Chapter 04

조건문과 반복문

04-1. 조건문 : if문, switch문

목차

0. 시작하기 전에

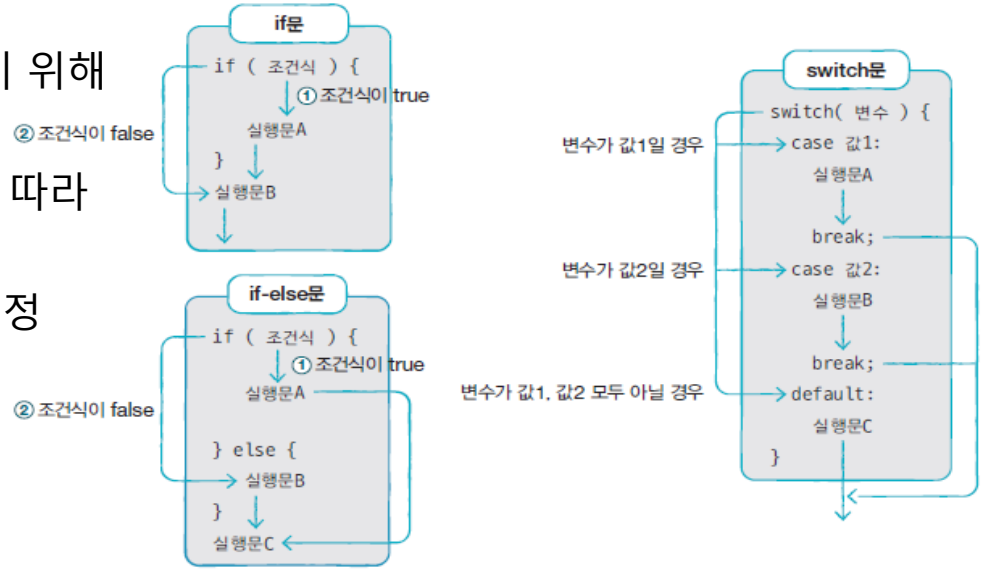
[핵심 키워드] : if문, if-else문, if-else if-else문, switch문

[핵심 포인트]

- 자바 프로그램은 main( ) 메소드의 시작 중괄호 {에서 끝 중괄호 }까지 위에서부터 아래로 실행하는 흐름을 가지고 있다. 이러한 실행 흐름을 개발자가 원하는 방향으로 바꿀 수 있도록 해주는 것을 흐름 제어문 혹은 제어문이라고 한다.
- 제어문의 종류에는 조건문과 반복문이 있다.

❖ 조건문

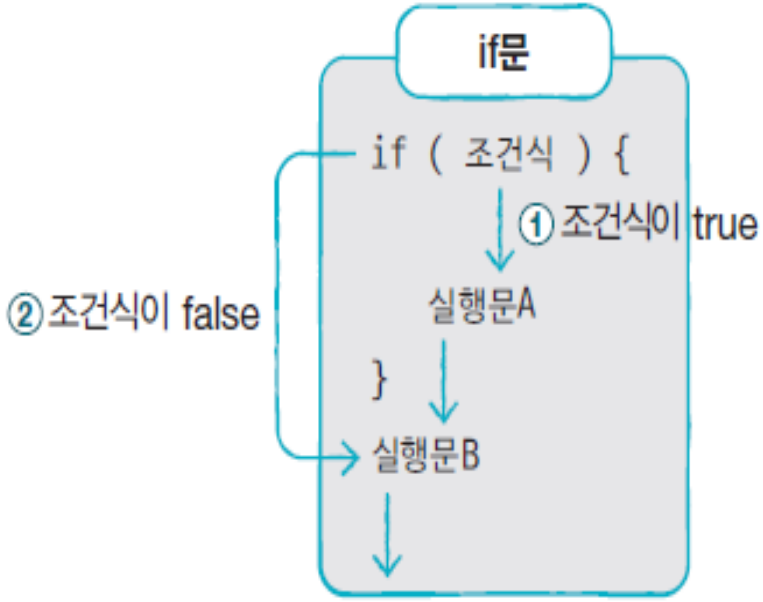
- 조건식에 따라 다른 실행문을 실행하기 위해 사용
- if문 : 조건식 결과의 true. false 여부에 따라 실행문 결정
- switch문 : 변수의 값에 따라 실행문 결정



1. if문

❖ if문

- 조건식 결과에 따라 블록 실행 여부가 결정
- 조건식에 올 수 있는 요소
  - true / false 값을 산출하는 연산식
  - boolean 타입 변수
- 중괄호 블록은 조건식이 true가 될 때 실행
  - 실행할 문장 하나뿐인 경우 생략 가능




```
if ( 조건식 ) {
    실행문;
    실행문;
    ...
}
```

```
if ( 조건식 )
    실행문;
```

