

2 학기

JAVA Class

'이것이 자바다'

2025.09

Cover Description

made by Lewis

01

제어문 이란?

조건문, 반복문

02

제어문 종류

if, switch, for, while, do-while, break, continue

컴퓨터가 이해하는 코드는
누구라도 작성할 수 있습니다.
뛰어난 프로그래머는 사람이
이해하는 코드를 작성합니다.



■ 연산자 복습

1. 사과의 수가 123개이고 하나의 바구니에는 10개의 사과를 담을 수 있다면 몇 개의 바구니가 필요한지 코딩하시오.

```
Console × Problems Debug Shell
<terminated> AppleBundle [Java Application] C:\Users\...
1.Total Need Bundle Count : 13 cnt
2.Total Need Bundle Count : 13 cnt
```



■ 연산자 복습

2. 첫번째 정수값 변수를 입력받아 iFstVal에
두번째 정수값 변수를 입력받아 iSecVal에 등록하고
두 입력 값 중 큰 수를 먼저 출력
하는 코드를 작성하시오

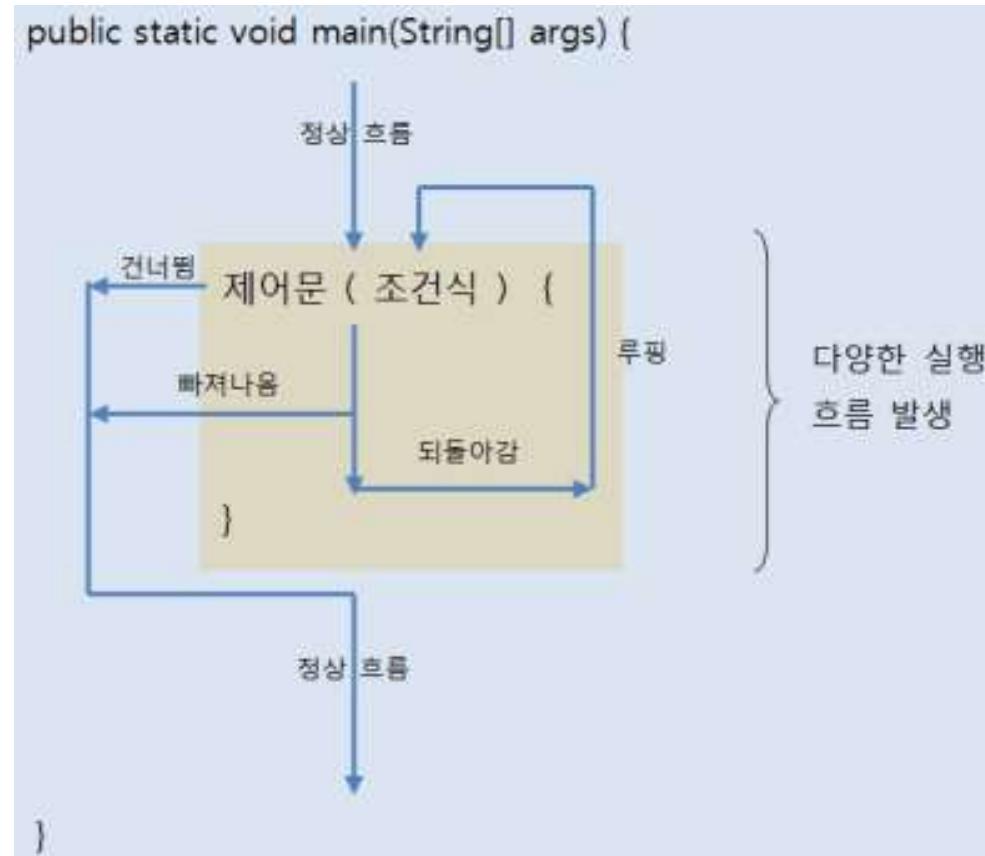
The screenshot shows a Java application running in an IDE. The console tab is active, displaying the following output:

```
Console × Problems J
<terminated> BetweenAB [Java Application]
1. Input Integer value :
4
2. Input Integer value :
5
Input iFstVal = 4
Input iSecVal = 5
Output iFstVal = 5
Output iSecVal = 4
```

The application prompts for two integer inputs. It then prints the input values and their corresponding output values, where the first input (4) is assigned to iFstVal and the second input (5) is assigned to iSecVal.



■ 제어문이란 ?



개발자가 원하는 방향으로 코드 실행 흐름 제어

■ 플로우차트

순서도, 흐름도 : 어떠한 일을 처리하는 과정을 순서대로
간단한 도형과 기호로 도식화 한 것
→ 플로우차트의 흐름은 코딩의 순서와 같다.

