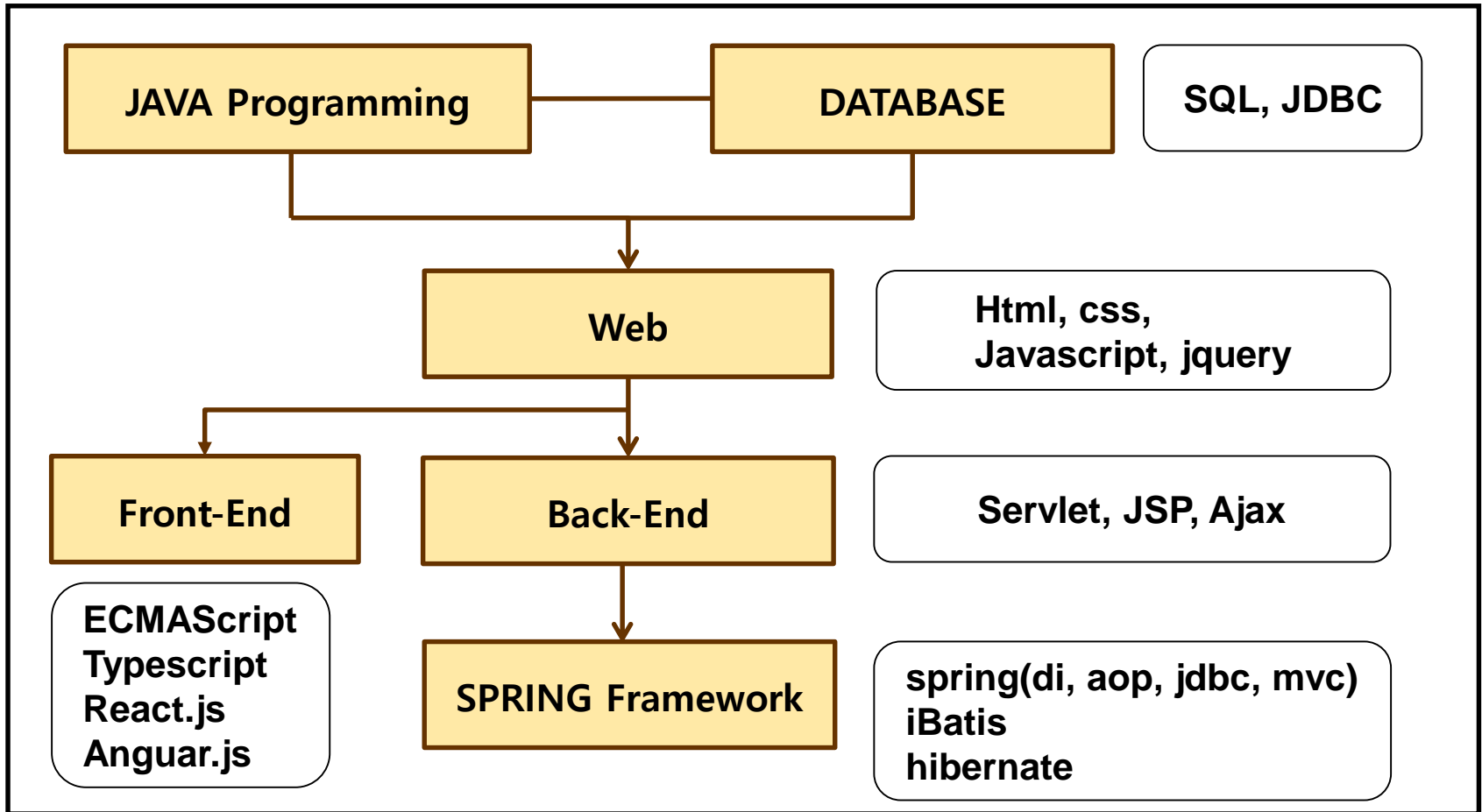


# 자바 기초

# 과정 로드맵





# 수업을 하기 위한 준비

0 절. 디렉토리 만들기

1 절. 필요파일 다운로드

- JDK : 1.8버전
- 편집기 : 이클립스 4.8 Photon

### - 수업 디렉토리 만들기

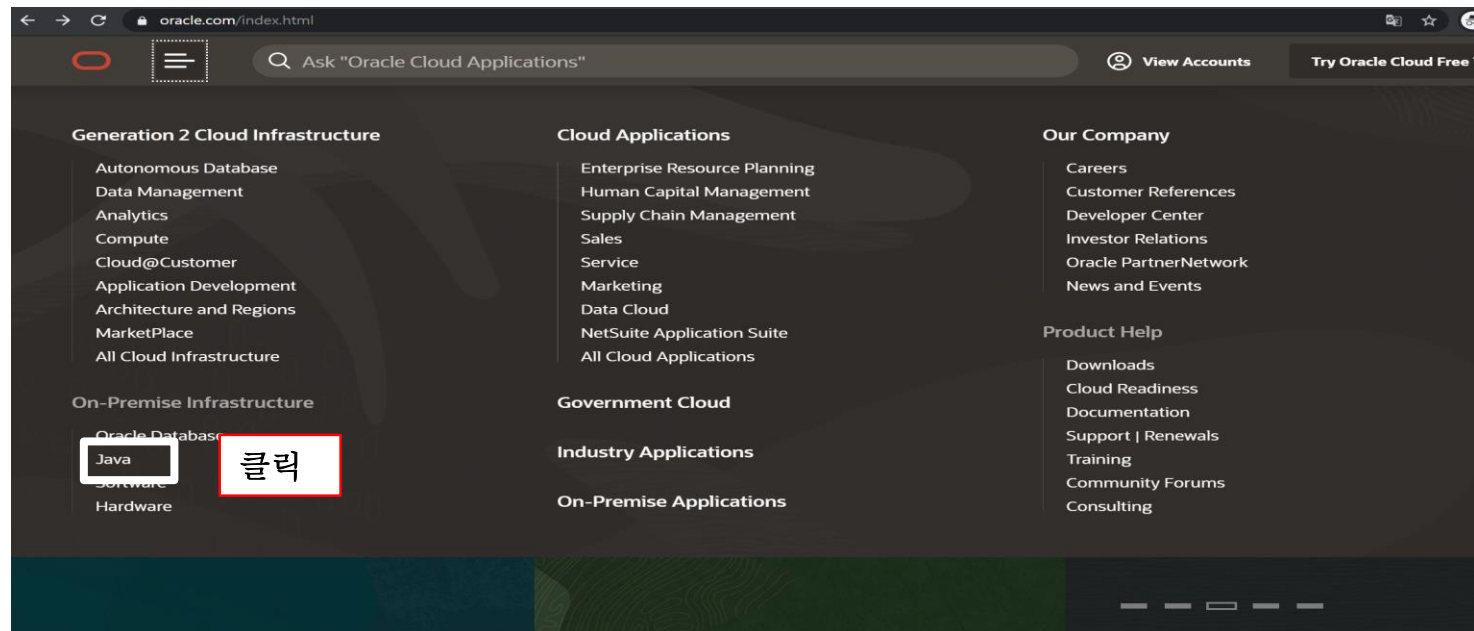
**d:\\ lecture** : 수업 동안 사용될 루트 폴더

- **bin** : 설치 경로 (**Eclipse**)
- **java-workspace** : 작업 소스 폴더
- **setup** : 수업에 필요한 설치파일

- 다운로드 링크

<http://www.oracle.com>

- 메뉴에서 **JAVA** 선택



### - Oracle Java Products 에서 Java SE 클릭 후 Lear more 버튼 클릭

The screenshot shows the Oracle Java Products page. The browser address bar displays 'oracle.com/java/'. The page features three main statistics: '3 Billion devices run Java—in your home, your car, and your office.', '12 Million Java developers worldwide.', and '25+ Billion Java Cards sold.' Below these is a green button labeled 'Learn how Safe Water Kenya uses Java'. The main content area is divided into sections. The first section, 'Oracle Java Products', is highlighted with a blue box and a red circle containing the number '1', with a red box labeled '클릭' (Click) next to it. The second section, 'Java SE', is highlighted with a blue box and a red circle containing the number '2', with a red box labeled '클릭' (Click) next to it. The third section, '#1 Development Platform', is highlighted with a blue box and a red circle containing the number '3', with a red box labeled '클릭' (Click) next to it. The page also includes an 'Open all +' link and a 'Java Card' link at the bottom.

1 Oracle Java Products 클릭

2 Java SE 클릭

3 Learn more 클릭

## - Downloads 탭 선택 후 JDK 다운

The screenshot shows the Oracle Java SE 8u241 download page. It includes a navigation bar with tabs: Overview, Downloads, Documentation, Community, Technologies, and Training. The Downloads tab is selected and highlighted with a blue box and a red '1' in a circle, with a red box labeled '클릭' (Click) next to it. Below the navigation bar, there is a section for 'Java SE at a Glance' and 'Java SE Development Kit 8u241'. To the right, there is a section for 'Oracle JDK' with a yellow 'JDK Download' button highlighted with a blue box and a red '2' in a circle, with a red box labeled '클릭' (Click) next to it. Below this, there are links for 'Server JRE Download' and 'JRE Download'. At the bottom, there is a table with columns: Product / File Description, File Size, and Download. The table lists two options: 'Windows x86' (200.86 MB) and 'Windows x64' (210.92 MB). The 'Windows x64' row is highlighted with a blue box and a red '3' in a circle, with a red box labeled '클릭' (Click) next to it.

