

1장. 자바(Java) 개발 환경 및 기본문법



JDK & Eclipse



프로그래밍과 자바

● 프로그래밍이란?

- 컴퓨터 프로그램을 만드는 일
- 컴퓨터에게 일을 하도록 명령어를 만드는 것

만약 사람이 원하는 바를 대략 말하면 컴퓨터가 알아서 프로그램을 만들어 준다면 좋은 것이다.
(현재, 인공지능과 빅데이터 기술로는 아직 먼 미래의 일)

● 프로그래밍 언어의 종류

- C언어, C++, C#, Java, Python, Javascript

● 자바(Java)

- 제임스 고슬링 등이 오크(Oak)언어에서 가전제품에 탑재할 용도로 만들어 냈
- 인터넷에 적합하고, 운영체제(플랫폼)에 독립적으로 변경함
- 썬 마이크로시스템즈, 1996년 1월 발표(지금, 오라클에 인수됨)



자바란?

● 자바 언어의 특징

- 운영 체제에 독립적이다. – JVM(자바가상머신)이 가능하게 함
- 객체지향 언어이다. – 유지보수가 쉽고, 확장성이 좋다.
- 풍부한 기능이 제공되는 오픈 소스이다.
- 네트워크와 멀티 쓰레드를 지원하는 다양한 API(라이브러리)
- 안드로이드용 스마트폰 App(앱) 개발 언어로 사용되고 있다.



자바(Java)로 개발한 프로그램

- 웹 사이트(서버)

- 웹 사이트를 운영하려면 반드시 서버(server)가 필요하다.
- 검색 사이트, 쇼핑몰, 금융 사이트 등 자바로 개발한 웹 서버 프로그램으로 운영

- 안드로이드 앱

- 안드로이드 폰에서 사용하는 앱을 만들 수 있다.

- 게임

- 게임을 만들때는 C++, C를 주로 사용하지만 마인크래프트처럼 게임을 구현하는데도 사용된다.



자바 가상 머신(JVM)

◆ JVM(Java Virtual Machine)

- 자바 프로그램 실행 환경을 만들어 주는 소프트웨어
- 자바 코드를 컴파일한 .class(바이트 코드)는 JVM 환경에서 실행됨
- 컴퓨터의 운영체제에 맞는 자바 실행 환경(JRE)가 설치되어 있다면 자바 가상머신이 설치되어 있는 것이다. (JRE > JVM)

◆ JDK와 JRE

JDK(Java Development Kit) – 자바 개발을 위해 설치하는 라이브러리이다.

JRE(Java Runtime Environment) – 자바 프로그램이 실행되는 자바실행환경이다.

▶ 컴파일(Compile)과 컴파일러

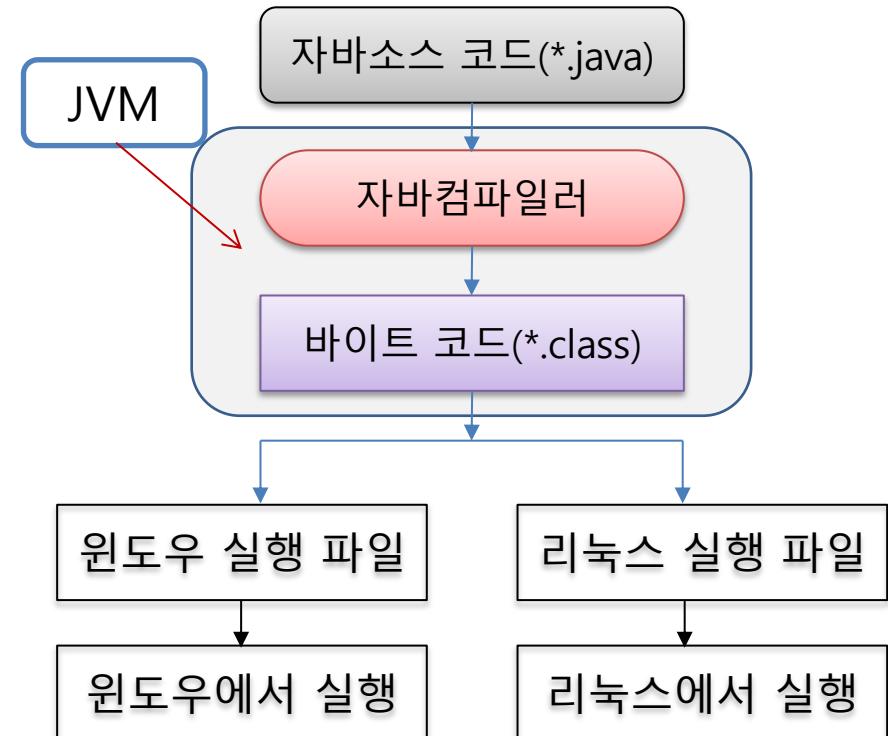
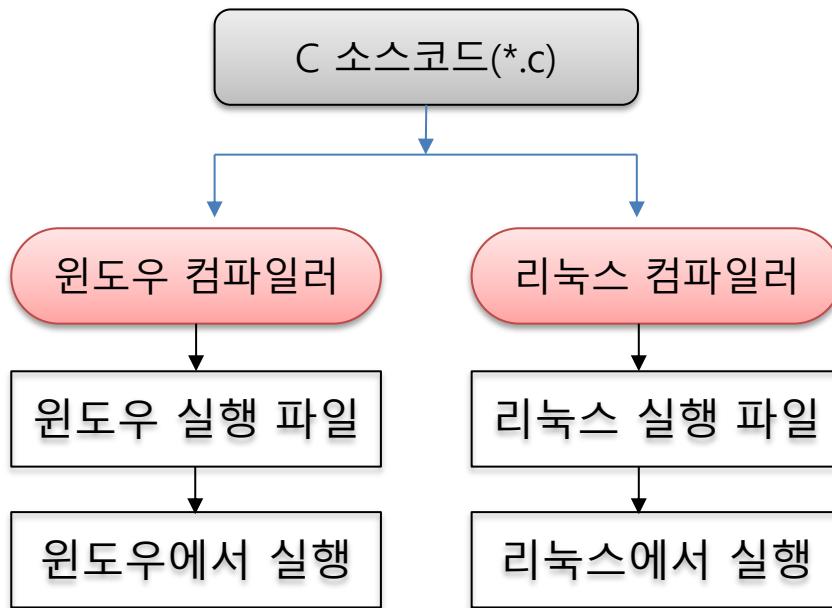
컴파일은 프로그램(코드)를 컴퓨터가 알 수 있는 언어(기계어)로 바꿔 주는 일

컴파일러는 프로그램 언어를 기계어로 번역해 주는 프로그램으로 자바(JDK)를 설치하면 자바 컴파일러도 설치



자바(Java) 언어

◆ JVM의 기능(역할)



자바 개발 환경 구축

◆ 자바 개발도구(JDK) 설치

- jdk 다운로드(검색)-> windows> Java SE11 다운로드 -> x64 인스톨러

Java SE Development Kit 11.0.15.1

Java SE subscribers will receive JDK 11 updates until at least **September of 2026**.

