JAVA 제어문

프로그램은 사람이 이해하는 코드를 작성.

느려도 꾸준하면 경기에서 이긴다.

작성자 : 홍효상
이메일 : hyomee@naver.com
소스 : https://github.com/hyomee/JAVA_EDU

Content

4. 제어문

1. 개념
2. If
3. switch
4. for
5. while
6. do-while
7. break, continue

1. 개념

"프로그램의 순서를 바꾸는 것"

제어문

- 위에서 아래로 1줄 씩 순차적으로 처리 하는 것은 조건에 따라 프로그램 순서를 바꾸는 것
- 선택 제어문 : if , switch
- 반복 제어문 : for, while, do..while
- 제어 키워드 : break, continue

제어문	문장	기능
if	• If • if – else • If – else if – else	주어진 조건에 따라서 실행문의 실행 여부 결정

switch	• switch (위치변수) {
	case 값1:
	.....
	case 값n:
	.....
	default :
	.....
	}
		위치변수의 값에 따라서 특정위치(case)의 실행문 실행

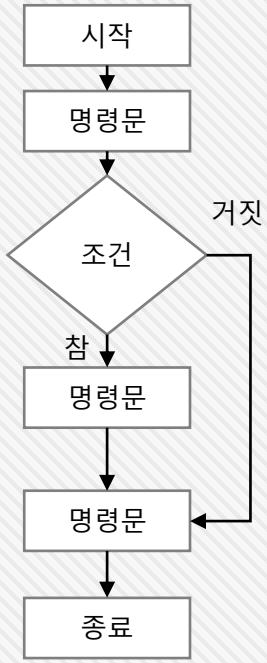
for	• for (초기값 ; 종료조건 ; 증감)	반복 횟수를 정해 놓고 반복 실행
while	• while (조건) { ..... }	조건이 만족 할 때 까지 반복 실행
do .. while	• do { ..... } while (조건)	한번 실행 후 조건이 만족하면 반복 실행

