

2025년 상반기 K-디지털 트레이닝

조건문과 반복문

[KB] IT's Your Life



☑ 코드가 실행되는 흐름 제어하기

○ 실행 흐름

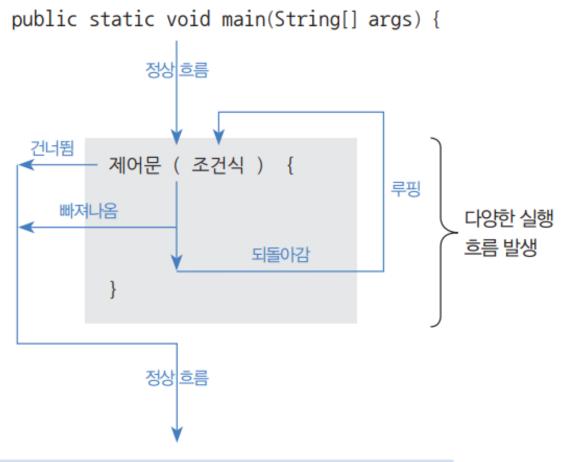
■ main() 메소드의 시작 중괄호({)에서 끝 중괄호(})까지 위부 터 아래로 실행되는 흐름

ㅇ 흐름 제어문

■ 실행 흐름을 개발자가 원하는 방향으로 바꿀 수 있도록 해주 는 것

ㅇ 루핑

■ 반복문이 실행 완료된 경우 제어문 처음으로 다시 되돌아가 반복 실행되는 것



조건문	반복문
if 문, switch 문	for 문, while 문, do-while 문

🧿 조건에 따라 실행되는 if 문

- 조건식의 결과에 따라 블록 실행 여부가 결정
- 조건식에는 true 또는 false 값을 산출할 수 있는 연산식이나 boolean 변수가 올 수 있음
- 조건식이 true이면 블록을 실행하고 false이면 블록을 실행하지 않음
- {} 블록 내에 실행문이 하나인 경우 중괄호 생략 가능 → 비권장

```
조건식이 false  조건식이 true }
```

```
if (조건식) {
실행문;
실행문;
...
}
```

```
if ( 조건식 )
실행문;
```

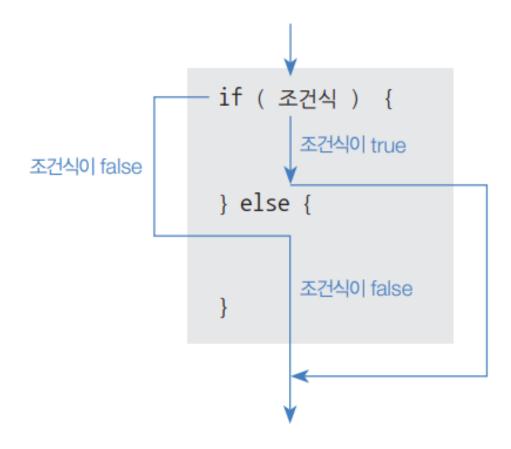
ch04.sec02.IfExample.java

```
package ch04.sec02;
public class IfExample {
 public static void main(String[] args) {
   int score = 93;
   if(score >= 90) {
    System.out.println("점수가 90보다 큽니다.");
    System.out.println("등급은 A입니다.");
   if(score < 90)
    System.out.println("점수가 90보다 작습니다.");
    System.out.println("등급은 B입니다."); // if문과는 상관없는 실행문
```

```
점수가 90보다 큽니다.
등급은 A입니다.
등급은 B입니다.
```