Java SE 8u241

Java SE 8u241 includes important bug fixes. Oracle strongly recommends that all Java SE 8 users upgrade to this release.

- Documentation
- Installation Instructions

Overview Downloads Documentation Community Technologies Training

Java SE at a Glance

Java SE Development Kit 8u241

This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE

Product / File Description	File Size	Download
Windows x86	200.86 MB	jdk-8u241-windows-i586.exe
Windows x64	210.92 MB	jdk-8u241-windows-x64.exe

- Windows x86 → 32bit, Windows x64 → 64bit 일 경우 선택

- 다운로드 링크

<http://eclipse.org>

- Download 버튼 클릭 -> Download Packages 클릭 ->

The screenshot shows the Eclipse IDE download page. It includes the following elements and annotations:

- Step 1:** A red circle with the number '1' is placed over the 'Download' button in the top navigation bar. A red box labeled '클릭' (Click) is next to it.
- Step 2:** A red circle with the number '2' is placed over the 'Download Packages' button in the 'Get Eclipse IDE 2019-12' section. A red box labeled '클릭' (Click) is next to it.
- Step 3:** A red circle with the number '3' is placed over the 'Windows 64-bit' download link. A red box labeled '클릭' (Click) is next to it.
- Navigation:** A red arrow points from the 'Download' button in Step 1 to the 'Download Packages' button in Step 2. Another red arrow points from the 'Download Packages' button in Step 2 to the 'Windows 64-bit' link in Step 3.
- Page Content:** The page header includes 'Log in' and 'Manage Cookies'. The main heading is 'Eclipse IDE for Enterprise Java Developers'. Below it, there is a '353 MB' size indicator and a list of tools: 'Tools for including more. Click here. Click here. Click here.' The 'Get Eclipse IDE 2019-12' section includes the text 'Install your favorite desktop IDE packages.' and a 'Download 64 bit' button. The right side of the page lists operating systems: 'Windows 64-bit', 'Mac Cocoa 64-bit', and 'Linux 64-bit'.