* 장점

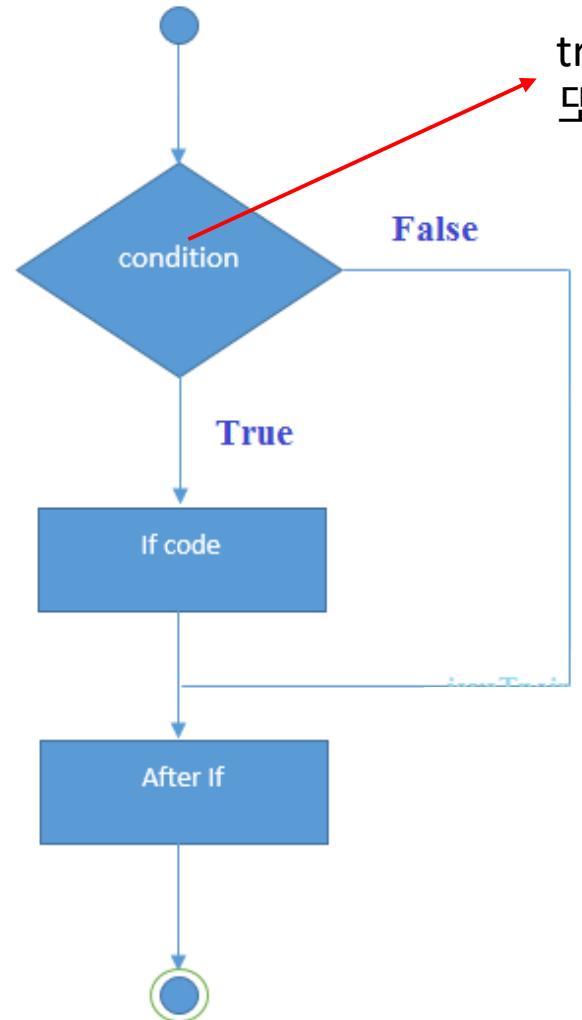
1. 프로그램의 흐름을 단순화 하여 분석이 명료해짐
2. 논리적 오류 분석 가능
3. 도식화된 기호로 다른 사람이 쉽게 이해
4. 프로그램의 작성성을 용이하게 하여 코딩 작업이 간단해짐

* 비교/판단 기호 사용 시 입/출력은 반드시 하나이고 결과는 Yes/No 여야 한다.

기호	명칭	설명
oval	단말	순서도의 시작과 끝을 나타냄.
→	흐름선	순서도 기호 간의 연결 및 작업의 흐름을 표시함.
◇	준비	작업 단계 시작 전 해야 할 작업을 명시함.
□	처리	처리해야 할 작업을 명시함.
□	입출력	데이터의 입출력 시 사용함.
◇	의사 결정	비교 및 판단에 의한 논리적 분기를 나타냄.
□	표시	화면으로 결과를 출력함.



if 문



true, false 값을 산출할 수 있는 연산식
또는 Boolean

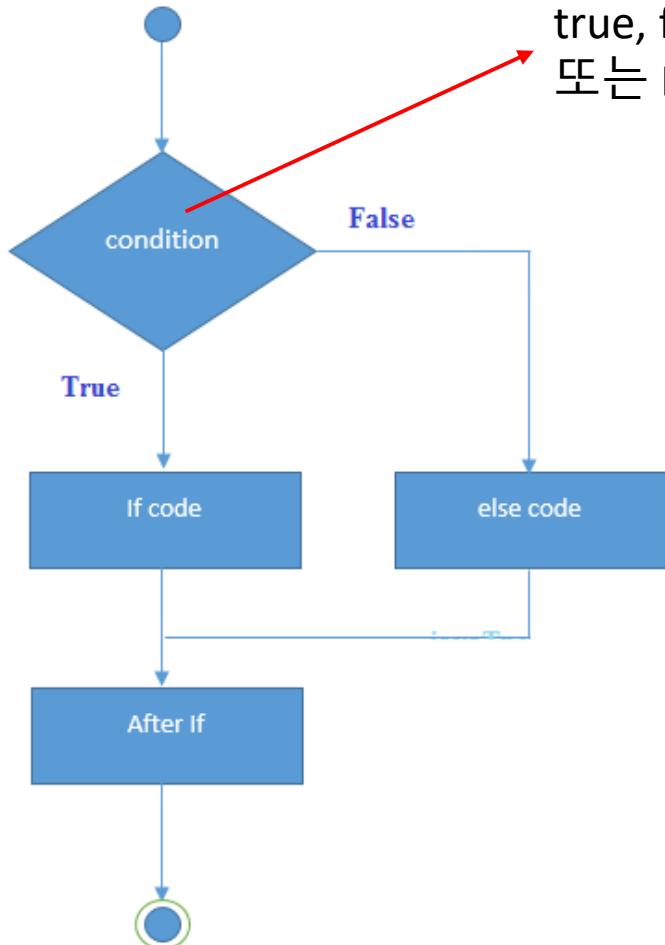
P.113 IfExample.java

```
1 package ch04.sec02;
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 }
17
```

A screenshot of a Java code editor showing the file "IfExample.java". The code defines a class "IfExample" with a main method. Inside the main method, an integer variable "score" is assigned the value 93. Two if statements are then evaluated. The first if statement checks if "score" is greater than or equal to 90. If true, it prints "점수가 90보다 큽니다." and "등급은 A 입니다.". The second if statement checks if "score" is less than 90. If true, it prints "점수가 90보다 작습니다." and "등급은 B 입니다.". Both if blocks have their corresponding closing braces and brackets correctly placed.



