

INTRODUCTION

JAVA CORE

INTRODUCTION

JAVA CORE

01

Giới thiệu về Java

IDE, JDK, JVM và JRE

02

Chương trình Java đầu tiên

03

Các khái niệm cơ bản

04

Biến, kiểu dữ liệu và toán tử

05

Lớp Math

06

JAVA

GIỚI THIỆU VỀ JAVA

NGÔN NGỮ LẬP TRÌNH MẠNH MẼ

AGE

01

JAVA LÀ GÌ?



NGÔN NGỮ LẬP TRÌNH

“

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.



(*) Wikipedia



MÃ CON NGƯỜI, MÃ MÁY



Ngôn ngữ của con 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 đổi phương biểu đạt.



Ngôn ngữ của máy tính – 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, ...)

NGÔN NGỮ LẬP TRÌNH BẬC CAO VÀ BẬC THẤP



Ngôn ngữ lập trình bậc cao
(High - level programming language)

- Java, C#, ...
- Python,



Ngôn ngữ lập trình bậc thấp
(Low – level programming)

- Mã máy
- C, Assembly



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** 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**.

ĐỊNH NGHĨA VỀ JAVA

“

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).

(*) Wikipedia



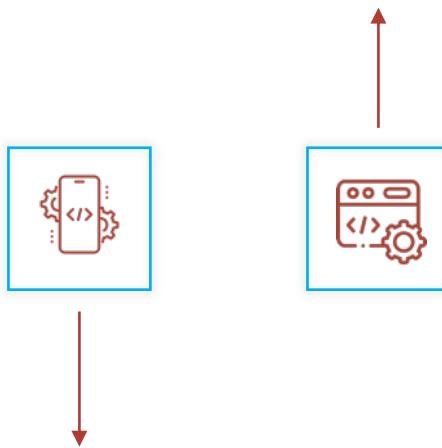
```

        ,
        // appear event when appropriate
        function() {
            if (element.hidden) {
                if (t.appeared) {
                    t.appeared = false;
                }
            }
            //is the element inside the visible window?
            var a = w.scrollLeft();
            var b = w.scrollTop();
            var o = t.offset();
            var x = o.left;
            var y = o.top;
            var ax = settings.accX;
            var ay = settings.accY;
            var th = t.height();
            var wh = w.height();
            var tw = t.width();
            var ww = w.width();
            if (y + th + ay >= b &&
                y <= b + wh + ay &&
                x + tw + ax >= a &&
                x <= a + ww + ax) {
                //trigger the custom event
                if (!t.appeared) t.trigger('appear', settings.data);
            } else {
                //it scrolled out of view
                t.appeared = false;
            }
        };
        //create a modified fn with some additional logic
        var modifiedFn = function() {
            //mark the element as visible
            t.appeared = true;
            //is this supposed to happen only once?
            if (settings.one) {
                //remove the check
                w.unbind('scroll', check);
                var i = $.inArray(check, $fn.appear.checks);
                if (i >= 0) $fn.appear.checks.splice(i, 1);
            }
            //trigger the original fn
            fn.apply(this, arguments);
        };
        //bind the modified fn to the element
        t.one('appear', settings.data, modifiedFn);
        //bind the modified fn to the element
        t.one('appear', settings.data, modifiedFn);
    }
}

