

Java Web Programming 입문

Java Basic #03

오늘의 키워드

- ▶ 형 변환 (type conversion)
 - 암시적 형 변환 (implicit type conversion)
 - 명시적 형 변환 (explicit type conversion)
- ▶ 주석 (comment)
 - 한 줄 주석 (single-line comment)
 - 여러 줄 주석 (multi-line comment)
- ▶ 연산자 (operator)
 - 단항 연산자
 - 산술 연산자
 - 비교 연산자
 - 논리 연산자
 - 삼항 연산자
 - 대입 연산자
- ▶ 제어문 (control statement)
 - 조건문 (conditional statement) / 분기문 (branch statement)
 - if
 - switch ~ case



형 변환 (type conversion)

▶ 의미

- 어떤 데이터의 타입이 다른 데이터 타입으로 바뀌는 현상
- 똑같은 값으로 보여도, 담기는 메모리의 크기/종류가 달라짐

▶ 종류

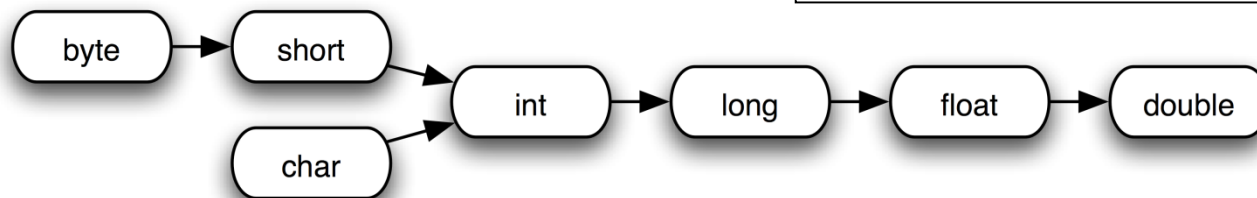
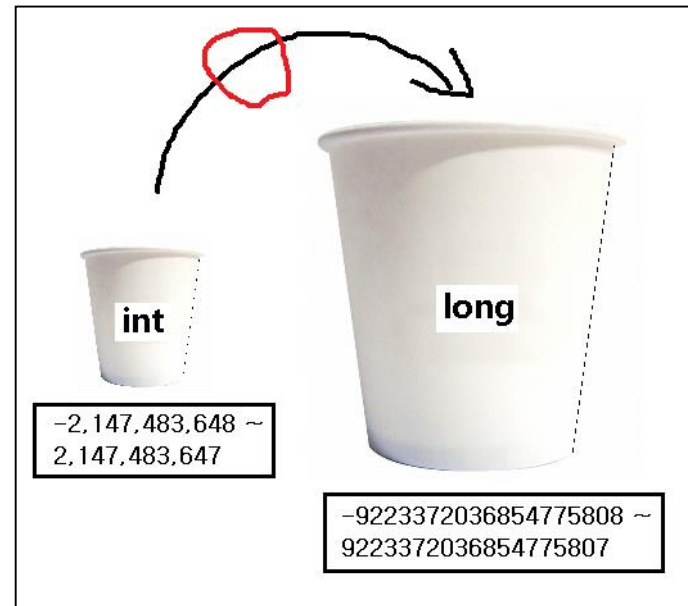
- 암시적 형 변환(implicit type conversion)
- 명시적 형 변환(explicit type conversion)



형 변환 (type conversion)

▶ 암시적 형 변환(implicit type conversion)

- 작은 컵의 물 -> 큰 컵
- No problem
- 자동



형 변환 (type conversion)

▶ 암시적 형 변환 (implicit type conversion)

