

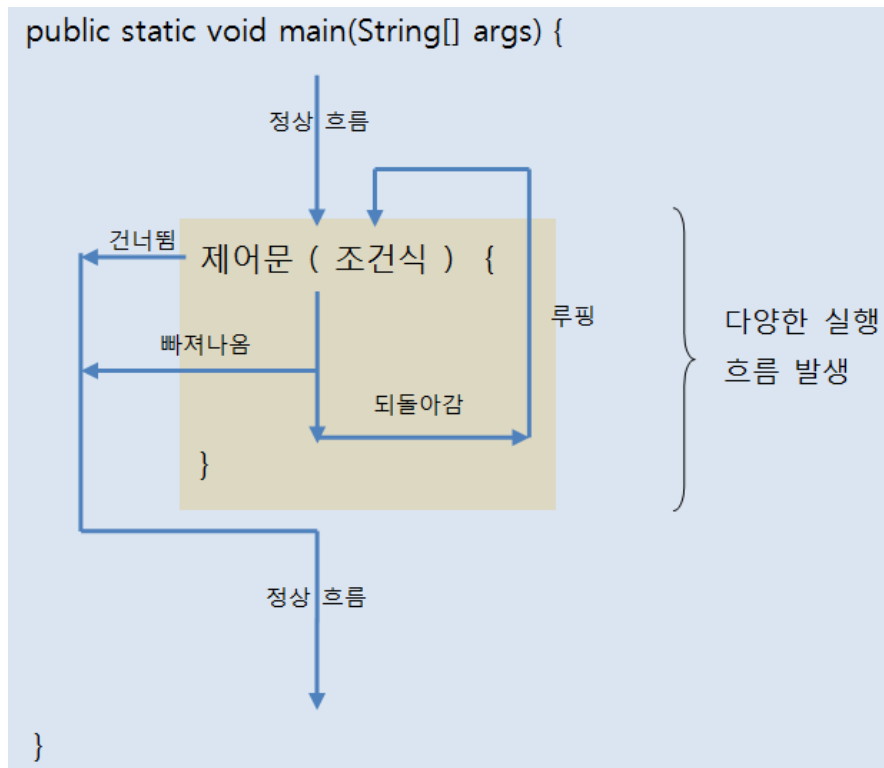
■ 코드 실행 흐름

● 정상적인 코드 실행 흐름

- main() 메소드의 시작인 중괄호 { 에서 끝 중괄호 } 까지
위 ➔ 아래 방향으로 실행

● 제어문의 역할

- 코드 실행 흐름을 개발자가 원하는 방향으로 변경할 수 있도록 도와줌



■ 제어문 종류

● 조건문

- if문, switch문

● 반복문

- for문, while문, do-while문

● break문, continue문

■ 제어문의 중첩

● 제어문의 중괄호 내부에 다른 제어문 작성 가능

- 다양한 흐름 발생 가능

■ 조건문

- 자바에서 조건문은 if / switch 두가지
- if문이 주로 사용되며, 경우의 수가 많고 단일 값인 경우 switch 사용을 고려
- 모든 switch 문은 if 문으로 변경 가능하지만,
if 문을 switch 문으로 변경할 수 없는 경우가 많음

```
if(dice >= 4) {  
    System.out.println("이김");  
} else {  
    System.out.println("짐");  
}
```

```
switch(dice) {  
    case 6 :  
        System.out.println("이김");  
        break;  
    case 5 :  
        System.out.println("이김");  
        break;  
    case 4 :  
        System.out.println("이김");  
        break;  
    case 3 :  
        System.out.println("짐");  
        break;  
    case 2 :  
        System.out.println("짐");  
        break;  
    case 1 :  
        System.out.println("짐");  
        break;  
}
```

■ if

- 조건식 결과 따라 중괄호 { } 블록을 실행할지 여부 결정할 때 사용
- 조건식
 - true 또는 false값을 산출할 수 있는 연산식
 - boolean 변수
 - 조건식이 true이면 블록 실행하고 false 이면 블록 실행하지 않음

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

```
if ( 조건식 )  
    실행문;
```

■ if

```
boolean isExist = true;
if(isExist) {
    System.out.println("파일삭제");
}
```

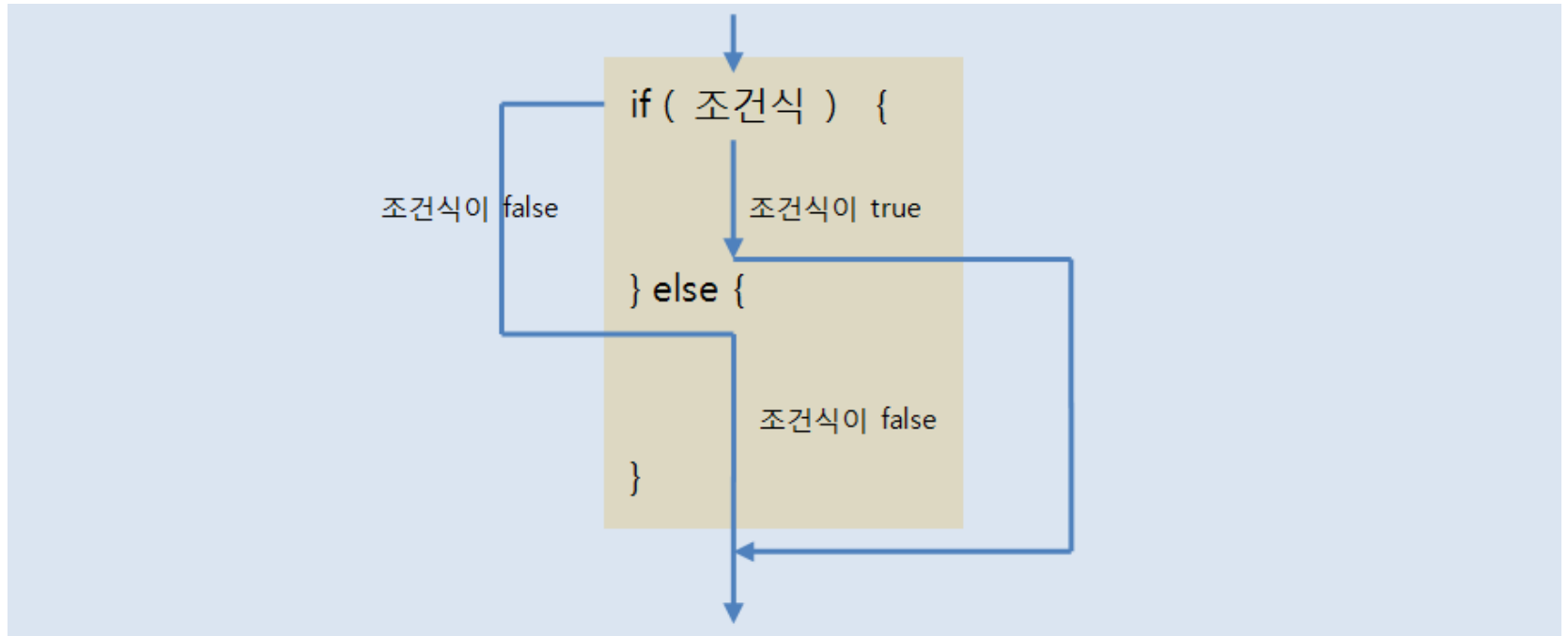
```
char gender = 'F';
if(gender == 'F') {
    System.out.println("여자");
}
```

```
int age = 28;
if(age >= 20 && age < 30) {
    System.out.println("20대");
}
```

```
int score = 61;
int cutline = 60;
if(score >= cutline) {
    System.out.println("점수 통과");
}
```