These downloads can be used for development, personal use, or to run Oracle licensed products. Use for other purposes, including production or commercial use, requires a Java SE subscriber license.

JDK 11 software is licensed under the [Oracle Technology Network License Agreement for Oracle Java SE](#).

JDK 11.0.15.1 [checksum](#)

Linux	macOS	Solaris	Windows
Product/file description	File size	Download	
x64 Installer	140.41 MB	jdk-11.0.15.1_windows-x64_bin.exe	
x64 Compressed Archive	158.1 MB	jdk-11.0.15.1_windows-x64_bin.zip	

```
C:\Users\김기용>java -version
java version "17.0.1" 2021-10-19 LTS
Java(TM) SE Runtime Environment (build 17.0.1+12-LTS-39)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.1+12-LTS-39,
```



자바 개발 환경 구축

◆ 자바 Documentation 설치

Java api 를 설명하고 있는 문서

JDK 11.0.15.1 [checksum](#)

Linux macOS Solaris **Windows**

Product/file description

x64 Installer

x64 Compressed Archive

Documentation Download

Java® Platform, Standard Edition & Java Development Kit Version 11 API Specification

This document is divided into two sections:

Java SE

The Java Platform, Standard Edition (Java SE) APIs define the core Java platform.

JDK

The Java Development Kit (JDK) APIs are specific to the JDK and will not necessarily have names start with jdk.

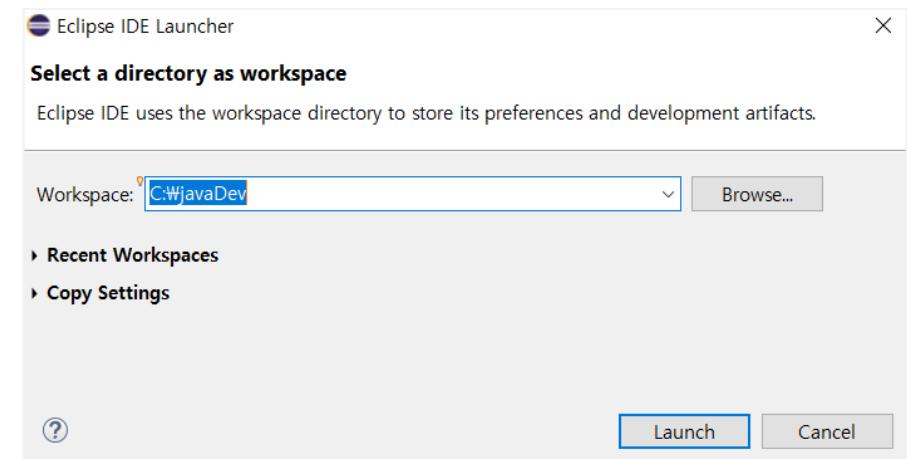
All Modules	Java SE	JDK	Other Modules
Module	Description		
java.base	Defines the foundational APIs of the Java SE Platform.		
java.compiler	Defines the Language Model, Annotation Processing, and Java Compiler APIs.		
java.datatransfer	Defines the API for transferring data between and within applications.		
java.desktop	Defines the AWT and Swing user interface toolkits, plus APIs for accessibility.		



이클립스(Eclipse) IDE 설치

◆ 이클립스 IDE(통합개발환경) 설치

- 검색 이클립스(<https://www.eclipse.org/downloads/>)
- 버전 - Eclipse IDE 2022-03



Workspace – C:\javaworks

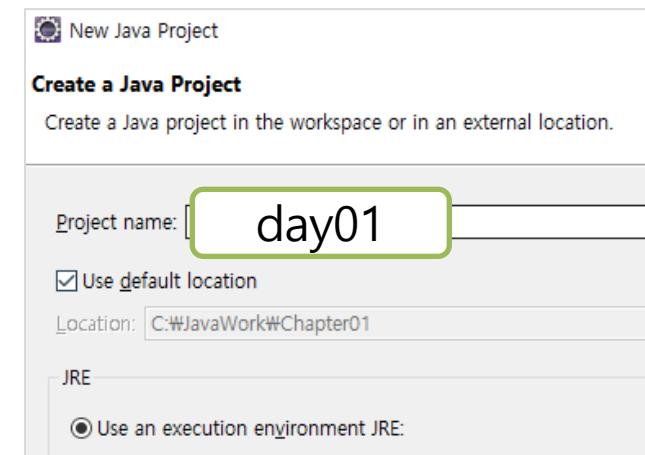
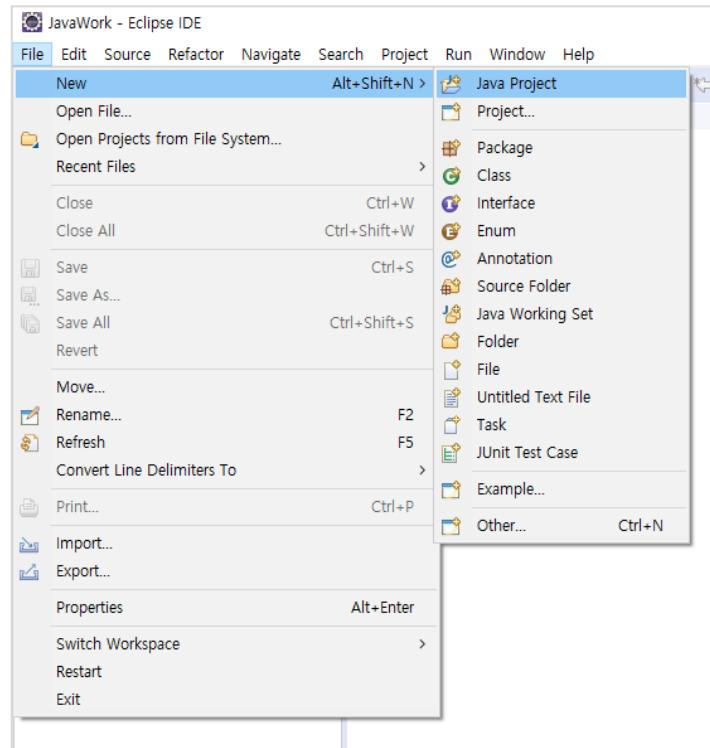