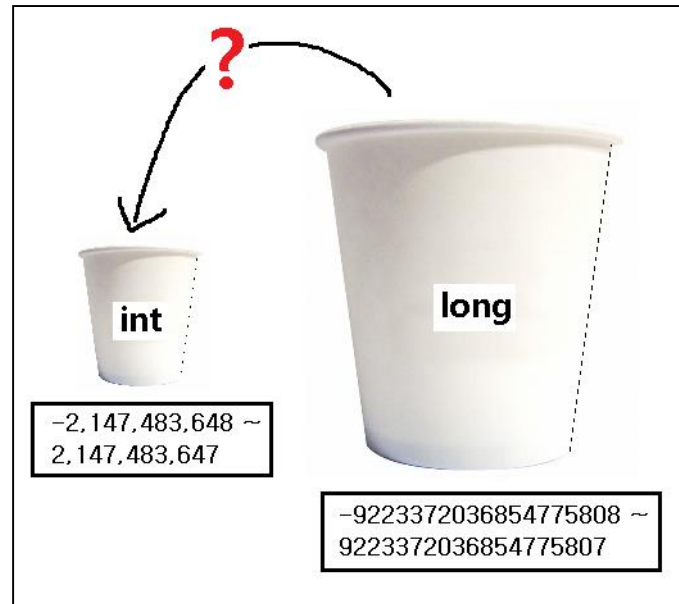
```
public class ImplicitTypeConversion{  
  
    public static void main(String[] args) {  
        byte byteNum = 100;  
        System.out.println("byte : " + byteNum);  
  
        short shortNum = byteNum;  
        System.out.println("short : " + shortNum);  
  
        int intNum = shortNum;  
        System.out.println("int : " + intNum);  
  
        long longNum = intNum;  
        System.out.println("long : " + longNum);  
  
        float floatNum = longNum;  
        System.out.println("float : " + floatNum);  
  
        double doubleNum = floatNum;  
        System.out.println("double : " + doubleNum);  
    }  
}
```



형 변환 (type conversion)

▶ 명시적 형 변환(explicit type conversion)

- 큰 겁의 물 -> 작은 컵
- 데이터의 **유실**
- 수동



형 변환 (type conversion)

▶ 명시적 형 변환(explicit type conversion)

```
public class ExplicitTypeConversion{  
  
    public static void main(String[] args){  
  
        double doubleNum = 3.14159265358979323846;  
        System.out.println("double : " + doubleNum);  
  
        float floatNum = (float) doubleNum;  
        System.out.println("float : " + floatNum);  
  
        long longNum = (long) floatNum;  
        System.out.println("long : " + longNum);  
  
        int intNum = (int) longNum;  
        System.out.println("int : " + intNum);  
  
        short shortNum = (short) intNum;  
        System.out.println("short : " + shortNum);  
  
        byte byteNum = (byte) shortNum;  
        System.out.println("byte : " + byteNum);  
  
    }  
}
```



```
double doubleNum2 = 987654.321;  
System.out.println("double : " + doubleNum2);  
  
float floatNum2 = (float) doubleNum2;  
System.out.println("float : " + floatNum2);  
  
long longNum2 = (long) floatNum2;  
System.out.println("long : " + longNum2);  
  
int intNum2 = (int) longNum2;  
System.out.println("int : " + intNum2);  
  
short shortNum2 = (short) intNum2;  
System.out.println("short : " + shortNum2);  
  
byte byteNum2 = (byte) shortNum2;  
System.out.println("byte : " + byteNum2);
```

주석 (comment)

- ▶ 컴파일 대상에서 제외



- ▶ 프로그래머를 위한

- 커뮤니케이션
- 설명
- 사용 방법
- 주의사항
- 메모

```
/*#####  
## 작성일 : 2014-01-12 ##  
## 작성자 : 홍길동 선임 ##  
## 연락처 : XX) XXX - XXXX (XX사 XX팀) ##  
## 작성내용 : 주석처리 예제 ##  
#####*/
```

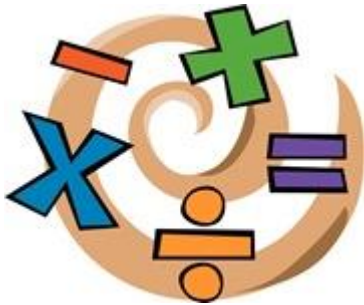
- ▶ 종류

- 한 줄 주석 (single-line comment)
- 여러 줄 주석 (multi-line comment)

```
// Single line comment1  
  
/* Multi  
   line  
   comment  
*/
```


연산자 (operator)

- ▶ 데이터의 연산
- ▶ 데이터 타입에 따라



- ▶ 단항 연산자
 - 증감 연산자
 - ++, --
 - 부호 연산자
 - +, -
 - 논리부정 연산자
 - !
 - 비트 전환 연산자
 - ~
- ▶ 산술 연산자
 - 오칙 연산자
 - +, -, *, /, %
 - 쉬프트 연산자
 - <<, >>, >>>
- ▶ 비교 연산자
 - 대소비교 연산자
 - <, >, <=, >=
 - 등가비교 연산자
 - ==, !=
 - 비트 연산자
 - &, |, ^
- ▶ 논리 연산자
 - &&, ||
- ▶ 삼항 연산자
 - ? :
- ▶ 대입 연산자
 - =, +=, -=
 - *=, /=, %=, <<=, >>=, >>>=, &=, ^=, |=

