

# INTRODUCTION

---

**JAVA CORE**

# INTRODUCTION

## TO JAVA CORE

**01**

IDE, JDK, JVM và JRE

**Giới thiệu về Java**

**02**

**Chương trình Java đầu tiên**

**03**

Các khái niệm cơ bản

**04**

**Biến, kiểu dữ liệu và toán tử**

**05**

Lớp Math

**06**

# GIỚI THIỆU VỀ JAVA

NGÔN NGỮ LẬP TRÌNH MẠNH MẼ



# **JAVA LÀ GÌ?**



## JAVA LÀ GÌ?

### NGÔN NGỮ LẬP TRÌNH

Theo Wikipedia, "ngôn ngữ lập trình là ngôn ngữ hình thức bao gồm một tập hợp các **lệnh** tạo ra nhiều loại đầu ra khác nhau. Ngôn ngữ lập trình được sử dụng trong lập trình máy tính để **thực hiện các thuật toán**. Hầu hết các ngôn ngữ lập trình bao gồm các lệnh cho máy tính."

Hay nói một cách dễ hiểu hơn, để có thể lập trình, chúng ta phải sử dụng các ngôn ngữ lập trình. **Lập trình**, một cách dân dã, là "**dạy máy tính học, dạy máy tính làm**".

→ chúng ta thông qua ngôn ngữ lập trình để có thể "**ra lệnh**" và "**sai bảo**" cho máy tính (một thiết bị vô tri, vô giác) có thể thực hiện được các công việc số hóa mà ta mong muốn.



### MÃ CON NGƯỜI, MÃ MÁY

**Ngôn ngữ con người** - hay mã người - là một thứ trung gian để con người có thể biểu đạt tư duy, suy nghĩ và tình cảm, miễn là sự giao tiếp giữa người và người đạt được mức hiểu ý đối phương biểu đạt.

**Mã máy tính**, hay gọi tắt là mã máy, là thứ trung gian để máy tính có thể thực hiện được các việc, thể hiện xử lý, giao tiếp với các thiết bị điện tử vô tri (như CPU, RAM, Disk, ...)



## JAVA LÀ GÌ?

---

### NGÔN NGỮ LẬP TRÌNH BẬC CAO VÀ THẤP

**Bậc của ngôn ngữ lập trình** được xác định bởi việc "gần" so với ngôn ngữ con người hay không. Cụ thể, nếu như ngôn ngữ lập trình đó càng **dễ hiểu với con người** (con người có thể đọc được càng dễ thì độ dễ hiểu càng cao) thì bậc của ngôn ngữ lập trình đó càng **cao**. Ngược lại, nếu như con người đọc càng **khó hiểu** thì bậc của ngôn ngữ lập trình đó càng **thấp**.

**Ngôn ngữ lập trình bậc cao (High-level programming language):** Java, Python, C#, PHP, ...

**Ngôn ngữ lập trình bậc thấp (Low-level programming language):** Mã máy, Assembly, C, ...



# JAVA LÀ GÌ?

## ĐỊNH NGHĨA VỀ JAVA

Theo wikipedia: "Java là một ngôn ngữ lập trình **hướng đối tượng**, dựa trên lớp được thiết kế để có càng ít phụ thuộc thực thi càng tốt. Nó là ngôn ngữ lập trình có mục đích chung cho phép các nhà phát triển ứng dụng **viết một lần, chạy khắp mọi nơi**", nghĩa là mã Java đã biên dịch có thể chạy trên tất cả các nền tảng hỗ trợ Java mà không cần biên dịch lại."

Java là ngôn ngữ lập trình có dấu chấm phẩy (semi-colon programming language).

## KHẢ NĂNG CỦA JAVA

- Làm ứng dụng di động
- Xây dựng web
- Lập trình nhúng
- Công nghệ big data
- Điện toán đám mây
- ...









## LỊCH SỬ

James Gosling, Mike Sheridan và Patrick Naughton khởi xướng dự án ngôn ngữ Java vào tháng 6/1991. Java ban đầu được thiết kế cho truyền hình tương tác, nhưng nó quá tiên tiến đối với ngành truyền hình cáp kỹ thuật số vào thời điểm đó.

Ban đầu, ngôn ngữ này được gọi là Oak (Cây sồi) bởi vì đây là cây được trồng rất phổ biến tại Mỹ, Đức, Romania,... và được xem là như biểu tượng của nước đó, biểu tượng của sức mạnh. Ngoài ra bên ngoài văn phòng của ông Gosling có rất nhiều cây sồi. Tuy nhiên, cái tên Oak này đã được đăng ký trước đó bởi một công ty có tên Oak Technologies.

Do đó, ngôn ngữ này được đổi tên thành Java, đây là tên của một hòn đảo thuộc Indonesia và cũng là tên của một loại cà phê đến từ đất nước này mà các kỹ sư của Sun Microsystem đã gắn bó suốt 4 năm làm việc để cho ra đời phiên bản Java đầu tiên. Đó cũng là lý do biểu tượng của Java là cốc cà phê bốc khói nghi ngút.