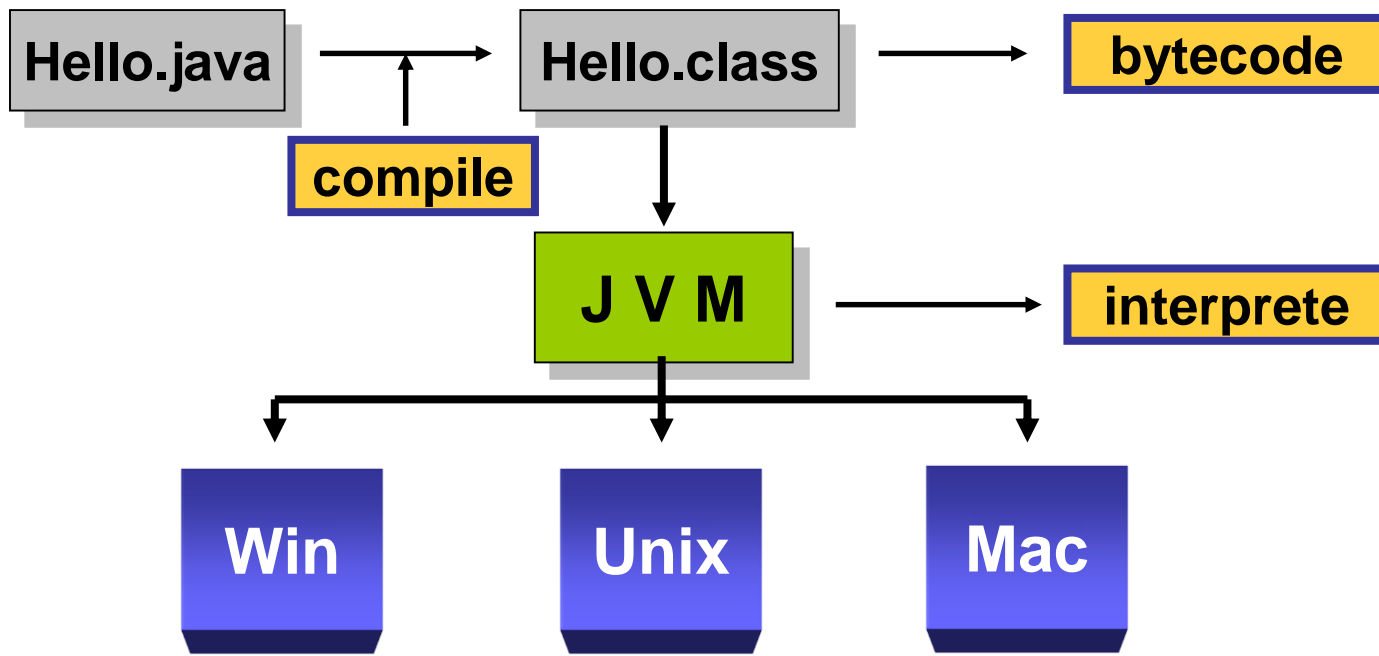




# 설치 및 실행

# 1. 자바언어의 특징

- 단순(Simple) : 메모리 관리(Garbage Collection)
- 객체지향적(Object-Oriented) : 재사용
- 컴파일 + 인터프리터 : OS에 독립적, WORA





## 2. 자바 환경 설정 및 설치

- Java 분야
  - J2SE
  - J2ME
  - J2EE
- Java 다운로드 : <http://www.oracle.com>
- JDK 설치 확인 : `java -version`
- 환경설정
  - JAVA\_HOME 설정
  - PATH 설정

## 2. 자바 환경 설정 및 설치

예제작성

- 편집기 이클립스 설치
  - 다운로드 : <http://eclipse.org/>

### 3. 메인 메소드

- 실행 명령인 java 를 실행 시 가장 먼저 호출 되는 부분
- 만약, Application 에서 main() 메소드가 없다면 절대로 실행 될 수 없음
- Application의 시작 = 특정 클래스의 main( ) 실행
- 형태 (고정된 형태)

```
public static void main(String [ ] args) { }
```

## 4. 출력문

예제작성

- print
- println
- printf
  - %d : 정수
  - %f : 실수
  - %c : 문자
  - %s : 문자열

```
System.out.println( "%s");
```

??

```
System.out.printf( "오늘은 날씨가 %s");
```

??

```
System.out.printf( "오늘은 날씨가 %s", "화창하다");
```

??

```
System.out.printf( "오늘 점심은 %d시에 %s" , 12, "강남식당");
```

??

```
System.out.printf( "오늘의 원달러 환율은 %f 원" , 1019.8);
```

??

```
System.out.printf( "오늘의 원달러 환율은 %7.1f 원" , 1019.8);
```

??