연산자 (operator)

▶ 단항 연산자

- 증감 연산자 (중요)

- ++
- --

- 부호 연산자

- +
- -

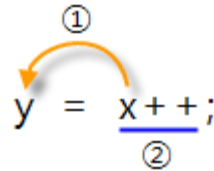
- 논리부정 연산자 (중요)

- !

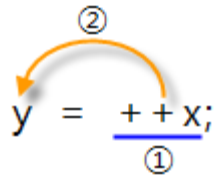
- 비트 전환 연산자 (pass)

- ~

①
y = x++;
②

A diagram illustrating the post-increment operator. It shows the code 'y = x++;'. An orange arrow labeled '①' points from the 'x' to the assignment operator '=', indicating the value of x is used in the assignment. Another orange arrow labeled '②' points from the '++' to the 'x', indicating the increment operation is performed on x after the assignment.

②
y = ++x;
①

A diagram illustrating the pre-increment operator. It shows the code 'y = ++x;'. An orange arrow labeled '②' points from the '++' to the assignment operator '=', indicating the increment operation is performed on x before the assignment. Another orange arrow labeled '①' points from the 'x' to the assignment operator '=', indicating the value of x is used in the assignment.

□ TRUE
□ FALSE

연산자 (operator)

증감 연산자

연산자	사용	설명
++	++ A	A의 값을 1증가시킨 후, A를 처리
	A ++	A를 처리한 후 A의 값을 1 증가시킴
--	-- A	A의 값을 1 감소시킨 후, A를 처리
	A --	A를 처리한 후, A의 값을 1 감소시킴

```
public class IncreaseDecrease{  
  
    public static void main(String[] args){  
  
        int a = 10;  
        System.out.println("현재 a의 값 : " + a);  
        System.out.println("[++a] 출력 : " + (++a));  
        System.out.println("현재 a의 값 : " + a);  
        System.out.println("[a++] 출력 : " + (a++));  
        System.out.println("현재 a의 값 : " + a);  
        System.out.println("[--a] 출력 : " + (--a));  
        System.out.println("현재 a의 값 : " + a);  
        System.out.println("[a--] 출력 : " + (a--));  
        System.out.println("현재 a의 값 : " + a);  
    }  
}
```

연산자 (operator)

▶ 산술 연산자

◦ 오직 연산자 (중요)

- +
- -
- *
- /
- %

◦ 시프트 연산자 (pass)

- <<
- >>
- >>>



연산자 (operator)

▶ 산술 연산자

연산자	사용	설명
+	A + B	A값과 B 값을 더한다.
-	A - B	A값에 B값을 뺀다.
*	A * B	A값과 B값을 곱한다.
/	A / B	A값에 B값을 나눈다.
%	A % B	A값을 B값으로 나눈 나머지를 구한다.

```
public class Arithmetic{  
  
    public static void main(String[] args){  
  
        int a = 10;  
        int b = 3;  
  
        System.out.println("a의 값 : " + a);  
        System.out.println("b의 값 : " + b);  
  
        System.out.println(a + " + " + b + " = " + (a + b));  
        System.out.println(a + " - " + b + " = " + (a - b));  
        System.out.println(a + " * " + b + " = " + (a * b));  
        System.out.println(a + " / " + b + " = " + (a / b));  
        System.out.println(a + " % " + b + " = " + (a % b));  
  
    }  
}
```

연산자 (operator)

▶ 비교 연산자

- 대소비교 연산자 (중요)

- $<$
- $>$
- $<=$
- $>=$

- 등가비교 연산자 (중요)

- $==$
- $!=$

- 비트 연산자 (pass)

- $\&$
- $|$
- \wedge



연산자 (operator)