## CÁC PHIÊN BẢN CỦA JAVA



Version	Release date
JDK Beta	1995
JDK 1.0	January 1996
JDK 1.1	February 1997
J2SE 1.2	December 1998
J2SE 1.3	May 2000
J2SE 1.4	February 2002
J2SE 5.0	September 2004
Java SE 6	December 2006
Java SE 7	July 2011
Java SE 8 (LTS)	March 2014
Java SE 9	September 2017
Java SE 10	March 2018
Java SE 11 (LTS)	September 2018
Java SE 12	March 2019
Java SE 13	September 2019
Java SE 14	March 2020
Java SE 15	September 2020
Java SE 16	March 2021
Java SE 17 (LTS)	September 2021





## ĐẶC TÍNH

### CÁC ĐẶC TÍNH CỦA JAVA

- Là **ngôn ngữ lập trình hướng đối tượng**, các chương trình Java được xây dựng dựa trên việc thiết kế các lớp và đối tượng.
- **Đơn giản**: Java được thiết kế với mục đích giúp người học dễ dàng hơn trong việc tiếp thu kiến thức. Vì vậy nếu bạn đã hiểu cơ bản Java là gì thì khi học, bạn có thể nắm bắt ngôn ngữ này rất nhanh.
- **Độc lập nền tảng**: Một chương trình Java có thể chạy trên nhiều máy tính khác nhau (Windows, Unix, Linux,...) với điều kiện máy đó có cài đặt JVM
- **Khả chuyển**: Chương trình ứng dụng viết bằng ngôn ngữ Java chỉ cần được chạy trên JVM là có thể chạy được trên bất kỳ máy tính, hệ điều hành nào có JVM. Write Once, Run Anywhere - Viết một lần, chạy mọi nơi
- **Đa nhiệm, đa luồng**: Java hỗ trợ đa nhiệm, đa luồng cho phép nhiều tiến trình, tiểu trình có thể chạy song song cùng một thời điểm và tương tác với nhau
- **An toàn**: Code luôn được kiểm tra trước khi thực thi, có nhiều mức độ bảo mật -> Môi trường thực thi an toàn

### CÁC NỀN TẢNG JAVA

- Java Standard Edition (Java SE - JSE)
- Java Enterprise Edition (Java EE - JEE)
- Java Micro Edition (Java ME - JME)
- ...



FUN  
DAM  
ENT  
AL

# IDE, JDK, JVM VÀ JRE

CÁC KHÁI NIỆM NỀN TẢNG

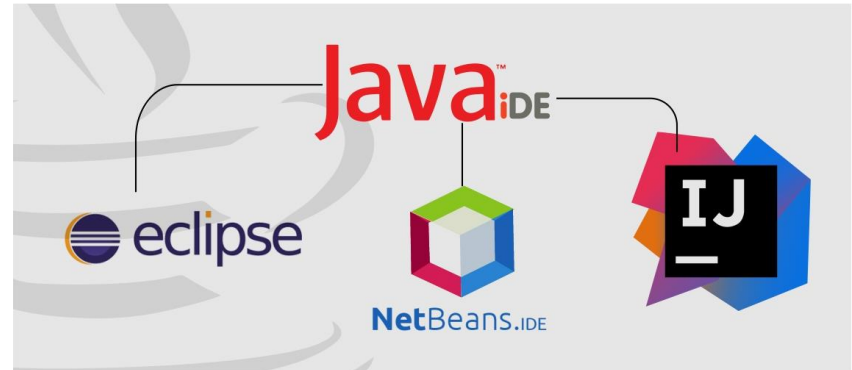


## ĐỊNH NGHĨA VỀ IDE

**Môi trường phát triển tích hợp (Integrated development environment)** còn được gọi là "Môi trường thiết kế hợp nhất" hay "Môi trường gỡ lỗi hợp nhất" là một loại phần mềm máy tính có công dụng giúp đỡ các lập trình viên trong việc phát triển phần mềm. Các môi trường phát triển hợp nhất thường bao gồm một trình soạn thảo mã nguồn dùng để viết mã.

## MỘT SỐ IDE PHỔ BIẾN CHO LẬP TRÌNH VIÊN JAVA

- IntelliJ IDEA
- Netbeans
- Eclipse
- Visual Studio Code (text editor)
- Apache Ant
- ...