제어 키워드	기능
break	• 제어문 탈출
continue	• 반복 실행 중 건너 뛰기

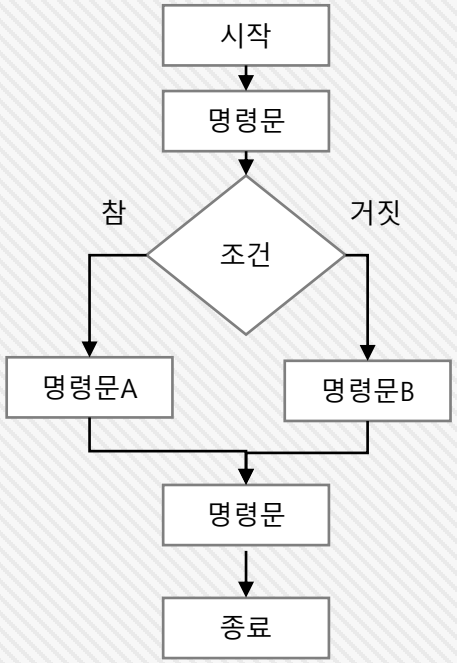
2. if

if

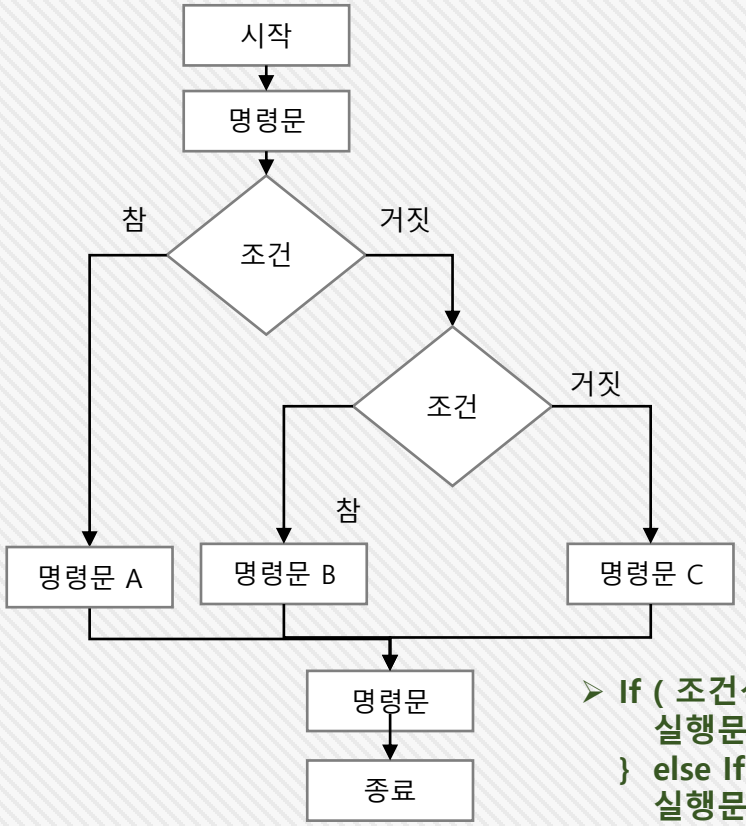
- 특정 조건에 따라서 실행문 결정



➢ If (조건식) 실행문



➢ If (조건식) {
  실행문 A;
} else {
  실행문 B;
}



➢ If (조건식) {
  실행문 A;
} else If (조건식) {
  실행문 B;
} else {
  실행문 C;
}

2. if

if

```
public class ControlIf {
   public static void main(String[] args) {
      int num01 = 10;
      int num02 = 20;

      if ( num01 < num02 )  {
         System.out.println(String.format("if :: %s < %s 는 만족 합니다.", num01, num02));
      }

      if ( num01 < num02 )  {
         System.out.println(String.format("if - else :: %s < %s 는 만족 합니다.", num01, num02));
      } else {
         System.out.println(String.format("if - else :: %s < %s 는 불만족 합니다.", num01, num02));
      }

      int score = 88;
      if ( score < 80 )  {
         System.out.println(String.format("if - else if - else  :: %s 는 80이하 입니다.", score));
      } else if ( score >= 80 && score < 90 )  {
         System.out.println(String.format("if - else if - else  :: %s 는 80이상 90이하 입니다.", score));
      } else if ( score >= 90 && score < 100 )  {
         System.out.println(String.format("if - else if - else  :: %s 는 90이상 1000이하 입니다.", score));
      } else if (  score == 100 )  {
         System.out.println(String.format("if - else if - else  :: %s 는 100 입니다.", score));
      } else {
         System.out.println(String.format("if - else if - else  :: %s 는 잘못 입력된 점수 입니다.", score));
      }
   }
}
```

```
if :: 10 < 20 는 만족 합니다.
if - else :: 10 < 20 는 만족 합니다.
if - else if - else  :: 88 는 80이상 90이하 입니다.
```

3. switch

switch

- 변수값에 따라서 특정 위치로 이동
- break 를 사용 해서 switch를 탈출 해야 함.

```
switch(수식 또는 변수) {
    case 상수1:
      명령문;
      break;
    case 상수2:
      명령문;
      break;
    default:
      명령문;
      break;
}
```

```java
public class ControlSwitch {
   public static void main(String[] args) {
      int num = 2;
      switch(num) {
         case 0:
         case 1:
         case 2:
            System.out.println("인위적을 1,2,3 수용");
            break;
         case 3:
            System.out.println("num = 3");
            break;
         case 4:
            System.out.println("num = 4");
            break;
         case 5:
            System.out.println("num = 5");
            break;
         default:
            System.out.println("num가 5 보다 큽니다.");
      }
   }
}
```

```
인위적을 1,2,3 수용
```

4. for

for

- 반복 횟수를 정해 놓고 실행 구문을 반복
- 반복문을 벗어나기 위해서 -> break 사용

➢ for (초기값 ; 조건식; 증감식)
➢ for (type 요소 : Object) : foreach

```
public class ControlFor {
   public static void main(String[] args) {
      int index ;
      int num = 0;
      System.out.println("==== for 문 안에서 후위형 증감");
      for ( index = 0; index < 5; index++) {
         num = ++num;
         System.out.println(String.format("i : %s , num : %s", index, num));
      }
      System.out.println("==== for 문 안에서 전위형 증감");
      num = 0;
      for ( int i = 0; i < 5; i++) {
         num = num++;
         System.out.println(String.format("i : %s , num : %s", i, num));
      }
      System.out.println("==== 문자열 한 단어 씩 출력");
      String str = new String("자바 언어");
      for ( int i = 0; i < str.length(); i++) {
         System.out.println(String.format("i : %s , num : %s", i, str.charAt(i)));
      }

      System.out.println("==== 문자열 한 단어 씩 출력 ( foreach ) ");
      char[] chstr = str.toCharArray();  // 문자열을 문자 배열로 변환
      for ( char c : chstr) {
         System.out.println(String.format("c : %s" , c));
      }

   }
}
```

전역

지역

Why : num은 0일까 ?

```
==== for 문 안에서 후위형 증감
i : 0 , num : 1
i : 1 , num : 2
i : 2 , num : 3
i : 3 , num : 4
i : 4 , num : 5
```

➢ 변수 index의 범위는 main 함수 내에서 유효 하므로 다른 for 문안에서 사용 불가
-> for 문은 블록 범위 이므로 index는 for 문에서 선언하는 것이 좋음
for (int index = 0; index < 5; i++) { ...)

```
==== for 문 안에서 전위형 증감
i : 0 , num : 0
i : 1 , num : 0
i : 2 , num : 0
i : 3 , num : 0
i : 4 , num : 0
```