▶ 비교 연산자

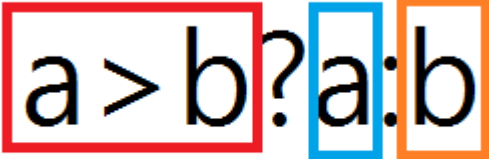
연산자	사용	설명
==	A == B	A의 값과 B의 값이 같은 경우 true, 다를 경우 false
!=	A != B	A의 값과 B의 값이 다를 경우 true, 같을 경우 false
>	A > B	A의 값이 B의 값보다 클 경우 true, 그렇지 않을 경우 false
>=	A >= B	A의 값이 B의 값보다 크거나 같으면 true, 그렇지 않으면 false
<	A < B	A의 값이 B의 값보다 작을 경우 true, 그렇지 않을 경우 false
<=	A <= B	A의 값이 B의 값보다 작거나 같으면 true, 그렇지 않으면 false

```
public class Compare{  
  
    public static void main(String[] args){  
  
        int a = 10;  
        int b = 3;  
        System.out.println("a : " + a + ", b : " + b);  
        System.out.println("a > b : " + (a > b));  
        System.out.println("a >= b : " + (a >= b));  
        System.out.println("a < b : " + (a < b));  
        System.out.println("a <= b : " + (a <= b));  
        System.out.println("a == b : " + (a == b));  
        System.out.println("a != b : " + (a != b));  
    }  
}
```

연산자 (operator)

▶ 논리 연산자 (중요)

- &&
- ||


비교문 TRUE FALSE

▶ 삼항 연산자 (중요: 좀 할 줄 아는 것처럼 보인다)

- (조건식) ? 식1 : 식2
- result = ((x > 0) ? x : -x);
- 행동 = ((니가 쓸래) ? 지갑을 꺼낸다 : 화장실에 간다);
- if 문으로 바꿔 쓸 수 있다.

▶ 대입 연산자 (중요)

- = (항상 쓴다)
- +=, -=, *=, /=, %= (꽤 쓴다)
- <<=, >>=, >>>=, &=, ^=, |= (pass)

연산자 (operator)

▶ 논리 연산자

연산자	사용	설명
&&	A && B	A의 값과 B의 값이 모두 true 일 경우 true, 그 외에는 false 반환
	A B	A의 값과 B의 값이 모두 false 일 경우 false, 그 외에는 true 반환
!	! A	A의 값이 true일 경우 false, false일 경우 true 반환

&& 연산자	true	false	연산자	true	false
true	true	false	true	true	true
false	false	false	false	true	false

```

public class Logical{

    public static void main(String[] args){

        int num1 = 10;
        int num2 = 5;
        System.out.println("num1 : " + num1 + ", num2 : " + num2);

        boolean bool1 = num1 > num2;
        boolean bool2 = num1 < num2;
        boolean bool3 = num1 == num2;
        boolean bool4 = num1 != num2;
        System.out.println("bool1 : " + bool1);
        System.out.println("bool2 : " + bool2);
        System.out.println("bool3 : " + bool3);
        System.out.println("bool4 : " + bool4);

        System.out.println("bool1 && bool2 : " + (bool1 && bool2));
        System.out.println("bool1 || bool2 : " + (bool1 || bool2));
        System.out.println("bool1 && bool4 : " + (bool1 && bool4));
        System.out.println("bool2 || bool3 : " + (bool2 || bool3));

    }
}
    
```

연산자 (operator)

▶ 대입 연산자

연산자	사용	동일 표현	설명
=	A = B		A에 B의 값을 대입
+=	A += B	A = A + B	A값에 B값을 더한 결과값을 A에 대입
-=	A -= B	A = A - B	A값에 B값을 뺀 결과값을 A에 대입
*=	A *= B	A = A * B	A값에 B값을 곱한 결과값을 A에 대입
/=	A /= B	A = A / B	A값에 B값을 나눈 결과값을 A에 대입
%=	A %= B	A = A % B	A값에 B값을 나눈 나머지 결과값을 A에 대입

```
public class Substitution{  
  
    public static void main(String[] args){  
  
        int a = 10;  
        int b = 3;  
        System.out.println("a : " + a + ", b : " + b);  
        a = b;  
        System.out.println("a = b 수행, " + "현재 a의 값 : " + a);  
        a += b;  
        System.out.println("a += b 수행, " + "현재 a의 값 : " + a);  
        a -= b;  
        System.out.println("a -= b 수행, " + "현재 a의 값 : " + a);  
        a *= b;  
        System.out.println("a *= b 수행, " + "현재 a의 값 : " + a);  
        a /= b;  
        System.out.println("a /= b 수행, " + "현재 a의 값 : " + a);  
        a %= b;  
        System.out.println("a %= b 수행, " + "현재 a의 값 : " + a);  
  
    }  
}
```

