# JAVA 기초 강의

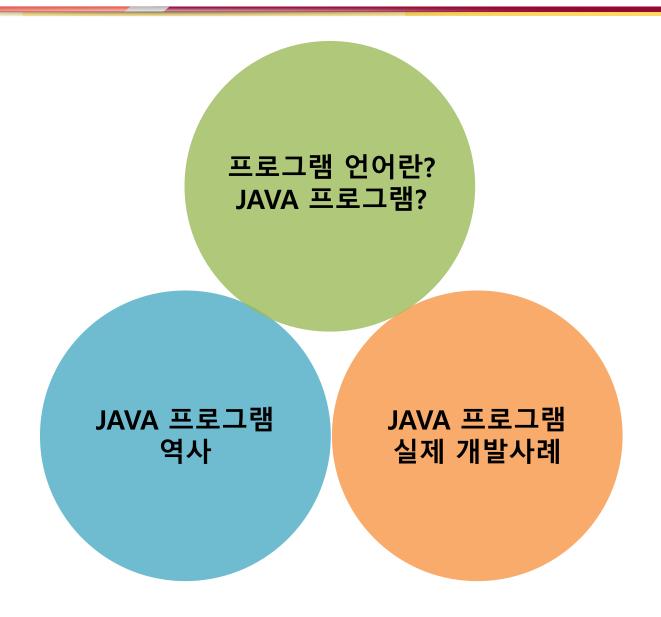
2022. 2

강 현 준

human@human.or.kr

# 1장. JAVA 시작하기

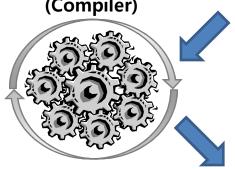
# 1.1 자바 개요



# 1.1.1. 프로그램 언어란?

#### 컴파일러





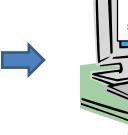
## 프로그램 작성

```
Class Calc {
 public void static main (String [] args) {
   int calc = 3+4;
   System.out.println ("결과 = " + calc);
```



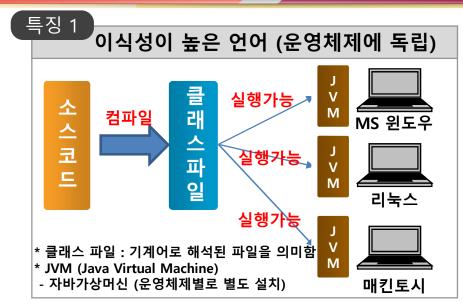
# 기계어로 변환

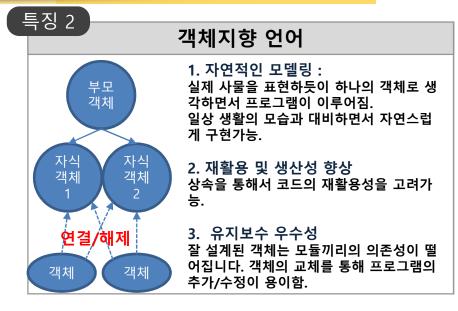
01010100010100100001001010.... 01011100010100100001001010.... 10001010010000100101000101.... 01010100010100100001001010.... 10100010100100001010100101.... 01010100010100100001001010.... 10010000100101001000010010....

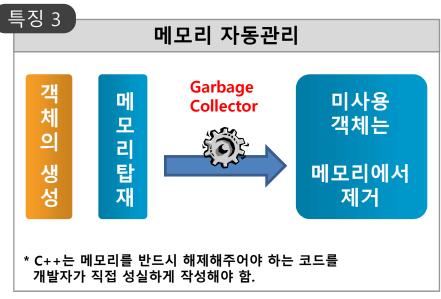




## 1.1.2. JAVA 프로그램의 특징?







## 1.1.2. JAVA 프로그램의 특징?

#### 특징 4

## 다양한 애플리케이션 개발 가능

명령어 기반의 Console / 사용자 화면 기반의 UI 프로그램 인터넷 기반의 서버 프로그램 / 안드로이드 모바일 프로그램

#### 「JAVA 개발도구 및 APII

#### JAVA SE

(Standard Edition)

JAVA EE

(Enterprise Edition)

JAVA ME

(Micro Edition)

- JAVA 프로그램 최소한의 JVM 제공
- 프로그램 개발도구와 API 정의
- JDK 형태로 제공함. (각자 PC 설치 가능)
- 서버 애플리케이션 개발시 사용함.
- WAS(Web Application Server)의 형태로 제공 (Tomcat, WebLogic 등)
- 임베디드 / 모바일 장비를 위한 개발도구 및 API 제공

## 특징 6

## 막강 Open 라이브러리 풍부

- JAVA는 Open 소스 기반이므로 Java로 개발된 라이브러리도 많음.
- 기 개발된 라이브러리 활용시 프로그램 개발에 들어가는 노력과 비용을 줄일 수 있고, 유지보수의 편의성을 위해서 사용이 가능함.