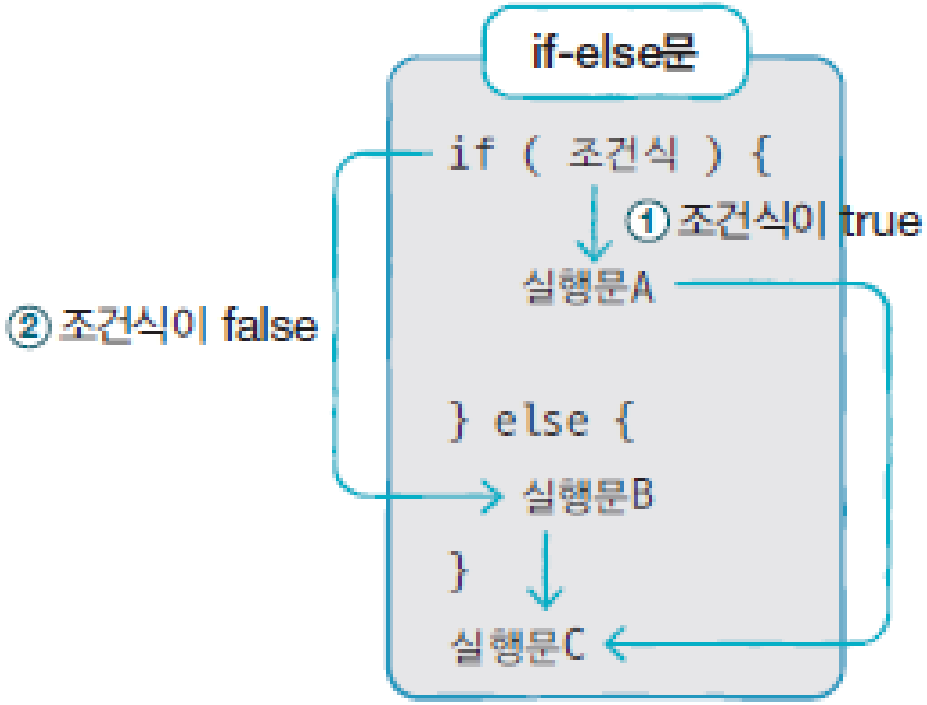
1. if문

```
1  package sec01.exam01;
2
3  public class IfExample {
4      public static void main(String[] args) {
5          int score = 93;
6
7          if(score>=90){
8              System.out.println("점수가 90보다 큽니다.");
9              System.out.println("등급은 A 입니다.");
10         }
11
12         if(score< 90)
13             System.out.println("점수가 90보다 작습니다.");
14             System.out.println("등급은 B 입니다.");
15     }
16 }
```

2. if-else문

❖ if-else문

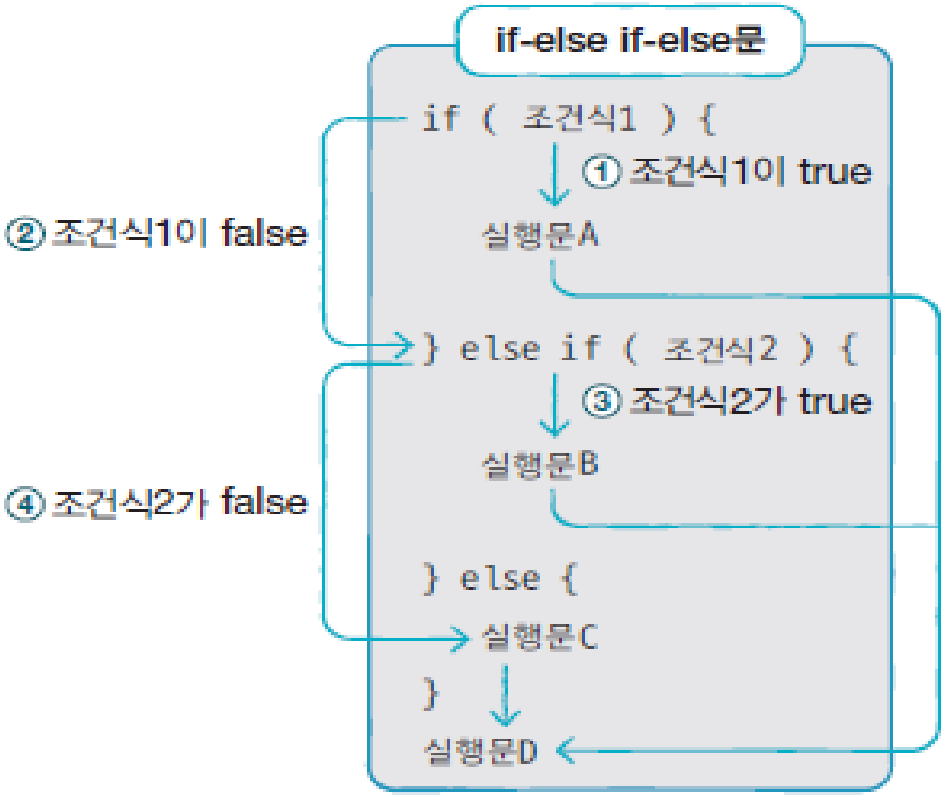
- if문을 else 블록과 함께 사용
- 조건식의 결과에 따라 실행 블록 선택
  - if문 조건식 true이면 if문 블록 실행
  - if문 조건식 false이면 else 블록 실행