```

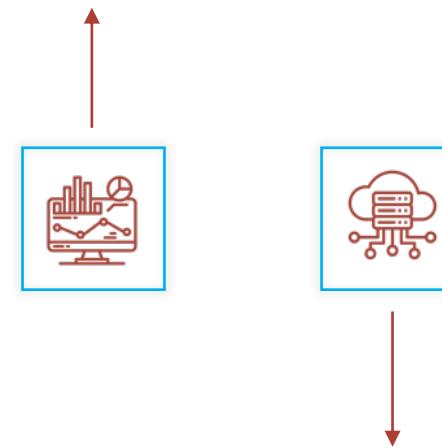
KHẢ NĂNG CỦA JAVA

Xây dựng web



Ứng dụng di động

Công nghệ big data



Điện toán đám mây

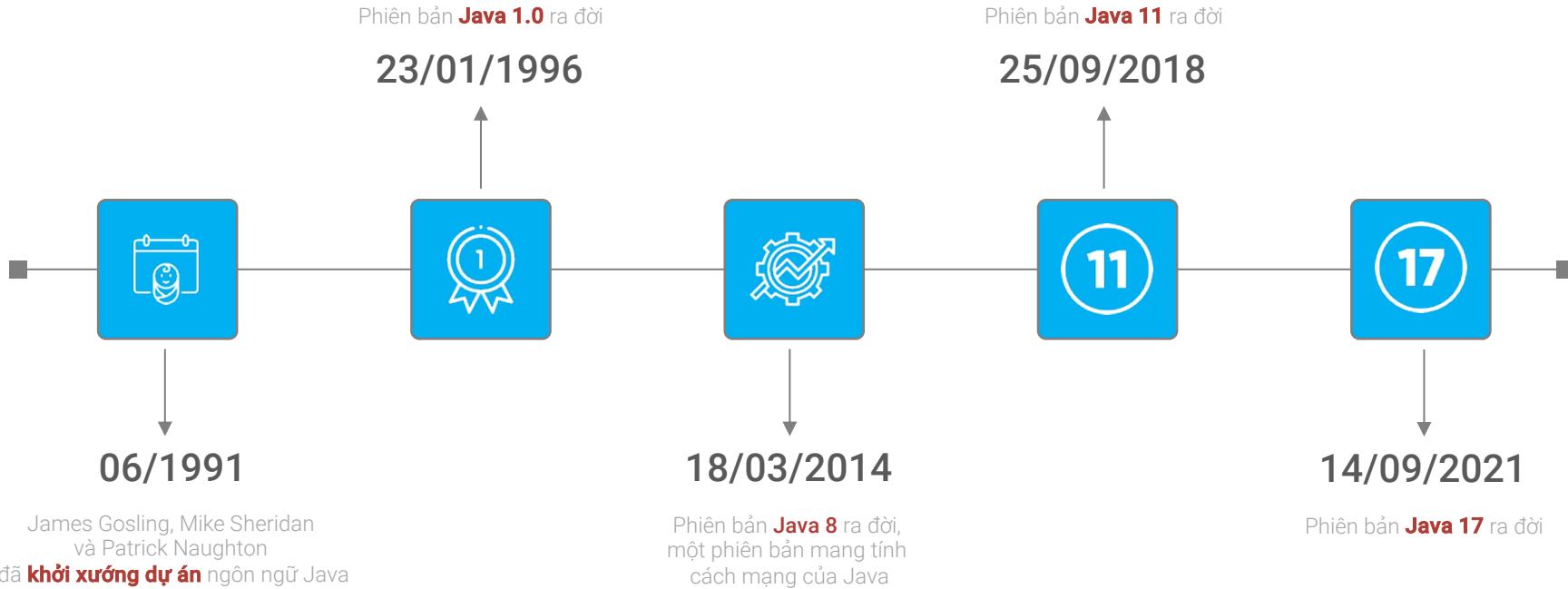
Lập trình nhúng



LỊCH SỬ



LỊCH SỬ CỦA NGÔN NGỮ JAVA



CÁC PHIÊN BẢN CỦA JAVA



- 1 Ngày 18/03/2014, Oracle đã phát hành phiên bản Java 8. Đây là một phiên bản mang tính cách mạng của Java cho nền tảng phát triển phần mềm khi so với các phiên bản trước đó. Các tính năng nổi bật như lambda, functional interface, Stream API, ...



- 2 Sau 6 tháng kể từ ngày Java 10 ra mắt, vào ngày 25/09/2018, Java 11 đã chào đón thế giới mới một loạt các thay đổi tuy không mang tính chuyển biến rõ rệt như Java 8, nhưng nó cũng chứa nhiều tính năng thú vị và đặc biệt nó là một bản LTS tiếp theo từ phiên bản thứ 8 trước đó.

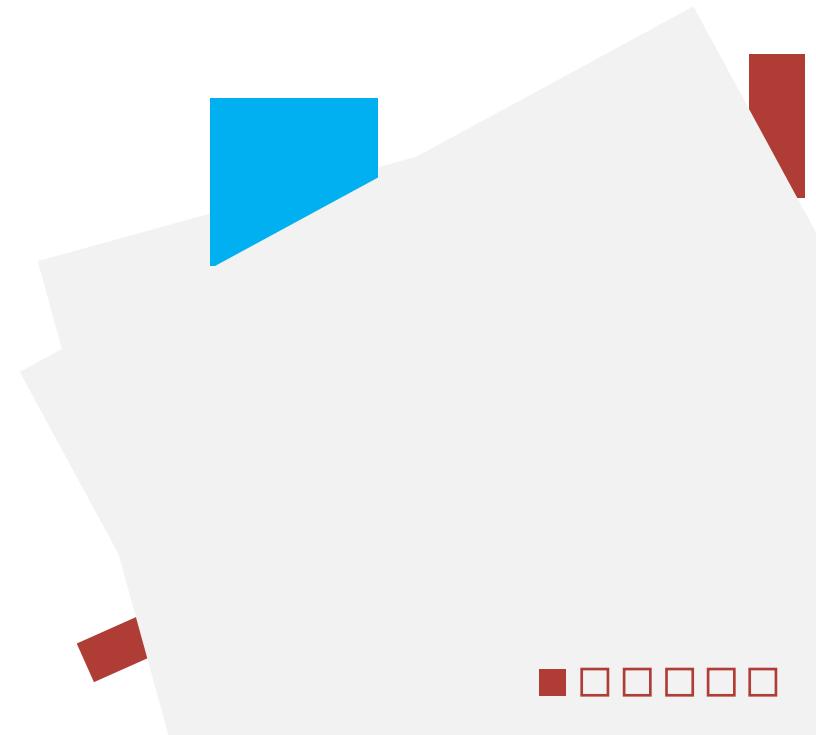


- 3 Java 17 được ra mắt sau 4 năm của "cuộc cách mạng" Java 8. Phiên bản này là một phiên bản LTS tiếp theo, đánh dấu sự "tiến hóa" của ngôn ngữ Java đi kèm với cả tiến mạnh mẽ về chuỗi, class, ...