연산자 (operator)

우선순위

Prio- rity	Opera- tor	Name	Asso- ciativity	Example
1	++	Increment	r	$x++$ $++x$
	--	Decrement	r	$x--$ $--x$
	+	Unary plus	r	$+x$
	-	Unary minus	r	$-x$
	!	Logical complement	r	<code>!isOpen</code>
	~	Bitwise complement	r	$\sim i$
	(type)	Cast	r	$i = (\text{int}) x$

2	*	Multiplication	l	$x * 2$
	/	Division	l	$x / 2$
	%	Remainder	l	$x \% 2$

3	+	Binary plus	l	$x + 2$ " " + x + i
	-	Binary minus	l	$x - i$

4	<<	Shift left	l	$i << 2$
	>>	Shift right	l	$-i >> 2$
	>>>	Shift right ignore sign	l	$-i >>> 2$

5	>	greater than	l	$i > x$
	<	less than	l	$i < x$
	>=	greater equal	l	$i >= x$
	<=	less equal	l	$i <= x$
	instanceof	Type check	l	<code>s instanceof String</code>

연산자 (operator)

Prio- rity	Opera- tor	Name	Asso- ciativity	Example
6	<code>==</code>	Equals	1	<code>i == j</code> <code>s == ""</code>
	<code>!=</code>	Not equal	1	<code>i != j</code> <code>s != null</code>
7	<code>&</code>	Bitwise and	1	<code>i & j</code>
8	<code>^</code>	Exclusive or	1	<code>i ^ 5</code>
9	<code> </code>	Bitwise or	1	<code>i j</code>
10	<code>&&</code>	Logical and	1	<code>isOpen && false</code>
11	<code> </code>	Logical or	1	<code>isOpen false</code>
12	<code>?:</code>	Conditional	r	<code>i < 0 ? -1 : 1</code>

13	<code>=</code>	Assignment	r	<code>j = i</code> <code>o = s;</code>
	<code>+=</code>	Plus assignment	r	<code>j += x</code>
	<code>-=</code>	Minus assignment	r	<code>j -= x</code>
13	<code>*=</code>	Multiplication assign.	r	<code>j *= x</code>
	<code>/=</code>	Division assign.	r	<code>j /= x</code>
	<code>&=</code>	Bitwise and assign.	r	<code>j &= i</code>
	<code> =</code>	Bitwise or assign.	r	<code>j = i</code>
	<code>^=</code>	Exclusive or assign.	r	<code>j ^= i</code>
	<code>%=</code>	Remainder assign.	r	<code>j %= i</code>
	<code><<=</code>	Shift left assign.	r	<code>j <<= i</code>
	<code>>>=</code>	Shift right assign.	r	<code>j >>= i</code>
	<code>>>>=</code>	Shift right i.s. assign.	r	<code>j >>>= i</code>



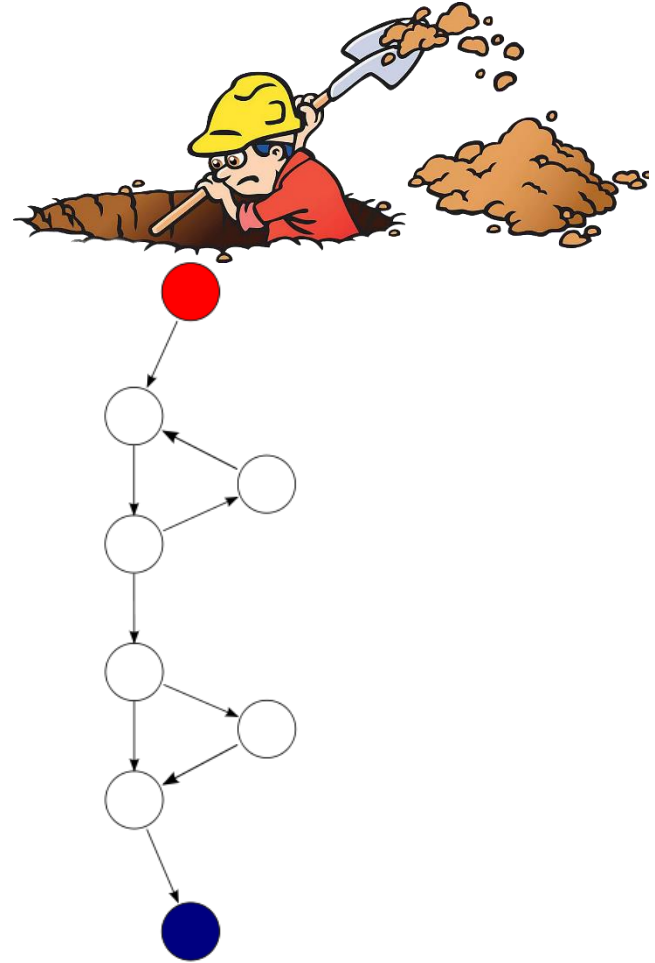
연산자 (operator)