## 1.1.3. JAVA 프로그램의 역사

- ➤ JAVA는 1996년부터 2~3년마다 새로운 버전이 발표됨. (안정성 때문)
- ▶ 그러나 JAVA SE 9 이후부터는 변화에 맞추기 위해 6개월마다 버전 발표함.
  - JDK 1.0 (1996. 01. 23)
  - JDK 1.1 (1996. 02. 19)
  - J2SE 1.2 (1998. 12. 08) :
- 이때부터 JAVA 2로 명명함. 이 때 JAVA의 많은 형태가 확정됨. - J2SE, J2EE, J2ME 의 형태로 완성됨.
  - J2SE 1.3 (2000. 05. 08)
  - J2SE 1.4 (2002. 02. 06)
  - J2SE 5.0 (2004. 09. 30):
    - JAVA 1.5라고도 함. 이 때 JAVA의 많은 기능이 추가됨.

- 이 때도 JAVA의 많은 기능이 추가됨.

- Java SE 6 (2006. 12. 11)
- Java SE 7 (2011. 07. 28)
- Java SE 8 (2014. 03. 18) :
- Java SE 9 (2017. 09. 21)
- Java SE 10 (2018. 03. 20)
- Java SE 11 (2018. 09. 25)
- Java SE 12 (2019. 03. 19)
- .. 현재도 지속 발표되고 있음 (현재기준 18Ver)

JDK: Java Development Kit (JAVA 개발 도구)



# 1.1.4. JAVA 실제 구현사례

PC 어플리케이션

PC에서 구동되는 프로그램 JAVA 개발도구



생산스케줄 프로그램



웹 애플리케이션

웹 서버를 이용한 웹사이트 프로그램 JAVA 개발 Framework



생산계획 솔루션



모바일 앱

안드로이드 프로그램

모바일 개발 Tool



모바일 솔루션



기타

빅데이터



게임



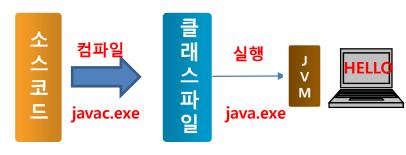
# 1.2.1. JAVA 개발 환경 구축

# JDK 및 JRE 다운로드

- 1. JDK와 JRE의 차이점
- JDK : JVM + 라이브러리 + 컴파일러
- JRE : JVM + 라이브러리
- 2. JAVA를 통해 개발하려면 JDK 설치하면 되고, JAVA로 개발된 파일을 통해서 구동만 하려면 JRE만 설치하면 됨
- 3. JDK는 오라클 웹사이트에서 다운로드 가능함. (www.oracle.com)

### 환경설정

- 1. 컴파일러 : javac.exe
- 2. JAVA 구동: java.exe
- 3. Path 설정: java설치파일의 bin 폴더 path 설정



#### 이클립스설치

- 1. Java 개발도구.
- 2. www.eclipse.org에서 무료 사용가능
- 3. 워크스페이스 설정 : 프로그램을 개발하여 관리할 주요 위치
- 4. 퍼스펙티브 및 뷰 설정

## 1.2.2. JAVA 소스분석

```
[ 파일명 HelloHuman.java ]

public class HelloHuman
{
public static void main (String [] args)
{
System.out.println ("Hello Human Computer");
}
}
```

# 1.2.3. 주석

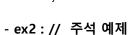
#### 3. 주석

- 실행되지 않는 코드문
- 코드에 설명을 붙일 때 주로 사용함.
- 형태

주석기호	설명
/* */	/* ~ */ 사이에 있는 모든 문장을 주석으로 인지함
//	// 가 있는 부분부터 라인 끝까지를 주석으로 인지함.

```
- ex1 : /* ... */ 주석 예제
```

```
public class HelloHuman {
  public static void main (String [] args)
  {
    System.out.println ("Hello Human Computer1");
/* 2022.5.8 주석 추가하여 임시 TEST
    System.out.println ("Hello Human Computer2");
    System.out.println ("Hello Human Computer3");
*/
  }
}
```



```
public class HelloHuman
{
  public static void main (String [] args)
  {
    System.out.println ("Hello Human Computer1");
    System.out.println ("Hello Human Computer2");
    System.out.println ("Hello Human Computer3");
  }
}
```



#### [출력예제]

Hello Human Computer1

\* 주석 처리된 Hello Human Computer2 Hello Human Computer3 은 출력안됨.