프로젝트 만들기

● 첫 자바 프로젝트(Project) 만들기

File->New->Java Project

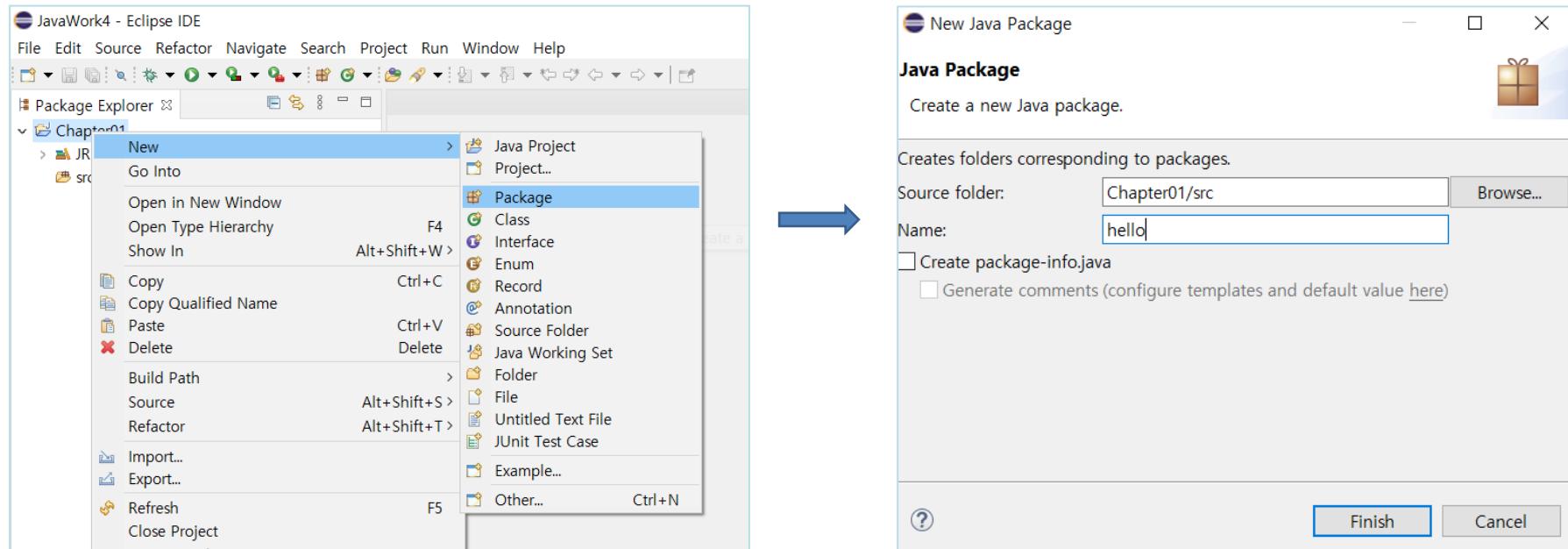
Project Name : day01



첫번째 패키지 만들기

첫번째 패키지 만들기

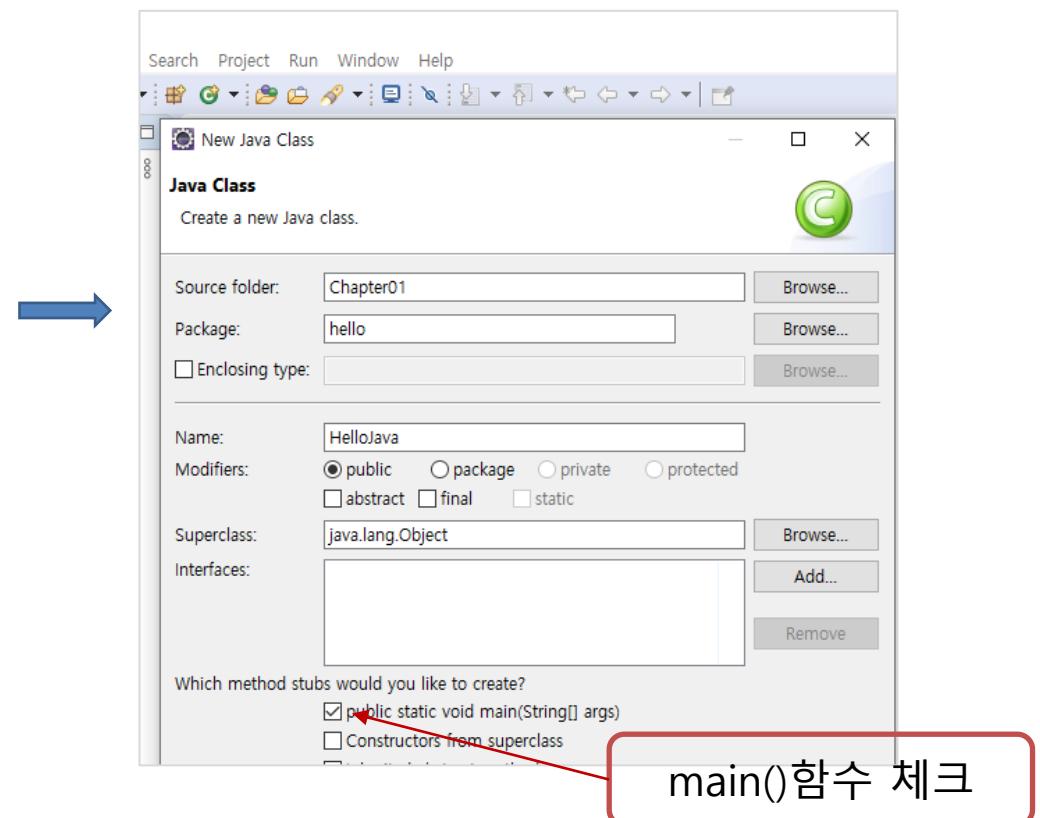
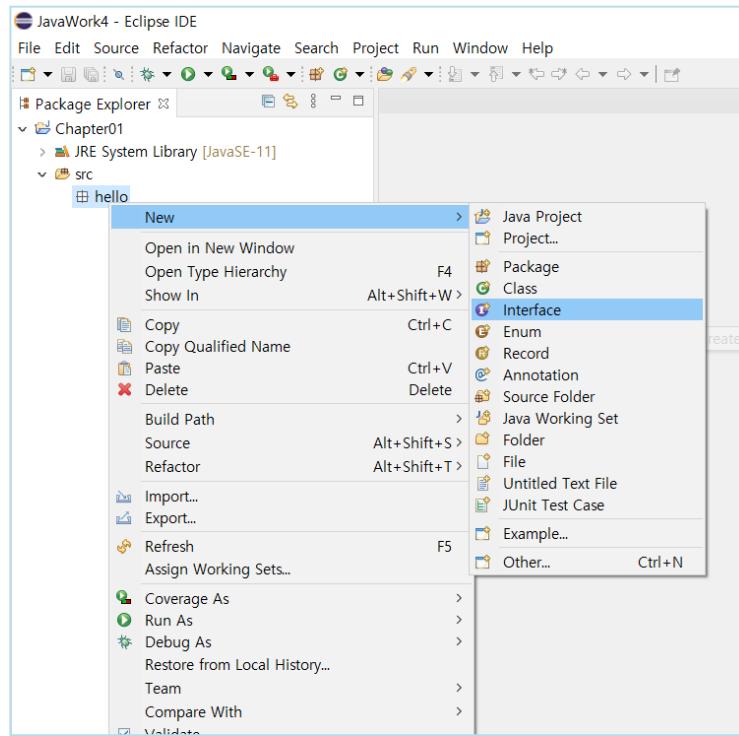
패키지 만들기 : day01(마우스우측) -> New package -> Name : hello