☑ 조건에 따라 실행되는 if 문

- o if-else 문
 - 조건식이 true이면 if 문 블록이 실행되고, false이면 else 블록이 실행



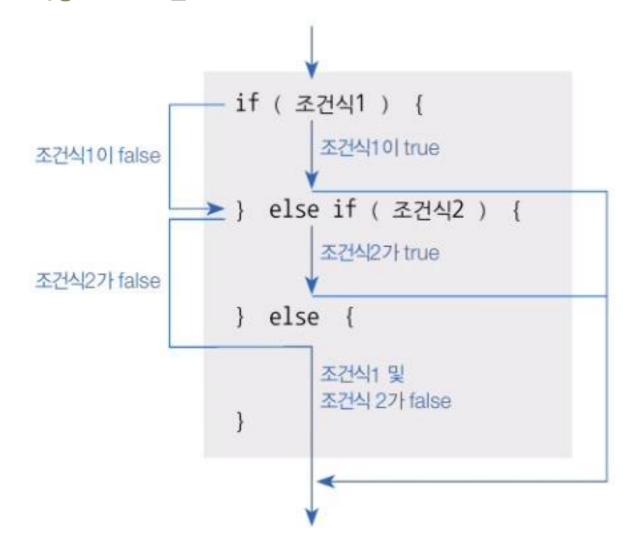
ch04.sec02.IfElseExample.java

```
package ch04.sec02;
public class IfElseExample {
 public static void main(String[] args) {
   int score = 85;
   if(score>=90) {
    System.out.println("점수가 90보다 큽니다.");
    System.out.println("등급은 A입니다.");
   } else {
    System.out.println("점수가 90보다 작습니다.");
    System.out.println("등급은 B입니다.");
```

```
점수가 90보다 작습니다.
등급은 B입니다.
```

☑ 조건에 따라 실행되는 if 문

○ 다중 if-else 문



ch04.sec02.IfElseIfElseExample..java

```
package ch04.sec02;
public class IfElseIfElseExample {
 public static void main(String[] args) {
   int score = 75;
   if(score>=90) {
    System.out.println("점수가 100~90입니다.");
    System.out.println("등급은 A입니다.");
  } else if(score>=80) { // 80 <= score < 90일 경우
    System.out.println("점수가 80~89입니다.");
    System.out.println("등급은 B입니다.");
  } else if(score>=70) { // 70 <= score < 80일 경우
    System.out.println("점수가 70~79입니다.");
    System.out.println("등급은 C입니다.");
  } else {
            // score < 70일 경우
    System.out.println("점수가 70 미만입니다.");
    System.out.println("등급은 D입니다.");
                                             점수가 70~79입니다.
                                             등급은 C입니다.
```

🧿 랜덤한 수 만들기

- Math.random()
 - 0.0 <= ~ < 1.0사이의 double 타입 난수 리턴
- 정수 범위의 랜덤한 수 만들기 (1 ~ 6)

$$(0.0 * 6) \le (Math.random() * 6) \le (1.0 * 6)$$

 (0.0)

(int)
$$0.0 \le \frac{\text{(int) (Math.random()} * 6)}{(0.1, 2, 3, 4, 5)} \le \text{(int) } 6.0$$

(0+1)
$$\leftarrow$$
 ((int) (Math,random() * 6) + 1) \leftarrow (6+1)
(0) (1, 2, 3, 4, 5, 6) (7)

```
int num = (int) (Math.random() \star n) + start;
```

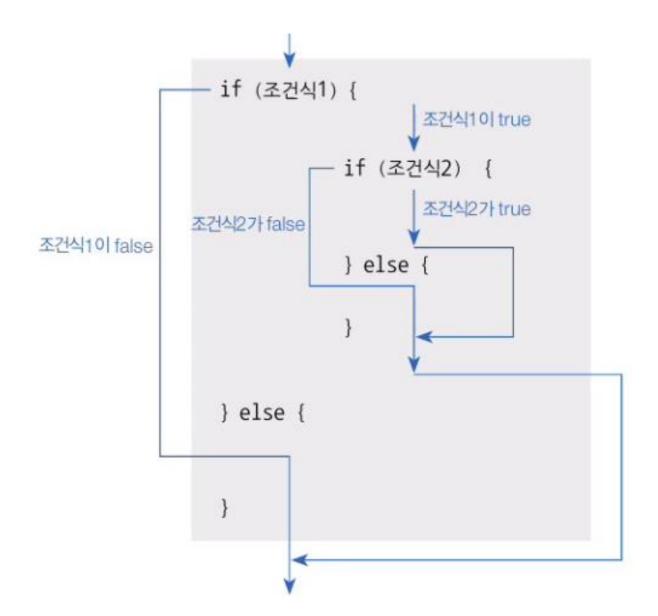
$$int num = (int) (Math.random() * 45) + 1;$$