if - else 문



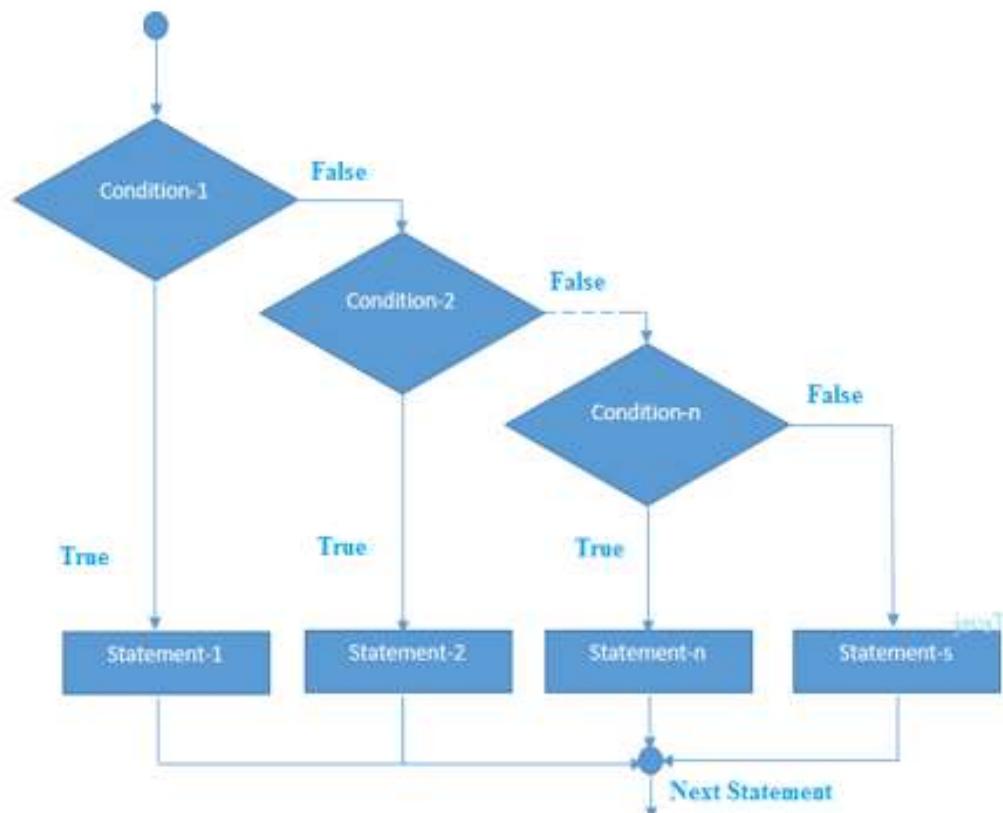
true, false 값을 산출할 수 있는 연산식
또는 Boolean

P.115 IfElseExample.java

```
1 package ch04.sec02;
2
3 public class IfElseExample {
4     public static void main(String[] args) {
5         int score = 85;
6
7         if(score>=90) {
8             System.out.println("점수가 90보다 큽니다.");
9             System.out.println("등급은 A 입니다.");
10        } else {
11            System.out.println("점수가 90보다 작습니다.");
12            System.out.println("등급은 B 입니다.");
13        }
14    }
15 }
```



if – else if 문



P.116 IfElseIfElseExample.java

```
1 package ch04.sec02;
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 }
22 }
```

* score = 90;
if(score >= 70)
else if (score >= 80)
else if (score >= 90)



■ if 문 주사위 출력

```
1 package ch04.sec02;
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 }
```

1. Math → java.Lang 패키지에 포함된
클래스로 수학과 관련된 일련의 작업들을
처리

abs(), random(), max(), min().....