첫번째 클래스 만들기

첫번째 클래스 만들기

클래스 만들기 : hello(우측)->New class->Name : HelloJava



main()함수 체크

첫번째 클래스 만들기

1. java 코드 작성

- 파일 이름 : HelloJava.java
- 클래스 : System , 메서드 : main(), print()
- 파일 실행하려면 main() 메서드가 필요함

```
package hello;           ← 패키지 이름

public class HelloJava{
    public static void main(String[] args) {
        System.out.println("Hello~ Java");
    }
}
```

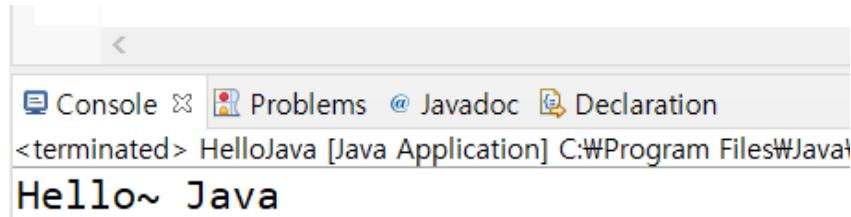
main()메서드



첫번째 클래스 만들기

2. 컴파일 및 실행

- 컴파일 하기 : 빌드 자동화 옵션 지정 -> 클래스 파일 생성
- 실행 : Run -> Run as -> Java Application [실행 단추(▶) 클릭]
- 실행 결과 : 콘솔(Console)



※ 클래스(.class) 파일의 위치는 어디일까?

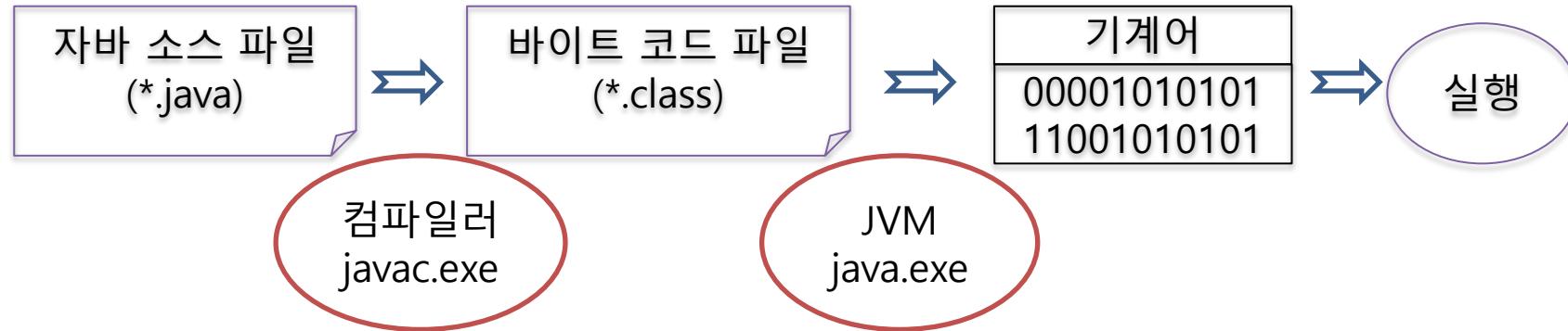
A screenshot of a Windows File Explorer window. The address bar shows the path: '내 PC > 로컬 디스크 (C:) > JavaWork4 > Chapter01 > bin > hello'. The search bar contains the text 'hello 검색'. The table below lists the contents of the 'hello' folder:

이름	수정한 날짜	유형	크기
HelloJava.class	2020-12-10 오전 8:59	CLASS 파일	1KB
PrintData.class	2020-12-10 오전 9:00	CLASS 파일	1KB



컴파일과 빌드

컴파일(compile)



빌드(build)

컴파일과 링크된 코드들을 실행가능한 파일로 만드는 일련의 과정

전처리, 컴파일, 패키징, 배포등이 포함된다.

Java 빌드 툴로는 Ant, Maven, Gradle 등이 있다.