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) 1	March 2014
Java SE 9	September 2017
Java SE 10	March 2018
Java SE 11 (LTS) 2	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) 3	September 2021





CÁC ĐẶC TÍNH CỦA JAVA



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, ta 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

Write Once, Run Anywhere - Viết một lần, chạy mọi nơi



Đa nhiệm

Java hỗ trợ đa nhiệm, đa luồng cho phép nhiều tiến trình, tiến 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



CÁC NỀN TẢNG JAVA



Java Standard Edition (Java SE - JSE)



Java Enterprise Edition (Java EE - JEE)



Java Micro Edition (Java ME - JME)

FUNDAMENTAL

IDE, JDK, JVM VÀ JRE

CÁC KHÁI NIỆM NỀN TẢNG

01

IDE



MÔI TRƯỜNG PHÁT TRIỂN TÍCH HỢP (IDE)



Môi trường phát triển tích hợp (Integrated development environment) 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, Netbeans, Eclipse, Visual Studio Code (text editor), Apache Ant, ...

TẢI VÀ CÀI ĐẶT IDE



IntelliJ IDEA



Netbeans



Eclipse



VS code



02

JDK, JRE VÀ JVM

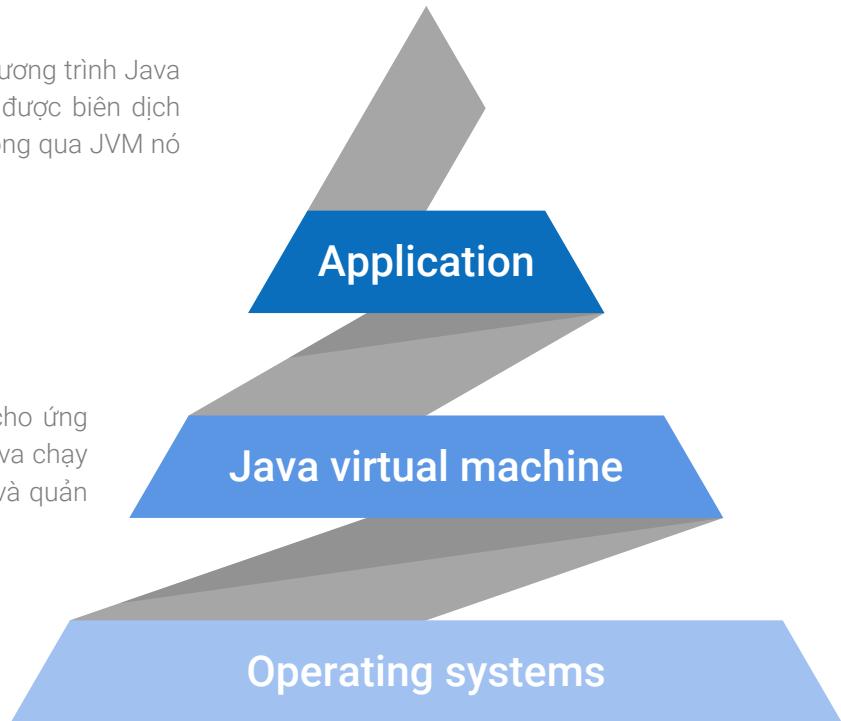
ĐỊNH NGHĨA VỀ JVM



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 nhung 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



JAVA RUNTIME ENVIRONMENT (JRE) & JAVA DEVELOPMENT KIT (JDK)

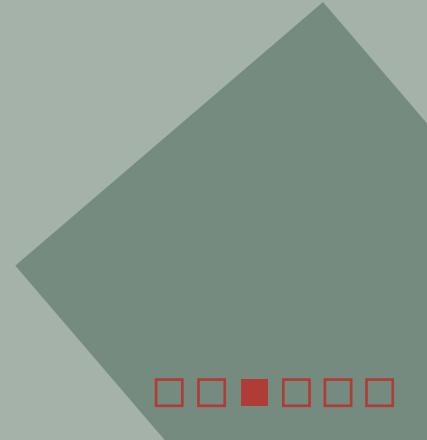


JRE (Java Runtime Environment) khởi tạo JVM và đảm bảo các phụ thuộc có sẵn cho chương trình. JRE chứa các thư viện Java, trình tải các class Java và JVM. Nó chịu trách nhiệm tải (load) chính xác các lớp và kết nối chúng với các thư viện Java cốt lõi.

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.



FIRST CHƯƠNG TRÌNH JAVA ĐẦU TIÊN PROGRAM



CHƯƠNG TRÌNH JAVA ĐẦU TIÊN

Từ khóa "class"

```
1 public class FirstJavaProgram {  
2  
3     public static void main(String[] args) {  
4  
5         System.out.println("Hello World");  
6  
7     }  
8  
9 }
```

Câu lệnh in
thông điệp mong muốn
ra màn hình

Cú pháp khai báo hàm main

CHƯƠNG TRÌNH JAVA ĐẦU TIÊN



Viết chương trình in ra dòng chữ:

I'm <your name>.

Với <your name> là tên của học viên.