3. if-else if-else문

❖ if-else if-else문

- 조건식이 여러 개인 if문
- 처음 if문의 조건식이 false일 경우 다른 조건식의 결과에 따라 실행 블록 선택
  - if 블록 끝에 else if문 추가
  - else if문 개수는 제한 없음



3. if-else if-else문

```
1  package sec01.exam03;
2
3  public class IfElseIfElseExample {
4      public static void main(String[] args) {
5          int score = 75;
6
7          if(score>=90){
8              System.out.println("점수가 100~90 입니다.");
9              System.out.println("등급은 A 입니다.");
10         } else if(score>=80){
11             System.out.println("점수가 80~89 입니다.");
12             System.out.println("등급은 B 입니다.");
13         } else if(score>=70){
14             System.out.println("점수가 70~79 입니다.");
15             System.out.println("등급은 C 입니다.");
16         } else {
17             System.out.println("점수가 70 미만 입니다.");
18             System.out.println("등급은 D 입니다.");
19         }
20     }
21 }
```

3. if-else if-else문

❖if-else if-else문

- 주사위 수 1~6 중에 하나의 수를 무작위로 뽑아 출력하는 프로그램
- 임의의 정수 뽑기 – Math.random() 메소드 활용
  - 이 매소드는 0.0과 1.0 사이의 난수 리턴 : 0.0 <= math.random() <1.0
  - 1~10 사이의 정수 중 하나 얻기 위해서는 *10을 양변에 해 줘야 함
  - 0.0 * 10 <= math.random() * 10 <1.0 * 10 => double 타입
  - int 형으로 변환 => (int) 0.0 <= int(math.random() * 10) < int(10.0)
  - 1을 더해줘야 1~10 범위가 됨 => <= int(math.random() * 10) < 10+1;
  - 위의 원리 응용한 임의의 정수를 하나 얻기 위한 연산
  - => start부터 시작하는 n 개의 정수 중에서 임의의 정수 하나를 얻기 위한 연산