2. Math.random()

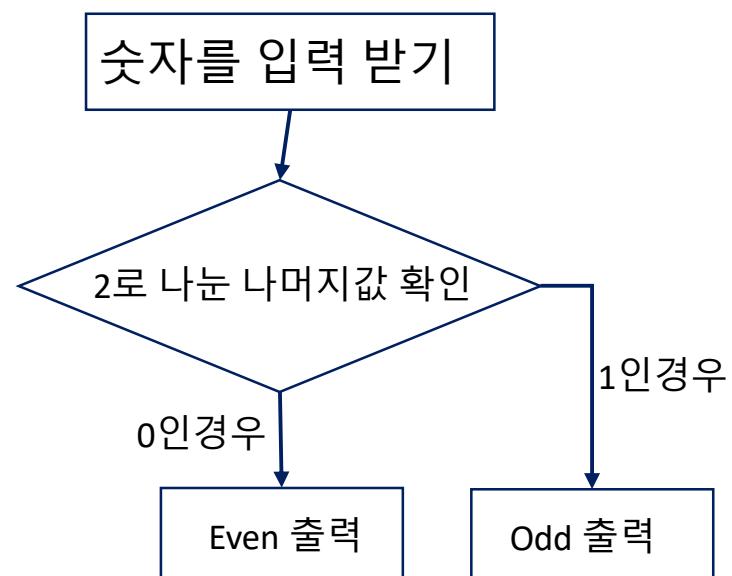
→ $0.0 \leq \text{Math.random()} < 1.0$

$0.0 * 6 \leq \text{Math.random()} * 6 < 1.0 * 6$



■ 테스트1

숫자를 입력 받아 짝수인지 홀수인지
확인하는 프로그램을 작성하시오





■ 테스트2

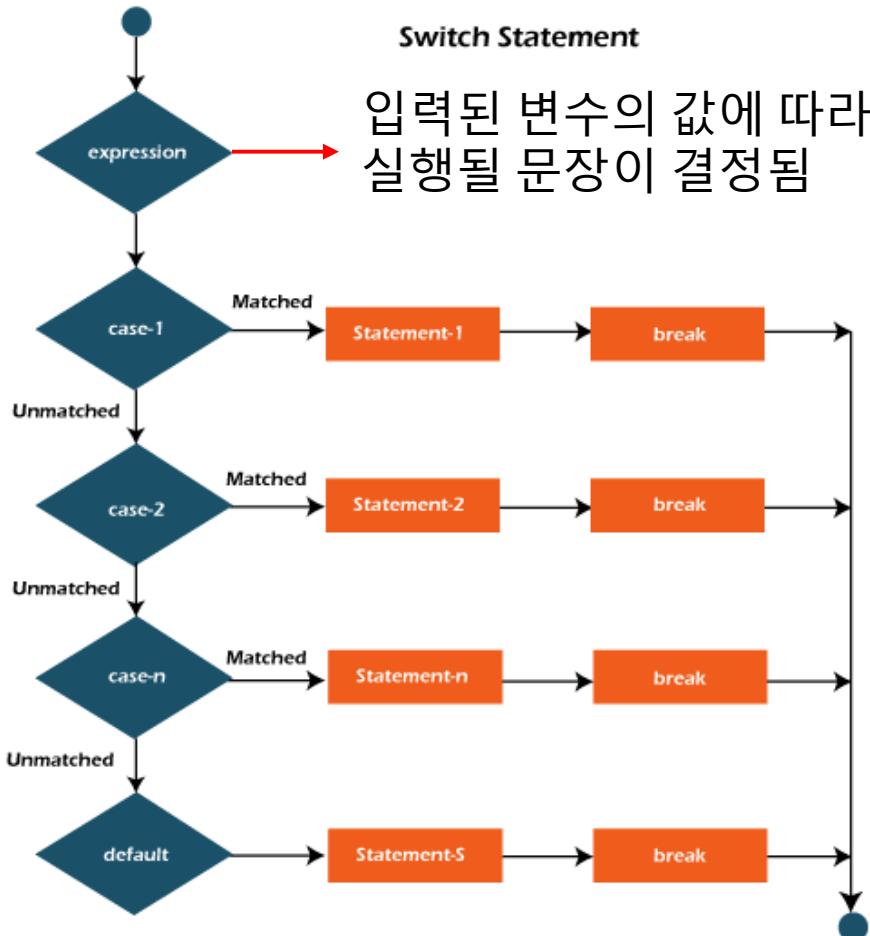
* 2개의 숫자를 입력 받아 작은 수 먼저
출력 하는 프로그램을 작성하시오

Example.

The screenshot shows a Java application window with a dark theme. The title bar includes tabs for 'Console' (selected), 'Problems', and 'D'. The main area displays the following text:

```
<terminated> CompareValue [Java Ap
1. Input Value : 3
2. Input Value : 4
3 < 4
Done..
```

switch 문



P.121 SwitchExample.java

```
1  package ch04.sec03;
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         }
26     }
27 }
```



switch 문

Case 문의 break 역할 확인

time == 8 인 경우

time == 10 인 경우

time == 11 인 경우

time == 3 인 경우 ??

```
int time = (int)(Math.random()*4) + 8;
```

```
int time = (int)(Math.random()*12) + 1;
```

P.122 SwitchNoBreakCaseExample.java

The screenshot shows the Eclipse IDE interface. The top part displays the Java code for `SwitchNoBreakCaseExample.java`. The code defines a class with a main method that generates a random time value between 8 and 11. It then uses a switch statement to print different messages based on the time value. The `default` case prints a message for time 3. The bottom part shows the `Console` tab with the program's output: "[현재시간: 9 시]", "회의를 합니다.", "업무를 봅니다.", and "외근을 나갑니다.". This demonstrates that the code executes all cases in the switch statement even if a break is not explicitly present.

```
1 package ch04.sec03;
2
3 public class SwitchNoBreakCaseExample {
4     public static void main(String[] args) {
5         int time = (int)(Math.random()*4) + 8;
6         System.out.println("[현재시간: " + time + " 시]");
7
8         switch(time) {
9             case 8:
10                 System.out.println("출근합니다.");
11             case 9:
12                 System.out.println("회의를 합니다.");
13             case 10:
14                 System.out.println("업무를 봅니다.");
15             default:
16                 System.out.println("외근을 나갑니다.");
17         }
18     }
19 }
```

Console × Problems Debug Shell
<terminated> SwitchNoBreakCaseExample [Java Application] C:\Users\Lewis\p2
[현재시간: 9 시]
회의를 합니다.
업무를 봅니다.
외근을 나갑니다.



P.123 SwitchCharExample.java

switch 문

Case 문의 조건값 확인

→ case 'A':

case 'a':

```
1 package ch04.sec03;
2
3 public class SwitchCharExample {
4     public static void main(String[] args) {
5         char grade = 'B';
6
7         switch(grade) {
8             case 'A':
9             case 'a':
10                System.out.println("우수 회원입니다.");
11                break;
12             case 'B':
13             case 'b':
14                System.out.println("일반 회원입니다.");
15                break;
16             default:
17                 System.out.println("손님입니다.");
18         }
19     }
20 }
21
```

'B' == 'b' ???

Console × Problems × Debug Shell
<terminated> SwitchCharExample [Java Application] C:\Users\Lewis\p2\po
일반 회원입니다.



P.124 SwitchExpressionsExample.java

switch 문

Case 문의 조건값 확인

- ➔ case 'A':
case 'a':
- ➔ case 'A', 'a' ->

* : -> 차이 구분

실행문이 한 줄인 경우
중괄호 생략 가능

```
1 package ch04.sec03;
2
3 public class SwitchExpressionsExample {
4     public static void main(String[] args) {
5         char grade = 'B';
6
7         switch(grade) {
8             case 'A', 'a' -> {
9                 System.out.println("우수 회원입니다.");
10            }
11            case 'B', 'b' -> {
12                System.out.println("일반 회원입니다.");
13            }
14            default -> {
15                System.out.println("손님입니다.");
16            }
17        }
18
19        switch(grade) {
20            case 'A', 'a' -> System.out.println("우수 회원입니다.");
21            case 'B', 'b' -> System.out.println("일반 회원입니다.");
22            default -> System.out.println("손님입니다.");
23        }
24    }
25}
```