#### [출력예제]

Hello Human Computer1 Hello Human Computer3

\* Hello Human Computer2는 출력안됨.

# *⑤* 휴먼교육센터

# 1.2.4. 실행문

#### 4. 실행문

- 실행문은 변수 선언, 값 저장, 메소드 호출 등에 해당하는 코드
- 실행문의 마지막에는 세미콜론(;)을 붙여서 문장의 끝을 알려야 함.
- 한 줄에 여려개의 문장끝을 알려도 문제없음
- 아래의 2개 예제는 같은 결과임. (단, 가독성을 위해서 ex1을 권장함.)

```
- ex1:
       public class HelloHuman
                                                                                     [출력예제]
        public static void main (String [] args)
                                                                                     285
          int kor = 100;
          int eng = 90;
          int math = 95;
          System.out.println (kor+eng+math);
- ex2:
                                                                                     [출력예제]
       public class HelloHuman
                                                                                     285
        public static void main (String [] args)
          int kor = 100; int eng = 90; int math = 95;
          System.out.println (kor+eng+math);
```



# 2장. 변수와 타입

# 2.1. 변수

- 변수란 하나의 값을 저장하는 메모리 공간이다.
- 변수를 사용하기 위해서는 변수명를 선언해야 함.

#### 변수 상세 설명

## 선언

- **1. 변수의 선언** - 타입 + 변수명 ;
- 2. 선언 예시

int age; // 정수값을 저장할 수 있는 age 변수의 선언 double average; // 실수값을 저장할 수 있는 avg변수의 선언 int x, y, z; // x,y,z의 변수를 정수형으로 선언

## 자바예약어

분류	예시		
기본타입	boolean, byte, int, long, float, double 등		
접근지정자	private, protected, public 등		
클래스 관련	class, abstract, interface, extends, implements 등		
객체 관련	new, instanceof, this, super, null 등		
메소드 관련	void, return 등		
제어문 관련	if, else, switch, break, continue, for, while 등		
기타	try, catch, finally, throw, true, false 등		





변수명 작성규칙	예시
첫번째 글자는 문자, \$, _ 이어야 하고, 숫자가 될 수 없음	가능 : human, \$human, _human 불가 : @human, \$#human
영어 대/소문자 구분함	Human과 human은 서로 다른 문자임
관례적으로 첫문자는 소문자로 시작하되, 다른 단어와 혼용할 경우 대문자로 구분	예시) maxPrice, minSpeed 등
자바 예약어는 사용 불가함	위의 표 참조
문자수는 제한없음	



## 2.1. 변수

- 변수를 사용하는 것은 변수에 값을 저장 및 활용하는 행위를 의미함.

#### 변수 사용 예제

## 변수저장

또는

```
int human; // 변수의 선언
human = 100; // human에 100이란 정수를 저장
```

int human = 100 ; // 위와 같은 문장임.

# 

## 변수활용

```
System.out.print(human); // 화면에 human에 담겨진 100 이란 숫자가 출력됨.
```

System.out.print(human+10); // 화면에 출력하되 human에 담긴 100이란 숫자에 10을 더해서 110을 출력함.

int result = human+10; // result란 변수에 110을 저장함. human(100)+10

## 2.2. 데이터 타입

#### - 데이터 타입: 정수타입, 실수타입, 논리타입 이 있음

#### 데이터 타입 특징

값의 종류	기본타입	메모리 사용크기	저장되는 값의 범위
정수	byte	1 byte ( 8 bit)	-2 <sup>7~(27</sup> -1) 에 해당하는 -128~127 사이의 값 처리 가능
	char	2 byte (16 bit)	0~(216-1) 에 해당하는 0~65535 사이의 값 처리 가능
	short	2 byte (16 bit)	-2 <sup>15~(</sup> 2 <sup>15</sup> -1) 에 해당하는 -33768~33767사이의 값 처리 가능
	int	4 byte (32 bit)	-2 <sup>31~(</sup> 2 <sup>31</sup> -1) 사이의 값 처리 가능
	long	8 byte (64 bit)	-2 <sup>63~(</sup> 2 <sup>63</sup> -1) 사이의 값 처리 가능
실수	float	4 byte (32 bit)	지수(8 bit) + 가수 (32 Bit) + 부호(1 Bit) 처리 가능 ex) 1.2345는 0.12345 x 10 <sup>1</sup> → 지수 (1), 가수 (0.12345)임
	double	8 byte (64 bit)	지수(11 bit) + 가수 (52 Bit) + 부호(1 Bit) 처리 가능
논리	boolean	1 byte ( 8 bit)	true, false