JAVA BASIC CÁC KHÁI NIỆM CƠ BẢN CỦA MỘT CHƯƠNG TRÌNH JAVA CONCEPTS

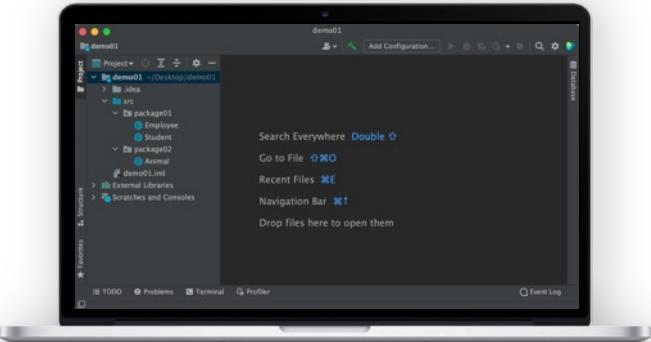


PROJECT, PACKAGE, CLASS

 **Project** (trong lập trình nói chung và lập trình Java nói riêng) là một dự án được sinh ra để xây dựng một hệ thống phần mềm sử dụng các ngôn ngữ lập trình (cụ thể trong trường hợp đang xét là ngôn ngữ Java). Một project bao gồm tập hợp các tệp tin (file), các thư mục (folder) một cách có cấu trúc để xử lý được các logic tính toán thông qua các thuật toán và tập lệnh.

 **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**



METHOD – MAIN METHOD



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 yêu cầu.

IMPORT - COMMENT

Import và comment trong Java giúp lập trình viên xử lý công việc một cách nhanh chóng hơn

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, ta có sẵn 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



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.



02 VARIABLES

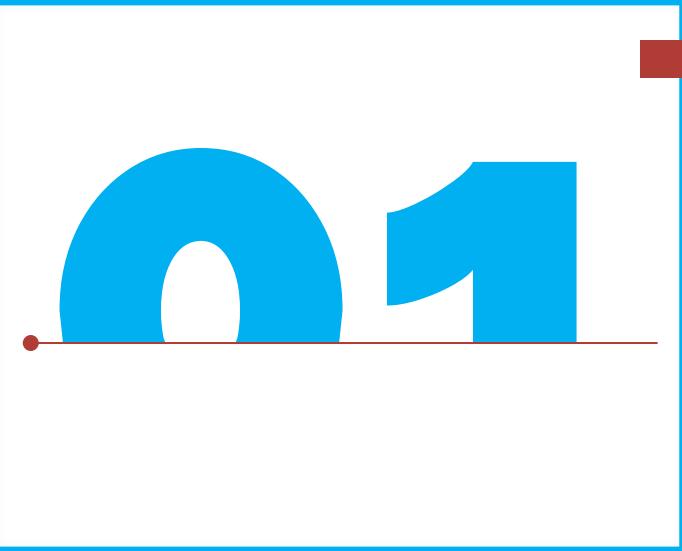


VARIABLES (BIẾN SỐ)

Variable (biến số)

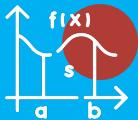
```
1 public class FirstJavaProgram {  
2  
3     public static void main(String[] args) {  
4  
5         int x = 10;  
6  
7         System.out.println(x);  
8  
9     }  
10  
11 }
```

VARIA BIẾN, KIỂU DỮ LIỆU, TOÁN TỬ TRONG NGÔN NGỮ LẬP TRÌNH JAVA BLES



BIẾN SỐ (VARIABLES)

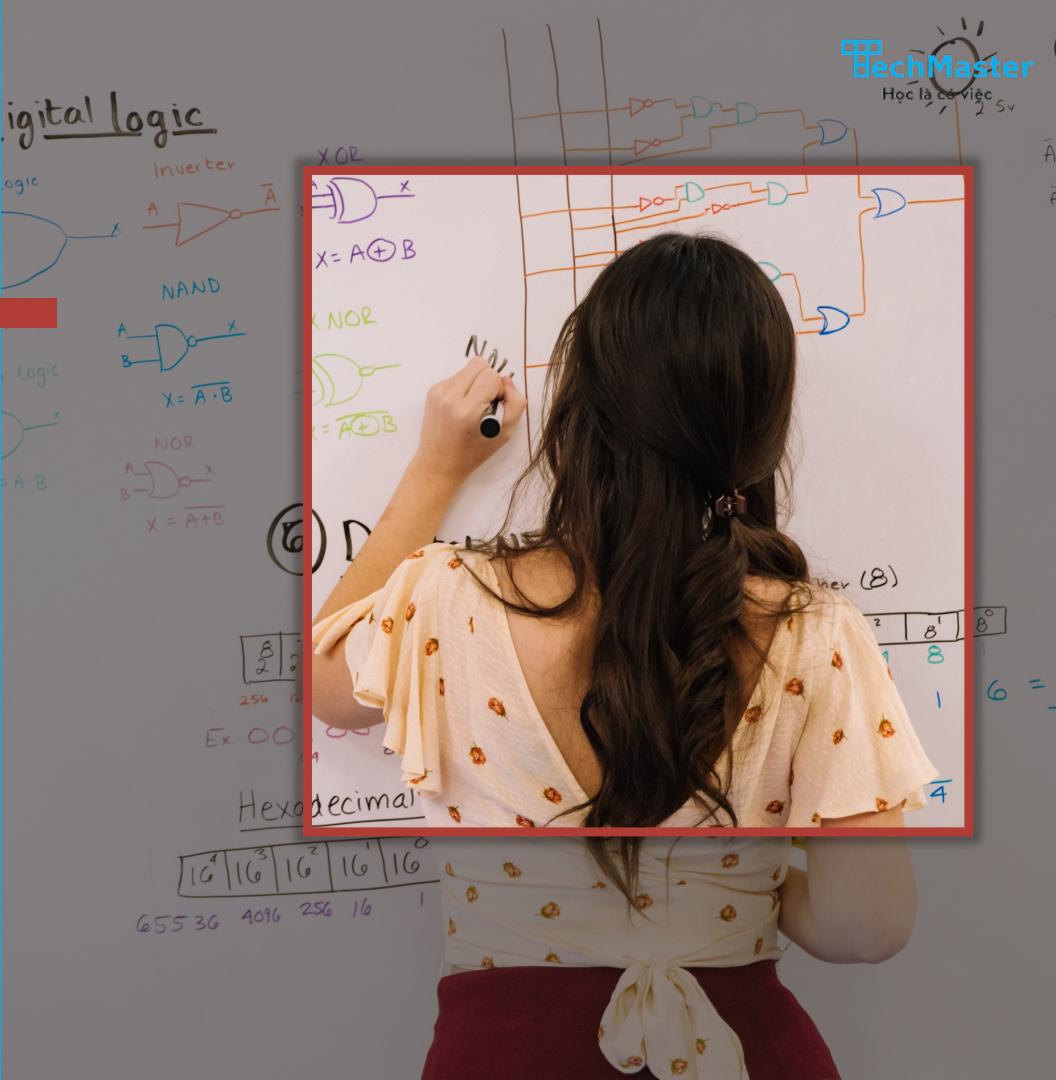
BIẾN SỐ (VARIABLES)



