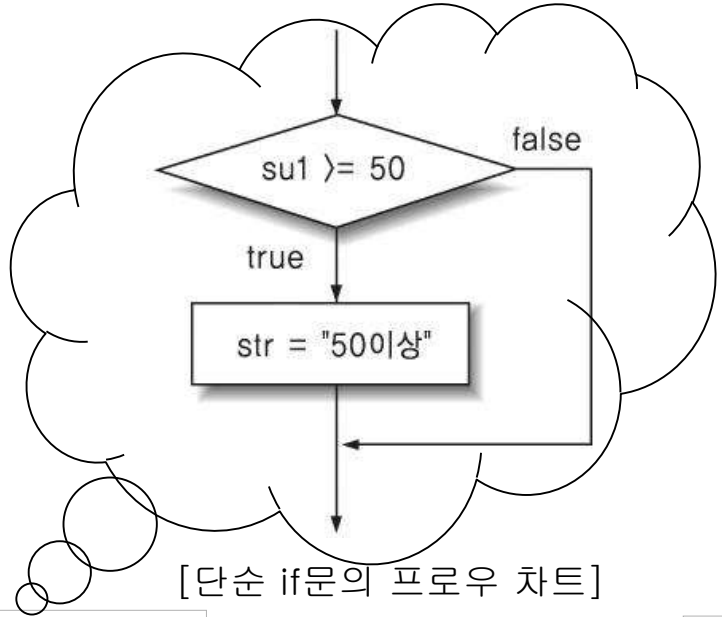
- 제어문의 종류
  - 분기문 (비교문)

    : 주어진 조건의 결과에 따라 실행 문장을 다르게 하여 전혀 다른 결과를 얻기 위해 사용되는 제어문이다.
    - if문, switch문
  - 반복문

    : 특정한 문장을 정해진 규칙에 따라 반복처리하기 위한 제어문이다.
    - for문, while문, do~while문
  - break문 : 반복문 내에서 쓰이며 반복문을 빠져나갈 때 쓰이는 제어문이다.
  - continue문 : 현재 진행되는 반복 회차를 포기하고 다음 회차로 이동 한다.

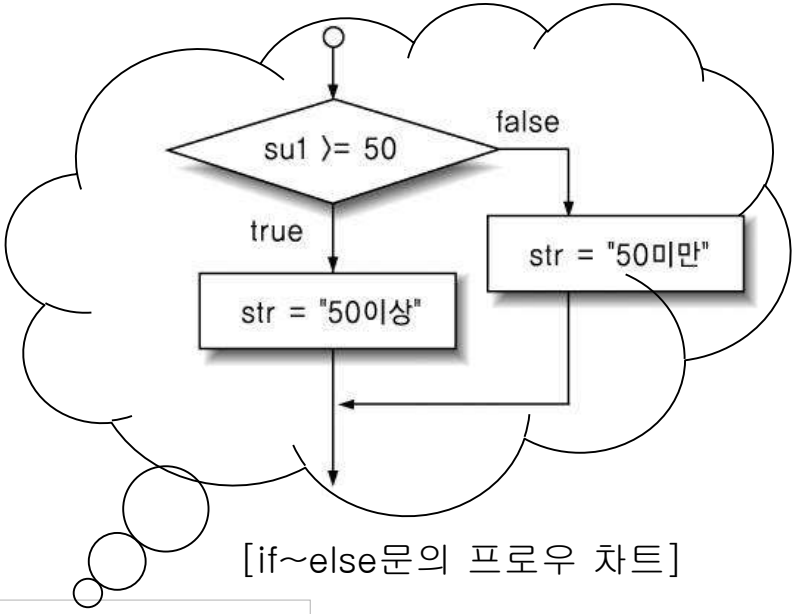
❍if문

- boolean형의 결과를 가지는 조건식이 있어야 하며 그 조건식의 결과로 수행하는 문장을 결정하게 되는 분기문이다.