■ if-else

- 조건식 결과 따라 실행 블록 선택



■ if-else

```
boolean isExist = true;
if(isExist) {
    System.out.println("파일삭제");
} else {
    System.out.println("삭제실패");
}
```

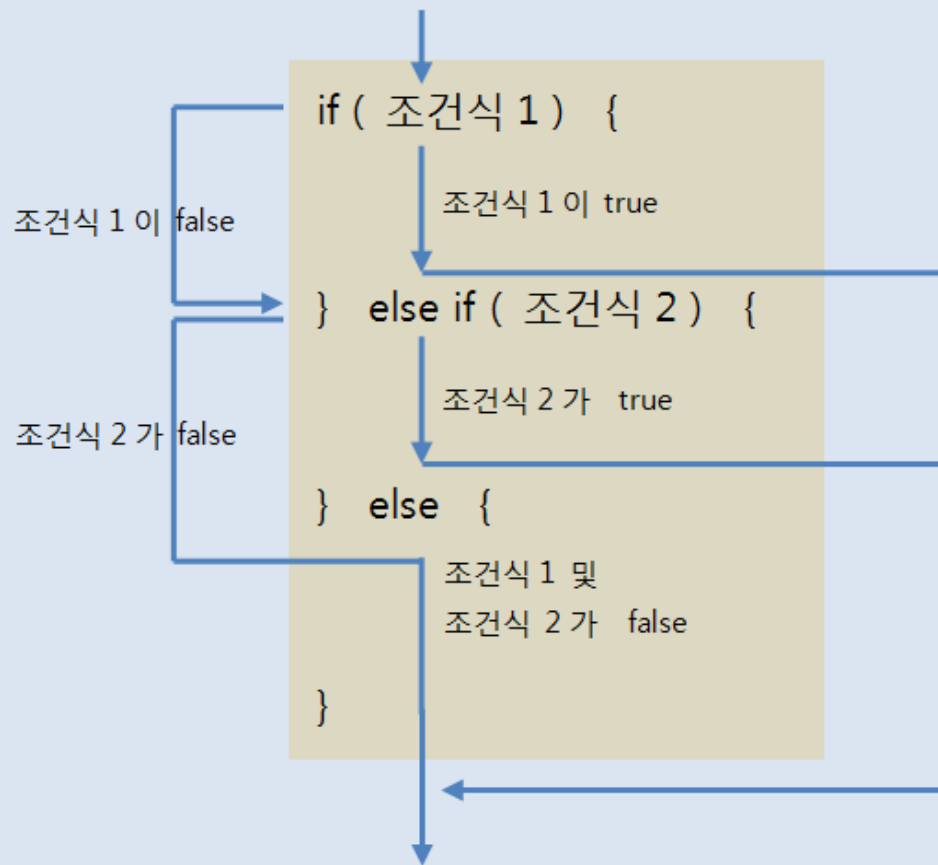
```
char gender = 'F';
if(gender == 'F') {
    System.out.println("여자");
} else {
    System.out.println("남자");
}
```

```
int age = 28;
if(age >= 20) {
    System.out.println("20대 이상");
} else {
    System.out.println("20대 미만");
}
```

```
int score = 61;
int cutline = 60;
if(score >= cutline) {
    System.out.println("합격");
} else {
    System.out.println("불합격");
}
```