## 5. 클래스의 구조

예제작성

- 클래스 선언부에 올 수 있는 것
  - 주석문
  - 패키지
  - 임포트
  - 클래스 선언
- 클래스 내용부에 올 수 있는 것
  - 멤버변수
  - 메소드
  - 내부클래스



# 식별자, 자료형, 연산자



- 클래스, 메소드, 변수의 이름
- 명명규칙
  1. 클래스 : 단어의 첫 글자를 대문자로 표기. 만약, 여러 개의 단어로 이루어져 있다면 각 단어의 첫글자를 대문자로 표기.  
예> Hello, HelloWorld, BoardMng
  2. 멤버변수, 메소드 : 단어의 첫 글자를 소문자로 표기. 만약, 여러 개의 단어로 이루어져 있다면 각 단어의 첫글자를 대문자로 표기  
예> name, cnt, main(), print(), juminNo, printName()
  3. 상수 : 모든 단어를 대문자로 표기. 만약, 여러 개의 단어로 이루어져 있다면 단어와 단어 사이를 '\_'로 구분한다.  
예> MAX, MIN, MAX\_VALUE, SERVER\_NAME



## 2. 변수

- 데이터를 저장할 메모리의 위치를 나타내는 이름
- 메모리 상에 데이터를 보관할 수 있는 공간을 확보
- 적절한 메모리 공간을 확보하기 위해서 변수의 타입 등장
- '=' 를 통해서 **CPU**에게 연산작업을 의뢰

## 2. 변수

메모리의 단위

- 0과 1을 표현하는 bit
- 8bit = 1byte
- 2bytes = word

bit의 데이터 처리 - 2진수

- 전구의 불이 들어오고 나가는 처리

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1
0	0	0	0	0	1	1	1
					4 +	2 +	1
							결과 7

## 2. 변수

- 변수

- 선언

- 1. 자료형 변수명;

- 2. 예> `int age; String name; ...`

- 초기화

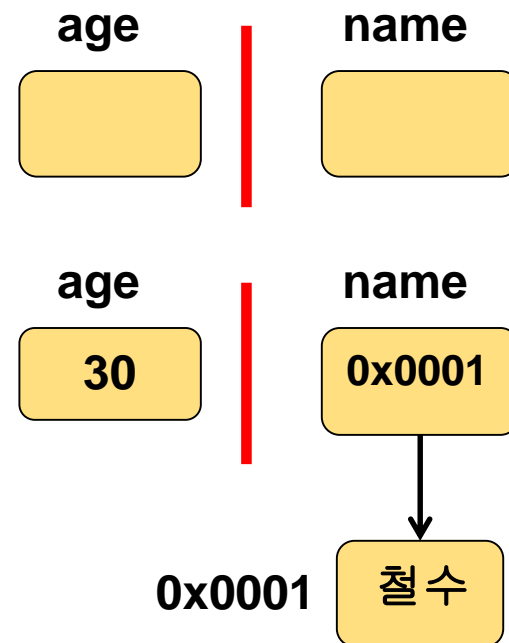
- 1. 변수명 = 저장할 값;

- 2. 예> `age = 30; name = "철수";`

- 선언과 초기화를 동시에

- 1. 자료형 변수명 = 저장할 값;

- 2. 예> `int age = 30;`



## 2. 자료형

- 자료형

- 기본 자료형과 참조 자료형(기본 자료형 8가지 외 모든 것)
- 기본 자료형 (맨 앞의 비트는 부호비트)

타입	세부타입	데이터형	크기	기본값	사용예
논리형		<b>boolean</b>	1bit	false	boolean b = true
문자형		<b>char</b>	2byte	null(\u0000)	char c = 'a', c1 = 65, c2 = '\uffff'
숫자형	정수형	<b>byte</b>	1byte	(byte)0	byte b = 100;
		<b>short</b>	2byte	(short)0	short s = 100;
		<b>int</b>	4byte	0	int i = 100;
		<b>long</b>	8byte	0L	long l = 100, l2 = 100L
	실수형	<b>float</b>	4byte	0.0f	float f = 3.1f, f2 = 3.1F;
		<b>double</b>	8byte	0.0d	double d = 3.1;

## 2. 형변환

예제작성

- 자료형의 크기 비교

byte < short < int < long < float < double

char < int < long < float < double

- 데이터 형변환

### 1) 묵시적(암묵적) : Implicit Casting

: 범위가 넓은 데이터 형에 좁은 데이터 형을 대입하는 것

: 예> `byte b = 100; int i = b;`

### 2) 명시적 : Explicit Casting

: 범위가 좁은 데이터 형에 넓은 데이터 형을 대입하는 것

: 형변환 연산자 사용 - (타입) 값;

: 예> `int i = 100; byte b = i; (X), byte b = (byte) i; (O)`

## 2. 자료형

예제작성

- 기본 자료형과 참조 자료형의 차이

	기본 자료형	참조 자료형
변수값	실제 사용할 값	객체 참조값
정의방식	Java 내부 이미 정의	클래스, 인터페이스, 배열, enum등..
생성방식	19, 3.14, true, 'a'	new 키워드 활용
초기화방식	default	생성자

```
int a = 100;
```

a

100

```
String s = new String("100");
```

s

0x0001

0x0001

"100"

## 2. 자료형

예제작성

### 기본 자료형 에서 꼭 기억해야 할 것들

모든 변수는 데이터를 담는 공간(상자)이다.  
기본 자료형의 타입은 무조건 소문자로 시작한다.  
상자의 타입은 담을 수 있는 데이터의 종류를 말한다.  
각 타입마다 상자의 크기가 있다.