주석 , 블록, 세미콜론(:)

기초 문법

- 주석은 소스코드에 설명을 추가하거나 특정 코드가 컴파일되지 않도록 처리할때 사용
- 한줄 주석 : 문장 앞에 '//' 표시
- 여러 줄 주석 : /*~ */ 기호 사용
- 문장이 종료는 세미콜론(:)을 사용
- { } 블록 안에 코드 작성



데이터(data) 출력하기

```
package hello;

public class PrintData {

    public static void main(String[] args) {
        //숫자
        System.out.println(100);
        System.out.println(3.3);
        System.out.println("-----");

        //문자
        System.out.println('A');           //문자
        System.out.println('가');
        System.out.println("apple");     //문자열
        System.out.println("-----");

        //연산
        System.out.println(4 + 5);
        System.out.println(4 + "5");

        //불리언
        System.out.println(true);
        System.out.println(5 < 4);
    }
}
```



System 클래스

Module java.base

Package java.lang

Class System

java.lang.Object
 java.lang.System

```
public final class System
extends Object
```

The **System** class contains several useful class fields and methods.
output streams; access to externally defined properties and environmen

Since:

1.0

void	print(Object obj)
void	print(String s)
PrintStream	printf(String format, Object... args)
PrintStream	printf(Locale l, String format, Object... args)
void	println()
void	println(boolean x)
void	println(char x)



연습 문제

실습 문제 : Java 개발 환경 구축

()안에 들어갈 적당한 말을 맞춰보세요.

1. 프로그램(코드)을 기계가 이해할 수 있는 언어로 바꾸는 일을 ()이라고 한다.
2. 자바로 만든 프로그램은 ()이 설치되어 있으면 운영체제와 상관없이 실행할 수 있다.
3. 자바 개발을 위해 설치하는 자바 라이브러리를 ()라고 한다.

1. 컴파일 2. JVM(자바 가상머신) 3. JRE



■ 변수란?

- 프로그램에서 사용되는 자료를 저장하기 위한 공간
- 할당받은 메모리의 주소 대신 부르는 이름
- 프로그램 실행 중에 값 변경 가능, variable 이라 함

■ 변수의 선언 및 초기화

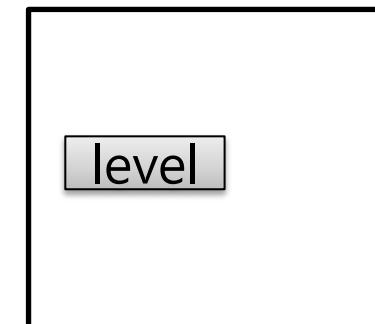
- 변수 선언은 어떤 타입의 데이터를 저장할 것인지 그리고 변수이름은 무엇인지를 결정한다.

- **자료형 변수이름;**

- **자료형 변수이름 = 초기값;**

```
int level;
```

```
double height;
```



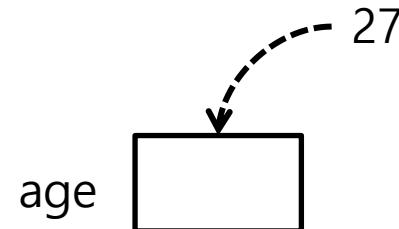
변수 사용하기

■ 변수의 초기화

```
int age = 27;
```

```
char c = 'k';
```

```
String fruit= "사과"
```



■ 변수 이름 선언시 유의점

- 변수의 이름은 알파벳, 숫자, _, \$로 구성된다.
- 대소문자를 구분한다.
- **숫자로 시작할 수 없고**, 키워드(예약어)도 변수의 이름으로 사용할 수 없다.
- 이름 사이에 공백이 있을 수 없다.
- 변수의 이름을 정할 때는 변수의 역할에 어울리는, 의미있는 이름을 지어야 한다.

예약어(reserved word)
프로그래밍 구문에 사용되는 명령어

break, int, const, if, for,
class, this 등



변수 사용하기

실습 예제

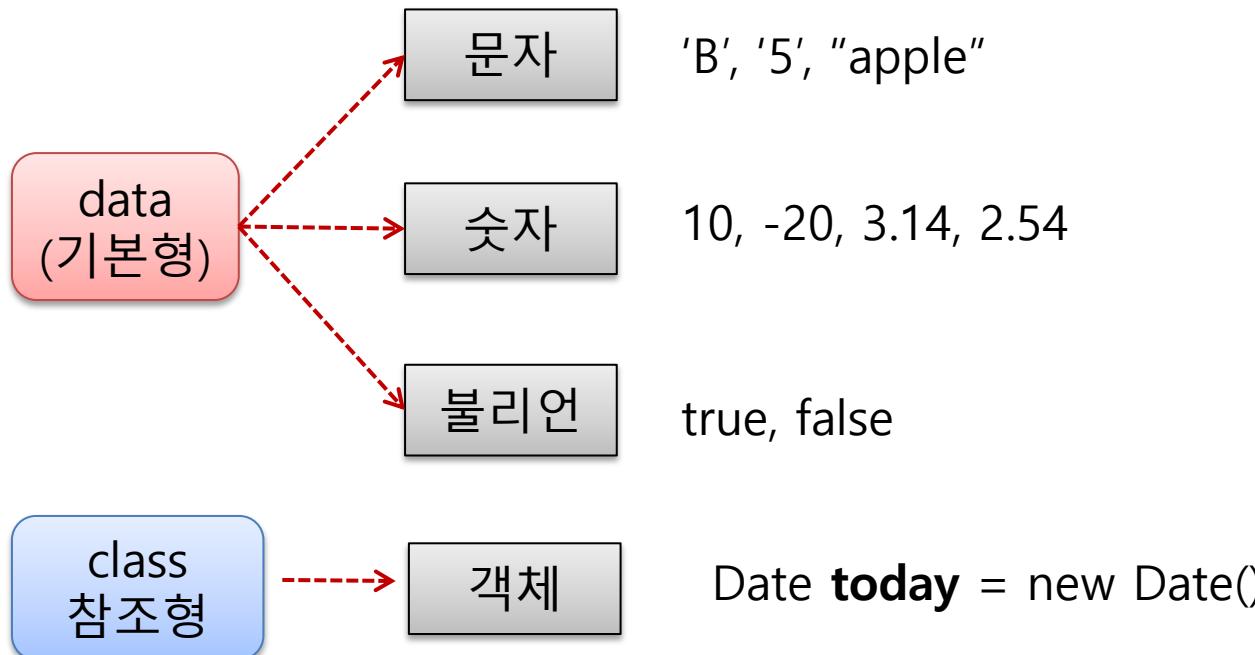
- 파일 이름 : Variable.java

```
public class Variable {  
    public static void main(String[] args) {  
        String name;  
        name = "한지수";  
  
        int grade;  
        grade = 2;  
  
        //int class = 3; //class는 예약어라 오류  
        int schoolClass = 3;  
  
        System.out.println(name + "는 " + grade + "학년 " + schoolClass + "반 입니다.");  
    }  
}
```



● 자료형이란?