[단순 if문의 프로우 차트]

```
if(su1 >= 50)
   str = “50이상”;
```

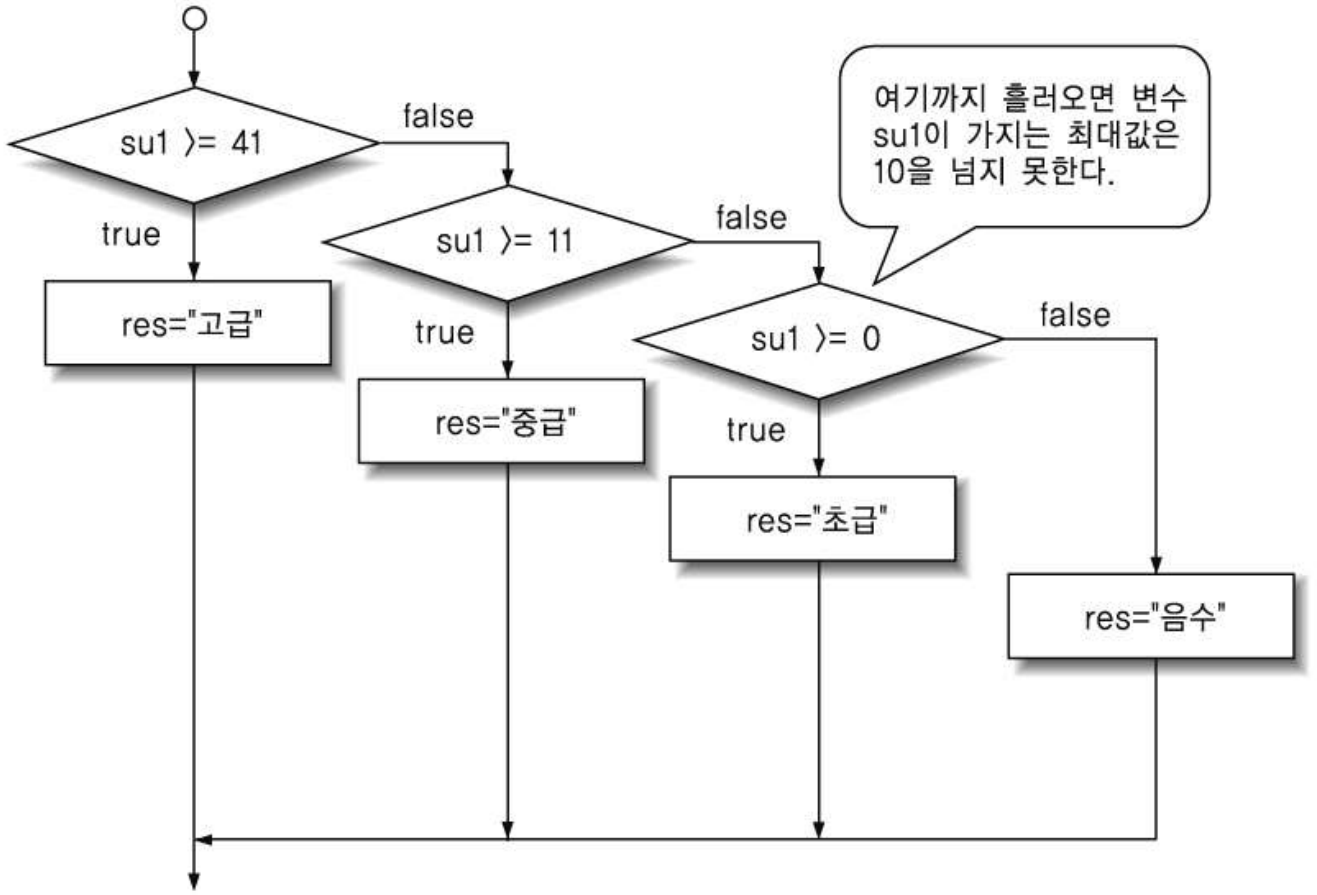


[if~else문의 프로우 차트]

```
if(su1 >= 50)
   str = “50이상”;
else
   str = “50미만”;
```

❑ 2) 제어문3 (101p)

```
if(su1 >= 41)
   res = “고급”;
else if(su1 >= 11)
   res = “중급”;
else if(su1 >= 0)
   res = “초급”;
else
   res = “음수”;
```