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

  - => 주사위 번호 하나를 뽑기 위해 사용하는 연산

```
int num = (int)(Math.random() * 6) + 1;
```

  - => 로또 번호 뽑기 위한 연산 : `int num = (int)(Math.random() * 45) + 1;`

3. if-else if-else문

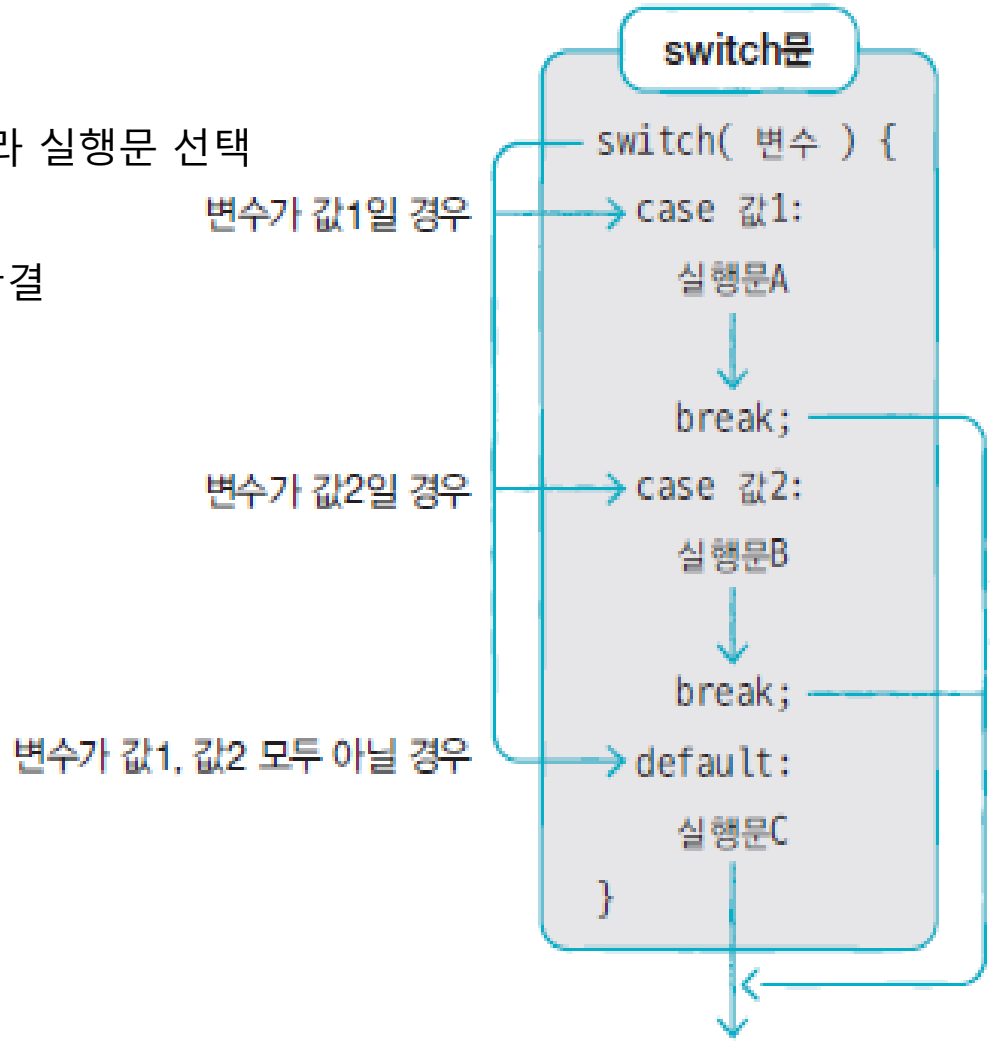
주사위의 번호 뽑기

```
1  package sec01.exam04;
2
3  public class IfDiceExample {
4      public static void main(String[] args) {
5          int num = (int)(Math.random()*6) + 1;
6
7          if(num==1){
8              System.out.println("1번이 나왔습니다.");
9          } else if(num==2){
10             System.out.println("2번이 나왔습니다.");
11         } else if(num==3){
12             System.out.println("3번이 나왔습니다.");
13         } else if(num==4){
14             System.out.println("4번이 나왔습니다.");
15         } else if(num==5){
16             System.out.println("5번이 나왔습니다.");
17         } else {
18             System.out.println("6번이 나왔습니다.");
19         }
20     }
21 }
```

4. swith문

❖ switch문

- 변수가 어떤 값을 갖는가에 따라 실행문 선택
- 같은 기능의 if문보다 코드가 간결



4. swith문

❖ switch문

```
1  package sec01.exam05;
2
3  public class SwitchExample {
4      public static void main(String[] args) {
5          int num = (int)(Math.random()*6) + 1;
6
7          switch(num) {
8              case 1:
9                  System.out.println("1번이 나왔습니다.");
10                 break;
11             case 2:
12                 System.out.println("2번이 나왔습니다.");
13                 break;
14             case 3:
15                 System.out.println("3번이 나왔습니다.");
16                 break;
17             case 4:
18                 System.out.println("4번이 나왔습니다.");
19                 break;
20             case 5:
21                 System.out.println("5번이 나왔습니다.");
22                 break;
23             default:
24                 System.out.println("6번이 나왔습니다.");
25                 break;
26         }
```

4. swith문

- switch문
- break가 없음
- 랜덤한 케이스 다음의 문장들이 다 실행됨

```
1  package sec01.exam06;
2
3  public class SwitchNoBreakCaseExample {
4      public static void main(String[] args) {
5          //8<= ... < 12(8+4) 사이의 정수 얻기
6          int time = (int)(Math.random()*4) + 8;
7          System.out.println("[현재시간: " + time + " 시]");
8
9          switch(time) {
10             case 8:
11                 System.out.println("출근합니다.");
12             case 9:
13                 System.out.println("회의를 합니다.");
14             case 10:
15                 System.out.println("업무를 봅니다.");
16             default:
17                 System.out.println("외근을 나갑니다.");
18         }
19     }
20 }
```

5. 키워드로 끝내는 핵심 포인트

- If문: if(조건식) { ... }을 말하며 조건식이 true가 되면 중괄호 내부를 실행

- If-else문: if(조건식) { ... } else { ... }를 말하며 조건식이 true가 되면 if 중괄호 내부를 실행하고, false가 되면 else 중괄호 내부를 실행

- if-else if-else문: if(조건식1) { ... } else if(조건식2) { ... } else { ... }를 말하며 조건식1이 true가 되면 if 중괄호 내부를 실행하고, 조건식2가 true가 되면 else if 중괄호 내부를 실행한다. 조건식1과 조건식2가 모두 false가 되면 else 중괄호 내부가 실행된다.