- 데이터를 저장하는 공간의 유형
- 사용할 데이터의 종류에 따라 메모리 공간을 적절하게 설정해 주는 것



- 기본 자료형의 크기

	정수형	문자형	실수형	논리형
1byte	byte	-	-	boolean
2byte	short	char	-	-
4byte	int	-	float	-
8byte	long	-	double	-



정수 자료형

- 정수 자료형의 종류 및 크기

자료형	바이트크기	수의 범위	설명
byte	1	$-2^7 \sim (2^7 - 1)$	1byte=8bit 1bit : 0, 1 - 2개 2bit : 00, 01, 10, 11 - 4개 3bit : 8개 ... 8bit : 256개
short	2	$-2^{15} \sim (2^{15} - 1)$	
int	4	$-2^{31} \sim (2^{31} - 1)$	
long	8	$-2^{63} \sim (2^{63} - 1)$	

- int로 5와 -5를 표현할 때(32비트)

5 -> 0000000000000000000000000000101



정수 자료형

■ byte

- 1 바이트 단위의 자료형
- 동영상, 음악, 이미지 파일등 이진(바이너리) 실행 파일의 자료

■ short

- 2바이트 단위의 자료형
- 주로 c/c++ 언어와의 호환 시 사용됨

■ int

- 4바이트 단위의 자료형
- 프로그램에서 사용하는 모든 숫자(리터럴)은 기본적으로 int로 저장됨

■ long

- 8바이트 자료형
- 가장 큰 정수 자료형으로 숫자 뒤에 'L' 또는 'L'을 써서 표시



숫자 자료형 실습

■ 정수 자료형

```
public class ByteShortType {  
    public static void main(String[] args) {  
        byte bData1 = -128;  
        System.out.println(bData1);  
  
        //byte bData2 = 128;      // -128 ~ 127  
  
        short sData1 = 32767;    // -32768 ~ 32767  
  
        //short sData2 = 32768;  
        System.out.println(sData1);  
    }  
}
```



숫자 자료형 실습

■ 정수 자료형

```
public class IntLongType {  
    public static void main(String[] args) {  
        int iNum = 1234567890;      // -21억 ~ 21억  
        System.out.println(iNum);  
  
        long lNum = 123456789012L;  //'L' or 'l'을 끝에 붙임  
        System.out.println(lNum);  
    }  
}
```



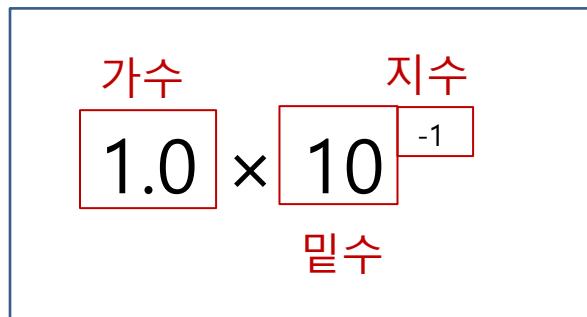
실수 자료형

- 실수를 컴퓨터 내부에서 어떻게 나타내야 할지 생각해 보자

- 실수값 3.14를 표현하면 3이라는 정수부분과 .14라는 소수 부분을 따로 표현할 수 있고, 0과 1사이에는 무한개의 실수가 있다.
 - 부동소수점 방식을 사용하면 실수를 좀 더 세밀하게 표현할 수 있다.

▪ float 와 double

- 부동 소수점 방식 : 실수를 지수부와 가수부로 표현함
무한의 실수를 표현하기 위한 방식
 - 0.1을 표현하는 방식



- float(4바이트) - 숫자 뒤에 F나 f를 써서 표시
 - double(8바이트)

예) double num = 3.14;
 float num = 3.14F;



숫자 자료형 실습

■ 실수 자료형

```
public class FloatDoubleType {  
    public static void main(String[] args) {  
        //실수 자료형  
        float fNum = 1.23456789F;    //'F' 나 'f'를 붙임  
        double dNum = 1.2345678901234567;  
  
        System.out.println(fNum);  
        System.out.println(dNum);  
    }  
}
```



문자 자료형

■ **char**

- 자바에서는 문자를 2바이트로 처리
- 문자 1개를 표현할때 홑따옴표(' ')로 감싸준다.

■ **문자 세트(charset)**

- 문자세트 – 문자를 위한 코드 값(숫자 값)들을 정해 놓은 세트
- 인코딩 – 각 문자에 따른 특정한 숫자 값(코드 값)을 부여
- 디코딩 – 숫자 값을 원래의 문자로 변환
- 아스키(ASCII) – 1 바이트로 영문자, 숫자, 특수 문자 등을 표현 함
- 유니코드(Unicode) – 한글과 같은 복잡한 언어를 표현하기 위한 표준 인코딩 UTF-8, UTF-16이 대표적, 2바이트

<https://www.unicode.org/charts/PDF/UAC00.pdf>



문자 자료형

```
public class CharType {  
    public static void main(String[] args) {  
        //문자 자료형  
        char ch1 = 'A';  
        System.out.println(ch1);  
        System.out.println((int)ch1);  
  
        char ch2 = 66;  
        System.out.println(ch2); 형변환  
  
        int ch3 = 67;  
        System.out.println(ch3);  
        System.out.println((char)ch3);  
  
        System.out.println("\n***** 유니코드 *****");  
  
        char uniCode1 = '한';  
        System.out.println(uniCode1);  
  
        char uniCode2 = '\uD55C';  
        System.out.println(uniCode2);  
  
        for(char c = 97; c < 123; c++) { //영어 소문자 출력(아스키코드)  
            System.out.print(c + " ");  
        }  
    }  
}
```



아스키 코드 vs 유니코드

★ 아스키 코드(ASCII Code)