ch04.sec02.IfDiceExample.java

```
package ch04.sec02;
public class IfDiceExample {
 public static void main(String[] args) {
   int num = (int)(Math.random()*6) + 1; // 주사위 번호 뽑기
   if(num==1) {
    System.out.println("1번이 나왔습니다.");
   } else if(num==2) {
    System.out.println("2번이 나왔습니다.");
   } else if(num==3) {
    System.out.println("3번이 나왔습니다.");
   } else if(num==4) {
    System.out.println("4번이 나왔습니다.");
   } else if(num==5) {
    System.out.println("5번이 나왔습니다.");
   } else {
    System.out.println("6번이 나왔습니다.");
                                               3번이 나왔습니다.
```

☑ 조건에 따라 실행되는 if 문

- 중첩 if 문
 - if문 블록안에 if문을 다시 사용



ch04.sec02.IfNestedExample.java

```
package ch04.sec02;
public class IfNestedExample {
 public static void main(String[] args) {
   int score = (int)(Math.random()*20) + 81;
   System.out.println("점수: " + score);
   String grade;
   if(score>=90) {
     if(score>=95) {
       grade = "A+";
     } else {
      grade = "A";
   } else {
     if(score>=85) {
       grade = "B+";
     } else {
      grade = "B";
```

```
System.out.println("학점: " + grade);
```

점수: 90 학점: A

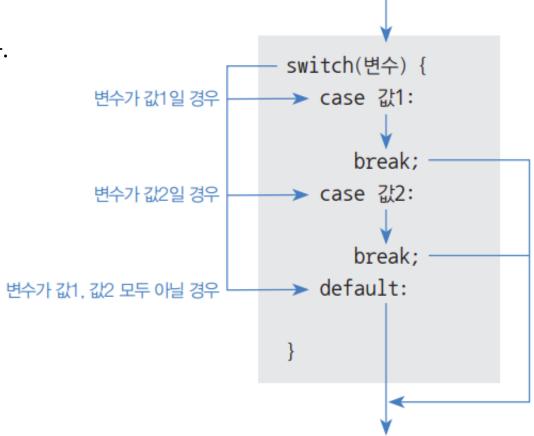
변수값에 따라 case를 실행하는 switch 문

o switch 문

- 괄호 안의 변수값에 따라 해당 case로 가서 실행문을 실행.
- 변수값과 동일한 값을 갖는 case가 없으면 default로 가서 실행문을 실행하며, default 생략 가능

o break

- 다음 case를 실행하지 않고 switch 문을 빠져나갈 때 사용.
- break가 없다면 다음 case가 연달아 실행



ch04.sec03.SwitchExample.java

```
package ch04.sec03;
public class SwitchExample {
 public static void main(String[] args) {
   int num = (int)(Math.random()*6) + 1;
   switch(num) {
    case 1:
      System.out.println("1번이 나왔습니다.");
      break;
    case 2:
      System.out.println("2번이 나왔습니다.");
      break;
    case 3:
      System.out.println("3번이 나왔습니다.");
      break;
    case 4:
      System.out.println("4번이 나왔습니다.");
      break;
```

```
case 5:
 System.out.println("5번이 나왔습니다.");
 break;
default:
 System.out.println("6번이 나왔습니다.");
```

5번이 나왔습니다.

ch04.sec03.SwitchNoBreakCaseExample.java

```
package ch04.sec03;
public class SwitchNoBreakCaseExample {
 public static void main(String[] args) {
   int time = (int)(Math.random()*4) + 8;
   System.out.println("[현재시간: " + time + " 시]");
   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.");
                                                    [현재시간: 9 시]
                                                    회의를 합니다.
                                                    업무를 봅니다.
                                                    외근을 나갑니다.
```

ch04.sec03.SwitchCharExample.java

```
package ch04.sec03;
public class SwitchCharExample {
 public static void main(String[] args) {
   char grade = 'B';
   switch(grade) {
    case 'A':
    case 'a':
      System.out.println("우수 회원입니다.");
      break;
    case 'B':
    case 'b':
      System.out.println("일반 회원입니다.");
      break;
    default:
                                                     일반 회원입니다.
      System.out.println("손님입니다.");
```