종 류	연산방향	연산자	우선순위
단항 연산자	←	++ -- + - ~ ! (타입)	높음
산술 연산자	→	* / %	
	→	+ -	
	→	<< >> >>>	
비교 연산자	→	< > <= >= instanceof	
	→	== !=	
논리 연산자	→	&	낮음
	→	^	
	→		
	→	&&	
	→		
삼항 연산자	→	?:	
대입 연산자	←	= *= /= %= += -= <<=	
	←	>>= >>>= &= ^= =	



제어문 (control statement)

- ▶ top-down X
- ▶ 흐름 제어 (flow control)
- ▶ 조건식 (condition)
- ▶ 참(true) / 거짓(false)
- ▶ 반복 (repetition)
- ▶ 이제부터 본격적인 프로그래밍



제어문 (control statement)

▶ 조건문 (conditional statement)

◦ 분기문 (branch statement)

◦ 실행 방식

- 조건이 맞아떨어진다면(true) 실행
- 조건이 맞지 않다면(false) 실행안하고 넘어감



◦ 종류

- if 문
- switch ~ case 문
- 연산자지만 삼항 연산자도 비슷한 느낌적인 느낌



제어문 (control statement)

▶ 조건문 (conditional statement)

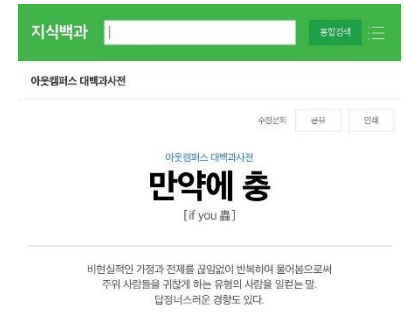
◦ if 문

- 조건식
- 범용성
- 사용방식

```
if ( condition ){  
    condition이 true일때 실행될 내용  
}
```

```
if ( condition1 ){  
    condition1이 true일때 실행될 내용  
}else{  
    condition1이 false일때 실행될 내용  
}
```

```
if ( condition1 ){  
    condition1이 true일때 실행될 내용  
  
}else if( condition2 ){  
    condition2이 true일때 실행될 내용  
  
}else if( condition3 ){  
    condition3이 true일때 실행될 내용  
    .  
    .  
    .  
}else{  
    if/else if의 조건식이 모두 false일때 실행될 내용  
}
```



제어문 (control statement)

▶ 조건문 (conditional statement)

◦ if 문

```
public class IfTest01{  
    public static void main(String[] args){  
        int num1 = 10;  
        int num2 = 3;  
  
        // 단일 if  
        if(num1 > num2){  
            System.out.println("num1이 num2보다 크다");  
        }  
        System.out.println("실행 끝!");  
  
        // if, else 구문  
        if(num1 < num2){  
            System.out.println("num1은 num2보다 작다");  
        }else{  
            System.out.println("num1은 num2보다 작지 않다");  
        }  
    }  
}
```

제어문 (control statement)

▶ 조건문 (conditional statement)

◦ if 문

```
public class IfTest02{  
    public static void main(String[] args){  
        int score = 83;  
  
        if(score > 90){  
            System.out.println("A학점");  
        }else if(score > 80){  
            System.out.println("B학점");  
        }else if(score > 70){  
            System.out.println("C학점");  
        }else if(score > 60){  
            System.out.println("D학점");  
        }else{  
            System.out.println("F학점");  
        }  
    }  
}
```