■ if-else if-else

- 복수의 조건식 두어 조건식을 만족하는 블록만 실행



■ if-else

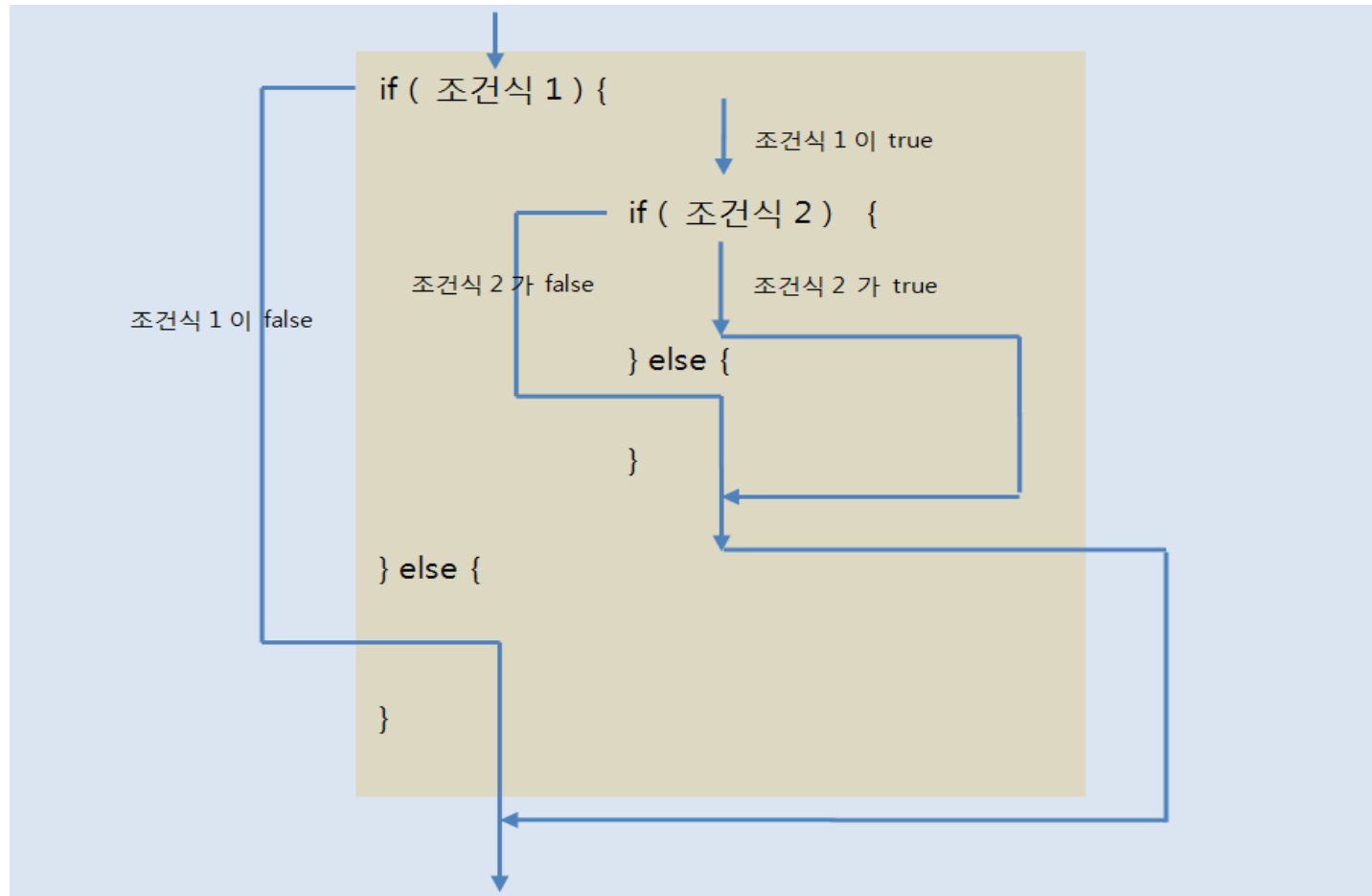
```
int score = 87;
if(score >= 90) {
    System.out.println("A");
} else if(score >= 80) {
    System.out.println("B");
} else if(score >= 70) {
    System.out.println("C");
} else if(score >= 60) {
    System.out.println("D");
} else {
    System.out.println("F");
}
```

```
int score = 87;
if(score >= 60) {
    System.out.println("D");
} else if(score >= 70) {
    System.out.println("C");
} else if(score >= 80) {
    System.out.println("B");
} else if(score >= 90) {
    System.out.println("A");
} else {
    System.out.println("F");
}
```

오류

■ 중첩 if

- 코드의 실행 흐름을 이해하는 것이 가장 중요



■ 중첩 if

```
int score = 95;
if(score >= 90) {
    if(score >= 95) {
        System.out.println("A+");
    } else {
        System.out.println("A");
    }
} else if(score >= 80) {
    System.out.println("B");
} else if(score >= 70) {
    System.out.println("C");
} else if(score >= 60) {
    System.out.println("D");
} else {
    System.out.println("F");
}
```

```
int number = -3;
if(number > 10) {
    System.out.println("10 초과");
} else if(number > 5) {
    System.out.println("5 초과");
} else {
    if(number >= 0) {
        System.out.println("양수");
    } else {
        System.out.println("음수");
    }
}
```

■ if 사용 - 1

```
public class IfExam1 {  
    public static void main(String[] args) {  
        int num = -1;  
  
        if (num > 10) {  
            System.out.println("10보다 큽니다.");  
        } else if (num > 5) {  
            System.out.println("5보다 큽니다.");  
        } else if (num == 5) {  
            System.out.println("5와 같습니다.");  
        } else {  
            if (num < 0) {  
                System.out.println("음수입니다.");  
            } else if (num == 0) {  
                System.out.println("0입니다.");  
            } else {  
                System.out.println("5보다 작은 양수입니다.");  
            }  
        }  
    }  
}
```