### 변수 선언시의 주의 점

적절한 변수의 타입을 써라.  
변수의 이름은 결국 메모리상에서 찾아가는 이름 - 충돌 조심  
상자를 최초로 만들 때에만 변수의 타입이 쓰인다.  
이클립스는 미리 프로파일링을 통해서 불필요한 변수 선언을 처리



## 2. 변수, 상수, 자료형

예제작성

- 상수

- 변경될 수 없는 고정된 데이터
- 코드의 이해와 변경이 쉬움
- 분산된 상수로 인한 에러를 방지
- **final** 키워드를 이용해서 정의

예> `int age;` -> 변수, **final** `double PI = 3.14` -> 상수

- 문자열 상수(이스케이프 문자)

문자상수	내용	문자상수	내용
<code>\n</code>	줄넘김	<code>\"</code>	“ 표시
<code>\t</code>	탭만큼 띄우기	<code>'</code>	‘ 표시
<code>\\</code>	<code>\</code> 화면에 표시		

# 3. 연산자

예제작성

## - 3항 연산자

### 형식

조건식 ? 수식-1 : 수식-2;

수식-1 : 조건식의 결과가 참(true) 일 때 수행되는 식

수식-2 : 조건식의 결과가 거짓(false) 일 때 수행되는 식

```
int a = 10;
```

```
int b = 5;
```

```
int max = ( a > b ) ? a : b ;
```

max 값은 ??

### 3. 연산자

예제작성

- 산술 연산자

연산자	사용법	설 명
+, -, *		
/	op1 / op2	op1을 op2로 나눈 몫을 구한다.
%	op1 % op2	op1을 op2로 나눈 나머지를 구한다.

정수와 정수의 연산 = 정수  
정수와 실수의 연산 = 실수

### 3. 연산자

예제작성

#### - 증감 연산자

연산자	사용법	설명	사용예
++	++op (선행처리)	1 증가	int a = 5;    int b = a++; ??
	op++ (후행처리)		int a = 5;    int b = ++a; ??
--	--op (선행처리)	1 감소	int a = 5;    int b = a--; ??
	op-- (후행처리)		int a = 5;    int b = --a; ??

```
int a = 5;
System.out.println(a++);
System.out.println(++a);
System.out.println(--a);
System.out.println(a  );
System.out.println(a--);
System.out.println(a++);
```

### 3. 연산자

예제작성

- 비교 연산자 : 결과값으로 참(true), 거짓(false)이 반환

연산자	사용법	설명	사용예
>, >=, <, <=			
==	op1 == op2	서로 같은 경우	boolean b = ( (10 % 2) == 0 )
!=	op1 != op2	서로 같지 않은 경우	boolean b = ( (10 % 2) != 0 )
instanceof	op1 instanceof op2	객체의 타입을 비교	

가장 많이 사용되는 것  
== , !=

### 3. 연산자

예제작성

- 조건 연산자 : 결과값으로 참(true), 거짓(false)이 반환

연산자	사용법	설명
&&	A && B	A와 B가 참일 경우만 참 반환 (A가 거짓일 경우 B는 실행하지 않는다)
	A    B	A 또는 B 둘 중에 하나가 참일 경우 참 반환 (A가 참일 경우 B는 실행하지 않는다)
!	! A	A가 참이면 거짓, A가 거짓이면 참을 반환

### 3. 연산자

예제작성

#### - 배정연산자

연산자	사용법	설명
+=	op1 +=op2	op1 = op1+op2
-=	op1 -=op2	op1 = op1-op2
*=	op1 *=op2	op1 = op1*op2
/=	op1 /=op2	op1 = op1/op2



## 4. 조건문

예제작성

- if 문
- switch 문



## 4. 조건문

예제작성

- 단일 if

```
- if ( 조건식 )
    실행문장;
- if ( 조건식 )
    실행문장;
else
    실행문장;
- if ( 조건식 ) {
    .....
} else {
    .....
}
```

주의사항

- 실행문장이 복수일 때에는 블록으로 처리
- 조건식 자리에는 반드시 참과 거짓을 구분해야 한다.

```
int a = 3;
if(a = 3) => 에러 발생
if( 0 )   => 에러 발생
```

- **else** 절은 필요에 따라 기술한다.

## 4. 조건문

예제작성

- 다중 if

```
형식 - if (조건식) {  
    실행문장;  
} else if (조건식) {  
    실행문장;  
} else if (조건식) {  
    실행문장;  
} else {  
    실행문장;  
}
```

## 4. 조건문

예제작성

- 내포된 if

형식 - if (조건식) {  
    if (조건식)  
        .....  
    if (조건식)  
        .....  
    else  
        .....  
}

```
int a = 10, b = 5;  
if ( a > b ) {  
    if ( a == 10 )  
        System.out.println("a = 10");  
        System.out.println("a가 b보다 크다");  
    else  
        System.out.println("b가 a보다 크다");  
}
```

## 4. 조건문

예제작성

- switch

```
switch (수식) {  
    case 값1:  
        처리문장들;  
        break;  
    case 값2:  
        처리문장들;  
        break;  
    case 값n:  
        처리문장들;  
        break;  
    default:  
        묵시적으로 처리해야 하는 문장들;  
}
```

주의사항

1. 수식에 올 수 있는 것
  - 1.4버전 까지  
byte, short, char, int
  - 1.5버전 부터  
enum 클래스 타입
  - 1.7 버전 부터  
String 클래스 타입
2. break문 없이도  
사용이 가능하다.
3. default => else의  
역할과 동일하다.