[그림 3-8] if ~ else if문의 플로우 차트

switch문

- If문의 조건값은 boolean형인데 비해 switch문의 조건값은 long형을 제외한 정수형(byte,short,int) 또는 char형인 것이 다르다. 구성은 다음과 같다.

```
switch(인자값) {
        case 조건값1 :      ← semicolon이 아닌 colon임을 기억하자
          수행문; break;
case 조건값2 :
          수행문; break;  ← break문은 하나의 조건 값마다 넣어주는 것이
case 조건값3 :        적당하며 만약 없을 시에는 다음 break문을
          수행문; break       만날 때가지 모든 수행문을 처리한다.
        default :           ← 받은 인자값이 case문의 조건값1에서 조건값3까지
          수행문;               일치하는 것이 단 하나도 없다면 default를 수행한다.
    }
```

※ 주의 사항

case뒤에 오는 조건값이 중복되지 않도록 해야 한다. 그렇지 않으면 case를 구분하는 값이 복제되어 중복되었다는 오류가 발생한다.

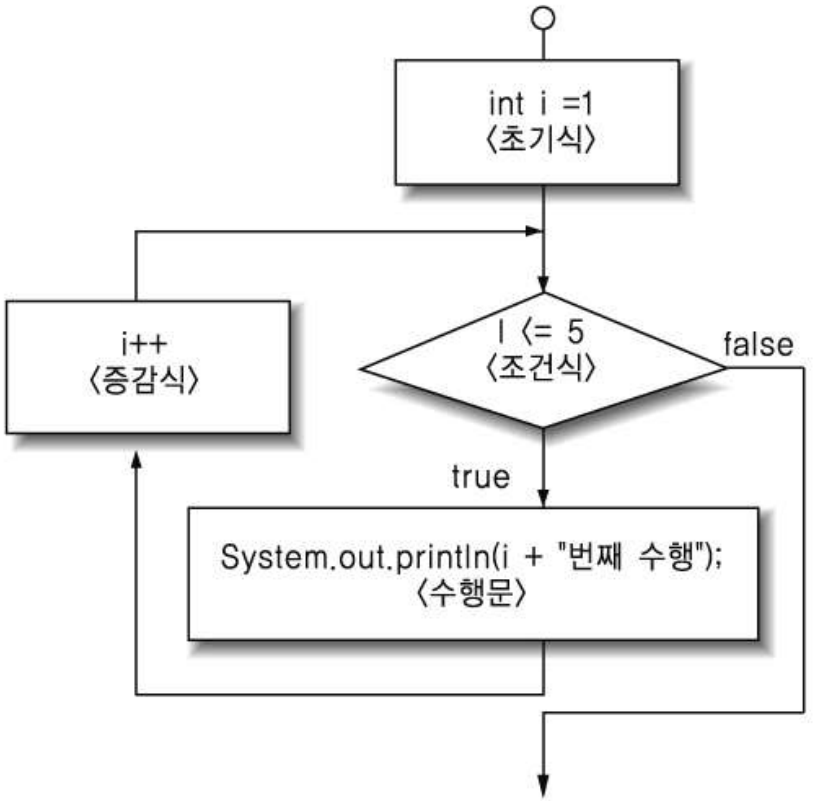
❍for문

- 특정한 명령들을 정해진 규칙에 따라 반복처리 할 때 사용하는 제어문이다. 다음은 for문의 구성이다.

```
for(초기식 ; 조건식 ; 증감식){
    수행문1;
    수행문2;
}
```