■ if 사용 - 2

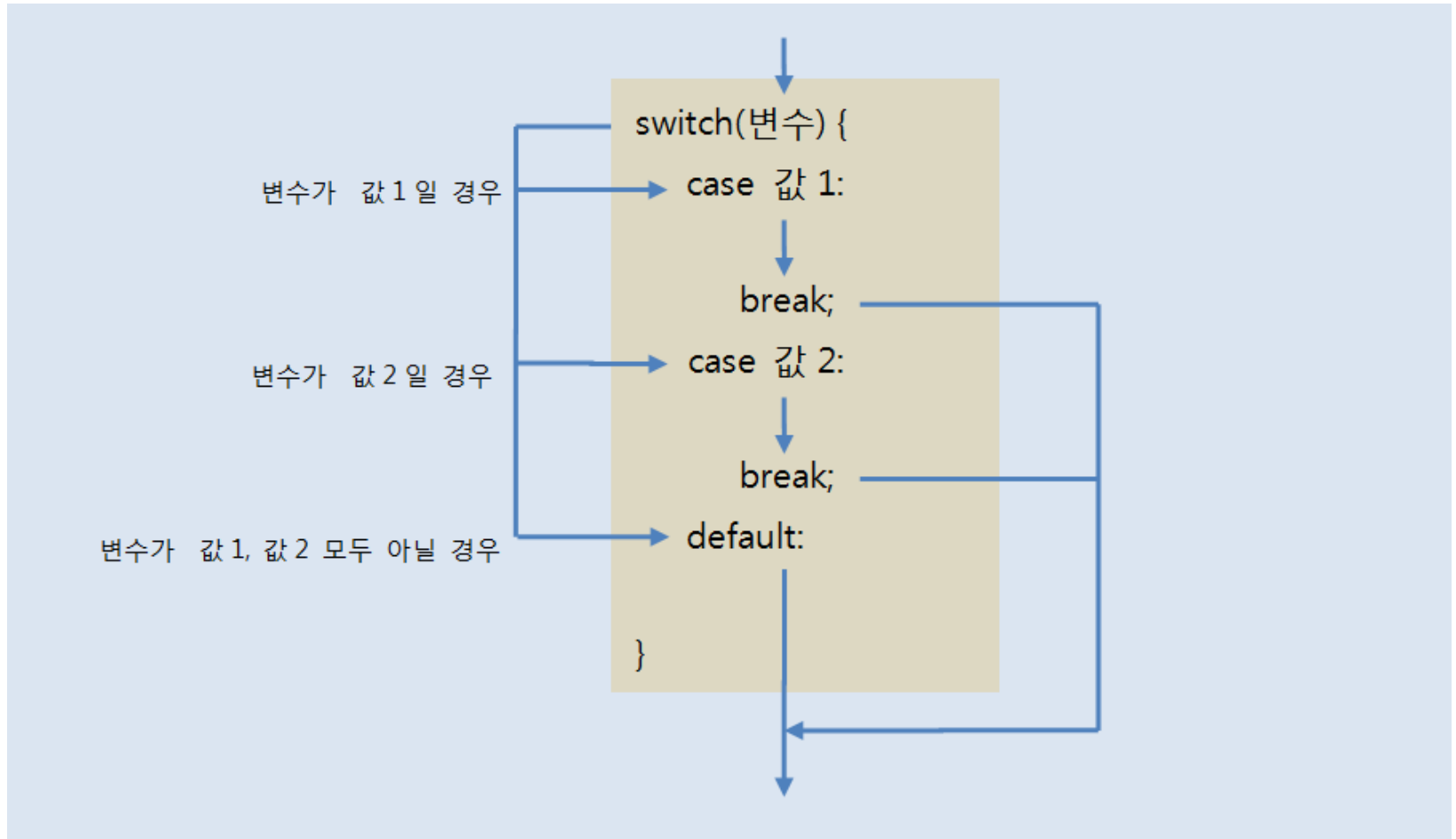
```
public class IfExam2 {  
    public static void main(String[] args) {  
        int a = 10;  
        int b = 20;  
  
        if(a++ == 10 && a == 11) {  
            System.out.println("1번");  
        }  
  
        System.out.println("a : " + a);  
  
        if(--a == 10 || ++b == 21) {  
            System.out.println("2번");  
        }  
  
        System.out.println("a : " + a);  
        System.out.println("b : " + b);  
    }  
}
```

■ if 사용 - 3

```
public class IfExam3 {  
    public static void main(String[] args) {  
        Scanner scan = new Scanner(System.in);  
        System.out.println("두 수를 입력해주세요. ex) 10 20");  
        int num1 = scan.nextInt();  
        int num2 = scan.nextInt();  
  
        System.out.println("입력결과");  
        System.out.println("첫번째 숫자 => " + num1);  
        System.out.println("두번째 숫자 => " + num2);  
  
        if(num1 > num2) {  
            System.out.println("첫번째 숫자가 더 큼니다.");  
        } else if(num1 < num2){  
            System.out.println("두번째 숫자가 더 큼니다.");  
        } else {  
            System.out.println("두 수가 같습니다.");  
        }  
    }  
}
```

■ switch

- 변수나 연산식의 값에 따라 실행문 선택할 때 사용



■ switch 사용 - 1 (정수)

```
public class SwitchExam1 {  
    public static void main(String[] args) {  
        int num = 1;  
  
        switch (num) {  
            case 1:  
                System.out.println("[1] 1입니다.");  
                break;  
            case 2:  
                System.out.println("[2] 2입니다.");  
                break;  
            default:  
                System.out.println("[3] 1, 2 아님");  
                break;  
        }  
    }  
}
```

```
        System.out.println("break가 없는 경우");  
  
        switch (num) {  
            case 1:  
                System.out.println("[4] 1입니다.");  
            case 2:  
                System.out.println("[5] 여기도 실행");  
            default:  
                System.out.println("[6] switch 종료");  
                break;  
        }  
    }  
}
```


■ switch 사용 - 2 (문자열)

```
public class SwitchExam2 {  
    public static void main(String[] args) {  
        String alphabet = "A";  
  
        switch (alphabet) {  
            case "A" : case "a" :  
                System.out.println("입력된 값은 A");  
                break;  
            case "B" :  
                System.out.println("입력된 값은 B");  
                break;  
            default:  
                System.out.println("A도 아니고 B도 아님");  
                break;  
        }  
    }  
}
```

■ switch 사용 - 3 (열거형)

```
public class SwitchExam3 {  
    public static void main(String[] args) {  
        Week w = Week.MON;  
  
        switch(w) {  
            case MON :  
                System.out.println("월요일");  
                break;  
            case TUE :  
                System.out.println("화요일");  
                break;  
        }  
    }  
}  
  
enum Week {  
    MON, TUE, WED, THU, FRI, SAT, SUN  
}
```

■ switch 사용 - 4

```
public class SwitchExam4 {  
    public static void main(String[] args) throws IOException {  
        System.out.println("한 글자만 입력해주세요.");  
        int num = System.in.read();  
        num = num - 48;  
  
        switch(num) {  
            case 0: case 1: case 2: case 3: case 4:  
            case 5: case 6: case 7: case 8: case 9:  
                System.out.println("0 ~ 9 숫자 입력!");  
                break;  
            default :  
                System.out.println("문자 입력!");  
        }  
    }  
}
```

