

# 열혈 Java 프로그래밍

## Chapter 05. 실행 흐름의 컨트롤

05-1. if 그리고 else

## if문

---

```
if(true or false) {
```

조건 true 시 실행되는 영역

```
}
```

ex1)

```
if(n1 < n2) {
```

```
    System.out.println("n1 > n2 is true");
```

```
}
```

ex2) *if문에 속한 문장이 하나일 경우 중괄호 생략 가능*

```
if(n1 < n2)
```

```
    System.out.println("n1 > n2 is true");
```

## if ~ else문

---

```
if(true or false) {
```

조건 true 시 실행되는 영역

```
} else {
```

조건 false 시 실행되는 영역

```
}
```

ex)

```
if(n1 == n2) {
```

```
    System.out.println("n1 == n2 is true");
```

```
}
```

```
else {
```


```
    System.out.println("n1 == n2 is false");
```

```
}
```

*if문과 마찬가지로 if절 또는 else 절에 속한 문장이*

*하나일 경우 중괄호 생략 가능*

```
public static void main(String[] args) {  
    int n1 = 5;  
    int n2 = 7;  
  
    // if문  
    if(n1 < n2) {  
        System.out.println("n1 > n2 is true");  
    }  
  
    // if ~ else 문  
    if(n1 == n2) {  
        System.out.println("n1 == n2 is true");  
    }  
    else {  
        System.out.println("n1 == n2 is false");  
    }  
}
```



명령 프롬프트

C:\JavaStudy>java IEBasic  
n1 > n2 is true  
n1 == n2 is false  
C:\JavaStudy>\_

if문, if ~ else문의 예

---

## if ~ else if ~ else 문

---

```
if(...)  
    System.out.println("...");
```

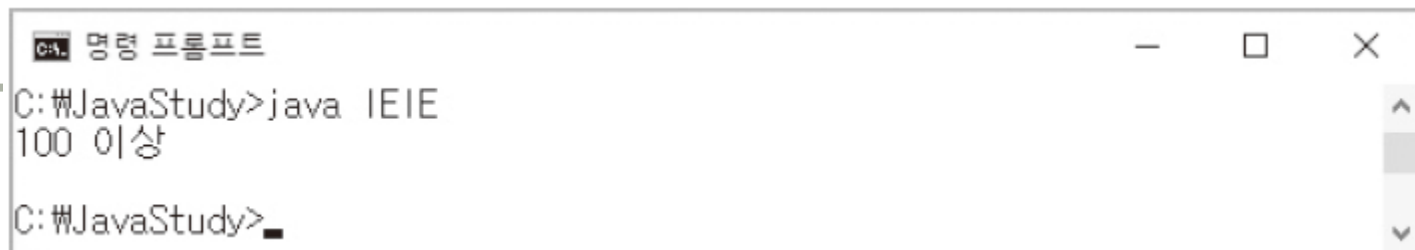
```
else if(...)  
    System.out.println("...");
```

```
else if(...)  
    System.out.println("...");
```

```
else  
    System.out.println("...");
```

*else if 절, 중간에 얼마든지 추가 가능*

```
public static void main(String[] args) {  
    int num = 120;  
  
    if(num < 0)  
        System.out.println("0 미만");  
    else if(num < 100)  
        System.out.println("0 이상 100 미만");  
    else  
        System.out.println("100 이상");  
}
```



```
명령 프롬프트  
C:\JavaStudy>java IEIE  
100 이상  
C:\JavaStudy>
```

if ~ else if ~ else 문의 예

---

## if ~ else if ~ else문과 if ~ else문의 관계

---

```
if(num < 0) {  
    System.out.println("...");  
}  
else {  
    if(num < 100)  
        System.out.println("...");  
    else  
        System.out.println("...");  
}
```



```
if(num < 0)  
    System.out.println("...");  
else  
    if(num < 100)  
        System.out.println("...");  
    else  
        System.out.println("...");
```

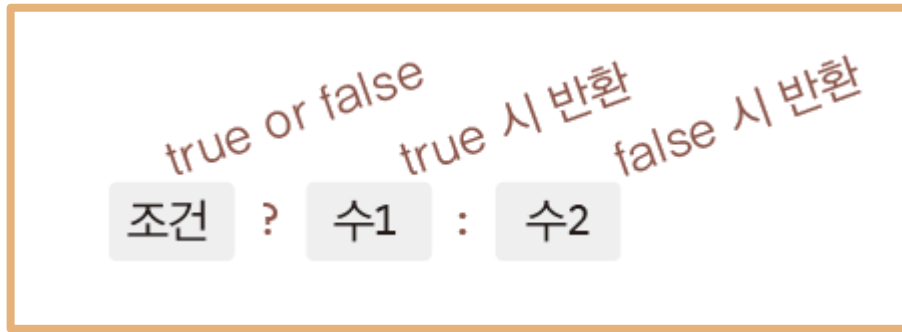


```
if(num < 0)  
    System.out.println("...");  
else if(num < 100)  
    System.out.println("...");  
else  
    System.out.println("...");
```