## 4. 반복문

예제작성

### - for

- 형식    **for**(1. 초기값 ; 2. 조건 ; 3. 증감) {  
          4. 반복문장들  
          }

5. 반복문 빠져나옴

### - 실행순서

- 1 - 2(조건이 참일 경우)    - 4 - 3
- 2(조건이 참일 경우)    - 4 - 3
- 2(조건이 거짓일 경우) - 5

## 4. 반복문

예제작성

### - while

- ♦ 조건절로 지정된 조건이 참일 동안 **while** 블록을 실행

```
while(조건절) {  
    반복문장들  
}
```

예>

```
int a=10, b=20;
```

```
while(a > b)
```

```
System.out.println("이 문장은 영원히 나타나지 않는다");
```

## 4. 반복문

### - do ~ while

**do-while**문은 조건을 나중에 평가한다  
**while** 블록이 적어도 한번은 수행

```
do {  
    반복문장들  
} while(조건절);
```

ex> int a = 5, b = 10;

```
do {  
    System.out.println("무조건 실행됨");  
} while (a > b)  => 1번 실행됨.
```

**while** 와 **do-while**의 차이점  
- **do-while** 은 무조건  
한번은 실행. **While**은  
실행이 안될 수 도 있다  
다시 말해서  
같은 조건일 경우 **do-while**  
문만 실행 될 수 있다.

### - break

**break**문의 3가지 역할

**switch**문에서 **switch**문을 벗어나는데 사용

반복문에서 반복루프를 벗어나는데 사용

중첩된 반복문을 한번에 빠져나갈때

### - continue

반복문의 특정지점에서 제어를 반복문의 처음으로 보낸다



### - break

```
int i = 1;
while(i < 100) {
    if(i == 10) break;
    System.out.println(i + "자바의 세계로 오세요! ");
    i++;
}
```

결과는 ??

## 4. 제어

### 예제 작성

#### - break

반복문이 중첩되었을 경우 가장 가까운 반복문을 빠져 나온다

```
int i, j;  
for(i=1 ; i<=5 ; i++) {  
    for(j=1 ; j<=i ; j++) {  
        if (j > 3) break;  
        System.out.print(" * ");  
    }  
    System.out.println();  
}
```

i = 1일 때

\*

i = 2일 때

\* \*

i = 3일 때

\* \* \*

i = 4일 때

\* \* \*

i = 5일 때

\* \* \*

결과

\*

\* \*

\* \* \*

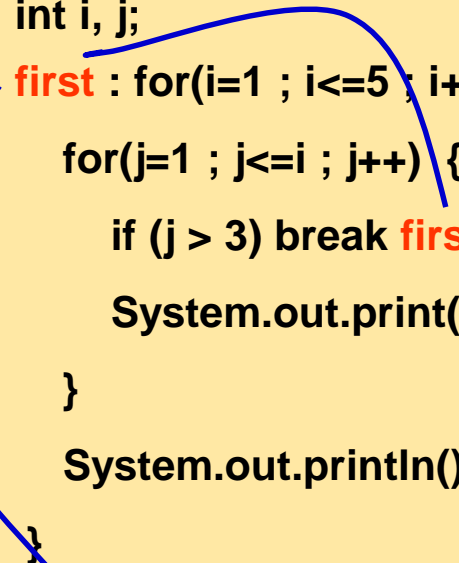
\* \* \*

\* \* \*

### - break

중첩된 반복문을 한번에 빠져 나오기

```
int i, j;  
first : for(i=1 ; i<=5 ; i++) {  
    for(j=1 ; j<=i ; j++) {  
        if (j > 3) break first; // first라는 이름의 블록을 벗어난다.  
        System.out.print(" * ");  
    }  
    System.out.println();  
}
```



### - continue

반복문의 특정지점에서 제어를 **반복문의 처음으로** 보낸다

```
예> class ContinueTest {  
    public static void main(String args[]) {  
        for(int i=0; i<10; i++) {  
            if (i%2 == 0) continue;  
            System.out.println(i + " 자바의 세계로 오세요! " );  
        }  
    }  
}
```

결과는??

1 자바의 세계로 오세요!  
3 자바의 세계로 오세요!  
5 자바의 세계로 오세요!  
7 자바의 세계로 오세요!  
9 자바의 세계로 오세요!



배 열

## - 배열이란?

나가수경연	임재범	김범수	박정현
1차 경연	40.1%	29.7%	30.2%
2차 경연	30.1%	35.7%	34.2%

**String name = “임재범”;**

**String name2 = “김범수”;**

**String name3 = “박정현”;**

**double first = 40.1;**

**double first2 = 29.7;**

**double first3 = 30.2;**

**double second = 30.1;**

**double second2 = 35.7;**

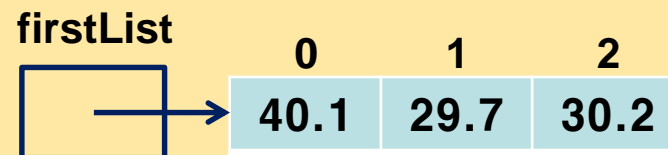
**double second3 = 34.2;**

## - 배열이란?

- 같은 종류의 데이터를 저장하기 위한 자료구조
- 크기가 고정되어 있다(한번 생성된 배열은 크기를 바꿀 수 없다)
- 배열을 객체로 취급
- 배열의 요소를 참조하려면 배열이름과 색인(index)이라고 하는
- **int** 유형의 정수값을 조합하여 사용한다.