## 체크사항

#### 1. char형은 한글자의 문자를 표현함

char human1 = 'A' // 한문장일 때는 홑따옴표 char human2 = 65
Sytem.out.println (human1); // A가 출력됨
Sytem.out.println (human2); // human2는 65가 저장되어 있지만 ASC코드인 A가 출력됨
Sytem.out.println ( (int)human2); // 65 출력
// 정수형으로 형변환시 65 출력됨.

#### 2. long형

- 8byte 이상의 큰 수로 초기화 시 L을 붙여야 함.
ex) long human3 = 20; // 에러 발생하지 않음
long human4 = 100000000000; // 에러 발생
long human5 = 10000000000L; // 에러 발생하지 않음

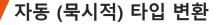
#### 3. Float형

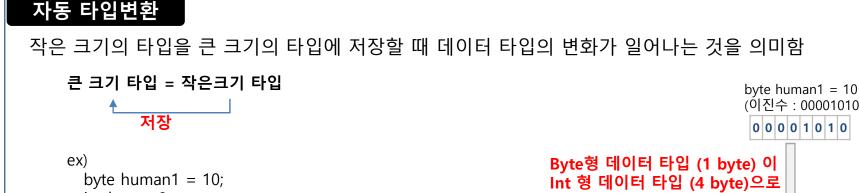
끝에 'F' / 'f' 를 붙여서 사용해야 함.
 ex) float human6= 3.14 (에러 발생)
 float human7= 3.14F (에러 발생하지 않음)



## 2.3. 데이터 타입의 변환

- 데이터 타입변환 : 데이터 타입을 다른 타입으로 변환하는 것 (byte<> int 등)
- 종류: 자동(묵시적) 타입변환, 강제(명세적) 타입변환





int human2 ;
human2 = human1;
int human2 = 10

int 형 데이터 타입 (4 byte)으로 자동변환되어 human2에 저장됨

int human2 = 10

3 byte 공간에 0으로 채워짐

### 주의사항

ex)
byte human1 = -10;
char human2;
human2 = human1;

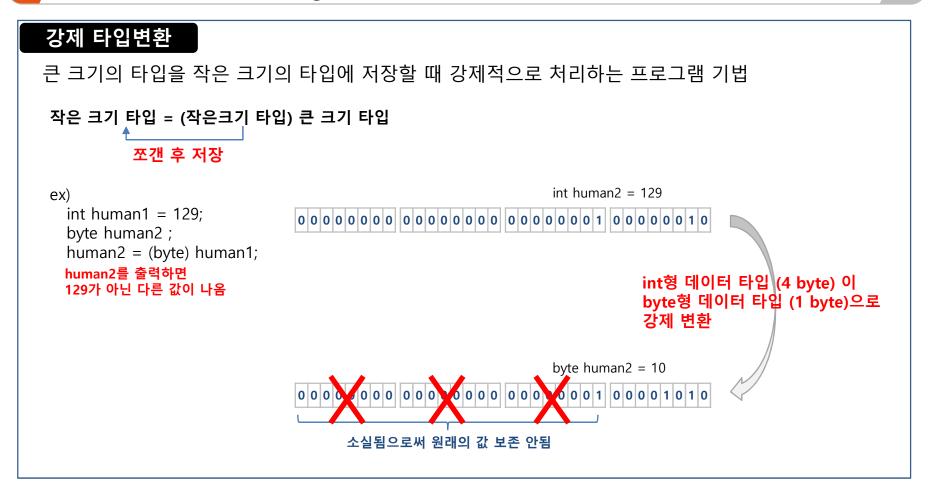


char 타입은 0~65535 의 값을 처리 가능합니다. 음수는 사용불가. 이 때는 에러 발생함.