## if ~ else문과 유사한 성격의 조건 연산자

---



ex1)

```
big = (num1 > num2) ? num1 : num2;
```

ex2)

```
diff = (num1 > num2) ? (num1 - num2) : (num2 - num1);
```

## 05-2. switch와 break

## switch문의 기본 구성

---

```
switch(n) {  
  case 1:  n이 1이면 여기서부터 실행  
    . . .  
  case 2:  n이 2이면 여기서부터 실행  
    . . .  
  case 3:  n이 3이면 여기서부터 실행  
    . . .  
  default: 해당하는 case 없으면 여기서부터 실행  
    . . .  
}
```

case와 default는 레이블!

따라서 실행 위치를 표시하는 용도로 사용될  
뿐!

```
public static void main(String[] args) {  
    int n = 3;  
  
    switch(n) {  
    case 1:  
        System.out.println("Simple Java");  
    case 2:  
        System.out.println("Funny Java");  
    case 3:  
        System.out.println("Fantastic Java");  
    default:  
        System.out.println("The best programming language");  
    }  
  
    System.out.println("Do you like Java?");  
}
```



명령 프롬프트

C:\JavaStudy>java SwitchBasic  
Fantastic Java  
The best programming language  
Do you like Java?  
C:\JavaStudy>

switch문의 예

## switch문 + break문

---

```
switch(n) {  
  case 1:      case 1 영역  
    . . . .  
    break;  
  
  case 2:      case 2 영역  
    . . . .  
    break;  
  
  case 3:      case 3 영역  
    . . . .  
    break;  
  
  default:    default 영역  
    . . . .  
}
```

break문이 실행되면 switch문을 빠져나간다.

```

public static void main(String[] args) {
    int n = 3;

    switch(n) {
    case 1:
        System.out.println("Simple Java");
        break;
    case 2:
        System.out.println("Funny Java");
        break;
    case 3:
        System.out.println("Fantastic Java");
        break;
    default:
        System.out.println("The best programming language");
    }

    System.out.println("Do you like Java?");
}

```

C:\ 명령 프롬프트

```

C:\JavaStudy>java SwitchBreak
Fantastic Java
Do you like Java?
C:\JavaStudy>

```

```

switch(n) {
case 1:
case 2:  switch + break 구성의 다른 예
case 3:
    System.out.println("case 1, 2, 3");
    break;
default:
    System.out.println("default");
}

```

## switch문 + break문의 예

05-3.

for, while 그리|고 do ~ while

# while문

---

```
    반복 조건
while(num < 5) {           반복 영역
    System.out.println("I like Java "+ num);
    num++;
}
```

먼저! 조건 검사

그리고 결과가 true이면 중괄호 영역 실행

```
public static void main(String[] args) {
    int num = 0;
    while(num < 5) {
        System.out.println("I like Java " + num);
        num++;
    }
}
```



## do ~ while문

---

```
do {  
    System.out.println("I like Java " + num);  
    num++;  
} while(num < 5);
```

반복 영역

반복 조건

먼저! 중괄호 영역 실행

그리고 조건 검사 후 결과가 true이면 반복 결정

```
public static void main(String[] args) {  
    int num = 0;  
    do {  
        System.out.println("I like Java " + num);  
        num++;  
    } while(num < 5);  
}
```

## for문 (while문과의 비교)

---

```
    ①  
int num = 0;  
    ②  
while( num < 5 ) {  
    System.out.println("...");  
    num++;  
}
```

```
    ①    ②    ③  
for( int num = 0 ; num < 5 ; num++ ) {  
    System.out.println("...");  
}
```

① → 반복의 횟수를 세기 위한 변수

② → 반복의 조건

③ → 반복의 조건을 무너뜨리기 위한 연산

## for문

---

```
for(int i = 0; i < 3; i++) {  
    System.out.println(. . .);  
}
```

첫 번째 루프의 흐름 [i=0]

① → ② → ③ → ④

두 번째 루프의 흐름 [i=1]