3 switch 문

ch04.sec03.SwitchExpressionsExample.java - Java 12 이후, 표현식 사용하기

```
public class SwitchExpressionsExample {
 public static void main(String[] args) {
   char grade = 'B';
   switch(grade) {
    case 'A', 'a' -> {
      System.out.println("우수 회원입니다.");
    case 'B', 'b' -> {
      System.out.println("일반 회원입니다.");
    default -> {
      System.out.println("손님입니다.");
   switch(grade) {
    case 'A', 'a' -> System.out.println("우수 회원입니다.");
    case 'B', 'b' -> System.out.println("일반 회원입니다.");
    default -> System.out.println("손님입니다.");
                                                    일반 회원입니다.
                                                    일반 회원입니다.
```

3 switch 문

switch expression

- o switc문을 대입문에서 사용 가능
- 중괄호를 사용하는 경우 yield로 배정값 지정
- 이경우 반드시 default가 존재해야 함

```
타입 변수 = switch(grade) {
 case "값1" -> 변수값;
 case "값2" -> {
   ...;
   yield 변수값;
 default -> 변수값;
```

ch04.sec03.SwitchValueExample.java

```
package ch04.sec03;
public class SwitchValueExample {
 public static void main(String[] args) {
   String grade = "B";
   //Java 11 이전 문법
   int score1 = 0;
   switch(grade) {
     case "A":
        score1 = 100;
        break;
     case "B":
        int result = 100 - 20;
        score1 = result;
        break;
     default:
        score1 = 60;
   System.out.println("score1: " + score1);
```

```
//Java 12부터 가능
int score2 = switch(grade) {
 case "A" -> 100;
 case "B" -> {
   int result = 100 - 20;
   //Java 13부터 가능
   yield result;
 default -> 60;
System.out.println("score2: " + score2);
score1: 80
score2: 80
```

♡ 반복문

```
int sum = 0;

sum = sum + 1;

sum = sum + 2;

sum = sum + 3;

sum = sum + 4;

sum = sum + 5;

System.out.println("1~5까지의 합:" + sum);
```

```
int sum = 0;

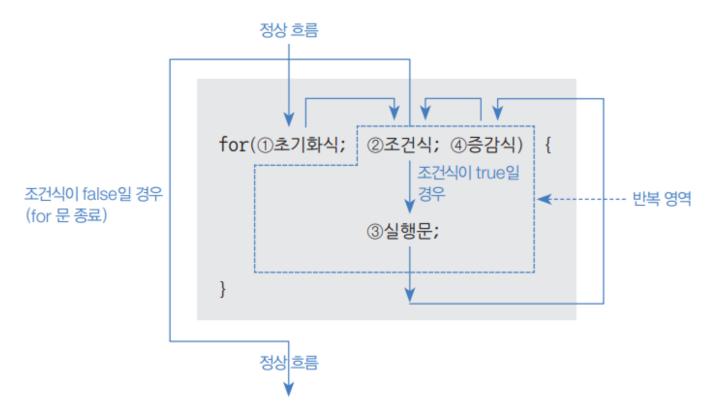
for (int i=1; i<=100; i++) {
    sum = sum + i; • 100번반복
}

System.out.println("1~100까지의 합:" + sum);
```

4 for 문

😕 실행문을 반복하는 for 문

o 실행문을 여러 번 반복 실행해주기 때문에 코드를 간결하게 만들어줌



- ①초기화식이 제일 먼저 실행
- ②조건식을 평가해서 true이면
- ③실행문을 실행시키고, false이면 for 문을 종료하고 블록을 건너뜀.
- ②조건식이 true가 되어 ③실행문을 모두 실행하면
- ④증감식이 실행.

다시 ②조건식을 평가.