## 2.3. 데이터 타입의 변환

- 데이터 타입변환 : 데이터 타입을 다른 타입으로 변환하는 것 (byte<> int 등)
- 종류 : 자동(묵시적) 타입변환, 강제(명세적) 타입변환

강제 (명시적) 타입 변환 (casting)



# 3장. 연산자

## 3.1. 연산자

- 연산자 : 산술, 부호, 문자열, 대입, 증감, 비교, 논리, 조건 등의 연산을 하는 기준

- 연산식 : 연산자를 통해 각 변수 또는 상수가 연산을 하는 식

#### 특징

## 연산자

연산자의 종류	연산자	기능설명	예시
산술연산자	+, -, *, /, %	사칙연산, 나머지 계산	* : 곱하기, / : 나누기
부호연산자	+, -	양수, 음수의 부호	Int a = -10 → 음수 10을 a의 변수에 저장함
문자열연산자	+	문자열 연결	System.out.println내에서 사용하는 예제
대입연산자	=, +=, -=	우변의 값을 좌변에 대입함	human += 1 → human = human+1
증감연산자	++,	1 증가 및 감소	human++ → human = human + 1
관계연산자	<,>, =>, =<, !=	값의 관계를 비교함	비교 → 결과는 true,false
논리연산자	&&,   , ^, !	and, or 및 논리부정 반환	
조건연산자	(조건)? A : B	조건이 참이면 A, 거짓이면 B 수행	

## 연산우선순위

1. 왼쪽에서 오른쪽으로 연산 수행 (일반적)

2. 대입연산자의 경우 오늘쪽에서 왼쪽으로 수행함

연산 순서

$$human1 = human2 = human3 = 100$$

## 3.2. 증감연산자

- 증감연산자 : 변수의 값을 1증가 또는 감소시키는 연산자
- 주의사항 : 순서를 확인함에 있음

### 증감연산자의 특징

#### 문법

human++ human = human + 1++human

human-human = human - 1--human

#### 주의사항

1. ++human의 경우 증감을 우선한 후 다른 연산자와 계산함 2. human++ 의 경우 다른 연산자와 계산 후 증감함

```
int human = 99;
int result = ++human * 10;
System.out.println ("human="+human);
System.out.println ("result="+result);
→
human = 100
result = 1000
++human을 통해 human에는 100이 우선 처리됨.
그래서 result = 100 * 10 = 1000
```

int human = 99; int result = human++ \* 10; System.out.println ("human="+human); System.out.println ("result="+result); human = 100result = 990

human++은 (human\* 10)를 연산한 이후에 처리됨.

그래서 result = 99 \* 10 = 990 이 되고, Human은 100이 된다.

## 3.3. 논리부정연산자 / 문자열연결연산자

- 논리부정연산자 : true → false로, false → true로 변환함 (boolean 타입만 사용가능)
- 문자열 연결연산자 : 문자열끼리 연결시킴 (String 타입에서 사용)

#### 특징

## 논리부정 연산자

boolean human1 = false;

! true → false ! false → true boolean human2 = !human1; // !human1은 false에서 true로 변경됨

System.out.println (human2); // true가 출력됨

! 는 Not을 의미함

## 문자열연결연산자

❖ 2개 이상의 연산자를 연결함

String human1 = "휴먼";

String human2 = "교육";

String human = human1 + human2 + "센터";

System.out.println (human); // "휴먼교육센터" 가 출력됨

# *⑤* 휴먼교육센터

## 3.4. 관계연산자 / 논리연산자

- 관계연산자 : 2개의 값의 관계를 비교를 하여 true, false 를 판별함

- 논리연산자 : 2개의 boolean 값의 관계를 비교하여 true, false 판별함

#### 특징

## 비교관계연산자

#### **⋄** '>, <, >=, <='

. 두 개의 피연산자 사이의 크기를 비교하여 논리형 결과 (true, false)를 반환

#### ex)

int human = 100;

boolean result= (human >=80) // result은 true

## 항등관계연산자

#### **❖** '==, !='

. == : 2개의 연산자가 동일할 경우 true 반환 .!= : 2개의 연산자가 동일할 경우 false 반환

#### ex)

int human 100;

boolean result= (human ==80) // result은 false

## 논리연산자

Exam\_05

구분	연산식	결과	설명
AND	true && true	true	2개의 연산자
(논리곱) <b>→</b>	true && false	false	모두가 true일 경우 true로 반환
~~이고	false && true	false	
	false && false	false	
OR	true    true	true	2개의 연산자 중 true가 있을 경우
│ (논리합) │ <b>→</b>	true    false	true	true가 있을 경우 true로 반환
~~ 또는	false    true	true	
	false    false	false	

구분	연산식	결과	설명
XOR - 배타적 논리합	true ^ true	false	2개의 연산자가
	true ^ false	true	같을 경우 false로 반환
	false ^ true	true	디크대 HL칭
	false ^ false	false	다르면 true반환
NOT - 논리 부정	! true	false	논리값을 바꿈
	! false	true	

# *⑤* 휴먼교육센터