② → ③ → ④

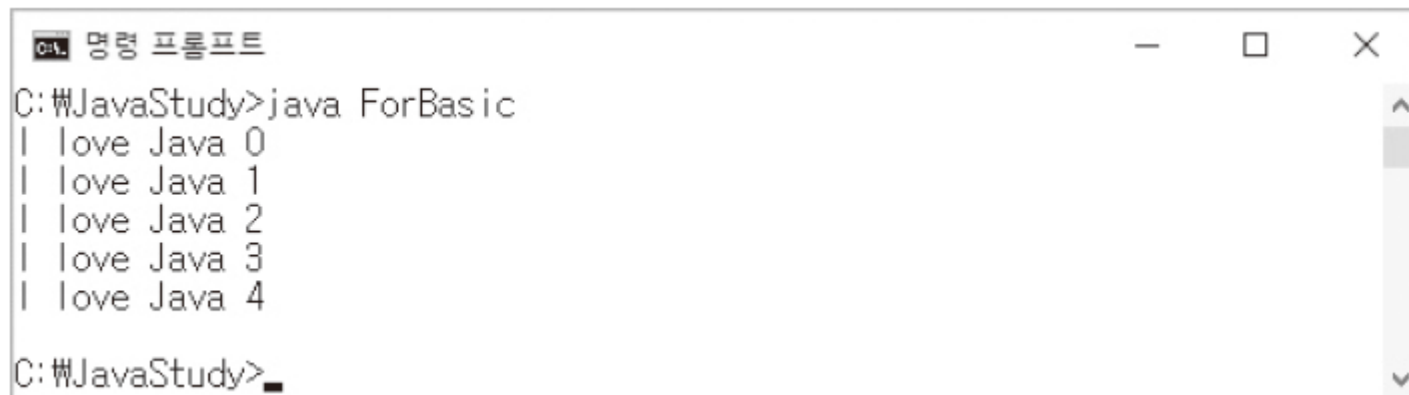
세 번째 루프의 흐름 [i=2]

② → ③ → ④

네 번째 루프의 흐름 [i=3]

② i=3이므로 탈출!

```
public static void main(String[] args) {  
    for(int i = 0; i < 5; i++)  
        System.out.println("I love Java " + i);  
}
```



```
C:\JavaStudy>java ForBasic  
I love Java 0  
I love Java 1  
I love Java 2  
I love Java 3  
I love Java 4  
C:\JavaStudy>
```

for문의 예

---

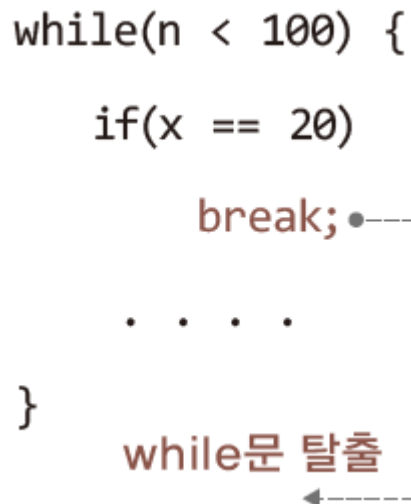
05-4. break & continue

## break와 continue

---

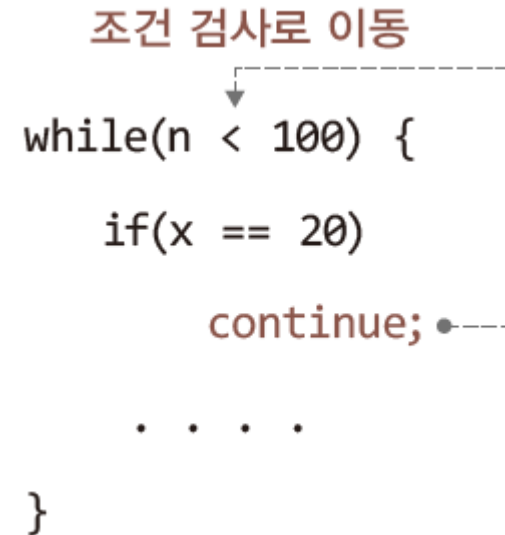
```
while(n < 100) {  
    if(x == 20)  
        break;  
    . . .  
}
```

while문 탈출

A diagram illustrating the use of the break statement. It shows a while loop with a condition (n < 100). Inside the loop, there is an if statement (if(x == 20)) followed by a break statement. A dashed line with an arrow points from the break statement to the closing brace of the while loop, indicating that the loop is exited. The text "while문 탈출" (Exit while loop) is written below the loop.