평가 결과가 다시 true이면 ③ \rightarrow ④ \rightarrow ②로 다시 진행하고, false이면 for 문이 끝남

초기화식에서 부동 소수점을 쓰는 float 타입을 사용하지 않도록 주의

ch04.sec04.PrintFrom1To10Example.java

```
package ch04.sec04;

public class PrintFrom1To10Example {
  public static void main(String[] args) {
    for(int i=1; i<=10; i++) {
       System.out.print(i + " ");
    }
  }
}</pre>
```

1 2 3 4 5 6 7 8 9 10

ch04.sec04.SumFrom1To100Example.java

```
package ch04.sec04;
public class SumFrom1To100Example {
 public static void main(String[] args) {
   int sum = 0;
   int i;
   for(i=1; i<=100; i++) {
     sum += i;
   System.out.println("1~" + (i-1) + " 합 : " + sum);
```

```
1~100 합 : 5050
```

☑ ch04.sec04.FloatCounterExample.java - 카운트 변수로 float 사용은 피해야 함

```
package ch04.sec04;

public class FloatCounterExample {
   public static void main(String[] args) {
     for(float x=0.1f; x<=1.0f; x+=0.1f) {
        System.out.println(x);
     }
   }
}</pre>
```

```
0.1

0.2

0.3

0.4

0.5

0.6

0.70000005

0.8000001

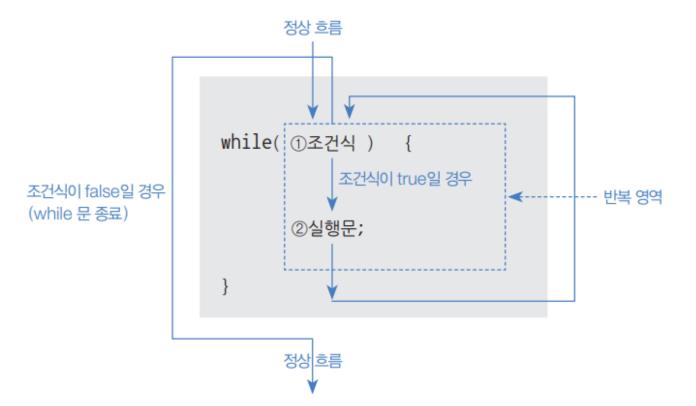
0.9000001
```

ch04.sec04.MultiplicationTableExample.java − 중첩

```
package ch04.sec04;
public class MultiplicationTableExample {
 public static void main(String[] args) {
    for (int m=2; m<=9; m++) {
     System.out.println("*** " + m + "단 ***");
     for (int n=1; n<=9; n++) {
        System.out.println(m + "x" + n + " = " + (m*n));
                *** 2단 ***
                2 \times 1 = 2
                2 \times 2 = 4
                2 \times 3 = 6
                *** 9타 ***
                9 \times 1 = 9
                9 \times 2 = 18
                9 \times 8 = 72
                9 \times 9 = 81
```

조건식에 따라 실행문을 반복하는 while 문

○ 조건식이 true일 경우에 계속해서 반복하고, false가 되면 반복을 멈추고 while 문을 종료



while 문이 처음 실행될 때 ①조건식을 평가. 평가 결과가 true이면 ②실행문을 실행한다.

- ②실행문이 모두 실행되면 조건식으로 되돌아가서
- ①조건식을 다시 평가.

다시 조건식이 true라면 $② \rightarrow ①로 진행하고,$ false라면 while 문을 종료.

조건식에 true를 사용하면 while(true) {...}가 되어서 무한 반복.

→ 이 경우 while 문을 빠져나가기 위한 코드 필요

ch04.sec05.PrintFrom1To10Example.java

```
package ch04.sec05;

public class PrintFrom1To10Example {
  public static void main(String[] args) {
    int i = 1;
    while (i<=10) {
      System.out.print(i + " ");
      i++;
    }
  }
}</pre>
```

```
1 2 3 4 5 6 7 8 9 10
```

ch04.sec05.SumFrom1To100Example.java

```
package ch04.sec05;
public class SumFrom1To100Example {
 public static void main(String[] args) {
   int sum = 0;
   int i = 1;
   while(i<=100) {
     sum += i;
     i++;
   System.out.println("1~" + (i-1) + " 합 : " + sum);
```

```
1~100 합 : 5050
```