- switch문 : switch(변수) { case 값1: ... case 값2: ... default: ... }를 말하며 변수의 값이 값1이면 첫 번째 case 코드를 실행하고, 값2이면 두 번째 case 코드를 실행한다. 값1과 값2가 모두 아니면 default 코드를 실행한다.

Chapter 04

조건문과 반복문

04-2. 반복문: for문, while문, do-while문

- 시작하기 전에
- for문
- while문
- do-while문
- break문
- continue문
- 키워드로 끝내는 핵심 포인트

시작하기 전에

[핵심 키워드] : for문, while문, do-while문, break문, continue문

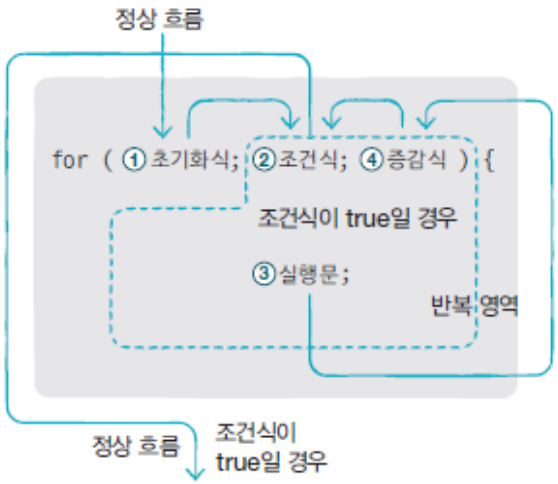
[핵심 포인트]

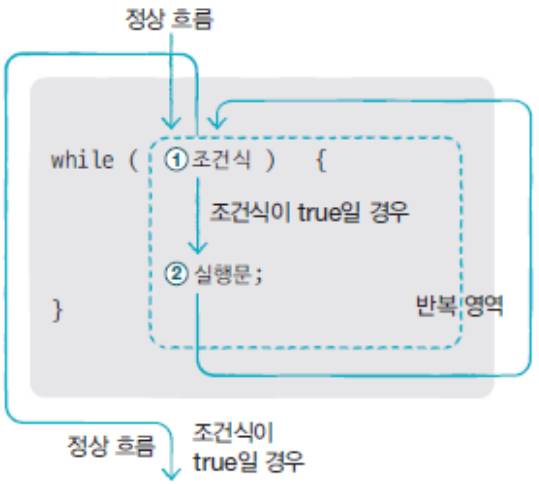
반복문에는 for문, while문, do-while문 있다.

반복문은 제어문 처음으로 되돌아가 반복 실행하는데 이것을 루핑(looping)이라고 한다.

❖ 반복문

- 어떤 작업을 반복적으로 실행하고 싶을 때 사용





- for문, while문, do-while문
  - for문은 반복 횟수 알고 있을 때 주로 사용
  - while문은 조건에 따라 반복할 때 주로 사용
  - do-while문은 while 문과 유사하나 조건을 나중에 검사