초기식	가장 먼저 수행하는 부분이며 두 번 다시 수행하지 않는다.(다중 for문에서는 예외)
조건식	초기식 다음으로 수행하는 부분이며 loop가 돌 때마다 한번씩 비교하여 반복을 수행해야 할지 반복을 벗어나야 할지를 결정한다.
증감식	증감식은 loop를 수행할 때마다 조건식에서 비교하기 전에 항상 수행하며 조건식에 사용되는 변수의 값을 증가 시키거나 감소 시켜 loop를 원활하게 수행하거나 무한 루프를 피하는데 바탕이 되는 부분이다.

```
04          for(int i = 1 ; i <= 5 ; i++)
05                  System.out.println(i+"번째 수행");
06
```



[그림 3-14] for문의 플로우 차트

❍ 다중 for문

- 단일 for문에서 끝나는 것이 아니라 그것을 다시 여러 번 반복하는 제어문이다. 다시 말해서 for문 안에 for문이 있는 경우를 다중 for문이라 한다.
- 예문 : 애국가 1절~4절까지를 3번 부르세요!

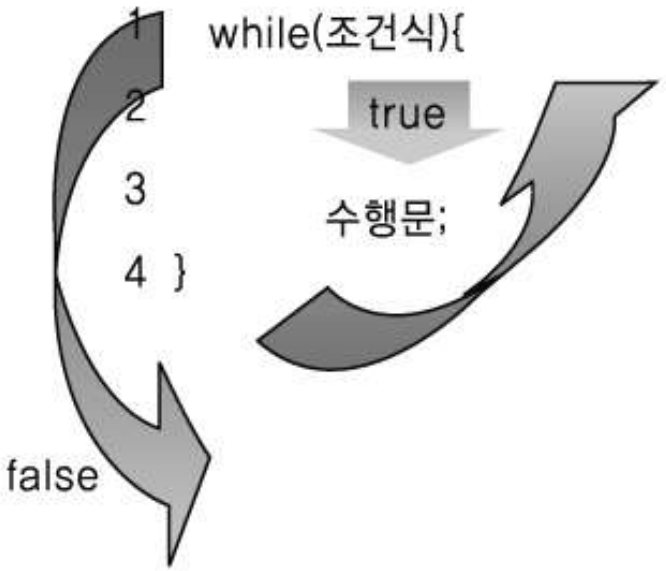
```
for(초기식1 ; 조건식1 ; 증감식1) {
    for(초기식2 ; 조건식2 ; 증감식2){
        명령어2;
    }
    명령어1;
}
명령어3;
```

3번 부르기

1절부터 4절까지 부르기

❍while문

- while문은 for문과 유사하며 조건비교에 만족 할 때에만 반복 처리하는 제어문이다. 다음은 구성과 동작이다



:: 先 비교, 後 처리

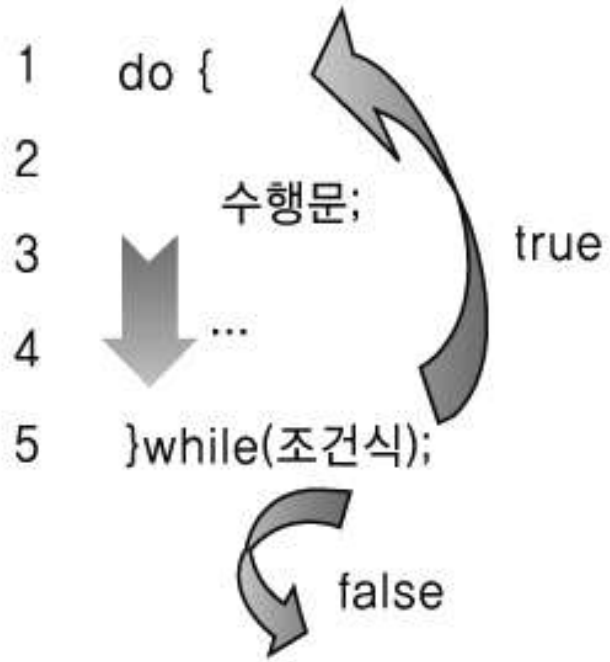
[그림 3-19] while문의 구성과 동작

※ 주의 사항

모든 반복문은 무한루프에 빠지지 않도록 주의해야 한다.