ch04.sec05.KeyControlExample.java

```
package ch04.sec05;
                                                                      System.(in out err)
import java.util.Scanner; // Scanner 객체를 사용하기 위해 필요
                                                                      표준 입력 출력 에러 장치
public class KeyControlExample {
 public static void main(String[] args) {
   Scanner scanner = new Scanner(System.in); // 키보드와 Scanner 연결
   boolean run = true; // while 루프 종료 조건 변수
   int speed = 0;
   while(run) {
    System.out.println("----");
    System.out.println("1. 증속 | 2. 감속 | 3. 중지");
    System.out.println("----");
    System.out.print("선택: ");
    String strNum = scanner.nextLine(); // 키보드에서 입력한 내용을 읽음(문자열)
```

ch04.sec05.KeyControlExample.java

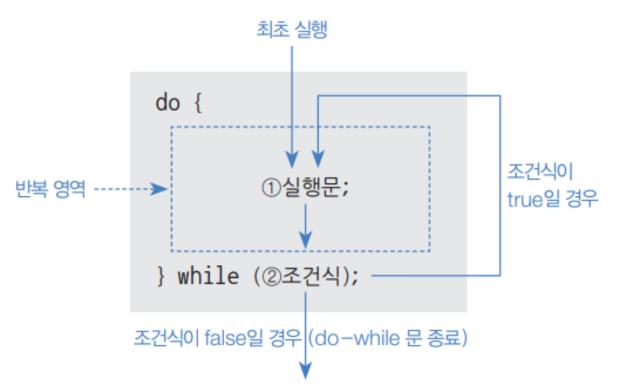
```
if(strNum.equals("1")) {
   speed++;
   System.out.println("현재 속도 = " + speed);
 } else if(strNum.equals("2")) {
   speed--;
   System.out.println("현재 속도 = " + speed);
 } else if(strNum.equals("3")) {
   run = false; // while문의 조건식을 false로
System.out.println("프로그램 종료");
```

```
1. 증속 | 2. 감속 | 3. 중지
선택: 1
현재 속도 = 1
1. 증속 | 2. 감속 | 3. 중지
선택: 2
현재 속도 = 0
1. 증속 | 2. 감속 | 3. 중지
선택: 3
프로그램 종료1
```

6 do-while 문

♡ 실행 결과에 따라 실행문을 반복하는 do-while 문

- 블록 내부를 먼저 실행시키고 실행 결과에 따라서 반복 실행을 계속할지 결정
- 작성 시 while() 뒤에 반드시 세미콜론(;)을 붙여야 하는 데 주의



do-while 문이 처음 실행될 때 ①실행문을 우선 실행한다.

①실행문이 모두 실행되면 ②조건식을 평가

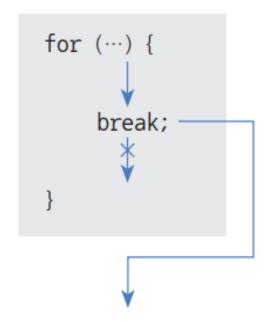
평가 결과가 true이면 ① \rightarrow ②와 같이 반복 실행을 하고, 조건식의 결과가 false이면 do-while 문을 종료한다.

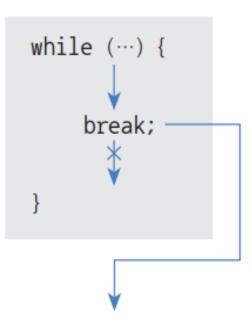
ch04.sec06.DoWhileExample.java

```
package ch04.sec06;
import java.util.Scanner;
public class DoWhileExample {
 public static void main(String[] args) {
   System.out.println("메시지를 입력하세요.");
   System.out.println("프로그램을 종료하려면 q를 입력하세요.");
   Scanner scanner = new Scanner(System.in);
                                                 메시지를 입력하세요.
   String inputString;
                                                 프로그램을 종료하려면 q를 입력하세요.
   do {
                                                 >안녕하세요
    System.out.print(">");
                                                 안녕하세요
    inputString = scanner.nextLine();
                                                 >반값습니다.
    System.out.println(inputString);
                                                 반값습니다.
   } while(! inputString.equals("q"));
                                                 >q
                                                 q
   System.out.println();
                                                 프로그램 종료
   System.out.println("프로그램 종료");
```