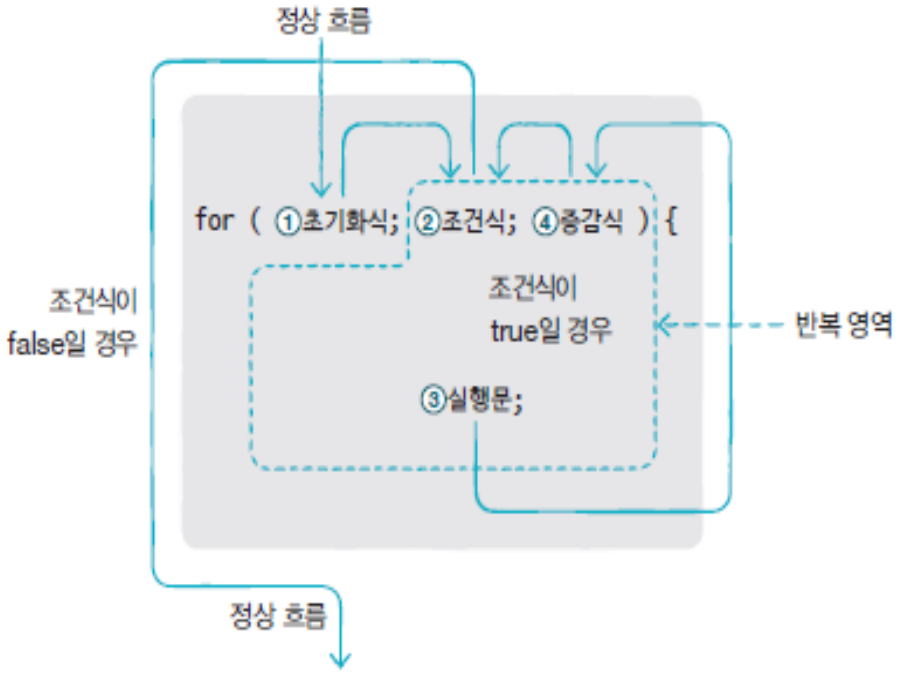
for문

❖ for문

- 주어진 횟수만큼 반복하고 싶을 경우 ㅅ

```
int sum = 0;
sum = sum + 1;
sum = sum + 2;
sum = sum + 3;
sum = sum + 4;
sum = sum + 5;
System.out.println("1~5의 합:" + sum);
```

5개의 실행문




```
int sum = 0;
for (int i=1; i<=100; i++) {
  sum = sum + i;
}
System.out.println("1~100의 합:" + sum);
```

100번 반복

for문

❖ 1부터 10까지 출력

```
1  package sec02.exam01;
2
3  public class ForPrintFrom1To10Example {
4      public static void main(String[] args) {
5          for(int i=1; i<=10; i++) {
6              System.out.println(i);
7          }
8      }
9  }
```

❖ 문제 : 1부터 100까지 합을 출력하시오.
힌트 : 합을 저장할 변수(sum) 이용
합에 이전합과 증가값을 더함...

for문

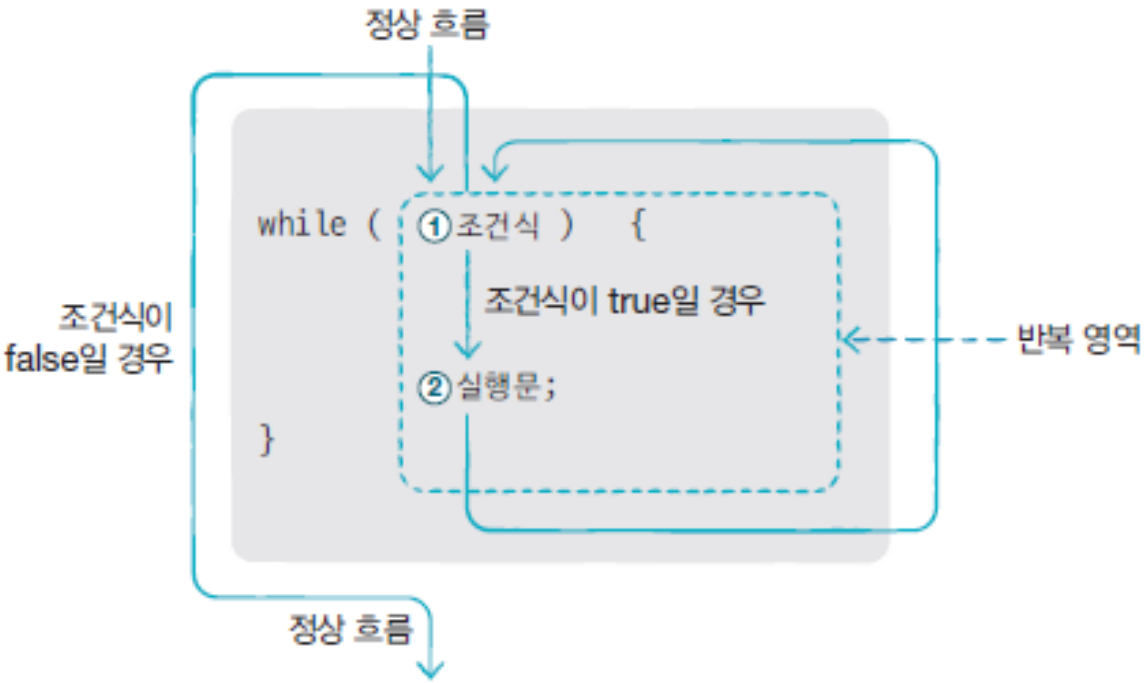
❖ 구구단 출력하기(중첩 for 문)

```
1  package sec02.exam05;
2
3  public class ForMultiplicationTableExample {
4      public static void main(String[] args) {
5          for (int m=2; m<=9; m++) {
6              System.out.println("*** " + m + "단 ***");
7              for (int n=1; n<=9; n++) {
8                  System.out.println(m + " x " + n + " = " + (m*n));
9              }
10         }
11     }
12 }
```