■ 반복문

- 문장 또는 문장들을 반복해서 수행할 때 사용
- 반복횟수가 중요한 경우 for문 사용
- 반복횟수를 모르는 경우 while문 사용
- for문과 while문은 서로 변경 가능
- do-while 문은 while 문의 변형으로 최소 한번은 수행될 것을 보장

```
System.out.println(1);  
System.out.println(2);  
System.out.println(3);  
System.out.println(4);  
System.out.println(5);
```

```
int i=0;
```

```
do {  
    i++;  
    System.out.println(i);  
} while(i<=5);
```

```
for(int i=1;i<=5;i++) {  
    System.out.println(i);  
}
```

```
int i=1;
```

```
while(i<=5) {  
    System.out.println(i);  
    i++;  
}
```

■ for

- 초기화, 조건식, 증감식 그리고 수행할 블록{} 또는 문장으로 구성

```
for (초기화;조건식;증감식) {  
    // 조건식이 true일 때 수행될 문장들을 적는다.  
}
```

[참고] 반복하려는 문장이 단 하나일 때는 중괄호{}를 생략할 수 있다.



예) 1부터 10까지의 정수를 더하기

```
int sum = 0;  
  
for(int i=1; i<=10; i++) {  
    sum += i; // sum = sum + i;  
}
```

i	sum
1	
2	
3	
4	
...	
10	

■ for 사용 – 1

```
public class ForExam1 {  
    public static void main(String[] args) {  
        for(int i = 1; i <= 10; i++) {  
            System.out.print(i);  
        }  
  
        System.out.println();  
        System.out.println("=====");  
  
        for(int i = 10; i >= 1; i--) {  
            System.out.print(i);  
        }  
    }  
}
```

```
12345678910  
=====  
10987654321
```

■ for 사용 - 2

```
public class ForExam2 {  
    public static void main(String[] args) {  
        int sum = 0;  
        for(int i = 1; i <= 10; i++) {  
            sum = sum + i;  
            System.out.print("i의 값 => " + i);  
            System.out.println(" sum의 값 => " + sum);  
        }  
    }  
}
```

i의 값 => 1	sum의 값 => 1
i의 값 => 2	sum의 값 => 3
i의 값 => 3	sum의 값 => 6
i의 값 => 4	sum의 값 => 10
i의 값 => 5	sum의 값 => 15
i의 값 => 6	sum의 값 => 21
i의 값 => 7	sum의 값 => 28
i의 값 => 8	sum의 값 => 36
i의 값 => 9	sum의 값 => 45
i의 값 => 10	sum의 값 => 55

■ for 사용 – 3

```
public class ForExam3 {  
    public static void main(String[] args) {  
        int sum = 0;  
        for(int i = 1; i <= 10; i = i + 2) {  
            sum = sum + i;  
            System.out.print("i의 값 => " + i);  
            System.out.println(" sum의 값 => " + sum);  
        }  
    }  
}
```

i의 값 => 1	sum의 값 => 1
i의 값 => 3	sum의 값 => 4
i의 값 => 5	sum의 값 => 9
i의 값 => 7	sum의 값 => 16
i의 값 => 9	sum의 값 => 25

■ for 사용 - 4

```
public class ForExam4 {  
    public static void main(String[] args) {  
        int sum = 0;  
        for(int i = 1; i <= 10; i++) {  
            if(i % 2 != 0) {  
                sum += i; // sum = sum + i;  
                System.out.print("i의 값 => " + i);  
                System.out.println(" sum의 값 => " + sum);  
            }  
        }  
        System.out.println("최종 sum의 값 => " + sum);  
    }  
}
```

```
i의 값 => 1 sum의 값 => 1  
i의 값 => 3 sum의 값 => 4  
i의 값 => 5 sum의 값 => 9  
i의 값 => 7 sum의 값 => 16  
i의 값 => 9 sum의 값 => 25  
최종 sum의 값 => 25
```

■ 중첩 for

- for문 안에 또 다른 for문을 포함시킬 수 있다.

```
for(int i=2; i<=9; i++) {  
    for(int j=1; j<=9; j++) {  
        System.out.println(i+" * "+j+" = "+i*j);  
    }  
}
```

```
for(int i=2; i<=9; i++)  
    for(int j=1; j<=9; j++)  
        System.out.println(i+" * "+j+" = "+i*j);
```

```
for(int i=1; i<=3; i++) {  
    for(int j=1; j<=3; j++) {  
        for(int k=1; k<=3; k++) {  
            System.out.println(""+i+j+k);  
        }  
    }  
}
```

```
for(int i=1; i<=3; i++)  
    for(int j=1; j<=3; j++)  
        for(int k=1; k<=3; k++)  
            System.out.println(""+i+j+k);
```

■ 중첩 for 사용 - 1

```
public class ForExam5 {  
    public static void main(String[] args) {  
        for(int i = 1; i <= 3; i++) {  
            System.out.println("i => " + i);  
  
            for(int j = 1; j <= 2; j++) {  
                System.out.println("    j => " + j);  
            }  
        }  
    }  
}
```

i => 1

j => 1

j => 2

i => 2

j => 1

j => 2

i => 3

j => 1

j => 2

■ 중첩 for 사용 - 1 (구구단)

```
public class Gugudan1 {  
    public static void main(String[] args) {  
        for(int i = 2; i <= 9; i++) {  
            for(int j = 1; j <= 9; j++) {  
                System.out.println(i + " * " + j + " = " + (i * j));  
            }  
        }  
    }  
}
```

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

3	*	1	=	3
3	*	2	=	6
3	*	3	=	9
3	*	4	=	12
3	*	5	=	15
3	*	6	=	18
3	*	7	=	21
3	*	8	=	24
3	*	9	=	27