Console × Problems Debug Shell
<terminated> SwitchExpressionsExample [Java Application] C:\Users\Lewis\p2\pool\plugins
일반 회원입니다.
일반 회원입니다.



switch 문

* 반환 값 사용 시 yield 확인

P.125 SwitchValueExample.java

```
SwitchValueExample.java ×
1 package ch04.sec03;
2
3 public class SwitchValueExample {
4     public static void main(String[] args) {
5         String grade = "B";
6
7         //Java 12 이전 문법
8         int score1 = 0;
9         switch(grade) {
10             case "A":
11                 score1 = 100;
12                 break;
13             case "B":
14                 int result = 100 - 20;
15                 score1 = result;
16                 break;
17             default:
18                 score1 = 60;
19         }
20         System.out.println("score1: " + score1);
21
22         int score2 = switch(grade) {
23             case "A" -> 100;
24             case "B" -> {
25                 int result = 100 - 20;
26                 yield result; //Java 13부터 가능
27             }
28             default -> 60;
29         };
30         System.out.println("score2: " + score2);
31     }
32 }
```

Console × Problems Debug Shell
<terminated> SwitchValueExample [Java Application] C:\Users\Lewis\p2Wi
score1: 80
score2: 80



Java

■ switch 문 테스트 1

- 임의의 값(1 ~ 12)을 생성하여 해당월의 마지막 날을 표시 하시오.
- 1, 3, 5, 7, 8, 10, 12 월은 31일
- 2 월은 28일
- 2, 4, 6, 9, 11 월은 30일



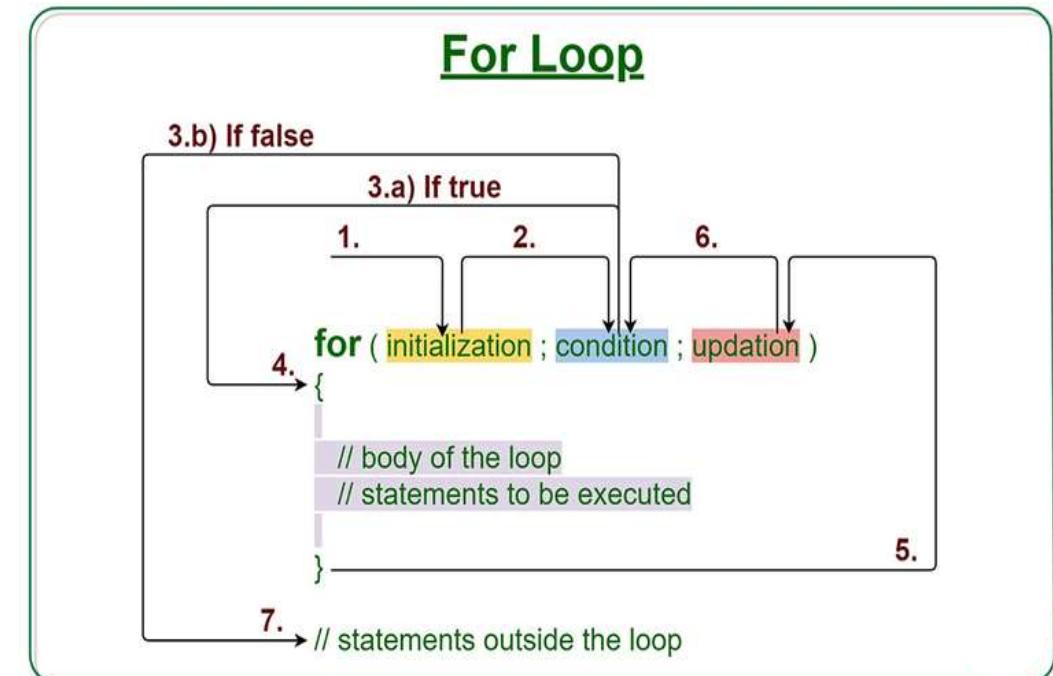
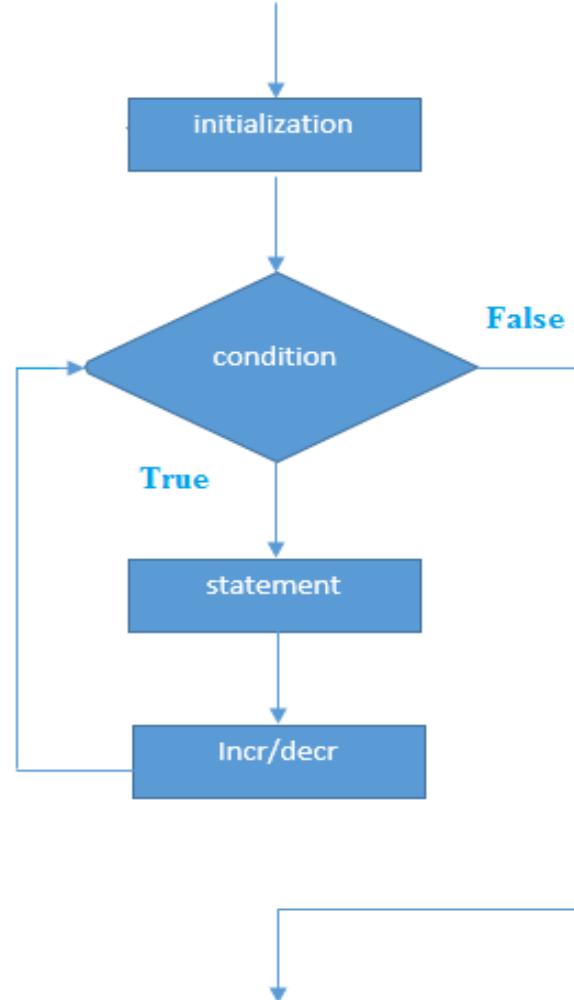
■ switch 문 테스트 2

두 정수 iVal_1, iVal_2 와 연산자 sOperator (+, -, *, /)를 입력받아,
switch문을 이용해 결과를 출력하는 프로그램을 작성하시오.