## IDE

---

### TẢI VÀ CÀI ĐẶT IDE

- **IntelliJ IDEA Official:** <https://www.jetbrains.com/idea/download/#section=windows>
- **IntelliJ IDEA crack:** <https://kipalog.com/posts/Crack-IntelliJ-IDEA-new-versions-2021>
- **Netbeans:** <https://netbeans.apache.org/download/index.html>
- **Eclipse:** <https://www.eclipse.org/downloads/>
- **Visual Studio Code:** <https://code.visualstudio.com/download>

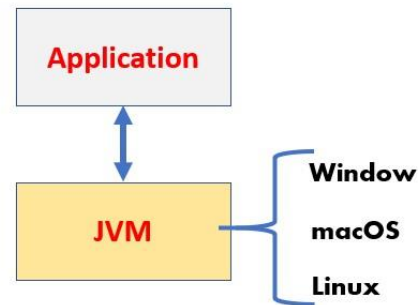




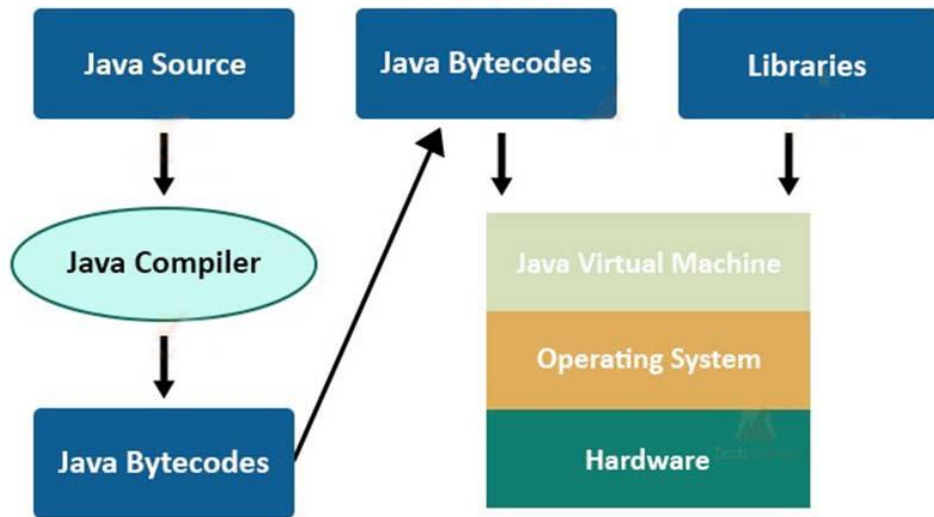
## ĐỊNH NGHĨA VỀ JVM

**JVM (Java Virtual Machine)** là một máy ảo cho phép chạy các chương trình Java cũng như các chương trình viết bằng các ngôn ngữ khác nhưng được biên dịch sang mã bytecode của Java. Do đó, từ một chương trình nhưng thông qua JVM nó có thể chạy được trên các nền tảng khác nhau.

JVM quản lý bộ nhớ hệ thống và cung cấp môi trường thực thi cho ứng dụng Java, nó có hai chức năng chính là cho phép chương trình Java chạy trên mọi thiết bị, nền tảng khác nhau (Write once, Run anywhere) và quản lý, tối ưu bộ nhớ chương trình.



## Cách hoạt động của JVM



# JDK, JRE VÀ JVM

## ĐỊNH NGHĨA VỀ JRE

**JRE (Java Runtime Environment)** khởi tạo JVM và đảm bảo các phụ thuộc có sẵn cho chương trình của chúng ta. JRE chứa các thư viện lớp Java, trình tải lớp Java và JVM. Nó chịu trách nhiệm tải chính xác các lớp và kết nối chúng với các thư viện lớp Java cốt lõi.

## ĐỊNH NGHĨA VỀ JDK

**JDK (Java Development Kit)** là một thành phần nền tảng chính để xây dựng các ứng dụng Java. Trái tim của nó là **trình biên dịch Java**. JRE có thể được sử dụng như một thành phần độc lập để chạy các chương trình Java, nhưng nó cũng là một thành phần của JDK. JDK yêu cầu JRE vì chạy các chương trình Java là một phần của việc phát triển chúng.

### JDK

java, javac, jdb, appletviewer, javah, javaw  
jar, rmi.....

### JRE

Class Loader, Byte Code Verifier  
Java API, Runtime Libraries

### JVM

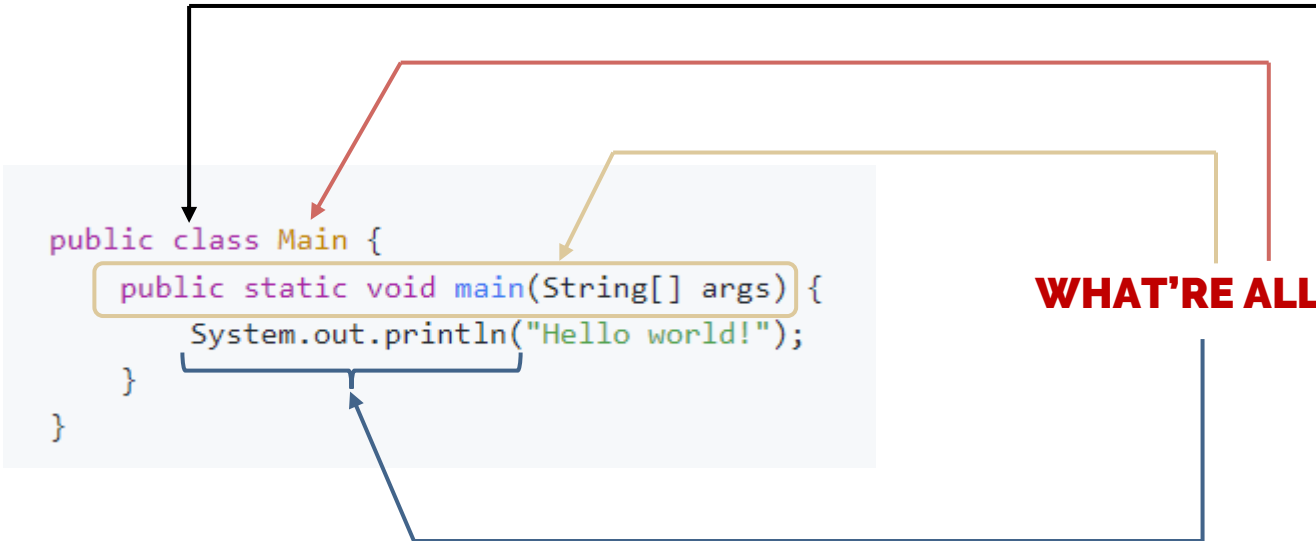
Java Interpreter  
JIT  
Garbage Collector  
Thread Sync.....