조건 검사로 이동

```
while(n < 100) {  
    if(x == 20)  
        continue;  
    . . .  
}
```

A diagram illustrating the use of the continue statement. It shows a while loop with a condition (n < 100). Inside the loop, there is an if statement (if(x == 20)) followed by a continue statement. A dashed line with an arrow points from the continue statement back to the condition check of the while loop, indicating that the loop body is skipped and the condition is re-evaluated. The text "조건 검사로 이동" (Move to condition check) is written above the loop.

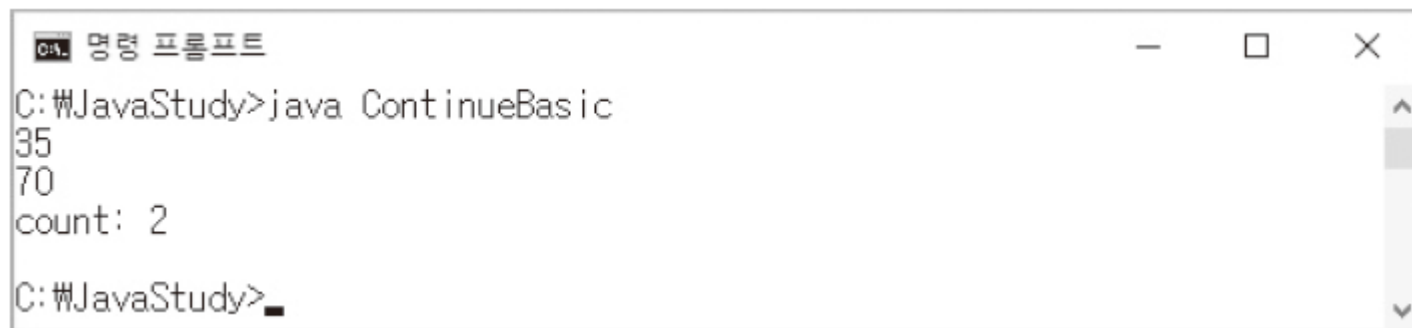
```
public static void main(String[] args) {  
    int num = 1;  
    boolean search = false;  
    // 처음 만나는 5의 배수이자 7의 배수인 수를 찾는 반복문  
    while(num < 100) {  
        if(((num % 5) == 0) && ((num % 7) == 0)) {  
            search = true;  
            break;    // while문을 탈출  
        }  
        num++;  
    }  
    if(search)  
        System.out.println("찾는 정수 : " + num);  
    else  
        System.out.println("5의 배수");  
}
```



```
명령 프롬프트  
C:\JavaStudy>java BreakBasic  
찾는 정수 : 35  
C:\JavaStudy>
```

break문의 예

```
public static void main(String[] args) {  
    int num = 0;  
    int count = 0;  
    while((num++) < 100) {  
        if(((num % 5) != 0) || ((num % 7) != 0))  
            continue;    // 5와 7의 배수 아니라면 나머지 건너뛰고 위로 이동  
        count++;    // 5와 7의 배수인 경우만 실행  
        System.out.println(num);    // 5와 7의 배수인 경우만 실행  
    }  
    System.out.println("count: " + count);  
}
```



```
명령 프롬프트  
C:\JavaStudy>java ContinueBasic  
35  
70  
count: 2  
C:\JavaStudy>
```

continue문의 예



```
for( ; ; ) {  
    ....  
}
```

```
while(true) {  
    ....  
}
```

```
do {  
    ....  
} while(true)
```

# 무한루프

---

```
int num = 1;

while(true) {
    if(((num % 6) == 0) && ((num % 14) == 0))
        break;
    num++;
}
```

'6의 배수이면서 14의 배수인 가장 작은 자연수'를 찾는 반복문

## 무한루프와 break문의 예

---

## 05-5. 반복문의 중첩

# 생각해 볼 수 있는 반복문의 중첩의 형태

```
for(...;...;...) {  
    for(...;...;...) {  
        . . .  
    }  
}
```

```
while(...) {  
    for(...;...;...) {  
        . . .  
    }  
}
```

```
do {  
    for(...;...;...) {  
        . . .  
    }  
} while(...);
```

```
for(...;...;...) {  
    while(...) {  
        . . .  
    }  
}
```

```
while(...) {  
    while(...) {  
        . . .  
    }  
}
```

```
do {  
    while(...) {  
        . . .  
    }  
} while(...);
```

```
for(...;...;...) {  
    do {  
        . . .  
    } while(...);  
}
```

```
while(...) {  
    do {  
        . . .  
    } while(...);  
}
```

```
do {  
    do {  
        . . .  
    } while(...);  
} while(...);
```