❍ do~while문

- while문이 [先 비교, 後 처리]라 하면 do ~ while문은 [先 처리, 後 비교]이다. 즉 조건비교에 불 만족하다 할지라도 무조건 한번은 수행하게 되어 있음을 기억하자!



:: 先 처리, 後 비교

[그림 3-20] do~while문의 구성과 동작

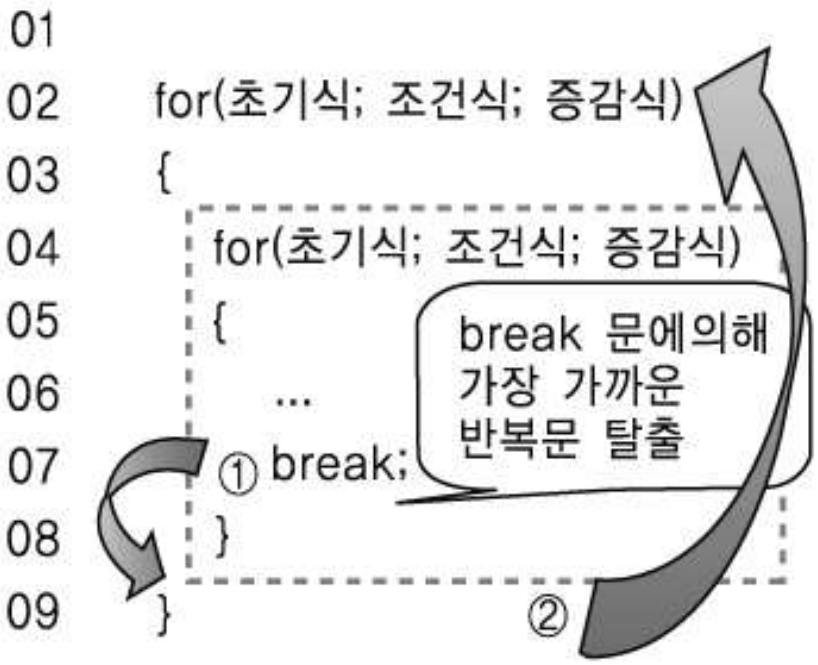
※ 주의 사항

조건식 후의 (;)를 잊지 말자!, 모든 반복문은 무한루프에 빠지지 않도록 주의해야 한다.

❍break문

- 가장 가까운 반복문을 탈출할 때 쓰이는 제어문이다.

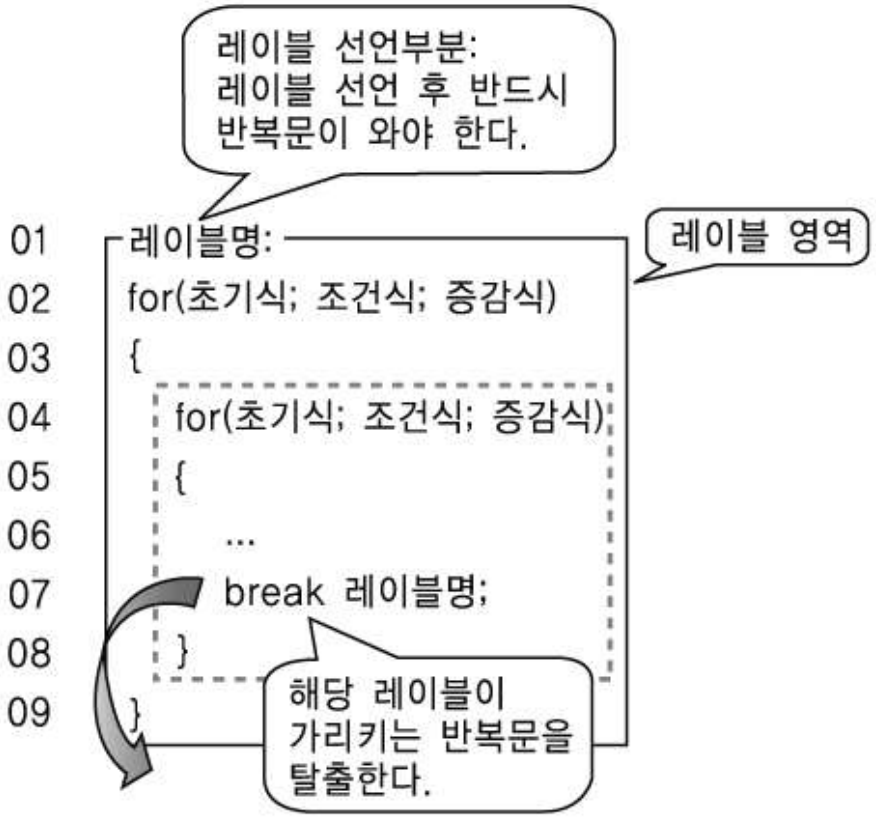
다음은 내부 for문에서 break를 사용 했으므로 내부 for문만 탈출한다는 뜻의 그림이다.