# JAVA

## CHƯƠNG TRÌNH JAVA

ĐẦU TIÊN

# RAM



```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello world!");  
    }  
}
```

**WHAT'RE ALL OF THESE?**



## CHƯƠNG TRÌNH JAVA ĐẦU TIÊN

Ví dụ: Viết chương trình in ra dòng chữ: *I'm <your name>*.

# JAVA BASIC CONCEPT

## CÁC KHÁI NIỆM CƠ BẢN

CỦA MỘT CHƯƠNG TRÌNH JAVA



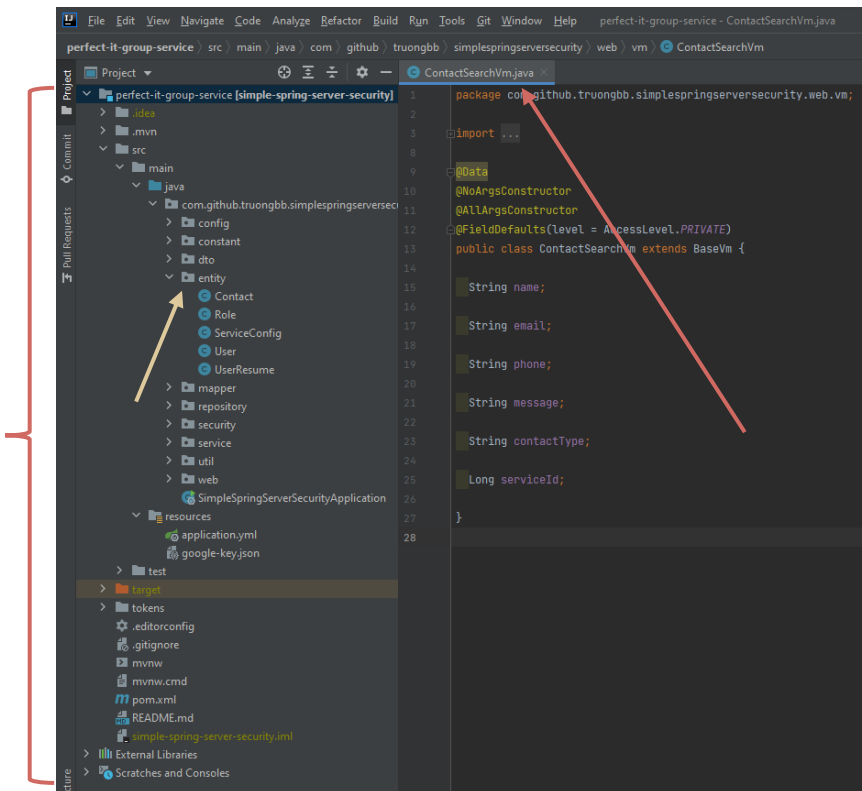


# **PROJECT, PACKAGE, CLASS**

## 01

**Một package (gói)** mô tả không gian có chứa các **lớp (class)** của java, chúng ta có thể xem package như một **thư mục**, còn class chính là các file thuộc thư mục đó.

**Class (lớp)** là một khái niệm sinh ra trong ngôn ngữ lập trình Java để chủ yếu xử lý các vấn đề liên quan tới lập trình hướng đối tượng (sẽ được đề cập trong các bài sau). Tuy nhiên, ở thời điểm hiện tại, ta quan niệm 1 lớp là một file **.java** có tác dụng giải quyết 1 bài toán hoặc 1 tác vụ cụ thể nào đó.





**MAIN,  
IMPORT,  
COMMENT**



## MAIN, IMPORT, COMMENT

---

**Phương thức - Hàm (method)** là một tập hợp các lệnh được viết bằng ngôn ngữ lập trình nhằm mục đích giải quyết một hành động hoặc một vấn đề nhỏ nào đó, hoặc một bài toán lớn (là tập hợp các vấn đề nhỏ lại với nhau). Một chương trình có thể chứa một hoặc nhiều phương thức - hàm, và chỉ có 1 phương thức - hàm duy nhất là hàm chính - phương thức chính (main method).