while문

❖ while문

- 조건식에 따라 반복 여부를 결정할 경우
  - true일 경우 계속해서 반복
  - false일 경우 반복 종료
- 조건식에는 주로 비교 연산식, 논리 연산식



while문

❖ 1부터 10까지 출력

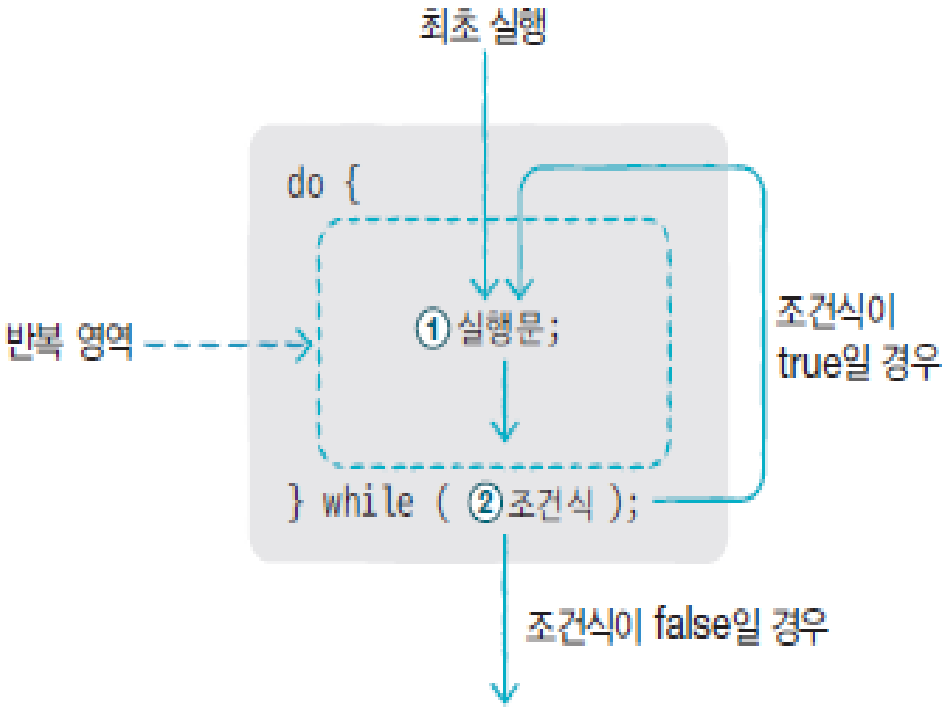
```
1  package sec02.exam06;
2
3  public class WhilePrintFrom1To10Example {
4      public static void main(String[] args) {
5          int i = 1;
6          while (i<=10){
7              System.out.println(i);
8              i++;
9          }
10     }
11 }
```

❖ while 문을 이용해 1부터 100까지 출력해 보세요.

do-while문

❖ do-while문

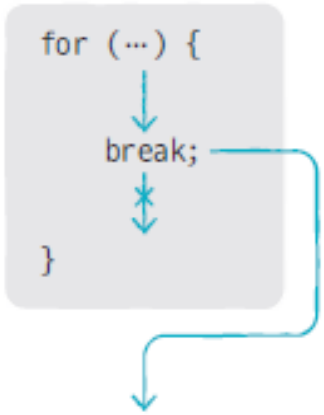
- 조건식에 의해 반복 실행하는 점에서 while문과 동일
- 블록 내부 실행문을 우선 실행하고 그 결과에 따라 반복 실행 여부를 결정