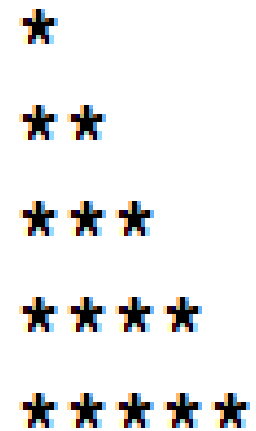
■ 중첩 for 사용 - 2 (구구단)

```
public class Gugudan2 {  
    public static void main(String[] args) {  
        for(int i = 1; i <= 9; i++) {  
            for(int j = 2; j <= 9; j++) {  
                System.out.print(j + " * " + i + " = " + (j * i) + "Wt");  
            }  
            System.out.println();  
        }  
    }  
}
```

2 * 1 = 2	3 * 1 = 3	4 * 1 = 4
2 * 2 = 4	3 * 2 = 6	4 * 2 = 8
2 * 3 = 6	3 * 3 = 9	4 * 3 = 12
2 * 4 = 8	3 * 4 = 12	4 * 4 = 16
2 * 5 = 10	3 * 5 = 15	4 * 5 = 20
2 * 6 = 12	3 * 6 = 18	4 * 6 = 24
2 * 7 = 14	3 * 7 = 21	4 * 7 = 28
2 * 8 = 16	3 * 8 = 24	4 * 8 = 32
2 * 9 = 18	3 * 9 = 27	4 * 9 = 36

■ 중첩 for 사용 – 3 (피라미드 별모양)

```
public class Star1 {  
    public static void main(String[] args) {  
        for (int a = 1; a <= 5; a++) {  
            for (int b = 1; b <= a; b++) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
    }  
}
```



A 5x5 pyramid of stars, where each row contains a number of stars equal to its row index (1 to 5). The stars are arranged in a right-angled triangle shape.

```
★  
★ ★  
★ ★ ★  
★ ★ ★ ★  
★ ★ ★ ★ ★
```

■ 중첩 for 사용 - 4 (피라미드 별모양)

```
public class Star2 {  
    public static void main(String[] args) {  
        for (int a = 5; a >= 1; a--) {  
            for (int b = a; b >= 1; b--) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
    }  
}
```

```
*****  
****  
***  
**  
*
```

■ while

- 조건식과 수행할 블록{} 또는 문장으로 구성

```
while (조건식) {  
    // 조건식의 연산결과가 true일 때 수행될 문장들을 적는다.  
}
```

```
int i=10;  
while(i >= 0) {  
    System.out.println(i--);  
}
```



```
for(int i=10;i>=0;i--) {  
    System.out.println(i);  
}
```


■ while 사용 - 1 (각 자리수 합 구하기)

```
public class WhileExam1 {  
    public static void main(String[] args) {  
        int num = 12345;  
  
        int total = 0;  
  
        while(num > 0) {  
            int n = num % 10;  
  
            total = total + n;  
  
            num = num / 10;  
        }  
  
        System.out.println("각 자리 숫자의 합 : " + total);  
    }  
}
```

■ while 사용 - 2 (랜덤값을 이용한 가위바위보)

```
public class WhileExam2 {  
    public static void main(String[] args) {  
        boolean isContinue = true;  
  
        Scanner scan = new Scanner(System.in);  
  
        while(isContinue) {  
            System.out.println("가위0, 바위1, 보2");  
            System.out.print("숫자 입력 > ");  
            int player = scan.nextInt();  
            int com = (int) (Math.random() * 3);  
            System.out.println(player + "/" + com);
```

```
                if(player == com) {  
                    System.out.println("비김");  
                } else {  
                    if((player + 1) % 3 == com) {  
                        System.out.println("짐");  
                    } else {  
                        System.out.println("이김");  
                        isContinue = false;  
                    }  
                }  
            }  
            scan.close();  
        }  
    }  
}
```

■ 중첩 while

- while문 안에 또 다른 while문을 포함시킬 수 있다.

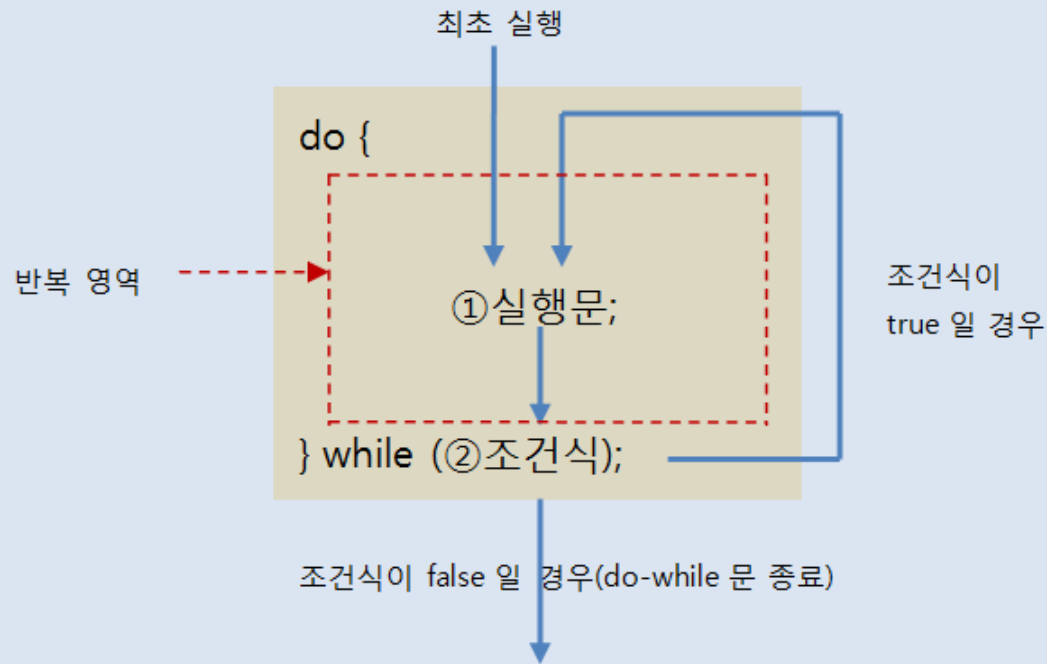
```
for(int i=2; i<=9; i++) {  
    for(int j=1; j<=9; j++) {  
        System.out.println(i+" * "+j+" = "+i*j);  
    }  
}
```



```
int i=2;  
while(i <= 9) {  
    int j=1;  
    while(j <= 9) {  
        System.out.println(i+" * "+j+" = "+i*j);  
        j++;  
    }  
    i++;  
}
```