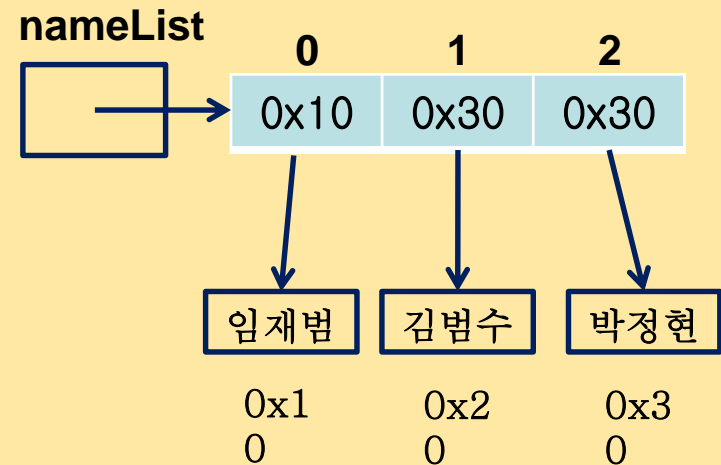
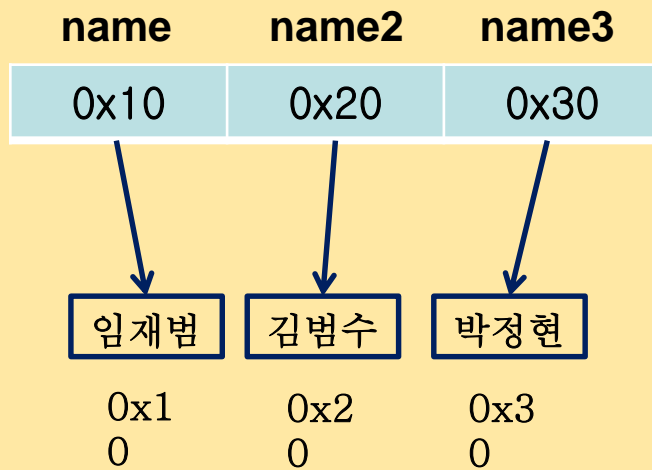
## - 배열

first	first2	first3
40.1	29.7	30.2





## - 배열



## - 배열의 선언

- [ ] 는 자바에서 배열을 나타낸다.
- [ ] 의 개수가 배열의 차원수를 나타낸다.  
예를 들어, [ ] 일 경우 1차원, [ ][ ] 일 경우 2차원이 된다.

### - 1차원 배열 선언

배열유형 배열이름 [ ] 또는 배열유형[] 배열이름

ex) int prime [ ], int [ ] prime

### - 다차원 배열 선언

배열유형 배열이름 [][] 또는 배열유형[][] 배열이름

ex) int prime [ ][ ], int [ ][ ] prime

\* 배열의 유형은 모든 것이 가능하다(기본형, 참조형)

## - 배열의 선언

타입	배열이름	선언
int	iArr	int [] iArr;
char	cArr	char [] cArr;
boolean	bArr	boolean [] bArr;
String	strArr	String [] strArr;
Date	dateArr	Date [] dateArr;

## - 배열의 선언

```
int [ ] dongList;
```

**dongList**



하나의 값을 저장할 수  
있는 메모리 생성

```
int [ ][ ] aptInfoList;
```

**aptInfoList**



하나의 값을 저장할 수  
있는 메모리 생성

## - 배열의 생성

### - 1차원 배열

배열의 이름 = **new** 배열유형 [배열크기];

ex) **prime = new int [10];**

### - 2차원 배열

배열의 이름 = **new** 배열유형[1차원배열개수][1차원배열의크기];

배열의 이름 = **new** 배열유형[1차원배열개수] **[ ]**;

ex) **prime = new int [3][2];**

**prime = new int [3][ ];**

## - 자동 초기화

- 배열이 생성되면 자동적으로 배열요소는 기본값으로 초기화된다.

ex) int : 0

boolean : false

char : '\u0000'

참조형 : null....

- 멤버변수와 로컬변수 모두 배열이 생성이 되면 자동 초기화된다.

## - 배열의 생성

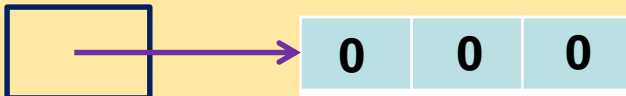
`int [ ] dongList;`

`dongList`



`dongList = new int[3];`

`dongList`



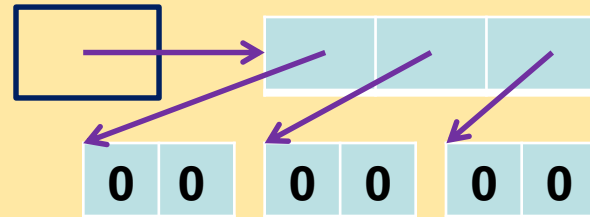
`int [ ][ ] aptInfoList;`

`aptInfoList`



`aptInfoList = new int[3][2];`

`aptInfoList`



## - 배열의 생성

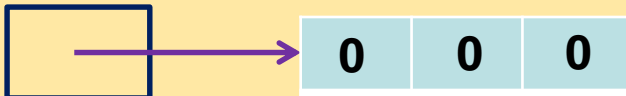
`int [ ] dongList;`

`dongList`



`dongList = new int[3];`

`dongList`



`int [ ][ ] aptInfoList;`

`aptInfoList`



`aptInfoList = new int[3][ ];`

`aptInfoList`





## - 초기화

### - 1차원 배열

배열이름[인덱스] = 값;

ex) `prime[0] = 100;`

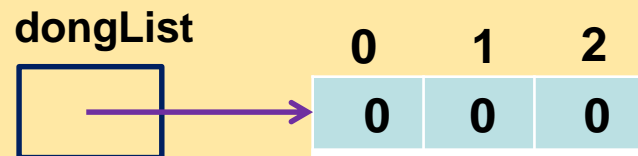
### - 2차원 배열

배열이름[인덱스][인덱스] = 값;