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.



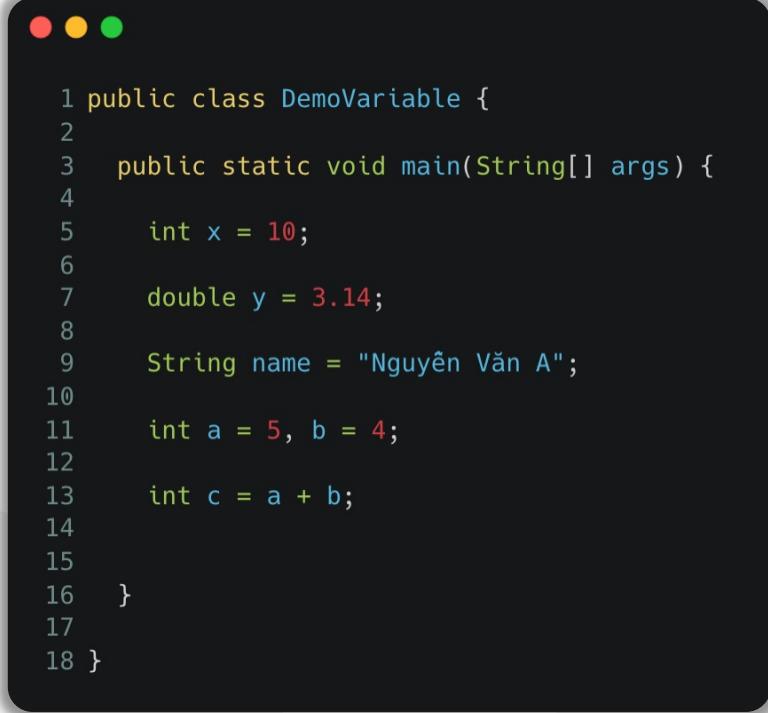
KHAI BÁO & GÁN GIÁ TRỊ

Cú pháp khai báo biến số

```
<Kiểu dữ liệu> <Tên biến số>;
```

Cú pháp gán giá trị cho biến số

```
<Tên biến số> = <Giá trị của biến số>;
```



```
1 public class DemoVariable {  
2  
3     public static void main(String[] args) {  
4  
5         int x = 10;  
6  
7         double y = 3.14;  
8  
9         String name = "Nguyễn Văn A";  
10  
11        int a = 5, b = 4;  
12  
13        int c = a + b;  
14  
15  
16    }  
17  
18 }
```

QUY TẮC KHAI BÁO BIỂN



Chỉ được bắt đầu bằng một ký tự (chữ), hoặc một dấu gạch dưới (_), hoặc một ký tự dollar (\$)



Bắt đầu từ ký tự thứ hai, có thể dùng ký tự (chữ), dấu gạch dưới (_), hoặc ký tự dollar (\$)



Tên biến không được chứa khoảng trắng và không được trùng với các từ khóa



Các biến phân biệt chữ hoa và chữ thường





Biến cục bộ (local variables)

Được khai báo trong các phương thức hoặc trong các block code (được quy định bởi dấu ngoặc nhọn {}).

Nên khởi tạo giá trị mặc định cho biến cục bộ trước khi có thể sử dụng.



Biến toàn cục (global variables)

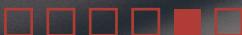
Được khai báo trong một lớp (class), bên ngoài các phương thức và các block.