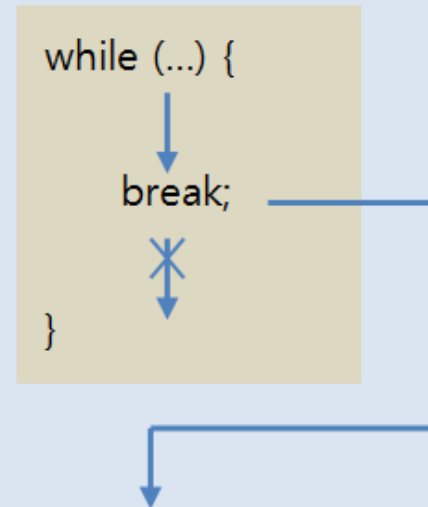
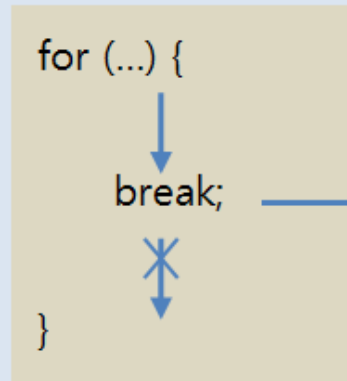
■ do-while

- while문의 변형. 블록{}을 먼저 수행한 다음에 조건식을 계산한다.
- 블록{}이 최소한 1번 이상 수행될 것을 보장한다.



■ break

- 반복문 (for, while, do-while) 종료
- switch 종료
- 대개 if 과 같이 사용
 - 조건식에 따라 반복문을 종료할때 사용



■ break 사용 – 1

```
public class BreakExam1 {  
    public static void main(String[] args) {  
        for(int i = 0; i < 10; i++) {  
            System.out.println("현재값 : " + i);  
            if(i == 5) {  
                System.out.println("5가 되어 종료!");  
                break;  
            }  
        }  
    }  
}
```

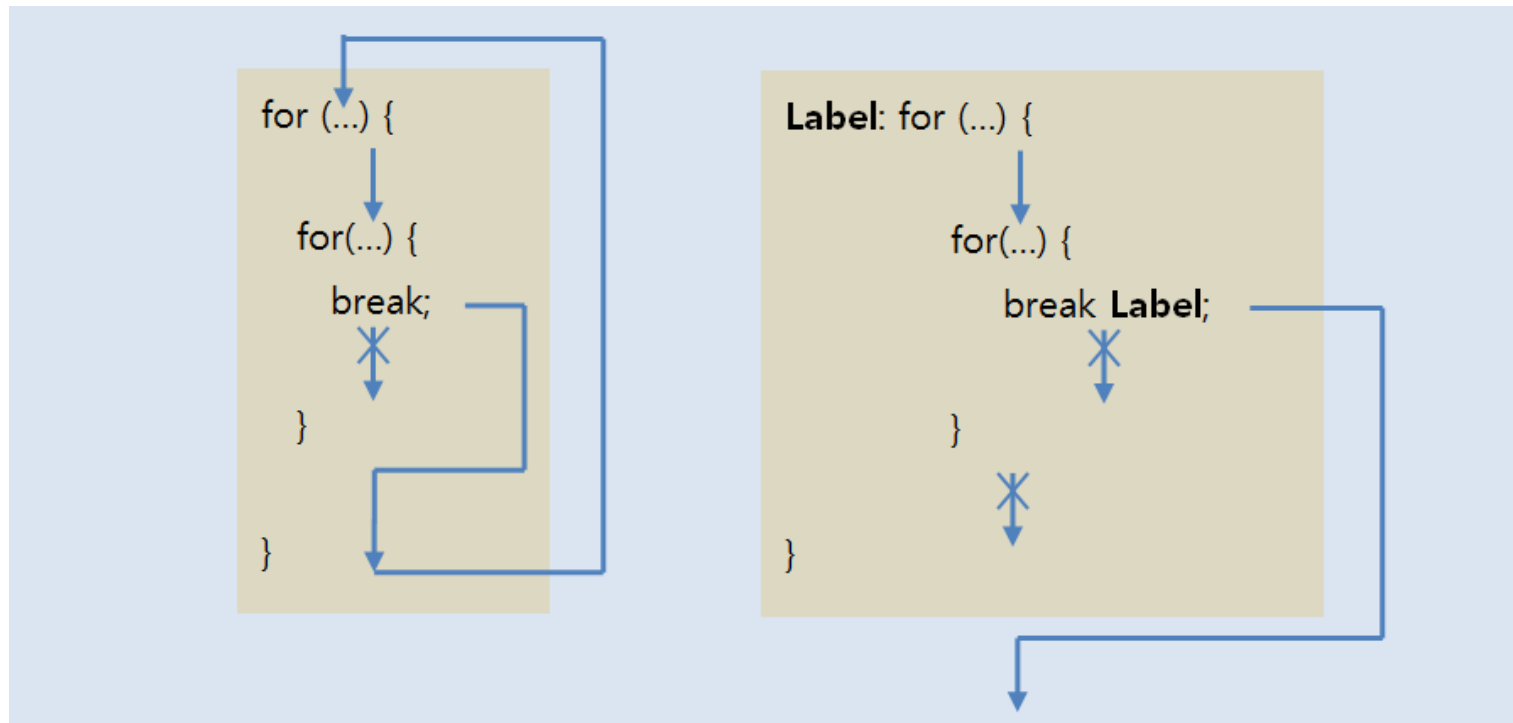
■ break 사용 - 2

```
public class BreadExam2 {  
    public static void main(String[] args) {  
        Scanner scan = new Scanner(System.in);  
  
        while(true) {  
            System.out.println("가위0, 바위1, 보2");  
            System.out.print("숫자 입력 > ");  
            int player = scan.nextInt();  
            int com = (int) (Math.random() * 3);  
            System.out.println(player + "/" + com);
```

```
                if(player == com) {  
                    System.out.println("비김");  
                } else {  
                    if((player + 1) % 3 == com) {  
                        System.out.println("짐");  
                    } else {  
                        System.out.println("이김");  
                        break;  
                    }  
                }  
            }  
            scan.close();  
        }  
    }  
}
```