아스키 코드는 미국 [ANSI](#)에서 표준화한 정보교환용 7비트 부호체계이다. 000(0x00)부터 127(0x7F) 까지 총 128개의 부호가 사용된다. 이는 영문 키보드로 입력할 수 있는 모든 기호들이 할당되어 있는 부호 체계이며, 2바이트 이상의 코드를 표현할 수 없기 때문에 국제표준의 위상은 [유니코드](#)에게 넘어갔다.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0 000	NUL	(null)	32	20 040	 	Space		64	40 100	@	Ø	96	60 140	`	`		
1	1 001	SOH	(start of heading)	33	21 041	!	!	!	65	41 101	A	A	97	61 141	a	a		
2	2 002	STX	(start of text)	34	22 042	"	"	"	66	42 102	B	B	98	62 142	b	b		
3	3 003	ETX	(end of text)	35	23 043	#	#	#	67	43 103	C	C	99	63 143	c	c		
4	4 004	EOT	(end of transmission)	36	24 044	$	\$	\$	68	44 104	D	D	100	64 144	d	d		
5	5 005	ENQ	(enquiry)	37	25 045	%	%	%	69	45 105	E	E	101	65 145	e	e		
6	6 006	ACK	(acknowledge)	38	26 046	&	&	&	70	46 106	F	F	102	66 146	f	f		
7	7 007	BEL	(bell)	39	27 047	'	'	'	71	47 107	G	G	103	67 147	g	g		
8	8 010	BS	(backspace)	40	28 050	(((72	48 110	H	H	104	68 150	h	h		
9	9 011	TAB	(horizontal tab)	41	29 051)))	73	49 111	I	I	105	69 151	i	i		
10	A 012	LF	(NL line feed, new line)	42	2A 052	*	*	*	74	4A 112	J	J	106	6A 152	j	j		
11	B 013	VT	(vertical tab)	43	2B 053	+	+	+	75	4B 113	K	K	107	6B 153	k	k		
12	C 014	FF	(NP form feed, new page)	44	2C 054	,	,	,	76	4C 114	L	L	108	6C 154	l	l		
13	D 015	CR	(carriage return)	45	2D 055	-	-	-	77	4D 115	M	M	109	6D 155	m	m		
14	E 016	SO	(shift out)	46	2E 056	.	.	.	78	4E 116	N	N	110	6E 156	n	n		
15	F 017	SI	(shift in)	47	2F 057	/	/	/	79	4F 117	O	O	111	6F 157	o	o		
16	10 020	DLE	(data link escape)	48	30 060	0	Ø	Ø	80	50 120	P	P	112	70 160	p	p		
17	11 021	DC1	(device control 1)	49	31 061	1	!	!	81	51 121	Q	Q	113	71 161	q	q		
18	12 022	DC2	(device control 2)	50	32 062	2	2	2	82	52 122	R	R	114	72 162	r	r		
19	13 023	DC3	(device control 3)	51	33 063	3	3	3	83	53 123	S	S	115	73 163	s	s		
20	14 024	DC4	(device control 4)	52	34 064	4	4	4	84	54 124	T	T	116	74 164	t	t		
21	15 025	NAK	(negative acknowledgement)	53	35 065	5	5	5	85	55 125	U	U	117	75 165	u	u		



아스키 코드 vs 유니코드

★ 유니 코드(Uni code)

전 세계의 모든 문자를 다루도록 설계된 표준 문자 전산 처리 방식. 주요 구성 요소는 ISO/IEC 10646 Universal Character Set과 UCS, UTF 등의 인코딩 방식, 문자 처리 알고리즘 등이다.

유니코드를 사용하면 한글과 간체자, 아랍 문자 등을 통일된 환경에서 깨뜨리지 않고 사용할 수 있다.

초창기에는 문자 코드는 ASCII의 로마자 위주 코드였고, 1바이트의 남은 공간에 각 나라가 자국 문자를 할당하였다.

하지만 이런 상황에서 다른 국가에 이메일을 보냈더니 글자가 와장창 깨졌던 것. 이에 따라 2~3바이트의 넉넉한 공간에 세상의 모든 문자를 할당한 결과물이 이것이다.

흔히 우리가 웹 브라우저의 인코딩을 설정하면서 자주 보는 UTF-8이라는 말이 이것이고, 바로 유니코드에 기반한 인코딩 방식 중 하나를 가리키는 것이다



문자열 자료형

■ String

- 문자열을 사용할 때는 String 자료형을 사용한다.
- 값은 쌍따옴표("")로 감싸준다.

```
public class StringType {  
  
    public static void main(String[] args) {  
        //문자열 자료형  
        String s = "k"; //문자열일 경우 "" 불임.  
  
        String name = "Ja" + "va";  
        String str = name + 8.0;  
  
        System.out.println(s);  
        System.out.println(name);  
        System.out.println(str);  
        System.out.println(3 + "4");  
    }  
}
```



▪ boolean

- 논리값 true(참), false(거짓)을 표현하는 자료형
- 프로그램 수행이 잘되었는지 여부, 값이 존재하는지 여부 등
- boolean으로 선언 예) **boolean** isMerried = true

```
// 회원 정보  
String name = "추신수";  
int age = 38;  
boolean isMerried = true;  
int numberofChildren = 3;
```

```
System.out.println("이름 : " + name);  
System.out.println("나이 : " + age);  
System.out.println("결혼 유무 : " + isMerried);  
System.out.println("자녀수 : " + numberofChildren);
```

Member.java

이름 : 추신수
나이 : 38
결혼 유무 : true
자녀수 : 3



형 변환(Type Conversion)

▪ 형 변환

- 자료형은 각각 사용하는 메모리 크기와 방식이 다름
- 정수와 실수를 더할때 하나의 자료형으로 통일한 후 연산을 해야함.

묵시적 형변환

- 작은 자료형에서 큰 자료형으로 변환

```
int iNum = 20;  
float fNum = iNum;
```

- 연산 중 자동변환

```
double dNum = fNum + iNum
```

명시적 형변환

- 큰 자료형에서 작은 자료형으로 변환
변환 자료형을 명시 : () 괄호 사용

```
double dNum = 12.34;  
int iNum = (int)dNum;
```



형 변환(Type Conversion)

```
public class TypeConversion {  
    public static void main(String[] args) {  
        //묵시적 형 변환  
        int iNum = 20;  
        float fNum = iNum;  
        System.out.println(iNum); //20  
        System.out.println(fNum); //20.0  
  
        double dNum;  
        dNum = iNum + fNum;  
        System.out.println(dNum); //40.0  
  
        //명시적 형변환  
        double dNum1 = 1.2;  
        float fNum2 = 0.9F;  
  
        int iNum3 = (int)dNum1 + (int)fNum2;  
        int iNum4 = (int)(dNum1 + fNum2);  
        System.out.println(iNum3); //1  
        System.out.println(iNum4); //2  
    }  
}
```



형 변환(Type Conversion)

```
public class TypeConversion2 {  
    public static void main(String[] args) {  
        //사칙연산  
        int x = 10, y = 20;  
  
        System.out.println(x + y);  
        System.out.println(x - y);  
        System.out.println(x * y);  
        System.out.println(x / y);  
  
        System.out.println(x / (double) y);  
    }  
}
```