[그림 3-21] break문의 구성과 동작

❍ break label문

- break label은 break문과 같지만 다중 반복문에서 한번에 바깥쪽 반복문을 탈출할 때 많이 쓰이는 제어문이다.

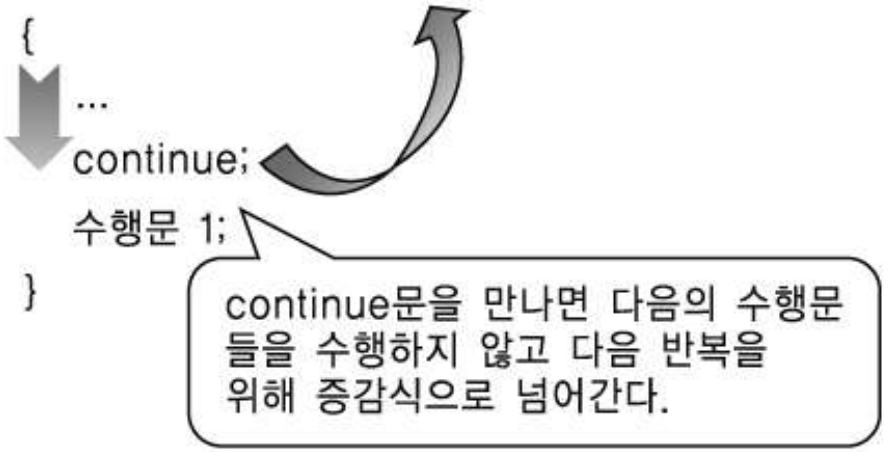


[그림 3-23] break label문의 구성과 동작

❍ continue문

- 반복문을 탈출하기위해 사용되는 것이 아니라 continue문 이하의 수행문들을 포기하고 다음 회차의 반복을 수행하기위한 제어문이다.

```
01
02    for(초기식; 조건식; 증감식)
03    {
04        for(초기식; 조건식; 증감식)
05        {
06            ...
07            continue;
08            수행문 1;
09        }
10    }
```



[그림 3-25] continue문의 구성과 동작

❍ continue label문

- 레이블을 가지는 continue문은 레이블이 지칭하는 반복문의 조건식 또는 증감식으로 프로그램상 수행 시점(제어권)이 이동한다.

```
01    레이블명:
02    for(초기식; 조건식; 증감식)
03    {
04       for(초기식; 조건식; 증감식)
05       {
06          ...
07          continue 레이블명;
08          수행문 1;
09       }
10    }
```

내부 반복문을 중단하고 외부 반복문의
다음 반복회차를 수행하기 위한 제어문이다.
continue문 아래의 수행문 1은 수행하지 못한다.

[그림 3-27] continue label문의 구성과 동작