Biến toàn cục có phạm vi sử dụng trong toàn bộ lớp (class) mà nó đang được khai báo. Bất kì hành động nào làm thay đổi giá trị của biến toàn cục ở bất kì vị trí nào trong class thì nó sẽ ảnh hưởng tới tất cả các vị trí khác.

PHẠM VI TRUY CẬP CỦA BIẾN SỐ

02

KIỂU DỮ LIỆU (DATA TYPES)



CÁC KIỂU DỮ LIỆU TRONG JAVA



Kiểu dữ liệu
nguyên thủy
(primitive data type)



Kiểu dữ liệu
đối tượng
(object data type)



Ví dụ
Về kiểu dữ liệu

Kiểu dữ liệu nguyên thủy thường gặp
là các số: số nguyên, số thực, ...

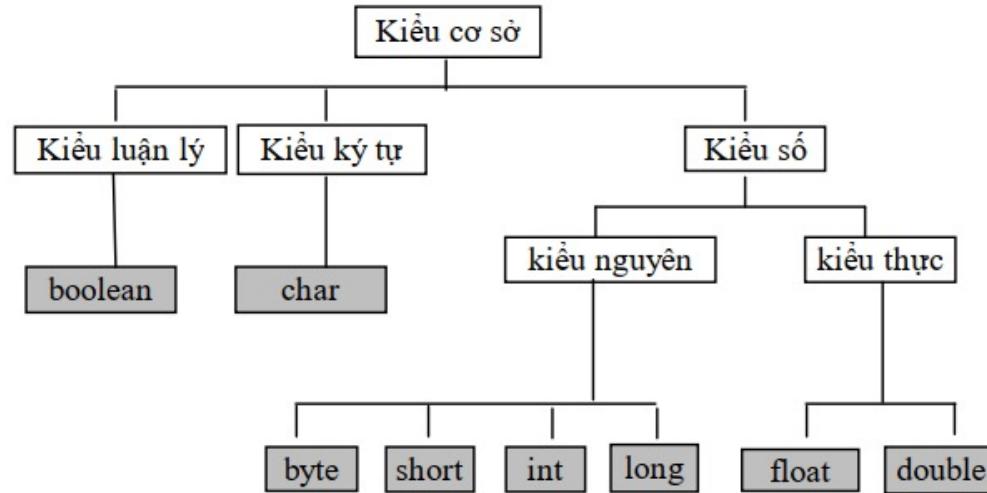
Kiểu dữ liệu đối tượng là các đối
tượng trong thực tế được tạo trong
Java

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,... sẽ được tìm hiểu kỹ
hơn trong các bài sau.

KIỂU DỮ LIỆU NGUYÊN THỦY



KIỂU SỐ NGUYÊN

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

KIỂU SỐ THỰC

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

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

Kí tự chỉ duy nhất một chữ cái hoặc chữ số hoặc ký tự đặc biệt, và được nằm trong **dấu nháy đơn**

```
1 public class DemoVariable {  
2  
3     public static void main(String[] args) {  
4  
5         char c = 'a';  
6  
7         char b = '1';  
8  
9         char c = '%';  
10  
11        char d = 65; // 65 là chữ 'a' trong bảng mã ASCII  
12  
13  
14    }  
15  
16 }
```

KIỂU LUẬN LÝ

ĐÚNG HOẶC SAI – TRUE OR FALSE

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**.



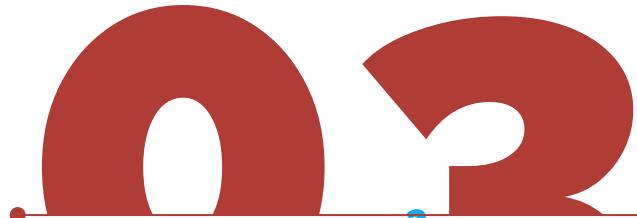
```
1 public class DemoVariable {  
2  
3     public static void main(String[] args) {  
4         boolean a = true;  
5         boolean b = false;  
6  
7         boolean c;  
8         System.out.println(c);  
9  
10    }  
11  
12 }  
13 }
```

THỰC HÀNH KIỂU DỮ LIỆU



1. Bài toán tính tổng và hiệu của hai số cùng nguyên hoặc cùng thực
2. Bài toán tính tích và thương của hai số cùng nguyên hoặc cùng thực





02

**ÉP KIỀU,
HÃNG SỐ**

ÉP KIỂU (CASTING)



Khái niệm

Ép kiểu (casting - 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

Trong quá trình xử lý dữ liệu, ta thường hay gặp rất nhiều trường hợp có sự chuyển đổi qua lại với nhau, ép kiểu là cách thức để phục vụ việc đưa kiểu dữ liệu đang sử dụng về đúng kiểu mà lập trình viên mong muốn



Phân loại

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

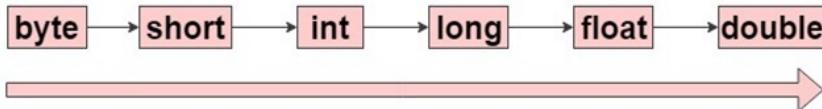


CASTING
or
DATA
PARSING



CHUYỂN ĐỔI KIỂU DỮ LIỆU NGẦM ĐỊNH

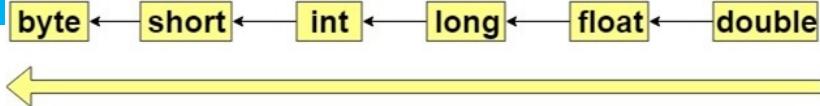
Automatic Type Conversion (Widening - implicit)



```
1 public class DemoVariable {  
2  
3     public static void main(String[] args) {  
4         int a = 5;  
5  
6         long b = a;  
7  
8         System.out.println(b);  
9  
10    }  
11  
12 }  
13 }
```