**Main method (phương thức chính)** là phương thức chính của chương trình, nó có nhiệm vụ thực thi các tác vụ của chương trình và toàn bộ dự án nhằm giải quyết bài toán đề ra.

**Import** là từ khóa được sử dụng trong java nhằm để xác định các class hoặc package được sử dụng trong lớp hiện tại. Cụ thể, khi có một lớp A có sẵn giải quyết công việc X nào đó; trong khi đó ta đang lập trình lớp B, mà lớp B chứa nhiều bước thực hiện để hoàn tất chương trình, trong đó có bước X; khi đó, lập trình viên sẽ không thực hiện lại việc X ở lớp B, mà họ sẽ tận dụng lớp A (đã giải quyết được việc X rồi) bằng cách import lớp A vào lớp B để sử dụng.

**Comment trong Java** là hành động chú thích trực tiếp vào chương trình, nhằm mục đích nêu ra dụng ý của người lập trình, hoặc các giải thích, chú ý về đoạn code chương trình mang lại. Các comment trong Java sẽ bị bỏ qua bởi trình biên dịch và không được thực thi khi chạy chương trình.





## VARIABLES

```
public class Main {  
    public static void main(String[] args) {  
        int x = 10;  
        System.out.println(x);  
    }  
}
```

Biến số (variable)

Lệnh in ra màn hình console

# BIẾN, KIỂU DỮ LIỆU, TOÁN TỬ

TRONG NGÔN NGỮ LẬP TRÌNH JAVA



# **BIẾN SỐ (VARIABLES)**





## BIẾN SỐ (VARIABLE)

**Biến số (Variable)** là vùng nhớ dùng để lưu trữ các giá trị của chương trình. Mỗi biến gắn liền với một kiểu dữ liệu và một định danh duy nhất gọi là tên biến.

Tên biến thông thường là một chuỗi các ký tự hoặc số. Trong java, biến có thể được khai báo ở bất kỳ nơi đâu trong chương trình Java.

### CÚ PHÁP KHAI BÁO BIẾN

```
<Kiểu dữ liệu> <tên biến>;
```

### CÚ PHÁP GÁN GIÁ TRỊ CHO BIẾN

```
<tên biến> = <giá trị>;
```

```
public class Demo {  
    public static void main(String[] args) {  
        int a = 5;  
  
        double b = 3.14;  
  
        String x = "Nguyễn Văn A";  
  
        int c = 7;  
  
        int z = a + c;  
    }  
}
```



## BIẾN SỐ (VARIABLE)

### QUY TẮC KHAI BÁO BIẾN

- Chỉ được bắt đầu bằng một ký tự(chữ), hoặc một dấu gạch dưới(\_), hoặc một ký tự dollar(\$).
- Tên biến không được chứa khoảng trắng.
- Bắt đầu từ ký tự thứ hai, có thể dùng ký tự(chữ), dấu gạch dưới(\_), hoặc ký tự dollar(\$).
- Không được trùng với các từ khóa.
- Có phân biệt chữ hoa và chữ thường.

```
public class Demo {  
    public static void main(String[] args) {  
        int a = 5;  
  
        double b = 3.14;  
  
        String x = "Nguyễn Văn A";  
  
        int c = 7;  
  
        int z = a + c;  
    }  
}
```




## BIẾN SỐ (VARIABLE)

---

### PHẠM VI TRUY CẬP CỦA BIẾN SỐ

- **Biến cục bộ (local variables)**: được khai báo trong các phương thức hoặc trong các block code (được quy định bởi dấu ngoặc nhọn `{ }`).
  - Cần khởi tạo giá trị mặc định cho biến cục bộ trước khi có thể sử dụng.
- 
- **Biến toàn cục (global variables)**: được khai báo trong một lớp (class), bên ngoài các phương thức và các block.
  - Biến toàn cục có giá trị mặc định phụ thuộc vào kiểu dữ liệu của nó. Ví dụ nếu là kiểu `int`, `short`, `byte` thì giá trị mặc định là 0, kiểu `double` thì là 0.0d, ... Vì vậy, bạn sẽ **không cần khởi tạo** giá trị cho biến instance trước khi sử dụng.