- sOperator 가 +면 덧셈, -면 뺄셈 ...
- 나눗셈은 0으로 나눌 경우 "Error: divide by zero" 출력



for 문





for 문

- 기본 for 문 형태

The screenshot shows a Java code editor with a file named `PrintFrom1To10Example.java`. The code contains a simple for loop that prints numbers from 1 to 10. Below the editor is a terminal window showing the execution of the program and its output.

```
1 package ch04.sec04;
2
3 public class PrintFrom1To10Example {
4     public static void main(String[] args) {
5         for(int i=1; i<=10; i++) {
6             System.out.print(i + " ");
7         }
8     }
9 }
10
```

Console × Problems Debug Shell
<terminated> PrintFrom1To10Example [Java Application] C:\Users\...
1 2 3 4 5 6 7 8 9 10

1 ~ 10 까지 출력

* 10 ~ 1 까지 출력하는
코드를 작성하시오



for 문

- 기본 for 문 형태

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor displays a Java file named 'SumFrom1To100Example.java' containing the following code:

```
1 package ch04.sec04;
2
3 public class SumFrom1To100Example {
4     public static void main(String[] args) {
5         int sum = 0;
6         int i;
7
8         for(i=1; i<=100; i++) {
9             sum += i;
10        }
11
12        System.out.println("1~" + (i-1) + " 합 : " + sum);
13    }
14 }
15
```

The terminal window at the bottom shows the output of the program:

```
<terminated> SumFrom1To100Example [Java Application] C:\Users\Lewis\p2\pool\SumFrom1To100Example.java
1~100 합 : 5050
```

1 ~ 100 까지 합을 구하고 그 결과 값을 출력

* for 문 사용 시 주의할 점
→ 초기화식에서 float 사용 금지
→ 연산 과정에서 소수점 연산이 부정확 할 수 있음으로



for 문

- 기본 for 문 형태

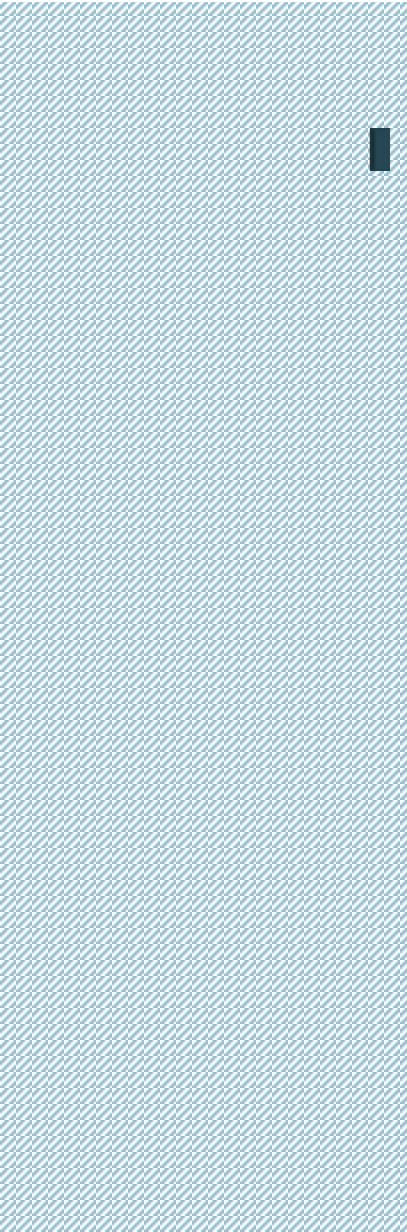
The screenshot shows an IDE interface with a code editor and a terminal window. The code editor displays a Java file named `MultiplicationTableExample.java`. The code contains a `main` method that prints multiplication tables for numbers 2 through 9. The terminal window below shows the output of the program, which is the multiplication table for the number 2.

```
1 package ch04.sec04;
2
3 public class MultiplicationTableExample {
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 }
```

Console

```
<terminated> MultiplicationTableExample [Java Application] C:\Users\Lewis\p2\pool\plugins\o
*** 2단 ***
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
*** 3단 ***
```

2~9 단 까지 구구단 출력



■ for 문 테스트 1

2~9 단 까지 구구단
출력하되 7단을 제외하고
출력 하시오



■ for 문 테스트 2

아래의 결과를 출력 하는 코드를 완성 하시오

A screenshot of a Java IDE's console window. The title bar shows "Console" and "Problems". The console output shows the following star pattern:

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

The pattern consists of nine lines of asterisks, with the last four lines being identical.