CHUYỂN ĐỔI KIỂU DỮ LIỆU TƯỜNG MINH

Narrowing (explicit)

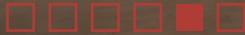


```
1 public class DemoVariable {  
2  
3     public static void main(String[] args) {  
4         long a = 5;  
5         int b = (int) a; // bảo toàn dữ liệu  
6         System.out.println(b);  
7  
8         double height = 7.52;  
9         int h = (int) a; // mất mát dữ liệu  
10        System.out.println(h);  
11  
12    }  
13  
14 }
```

THỰC HÀNH ÉP KIỂU DỮ LIỆU



1. Sử dụng kiến thức ép kiểu dữ liệu để giải bài toán tính thương của hai số nguyên với mong muốn kết quả cho ra là một số thực với độ chính xác lấy tới 2 chữ số thập phân.
2. Sử dụng kiến thức ép kiểu dữ liệu để giải bài toán tính sin và cos của 1 góc trong tam giác vuông khi biết độ dài 3 cạnh (với độ chính xác lấy tới 2 chữ số thập phân).



HẰNG SỐ



Hằng số là một giá trị không đổi trong suốt chương trình, được khởi tạo giá trị ngay từ ban đầu.

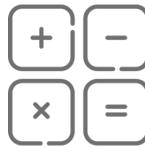
Quy ước: Tên hằng số luôn được viết in hoa và phân cách các từ bằng dấu gạch dưới (_)

```
1 public class DemoVariable {  
2  
3     public static void main(String[] args) {  
4         // final <Kiểu dữ liệu> <Tên hằng số> = <Giá trị>  
5  
6         final PI = 3.14;  
7  
8     }  
9  
10 }
```



TOÁN TỬ

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



Toán tử số học

01100
10110
11110

Toán tử trên bit



Toán tử quan hệ



Toán tử logic



Toán tử gán



Toán tử ba
ngôi



Giả sử với $a = 30$ và $b = 10$

TOÁN TỬ SỐ HỌC

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



Giả sử với $a = 30$ (00011110) và $b = 10$ (00001010)

TOÁN TỬ TRÊN BIT

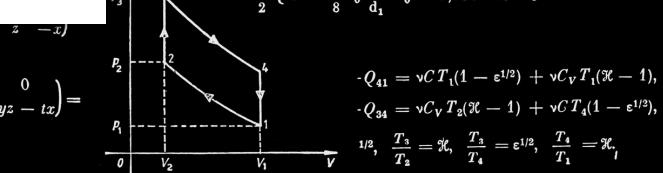
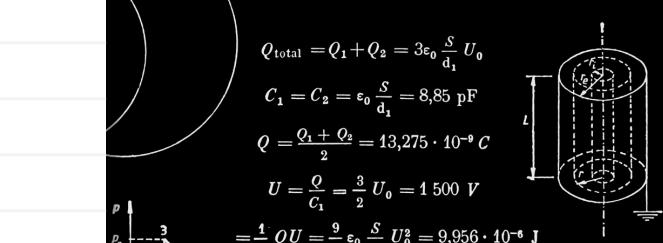
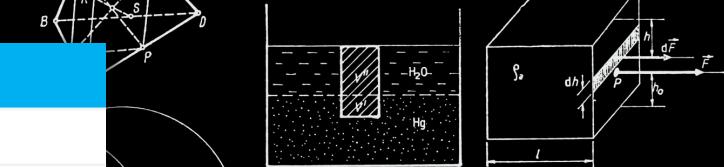
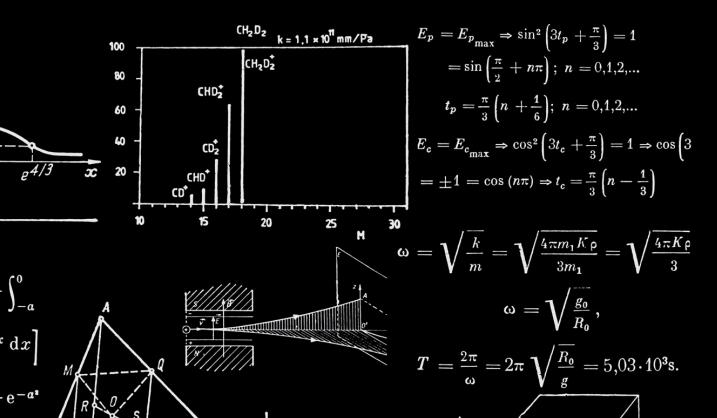
Toán tử	Ý nghĩa	Ví dụ
&	AND	$a \& b \Rightarrow 10$ (00001010)
	OR	$a b \Rightarrow 30$ (00010100)
^	XOR	$a ^ b \Rightarrow 20$ (00010100)
<<	Dịch trái	$a << 2 \Rightarrow 120$ (01111000)
>>	Dịch phải	$a >> 2 \Rightarrow 7$ (111)
>>>	Dịch phải và điền 0 vào bit trống	$a >>> 2 \Rightarrow 7$ (000000111)
~	Bù bit	$\sim a \Rightarrow -31$