## ◆ ForInFor.java

```
1. class ForInFor {
2.     public static void main(String[] args) {
3.         for(int i = 0; i < 3; i++) {      // 바깥쪽 for문
4.             System.out.println("-----");
5.             for(int j = 0; j < 3; j++) {    // 안쪽 for문
6.                 System.out.print "[" + i + ", " + j + " ] ";
7.             }
8.             System.out.print('\n');
9.         }
10.    }
11. }
```

명령 프롬프트

C:\JavaStudy>java ForInFor

[0, 0] [0, 1] [0, 2]

-----

[1, 0] [1, 1] [1, 2]

-----

[2, 0] [2, 1] [2, 2]

-----

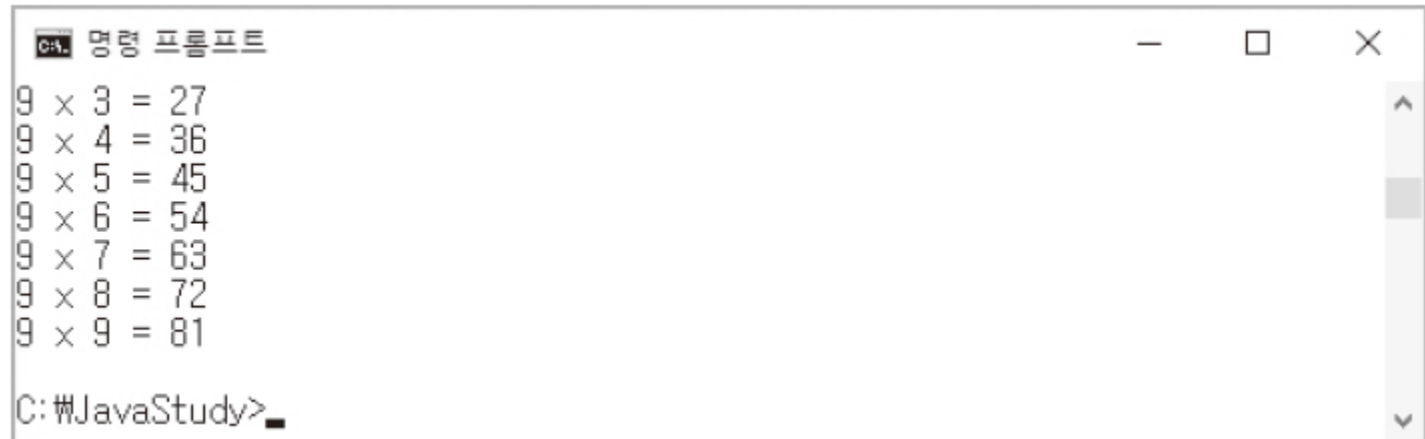
C:\JavaStudy>\_

for문 중첩의 예

		바깥쪽 for문 담당 →						
안쪽 for문 담당 ↓	$2 \times 1 = 2$	$3 \times 1 = 3$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 1 = 9$
	$2 \times 2 = 4$	$3 \times 2 = 6$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 2 = 18$
	$2 \times 3 = 6$	$3 \times 3 = 9$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 3 = 27$
	$2 \times 4 = 8$	$3 \times 4 = 12$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 4 = 36$
	$2 \times 5 = 10$	$3 \times 5 = 15$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 5 = 45$
	$2 \times 6 = 12$	$3 \times 6 = 18$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 6 = 54$
	$2 \times 7 = 14$	$3 \times 7 = 21$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 7 = 63$
	$2 \times 8 = 16$	$3 \times 8 = 24$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 8 = 72$
	$2 \times 9 = 18$	$3 \times 9 = 27$	$4 \times \dots$	$5 \dots$	$6 \dots$	$7 \dots$	$8 \dots$	$9 \times 9 = 81$

# 구구단 전체 출력을 위한 관찰

```
for(int i = 2; i < 10; i++) {    // 2단부터 9단까지 진행 위한 바깥쪽 for문
    for(int j = 1; j < 10; j++)    // 1부터 9까지의 곱을 위한 안쪽 for문
        System.out.println(i + " x " + j + " = " + (i * j));
}
```



```
C:\JavaStudy>
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
C:\JavaStudy>
```

구구단 출력 예제

*The End*

Chapter 05의 강의를 마칩니다.