■ break

- 반복문이 중첩된 경우 가장 가까운 반복문만 종료
- 바깥쪽 반복문까지 종료시키려면 반복문에 이름을 붙여서 사용



■ break 사용 - 3

```
public class BreakExam3 {  
    public static void main(String[] args) {  
        for(int i = 1; i <= 9; i++) {  
            for(int j = 2; j <= 9; j++) {  
                if(i == j) break;  
                System.out.print(j + " * " + i + " = " + (j * i) + " ");  
            }  
            System.out.println();  
        }  
    }  
}
```

2 * 1 = 2	3 * 1 = 3	4 * 1 = 4	5 * 1 = 5	6 * 1 = 6
2 * 3 = 6				
2 * 4 = 8	3 * 4 = 12			
2 * 5 = 10	3 * 5 = 15	4 * 5 = 20		
2 * 6 = 12	3 * 6 = 18	4 * 6 = 24	5 * 6 = 30	
2 * 7 = 14	3 * 7 = 21	4 * 7 = 28	5 * 7 = 35	6 * 7 = 42
2 * 8 = 16	3 * 8 = 24	4 * 8 = 32	5 * 8 = 40	6 * 8 = 48
2 * 9 = 18	3 * 9 = 27	4 * 9 = 36	5 * 9 = 45	6 * 9 = 54

■ break 사용 - 4 (난수 맞추기)

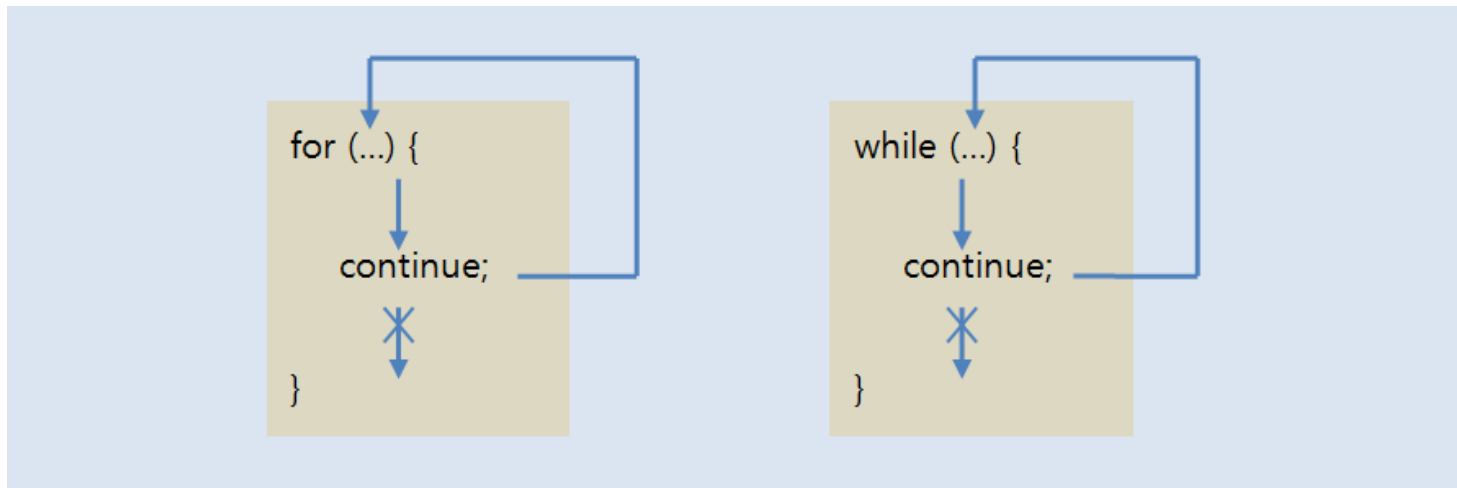
```
public class FindNumber {  
    public static void main(String[] args) {  
        int count = 0;  
        int random = (int) (Math.random() * 100 + 1);  
        Scanner scan = new Scanner(System.in);  
        while(true) {  
            System.out.print("입력 : ");  
            int num = scan.nextInt();  
            count++;  

```

```
            if(random > num) {  
                System.out.println("입력값보다 큼");  
            } else if(random < num) {  
                System.out.println("입력값보다 적음");  
            } else {  
                System.out.println("맞혔음");  
                break;  
            }  
        }  
        System.out.println(  
            "시도한 횟수 : " + count);  
        scan.close();  
    }  
}
```

■ continue

- 자신이 포함된 반복문의 끝으로 이동 (다음 반복으로 넘어감)
- continue문 이후의 문장들은 수행되지 않음



```
for(int i=1; i<=10; i++) {  
    if(i%2 != 0) { ●----- 2로 나눈 나머지가 0 이 아닐 경우  
                        즉 홀수인 경우  
        continue;  
    }  
    System.out.println(i); ●----- 홀수는 실행되지 않는다.  
}
```

■ continue 사용

```
public class ContinueExam {  
    public static void main(String[] args) {  
        File file = new File("C:\\Windows\\System32\\drivers\\etc");  
        File[] files = file.listFiles();  
        for(int i = 0; i < files.length; i++) {  
            long fileSize = files[i].length();  
            if(fileSize < 1000) { // 파일크기가 1000Byte 미만이면 아래 코드 무시  
                continue;  
            }  
            System.out.println(files[i].getName() + " / " + fileSize);  
        }  
    }  
}
```

```
hosts / 824  
hosts_tmp / 824  
lmhosts.sam / 3683  
networks / 407  
protocol / 1358  
services / 17463
```

```
lmhosts.sam / 3683  
protocol / 1358  
services / 17463
```