➢ 후위 증감 연산자를 사용 하였으므로 언제나 0

```
==== 문자열 한 단어 씩 출력
i : 0 , num : 자
i : 1 , num : 바
i : 2 , num :
i : 3 , num : 언
i : 4 , num : 어
```

➢ String은 forEach 사용 불가
➢ 문자 하나씩 얻기 위해서 charAt 사용

```
==== 문자열 한 단어 씩 출력 ( foreach )
c : 자
c : 바
c :
c : 언
c : 어
```

➢ 문자열을 한 단어 씩 처리 하기 위해서는 문자배열로 변환 (toCharArray())
➢ 참조형 (Object)는 forEach 사용

5. while

while

- 조건식이 참인 동안 중괄호안의 실행문 처리
- 한번도 실행이 되지 않을 수 있음
- 무한 루프 처리 될 수 있으므로 주의 -> break 사용

➢ while(조건식) { 실행문 }

```
public class ControlWhile {
   public static void main(String[] args) {
      int num = 0;
      System.out.println(String.format("시작 : num : %s", num));
      while ( num < 5) {
         num = ++num;
         System.out.println(String.format("num : %s", num));
      }
      System.out.println(String.format("종료 : num : %s", num));
      while ( num < 10) {
         num = ++num;
         if ( num == 8 ) {
            System.out.println(String.format("beak 탈출 num : %s", num));
            break;
         }
         System.out.println(String.format("num : %s", num));
      }
      System.out.println(String.format("종료 : num : %s", num));
   }
}
```

```
시작 : num : 0
num : 1
num : 2
num : 3
num : 4
num : 5
종료 : num : 5
```

➢ 후위 증감 함수로 변경 하면 무한 루프

```
num : 6
num : 7
beak 탈출 num : 8
종료 : num : 8
```

➢ break로 탈출

6. do .. while

do .. while

- 한번 실행 후 while의 조건식이 만족 하지 않으면 탈출
- 무조건 한번 실행
- 무한 루프 처리 될 수 있으므로 주의 -> break 사용

➢ do { 실행문 } while(조건식)

```
public class ControlDoWhile {
   public static void main(String[] args) {
      int num = 0;
      System.out.println(String.format("시작 : num : %s", num));
      do {
         num = ++num;
         System.out.println(String.format("num : %s", num));
      } while ( num < 1);
      System.out.println(String.format("종료 : num : %s", num));

      do {
         num = ++num;
         System.out.println(String.format("num : %s", num));
      } while ( num < 5);
      System.out.println(String.format("종료 : num : %s", num));

      do {
         num = ++num;
         if (num == 8) {
            System.out.println(String.format("탈출 num : %s", num));
            break;
         }
         System.out.println(String.format("num : %s", num));
      } while ( num < 10);
      System.out.println(String.format("탈출 종료 : num : %s", num));
   }
}
```

```
시작 : num : 0
num : 1
종료 : num : 1
```

➢ 무조건 한번 실행

```
num : 2
num : 3
num : 4
num : 5
종료 : num : 5
```

➢ 조건식 만족 할 때 까지 실행

```
num : 6
num : 7
탈출 num : 8
탈출 종료 : num : 8
```

➢ break로 탈출

7. break, continue

break, continue

- break : 반복문 탈출 -> 중첩인 경우 실행 중인 반복문 탈출
- continue : 다시 반복문 실행

```
public class ControlBreakContinue {
   public static void main(String[] args) {
      int num = 0;
      for (int x = 0; x < 5; x++) {
         if (x == 4) {
            System.out.println(String.format("스킵 x : %s ,  num : %s", x, num));
            continue;
         }
         for (int y = 0; y < 5; y++) {
            num = ++num;
            if (y == 3) {
               System.out.println(String.format("탈출 x : %s , y : %s,  num : %s", x, y, num));
               break;
            }
            System.out.println(String.format("x : %s , y : %s,  num : %s", x, y, num));
         }
      }
   }
}
```

```
x : 0 , y : 0,  num : 1
x : 0 , y : 1,  num : 2
x : 0 , y : 2,  num : 3
탈출 x : 0 , y : 3,  num : 4
x : 1 , y : 0,  num : 5
x : 1 , y : 1,  num : 6
x : 1 , y : 2,  num : 7
탈출 x : 1 , y : 3,  num : 8
x : 2 , y : 0,  num : 9
x : 2 , y : 1,  num : 10
x : 2 , y : 2,  num : 11
탈출 x : 2 , y : 3,  num : 12
x : 3 , y : 0,  num : 13
x : 3 , y : 1,  num : 14
x : 3 , y : 2,  num : 15
탈출 x : 3 , y : 3,  num : 16
스킵 x : 4 ,  num : 16
```