상수(Constant)

■ 상수(constant)

- 상수 : 변하지 않는 값(1년은 12개월, 원주율은 3.14 등)
- 상수 선언하기 : **final** 키워드 사용, 대문자 사용

```
final double PI = 3.14;
```

```
final int MAX_NUM = 100;
```

final로 선언된 상수는 다른 값을 대입할 수 없음

PI = 3.15 //에러



상수와 리터럴

■ 상수 실습 예제

```
public class ConstantEx {
    public static void main(String[] args) {
        //상수로 선언하기
        final int MAX_NUM = 100;
        final int MIN_NUM;

        MIN_NUM = 0;
        //MAX_NUM = 1000; //final로 선언하면 변경할 수 없음

        System.out.println(MAX_NUM);
        System.out.println(MIN_NUM);

        // 원의 넓이 구하기(반지름*반지름*3.14)
        final double PI = 3.14;
        int radius = 5;
        double area = PI * radius * radius;
        System.out.println("원의 넓이는 " + area + "입니다.");
    }
}
```



컴퓨터에서 데이터 표현하기

- 컴퓨터는 0과 1로만 데이터를 저장함(0-> 신호꺼짐, 1-> 신호켜짐)
- Bit(비트) : 컴퓨터가 표현하는 데이터의 최소 단위로 2진수 하나의 값
을 저장할 수 있는 메모리의 크기
- 숫자, 문자 표현 – 컴퓨터 내부에서는 숫자뿐만 아니라 문자도 2진수로 표현
예) 'A'는 65로 정함(아스키코드) → 이진수로 01000001, 한글 'ㄱ'은 12593(유니코드)

10진수	2진수
0	00000000
1	00000001
2	00000010
3	00000011
...	...
9	00001001
10	00001010

문자	코드값	2진수
A	65	01000001
B	66	01000010
C	67	01000011
...
0	48	00110000
1	49	00110001
2	50	00110010



컴퓨터에서 데이터 표현하기

▪ 비트로 표현할 수 있는 수의 범위

비트수	표현할 수 있는 범위(십진수)	
1bit	0, 1(0~1)	2^1
2bit	00, 01, 10, 11(0~3)	2^2
3bit	000, 001, 010, 011, 100, 101, 110, 111(0~7)	2^3

16진수	2진수
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111
10	10000

※ 10진수를 2진수로 바꾸기

가중치 방식

$$10 = 1010_{(2)}$$

8 4 2 1

$$1 \ 0 \ 1 \ 0 (1 \times 2^3 + 1 \times 2^1 \rightarrow 8 + 2)$$

자리 올림 발생



이진수, 16진수 표기

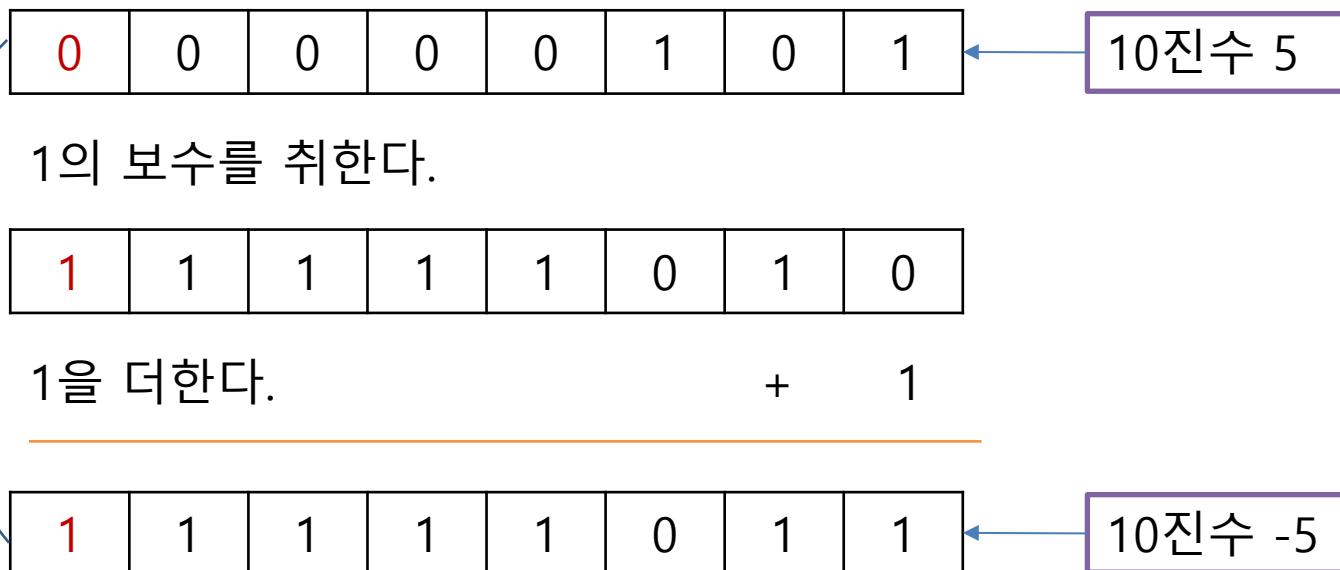
```
public class BinHexPrint {  
  
    public static void main(String[] args) {  
        //2진수, 16진수 표기  
        int num = 10;  
        int bNum = 0b1010;  
        int hNum = 0xA;  
  
        System.out.println(num);      //10  
        System.out.println(bNum);     //10  
        System.out.println(hNum);     //10  
    }  
}
```



부호 있는 수를 표현하는 방법

● 음의 정수는 어떻게 표현할까?

- 정수의 가장 왼쪽에 존재하는 비트는 부호비트입니다.
MSB(Most Significant Bit) 가장 의미있는 비트라는 뜻
- 음수를 만드는 방법은 2의 보수(1의 보수에 1을 더함)를 취한다.
- 1의 보수 표기법 - 0을 1로, 1을 0으로 표기



부호 있는 수를 표현하는 방법

```
public class BinaryTest {  
  
    public static void main(String[] args) {  
        //2진수를 음의 정수로 만드는 방법 - 2의 보수 법  
        int num1 = 0b0000000000000000000000000000000101;  
        int num2 = 0b111111111111111111111111111111011; //보수에 1을 더함  
  
        System.out.println(num1); //5  
        System.out.println(num2); // -5  
  
        int sum = num1 + num2;  
        System.out.println(sum);  
    }  
}
```