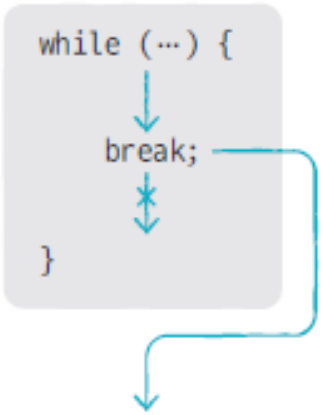


break문

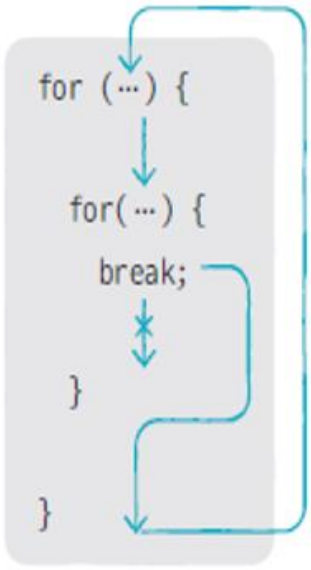
❖ break문

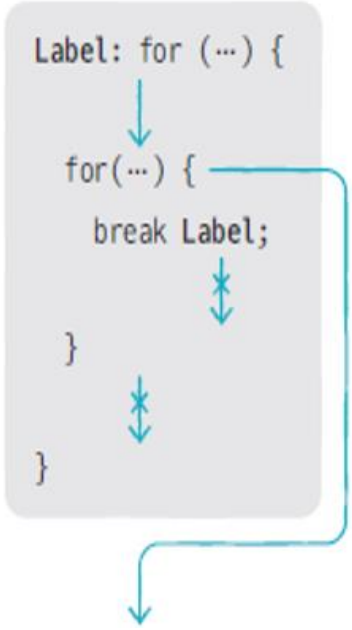
- for, while, do-while, switch문의 실행을 중지할 때 사용
- 주로 if문과 함께 사용





- 반복문이 중첩되어 있을 경우
  - Label을 이용해서 바깥 반복문을 빠져나감





break문

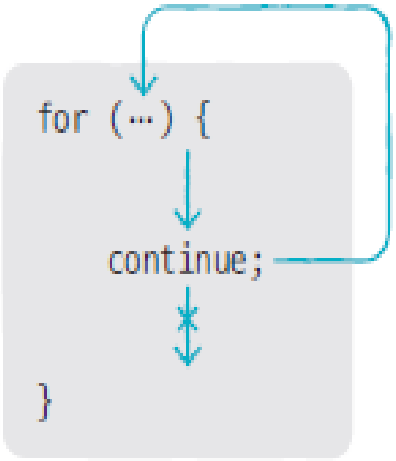
❖ break를 이용해 while 문 종료

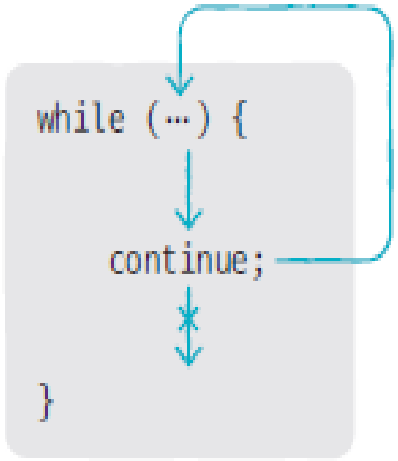
```
1  package sec02.exam08;
2
3  public class BreakExample {
4      public static void main(String[] args) throws Exception {
5          while(true) {
6              int num = (int)(Math.random()*6) + 1;
7              System.out.println(num);
8              if(num == 6){
9                  break;
10             }
11         }
12         System.out.println("프로그램 종료");
13     }
14 }
```

continue문

❖ continue문

- for, while, do-while문에서만 사용
- for문의 증감식이나 while, do-while문의 조건식으로 이동
- 주로 if문과 함께 사용





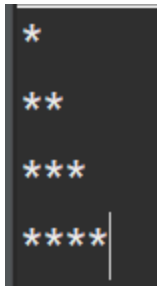
continue문

❖ continue 사용한 for 문

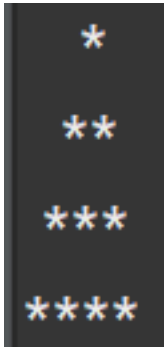
```
1  package sec02.exam10;
2
3  public  class    ContinueExample  {
4          public  static  void  main(String[]  args)  throws  Exception  {
5                  for(int  i=1;  i<=10;  i++)  {
6                          if(i%2  !=  0){
7                                  continue;
8                          }
9                          System.out.println(i);
10                 }
11         }
12 }
```

실습(연습문제)

❖ for 문을 이용해서 다음과 같이 *을 출력하는 코드를 작성해 보세요.



❖ for 문을 이용해서 다음과 같이 *을 출력하는 코드를 작성해 보세요.



키워드로 끝내는 핵심 포인트

- for문: for(초기화식; 조건식; 증감식;) ~ 을 말하며 지정한 횟수만큼만 반복할 때 주로 사용
- while문: while(조건식;) ~ 을 말하며 조건식이 true가 될 때까지 반복 실행
- do-while문 : do ~ while(조건식); 을 말하며 while문과 동일하나 조건식이 뒤에 있음
- break문 : for문, while문, do-while문의 반복을 종료할 때 사용
- continue문 : for문, while문, do-while문의 증감식 또는 조건식으로 돌아감

Thank You!