☑ 제어문을 종료하는 break 문

- 반복문인 for 문, while 문, do-while 문을 실행 중지하거나 조건문인 switch 문을 종료할 때 사용
- o break 문은 대개 if 문과 같이 사용되어 조건식에 따라 for 문과 while 문을 종료





ch04.sec07.BreakExample.java

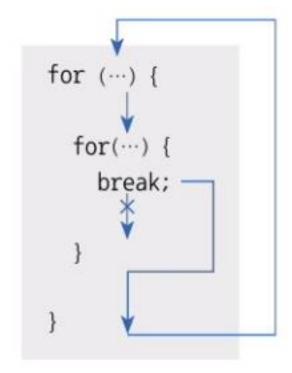
```
package ch04.sec07;
public class BreakExample {
 public static void main(String[] args) throws Exception {
   while(true) {
     int num = (int)(Math.random()*6) + 1;
     System.out.println(num);
     if(num == 6) {
      break;
   System.out.println("프로그램 종료");
```

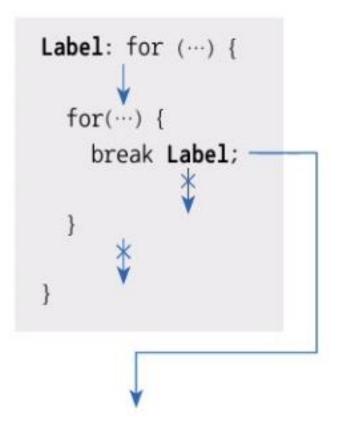
```
4
6
프로그램 종료
```

7 break 문

💟 제어문을 종료하는 break 문

- o break는 현재 루프만 종료
- 중첩 루프문인 경우 안쪽 루프에서 break하면 안쪽 루프 종료
- o 바깥 루프까지 완전히 종료하고 싶은 경우
 - 라벨 사용





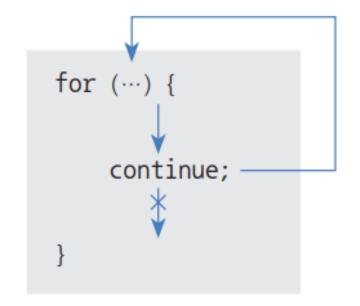
break 문

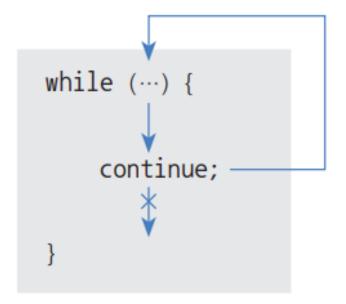
ch04.sec07.BreakOutterExample.java

```
package ch04.sec07;
public class BreakOutterExample {
 public static void main(String[] args) throws Exception {
   Outter: for(char upper='A'; upper<='Z'; upper++) {</pre>
     for(char lower='a'; lower<='z'; lower++) {</pre>
       System.out.println(upper + "-" + lower);
       if(lower=='g') {
        break Outter; -
                                                       A-a
                                                       A-b
                                                       A-c
   System.out.println("프로그램 실행 종료");←
                                                       A-d
                                                       A-e
                                                       A-f
                                                       A-g
                                                       프로그램 실행 종료
```

조건식으로 이동하는 continue 문

- 반복문인 for 문, while 문, do-while 문에서만 사용
- 블록 내부에서 continue 문이 실행되면 for 문의 증감식 또는 while 문, do-while 문의 조건식으로 바로 이동
- o break 문과 달리 반복문을 종료하지 않고 계속 반복을 수행
- 대개 if 문과 같이 사용되며, 특정 조건을 만족하는 경우에 continue 문을 실행해서 그 이후의 문장을 실행하 지 않고 다음 반복으로 넘어감





8 continue 문

ch04.sec08.ContinueExample.java

```
package ch04.sec08;
public class ContinueExample {
 public static void main(String[] args) throws Exception {
   for(int i=1; i<=10; i++) {
     if(i%2 != 0) {
       continue;
     System.out.print(i + " ");
```

2 4 6 8 10