# KIỂU DỮ LIỆU (DATA TYPES)



## Kiểu dữ liệu (DATA TYPES)

---

### CÁC KIỂU DỮ LIỆU TRONG JAVA

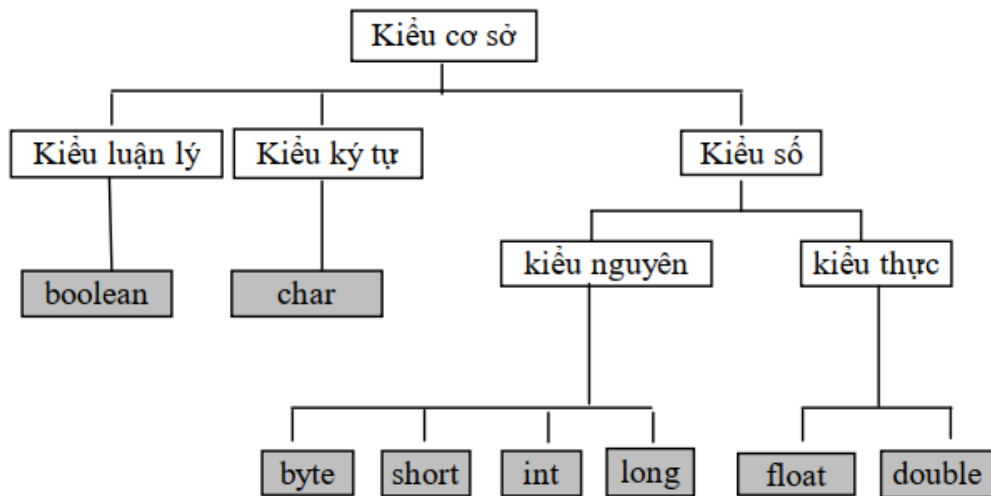
- Kiểu nguyên thủy (primitive data type)

- Kiểu dữ liệu đối tượng (object data type): Biến của kiểu dữ liệu này chỉ chứa địa chỉ của đối tượng tại bộ nhớ stack. Đối tượng dữ liệu lại nằm ở bộ nhớ Heap. Một số kiểu dữ liệu cụ thể có thể kể đến như: mảng (Array), class, interface,... Chúng ta sẽ tìm hiểu kỹ hơn trong các bài sau.



## Kiểu dữ liệu (DATA TYPES)

### Kiểu dữ liệu NGUYÊN THỦY





## Kiểu dữ liệu (DATA TYPES)

### Kiểu số nguyên

Java cung cấp 4 kiểu số nguyên khác nhau là `byte`, `short`, `int`, `long`

Kiểu dữ liệu	Miền giá trị	Giá trị mặc định	Kích cỡ mặc định
byte	-128 đến 127	0	1 byte
short	-32768 đến 32767	0	2 byte
int	$-2^{31}$ đến $2^{31}-1$	0	4 byte
long	$-2^{63}$ đến $2^{63}-1$	0L	8 byte

```
int age;  
age = 35;  
long salary = 4000000L;
```



## Kiểu dữ liệu (DATA TYPES)

### Kiểu số thực

- Java hỗ trợ hai kiểu dữ liệu là **float** và **double**. Kiểu số thực không có giá trị nhỏ nhất và giá trị lớn nhất

Kiểu dữ liệu	Giá trị mặc định	Kích cỡ mặc định
float	0.0f	4 byte
double	0.0d	8 byte

```
float weight;  
weight = 40f;  
double height = 1.6;
```





## KIỂU DỮ LIỆU (DATA TYPES)

### KIỂU KÝ TỰ

- Kiểu ký tự trong Java có kích thước là **2 byte** và chỉ dùng để biểu diễn các ký tự trong bộ mã Unicode. Như vậy char trong Java có thể biểu diễn tất cả  $2^{16} = 65536$  ký tự khác nhau.

- Giá trị nhỏ nhất của một biến kiểu ký tự là 0 và giá trị lớn nhất là 65535

```
char a = 'a';  
char b = '5';  
char c = 65; //theo bảng mã ASCII c == 'A'
```



## Kiểu dữ liệu (DATA TYPES)

### Kiểu luận lý

- Kiểu boolean chỉ nhận 1 trong 2 giá trị: **true** hoặc **false**.
- Kiểu boolean không thể chuyển thành kiểu nguyên và ngược lại.
- Giá trị mặc định của kiểu boolean là **false**.

```
boolean isCheck = false;  
  
int x = 10;  
boolean flag = x % 2 == 0; //Trả về true do x chia hết cho 2
```



## Kiểu dữ liệu (DATA TYPES)

---

Ví dụ: Bài toán tính tổng và hiệu của hai số nguyên, thực.

- Bài toán tính tích và thương của hai số nguyên, thực.
- Bài toán tính thương của 1 số nguyên và 1 số thực.



**ÉP KIỂU,  
HẰNG SỐ**



## ÉP KIỂU, HÀNG SỐ

---

- **Ép kiểu (data parsing)** là cách chuyển đổi kiểu dữ liệu này thành biến thuộc kiểu dữ liệu khác. Trong bài học này, chúng ta chỉ xét tới việc ép kiểu đối với dữ liệu nguyên thủy.

- Ý nghĩa:

- + Việc chuyển kiểu dữ liệu sẽ đến lúc phải cần trong quá trình xử lý chương trình.
- + Có thể định dạng đúng kiểu dữ liệu mình mong muốn.