제어문 (control statement)

▶ 조건문 (conditional statement)

◦ if 문

• 중첩 if 문

```
if ( condition1 ){
```

condition1이 true일때 실행될 내용

```
    if ( condition2 ){
```

condition1과 condition2가 true일때 실행될 내용

```
    }else{
```

condition1이 true이고,

condition2는 false일때 실행될 내용

```
    }
```

```
}else {
```

condition1이 false일때 실행될 내용

```
}
```



제어문 (control statement)

▶ 조건문 (conditional statement)

◦ if 문

• 중첩 if 문

```
public class IfTest03{  
    public static void main(String[] args){  
        int num = 7;  
  
        if( num % 2 == 0 ){  
            if( num > 0 ){  
                System.out.println(num + "은 짝수인 양수");  
            }else if( num > 0 ){  
                System.out.println(num + "은 0");  
            }else{  
                System.out.println(num + "은 짝수인 음수");  
            }  
        }else{  
            if( num > 0 ){  
                System.out.println(num + "은 홀수인 양수");  
            }else{  
                System.out.println(num + "은 홀수인 음수");  
            }  
        }  
    }  
}
```

제어문 (control statement)

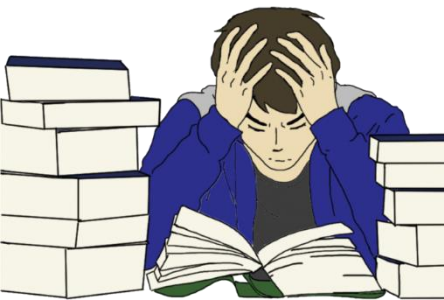
▶ 조건문 (conditional statement)

◦ if 문

- 중첩 if 문
- 숙제

```
public class IfTestHW{
    public static void main(String[] args){
        int num = 7;

        if( num % 2 == 0 && num > 0 ){
            System.out.println(num + "은 짝수인 양수");
        }else if( num % 2 == 0 && num == 0 ){
            System.out.println(num + "은 0");
        }else if( num % 2 == 0 && !(num > 0 && num == 0) ){
            System.out.println(num + "은 짝수인 음수");
        }else if( !(num % 2 == 0) && num > 0 ){
            System.out.println(num + "은 홀수인 양수");
        }else{
            System.out.println(num + "은 홀수인 음수");
        }
    }
}
```



제어문 (control statement)

- ▶ 조건문 (conditional statement)
 - switch ~ case
 - 하나의 값에 대한 여러가지 케이스
 - If문으로 변경 가능

```
switch( 어떤값이 나올 수 있는 코드 혹은 연산식 ) {  
  case [case1]:  
    코드 혹은 연산식의 결과가 case1 일때 실행될 내용  
    break;  
  case [case2]:  
    코드 혹은 연산식의 결과가 case2 일때 실행될 내용  
    break;  
  .  
  .  
  .  
  default :  
    코드 혹은 연산식의 결과가  
    위 case들의 값중 해당사항이 없을때 실행될 내용  
    break;  
}
```

국어사전 검색 검색

전체 단어 속담/관용구 예문 본문 발음법/표기법

*케바케*에 대한 검색 결과입니다.

단어 (1)

케바케 지식백과사전

‘케바케’란 ‘case by case (케이스 바이 케이스)’의 줄임말. 사전적인 의미로는 ‘경우에 따라서, 개별적으로, 사례별로’라는 뜻. ‘원칙이 없이, 그때 그때 다르다’라는 의미로 사용되고 있는 말.

제어문 (control statement)

- ▶ 조건문 (conditional statement)
 - switch ~ case

```
public class SwitchCaseTest01 {  
    public static void main(String[] args) {  
        int jumsoo = 87;  
  
        switch( jumsoo / 10 ){  
            case 10:  
                System.out.println("A학점입니다. ");  
                break;  
            case 9:  
                System.out.println("A학점입니다. ");  
                break;  
            case 8:  
                System.out.println("B학점입니다. ");  
                break;
```

```
            case 7:  
                System.out.println("C학점입니다. ");  
                break;  
            case 6:  
                System.out.println("D학점입니다. ");  
                break;  
            default :  
                System.out.println("F학점입니다. ");  
                break;  
        }  
    }  
}
```