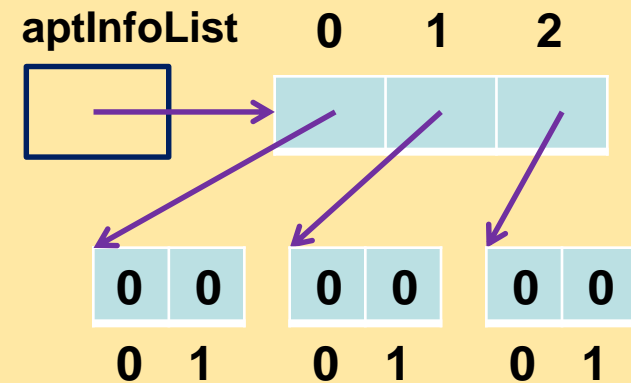
ex) `twoArr[0][1] = 100;`

## - 배열의 초기화

```
int [ ] dongList = new int[3];
```



```
int [ ][ ] aptInfoList =  
    new int[3][2];
```



## - 배열의 초기화

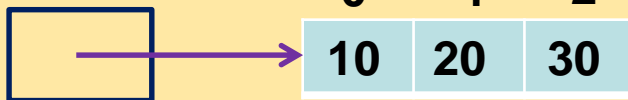
```
int [ ] dongList = new int[3];
```

```
dongList[0] = 10;
```

```
dongList[1] = 20;
```

```
dongList[2] = 30;
```

dongList

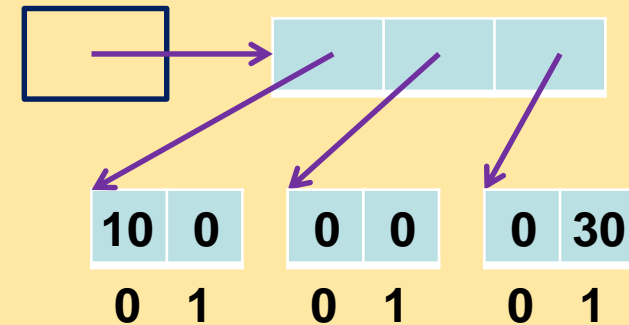


```
int [ ][ ] aptInfoList =  
    new int[3][2];
```

```
aptInfoList[0][0] = 10;
```

```
aptInfoList[2][1] = 30;
```

aptInfoList      0      1      2



## - 배열의 초기화

```
int [ ] dongList = new int[3];
```

- 배열의 인덱스는 0부터 시작
- 배열의 크기 : 배열이름.length
- 마지막 요소 인덱스 : 배열크기 - 1

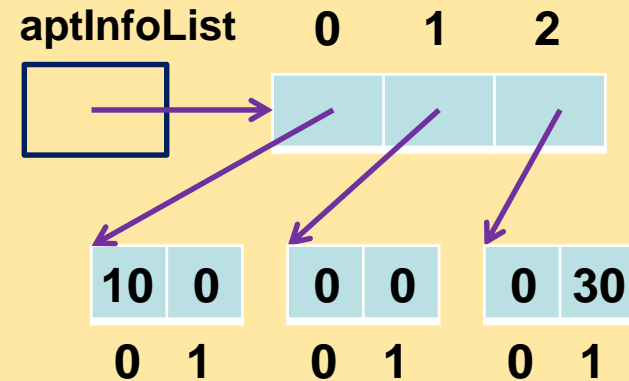
aptInfoList.length ??

aptInfoList[0].length ??

```
int [ ][ ] aptInfoList =  
    new int[3][2];
```

```
aptInfoList[0][0] = 10;
```

```
aptInfoList[2][1] = 30;
```



## - 초기화

- { } 를 활용하는 방식 : 배열 선언 시에만 설정 가능

1차원 배열 : 배열유형 [ ] 배열명 = { 값, .. 값 };

ex) int [ ] prime = { 1, 2, 3 };

2차원 배열 : 배열유형 [ ][ ] 배열명 = { { 값1, 값2 }, { 값3, 값4 } };

ex) int [ ][ ] twoArr = { { 1, 2 }, { 3, 4 }, { 5, 6 } };

- new 배열타입[ ] { 값, ....}

ex) int [ ] prime = new int[ ] { 1, 2 };

## - 배열관련 제공 API

### - **`System.arraycopy(src, srcPos, dest, destPos, length)`**

**src** : 원본배열      **srcPos** : 원본배열의 복사 시작 위치( 0부터 시작 )

**dest** : 복사할 배열    **destPos** : 복사 받을 시작 위치

**length** : 복사할 크기

예) `String [ ] oriArr = {“봄”, “여름”, “가을”};`

`String [ ] destArr = new String[oriArr.length + 1];`

`System.arraycopy( oriArr, 0, destArr, 0, oriArr.length );`

`destArr[ 3 ] = “겨울”;`

`for( int i = 0; i < destArr.length; i++)`

`System.out.println(destArr[ i ]);`

## - 배열관련 제공 API

### - **Arrays.toString(배열객체)**

: 배열안의 요소를 [ 요소, 요소, ..] 의 형태로 출력

예)

```
for( int i = 0; i < destArr.length; i++)
```

```
    System.out.println(destArr[ i ]);
```

=====

단순 배열값 확인일 경우

```
System.out.println( Arrays.toString(destArr) );
```