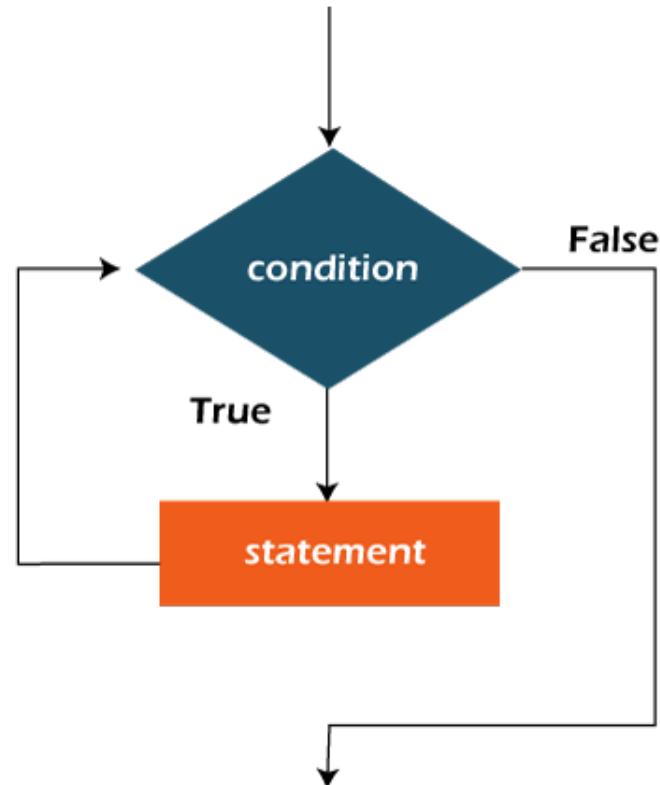
■ for 문 테스트

아래의 결과를 출력 하는 코드를 완성 하시오

```
Console <terminated> ShowSt
*
**
***
****
*****
******
*****
*****
*****
```



while문



A screenshot of an IDE interface showing a Java file named "PrintFrom1To10Example.java". The code is as follows:

```
1 package ch04.sec05;
2
3 public class PrintFrom1To10Example {
4     public static void main(String[] args) {
5         int i = 1;
6         while (i<=10) {
7             System.out.print(i + " ");
8             i++;
9         }
10    }
11 }
```

Below the code editor, the "Console" tab is active, displaying the output of the program: "1 2 3 4 5 6 7 8 9 10". Other tabs visible include "Problems" and "Debug Shell".



■ while 문 합 구하기

1 ~ 100 까지의 합을 구하시오

The screenshot shows an IDE interface with a code editor and a terminal window.

Code Editor:

```
1 package ch04.sec05;
2
3 public class SumForm1To100Example {
4     public static void main(String[] args) {
5         int sum = 0;
6         int i = 1;
7
8         while(i<=100) {
9             sum += i;
10            i++;
11        }
12
13        System.out.println("1~" + (i-1) + " Sum : " + sum);
14    }
15 }
16
```

Terminal (Console):

```
<terminated> SumForm1To100Example [Java Application] C:\Users\Lewis\p2\pool\1~100 Sum : 5050
```



■ while 문 Key Control

키보드 값으로 반복문 제어 테스트

The screenshot shows a Java application running in a terminal window. The window title is 'Console' and the application title is '<terminated> KeyControlExample [Java Appl]'. The application displays a menu loop with three options: 1. 증속, 2. 감속, and 3. 중지. It also shows the current speed value. The user has selected option 1 twice, changing the speed from 1 to 0.

```
Console × Problems Debug S
<terminated> KeyControlExample [Java Appl]
-----
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 종료
```



■ while 문 테스트

아래의 결과를 출력 하는 while 문 코드를 완성 하시오

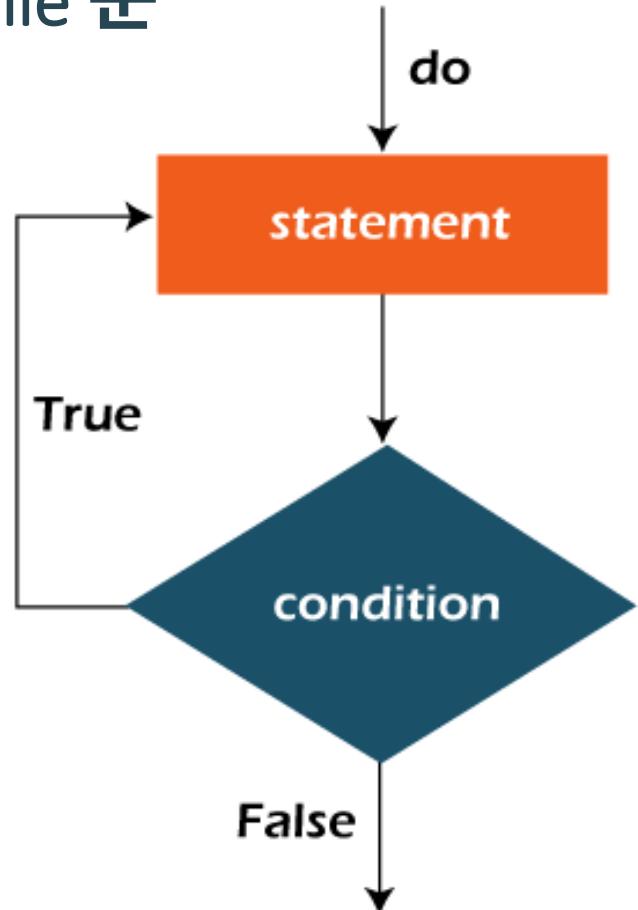
A screenshot of a Java IDE's console window. The title bar shows "Console" and "Problems". The console output shows the following star pattern:

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

The pattern consists of nine lines of asterisks, starting with one asterisk and increasing by one each line, then decreasing back to one asterisk.



do-while 문



do-while문이 끝날 때는 세미콜론을 붙인다

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor contains a Java file named 'DoWhileExample.java' with the following content:

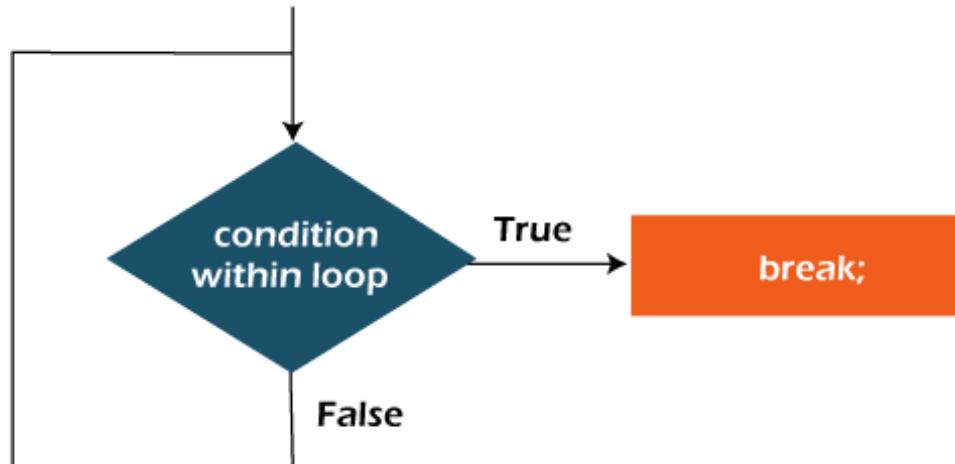
```
*DoWhileExample.java ×
1 package ch04.sec06;
2
3 import java.util.Scanner;
4
5 public class DoWhileExample {
6     public static void main(String[] args) {
7         System.out.println("메시지를 입력하세요");
8         System.out.println("프로그램을 종료하려면 q를 입력하세요.");
9
10    Scanner scanner = new Scanner(System.in);
11    String inputString;
12
13    do {
14        System.out.print(">");
15        inputString = scanner.nextLine();
16        System.out.println(inputString);
17    } while( ! inputString.equals("q") );
18
19    System.out.println();
20    System.out.println("프로그램 종료");
21 }
22 }
```

The terminal window below shows the program's interaction with the user:

```
Console × Problems Debug Shell
<terminated> DoWhileExample [Java Application] C:\Users\Lewis\p2\pool\plugins
메시지를 입력하세요
프로그램을 종료하려면 q를 입력하세요.
>hello
hello
>q
|
프로그램 종료
```



break 문



For, while 문 모두 사용 가능

```
1 package ch04.sec07;  
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}  
15  
16
```

The screenshot shows the Eclipse IDE interface with three tabs: "Console", "Problems", and "Debug Shell". The "Console" tab displays the output of the application, which consists of five integers (3, 4, 5, 2, 6) followed by the message "프로그램 종료".

```
<terminated> BreakExample [Java Application] C:\Users\Lewis\p2\pool\plugins\org.eclips...
```

```
3  
4  
5  
2  
6  
프로그램 종료
```



break 문

* 중첩된 반복문의 경우 바깥쪽
반복문까지 종료 시키려는 경우

* Outer 이란 라벨 설정

```
1 package ch04.sec07;
2
3 public class BreakOuterExample {
4     public static void main(String[] args) throws Exception {
5         Outer: for(char upper='A'; upper<='Z'; upper++) {
6             for(char lower='a'; lower<='z'; lower++) {
7                 System.out.println(upper + "-" + lower);
8                 if(lower=='g') {
9                     break Outer;
10                }
11            }
12        }
13        System.out.println("프로그램 실행 종료");
14    }
15 }
16 }
```

```
Console × Problems × Debug Shell
<terminated> BreakOuterExample [Java Application] C:\Users\Lewis\p2\pool\plugins\or
A-a
A-b
A-c
A-d
A-e
A-f
A-g
프로그램 실행 종료
```



■ while 문 테스트

1. 1 ~ 10 까지의 랜덤 숫자를 발생시키고 숫자를 입력 받아 맞추는 프로그램을 작성하시오.
2. 입력한 숫자가 랜덤 숫자보다 작으면 Up
입력한 숫자가 랜덤 숫자보다 크면 Down
맞으면 Bingo 표시
3. 맞추는 기회는 3번 까지

```
Console × Problems Debug Shell
<terminated> GuessNumber [Java Application] C:\WUSe...
Input a number between 1 and 10.
4
Up!
Input a number between 1 and 10.
2
Bingo!
```



■ do-while 문 테스트

1. 1 ~ 10 까지의 랜덤 숫자를 발생시키고 숫자를 입력 받아 맞추는 프로그램을 do-while 문을 이용하여 작성하시오.
2. 입력한 숫자가 랜덤 숫자보다 작으면 Up
입력한 숫자가 랜덤 숫자보다 크면 Down
맞으면 Bingo 표시
3. 맞추는 기회는 3번 까지

```
Console × Problems Debug Shell
<terminated> GuessNumber [Java Application] C:\WUSe...
Input a number between 1 and 10.
4
Down!
Input a number between 1 and 10.
2
Bingo!
```



■ continue 문

* For, while 문 모두 사용 가능
반복문을 종료 하지 않고 계속
수행한다.

```
1 package ch04.sec08;
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.print(i + " ");
10        }
11    }
12 }
13 }
```

The screenshot shows an IDE interface with the following details:

- Code Area:** Displays the Java code for ContinueExample.java.
- Console Tab:** Shows the output of the program: "2 4 6 8 10".
- Bottom Status Bar:** Shows the path: <terminated> ContinueExample [Java Application] C:\Users\Lewis\p2\pool\plugins\org.e...

P.139 ContinueExample.java