- Có 2 cách ép kiểu dữ liệu nguyên thủy:

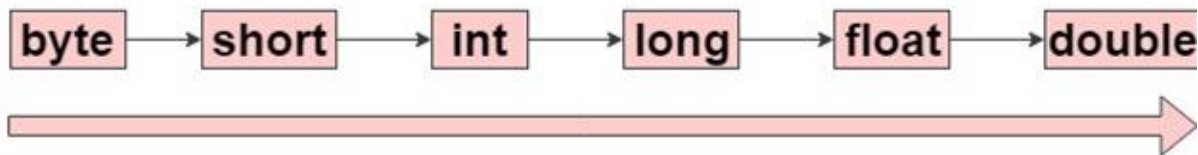
- + Chuyển đổi kiểu ngầm định.
- + Chuyển đổi kiểu tường minh.



## ÉP KIỂU, HÀNG SỐ

### CHUYỂN ĐỔI KIỂU DỮ LIỆU NGẦM ĐỊNH

# Automatic Type Conversion (Widening - implicit)



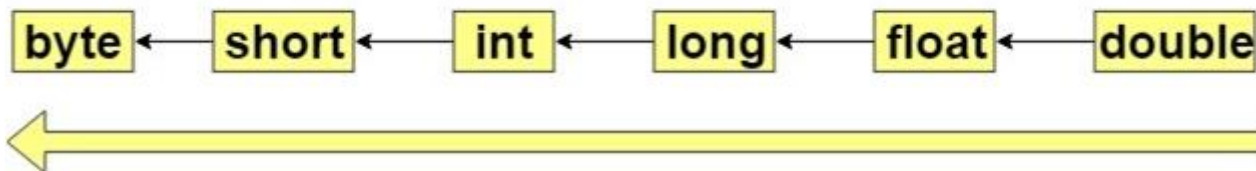
```
int a = 5;  
long b = a;  
System.out.println(b);
```



## ÉP KIỂU, HẰNG SỐ

### CHUYỂN ĐỔI KIỂU DỮ LIỆU TƯỜNG MINH

## Narrowing (explicit)



```
long a = 6;  
int b = (int) a;  
System.out.println(b);
```

**Bảo toàn dữ liệu**

```
double height = 1.7;  
int h = (int) height;  
System.out.println(h);
```

**Mất mát dữ liệu**



## ÉP KIỂU, HÀNG SỐ

---

Ví dụ: Bài toán tính tổng và hiệu của hai số nguyên, thực.

→ Bài toán tính tích và thương của hai số nguyên, thực.

→ Bài toán tính thương của 1 số nguyên và 1 số thực.

→ Bài toán tính sin, cos của một góc trong tam giác vuông khi biết độ dài 3 cạnh.





## ÉP KIỂU, HẲNG SỐ

### HẲNG SỐ

- Hằng số là một giá trị không đổi trong suốt chương trình, được khởi tạo giá trị ngay từ ban đầu.
- Cú pháp khai báo:

```
final <Kiểu dữ liệu> <Tên hằng> = <Giá trị>;
```

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

- Quy ước: Tên hằng số luôn được viết in hoa và phân cách các từ bằng dấu gạch dưới (\_)





## TOÁN TỬ (OPERATOR)

---

### ĐỊNH NGHĨA

- Là các ký hiệu để thể hiện các phép toán học trong Java như cộng, trừ, nhân, chia, so sánh, gán, ...

### PHÂN LOẠI

- Toán tử số học
- Toán tử trên bit
- Toán tử quan hệ
- Toán tử logic
- Toán tử gán
- Toán tử ba ngôi



## TOÁN TỬ (OPERATOR)

### TOÁN TỬ SỐ HỌC

Giả sử với  $a = 30$  và  $b = 10$ , ta có:

Toán tử	Ý nghĩa	Ví dụ
+	Cộng	$a + b = 40$
-	Trừ	$a - b = 20$
*	Nhân	$a * b = 300$
/	Chia lấy nguyên	$a / b = 3$
%	Chia lấy dư	$a \% b = 0$
++	Tăng 1	$a++ = 31$
--	Giảm 1	$b-- = 9$



## TOÁN TỬ (OPERATOR)

### TOÁN TỬ TRÊN BIT

Giả sử với  $a = 30$  ( $00011110$ ) và  $b = 10$  ( $00001010$ ), ta có:

Toán tử	Ý nghĩa	Ví dụ
&	AND	$a \& b = 10$ (00001010)
	OR	$a   b = 30$ (00010100)
^	XOR	$a ^ b = 20$ (00010100)
<<	Dịch trái	$a << 2 = 120$ (01111000)
>>	Dịch phải	$a >> 2 = 7$ (111)
>>>	Dịch phải và điền 0 vào bit trống	$a >>> 2 = 7$ (00000111)
~	Bù bit	$\sim a = -31$



## TOÁN TỬ (OPERATOR)

### TOÁN TỬ QUAN HỆ

Giả sử với  $a = 30$  và  $b = 10$ , ta có:

Toán tử	Ý nghĩa	Ví dụ
<code>==</code>	So sánh bằng	$a == b \Rightarrow \text{false}$
<code>!=</code>	So sánh khác	$a != b \Rightarrow \text{true}$
<code>&gt;</code>	So sánh lớn hơn	$a > b \Rightarrow \text{true}$
<code>&lt;</code>	So sánh nhỏ hơn	$a < b \Rightarrow \text{false}$
<code>&gt;=</code>	So sánh lớn hơn hoặc bằng	$a >= b \Rightarrow \text{true}$
<code>&lt;=</code>	So sánh nhỏ hơn hoặc bằng	$a <= b \Rightarrow \text{false}$



## TOÁN TỬ (OPERATOR)

### TOÁN TỬ LOGIC

Giả sử với  $c = \text{true}$  và  $d = \text{false}$ , ta có:

Toán tử	Ý nghĩa	Ví dụ
<code>&amp;&amp;</code>	Toán tử và	$c \ \&\& \ d \Rightarrow \text{false}$
<code>  </code>	Toán tử hoặc	$c \    \ d \Rightarrow \text{true}$
<code>!</code>	Toán tử phủ định	$!c \Rightarrow c = \text{false}$



## TOÁN TỬ (OPERATOR)

### TOÁN TỬ GÁN

Toán tử	Ý nghĩa	Ví dụ
=	Toán tử đơn giản, gán giá trị toán hạng bên phải cho toán hạng trái	
+=	Thêm giá trị toán hạng phải tới toán hạng trái và gán giá trị đó cho toán hạng trái	$c += a \Rightarrow c = c + a$
-=	Trừ đi giá trị toán hạng phải từ toán hạng trái và gán giá trị này cho toán hạng trái	$c -= a \Rightarrow c = c - a$
*=	Nhân giá trị toán hạng phải với toán hạng trái và gán giá trị này cho toán hạng trái	$c *= a \Rightarrow c = c * a$
/=	Chia toán hạng trái cho toán hạng phải và gán giá trị này cho toán hạng phải	$c /= a \Rightarrow c = c / a;$
%=	Lấy phần dư của phép chia toán hạng trái cho toán hạng phải và gán cho toán hạng trái	$c \% = a \Rightarrow c = c \% a$
<<=	Dịch toán hạng trái sang số vị trí là giá trị toán hạng phải	$c << = a \Rightarrow c = c << a$
>>=	Dịch toán hạng trái sang số vị trí là giá trị toán hạng phải	$c >> = a \Rightarrow c = c >> a$
&=	Phép AND bit	$c \& = a \Rightarrow c = c \& a$
^=	Phép OR loại trừ bit	$c \wedge = a \Rightarrow c = c \wedge a$
=	Phép OR bit	$c   = a \Rightarrow c = c   a$





## TOÁN TỬ (OPERATOR)

### TOÁN TỬ BA NGÔI

**Toán tử ba ngôi** hay còn gọi là **toán tử điều kiện**, là một toán tử được cấu tạo bởi ba đối số gồm biểu thức điều kiện, kết quả khi điều kiện đúng và kết quả khi điều kiện sai. Kết quả ở đây có thể là một giá trị được trả về, cũng có thể là một xử lý sẽ được thực hiện sau đó tùy thuộc điều kiện chỉ định là đúng hay sai.

<điều kiện> ? <Biểu thức 1> : <Biểu thức 2>;

```
int a = 4;  
int b = 2;  
  
String s = (a % b == 0) ? "a chia hết cho b" : "a không chia hết cho b";  
System.out.println(s);
```



a chia hết cho b



## TOÁN TỬ (OPERATOR)

---

Ví dụ: Giải phương trình bậc nhất một ẩn  $ax + b = 0$  (với  $a$  khác  $0$ ).

→ Giải phương trình bậc nhất hai một ẩn  $ax^2 + bx + c = 0$  (với  $a$  khác  $0$ ).

# MATH UTILITY METHODS

## LỚP MATH TRONG JAVA

GIẢM NHẸ CÔNG SỨC TÍNH TOÁN



## LỚP MATH

---

### ĐỊNH NGHĨA

- JDK định nghĩa sẵn một số lớp tiện dụng, một trong số đó là lớp Math cung cấp các hàm về toán học.
- Lớp này nằm trong package java.lang, chứa các phương thức static (Phương thức static sẽ được giải thích kỹ hơn trong các bài sau), do đó ta không cần tạo đối tượng mà vẫn có thể sử dụng các phương thức có trong lớp đó

### MỘT SỐ TIỆN ÍCH CỦA LỚP MATH

- Hằng số: Math.PI
- Hàm lượng giác: sin, cos, tan, cotan, toDegrees, toRadians, ...
- Hàm làm tròn: ceil, round, floor
- Hàm tính toán: sqrt, pow, log, ...
- Hàm tìm kiếm lớn nhất, nhỏ nhất: max, min
- Hàm sinh số ngẫu nhiên: random



Ví dụ: Tính lũy thừa mũ  $n$  của cơ số  $a$ .

→ Tìm số lớn nhất trong 3 số nguyên.

→ Tìm thương của 2 số nguyên, làm tròn kết quả tới 3 chữ số thập phân.