TOÁN TỬ QUAN HỆ

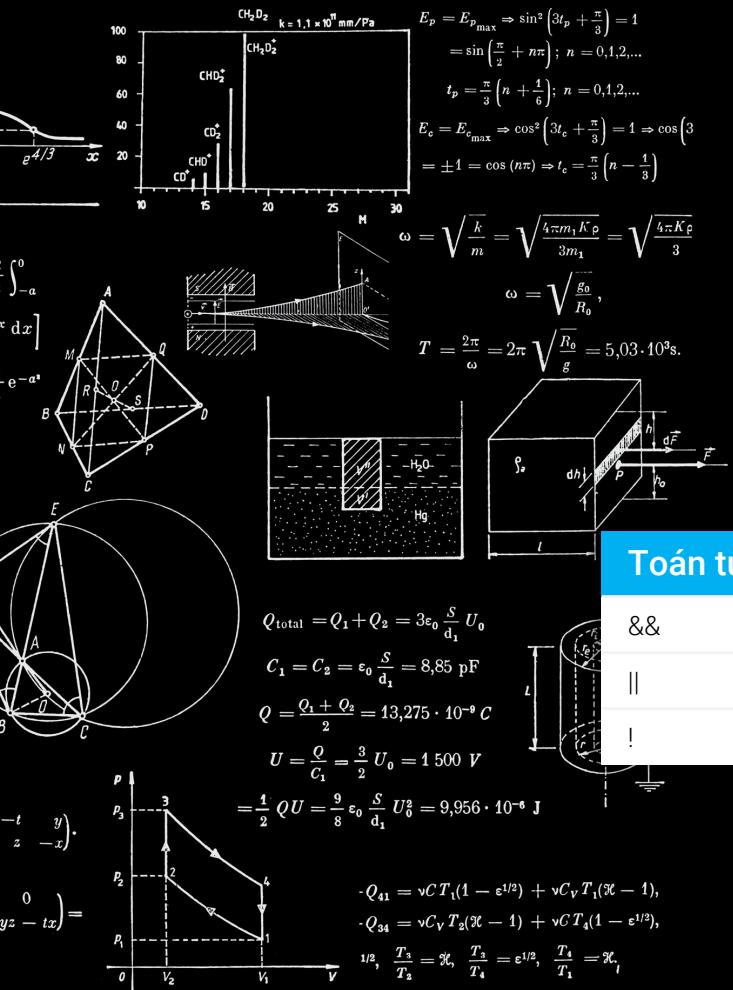


Giả sử với $a = 30$ và $b = 10$

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



TOÁN TỬ LOGIC



Giả sử với $c = \text{true}$ và $d = \text{false}$

Toán tử

&&

||

!

Ý nghĩa

Toán tử và

Toán tử hoặc

Toán tử phủ định

Ví dụ

$c \&& d \Rightarrow \text{false}$

$c || d \Rightarrow \text{true}$

$!c \Rightarrow c = \text{false}$

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	$c -= a \Rightarrow a = c - a$
*=	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	$c ^= a \Rightarrow c = c ^ a$
=	Phép OR bit	$c = a \Rightarrow c = c a$

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>;
```

```
1 public class DemoOperator {  
2  
3     public static void main(String[] args) {  
4         int a = 4;  
5         int b = 2;  
6  
7         String result = (a % b == 0) ? "a chia hết cho b" : "a không chia hết cho b";  
8  
9         System.out.println(result);  
10    }  
11  
12 }  
13 }
```

THỰC HÀNH SỬ DỤNG TOÁN TỬ



1. Sử dụng kiến thức toán tử để giải phương trình bậc nhất một ẩn $ax + b = 0$ (với a khác 0)

2. Sử dụng kiến thức toán tử để giải phương trình bậc hai một ẩn $ax^2 + bx + c = 0$ (với a khác 0 và giả sử phương trình luôn có 2 nghiệm phân biệt)

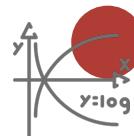


MATH LỚP MATH TRONG JAVA GIẢM NHẸ CÔNG SỨC TÍNH TOÁN METHODS





ĐỊNH NGHĨA LỚP MATH

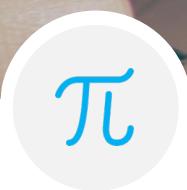


JDK định nghĩa sẵn một số lớp tiện dụng, một trong số đó là lớp Math cung cấp các hàm về toán học.



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), vì vậy 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 TRONG JAVA



Hằng số



Lượng giác



Làm tròn



Tính toán



Tìm kiếm số



Sinh số
ngẫu nhiên

số PI, số siêu việt, ...

sin, cos, tan, cotan,
toDegrees,
toRadians, ...

ceil, floor, round, ...

sqrt, pow, log, ...

max, min, ...

random

THỰC HÀNH SỬ DỤNG LỚP MATH



1. Sử dụng kiến thức của lớp Math trong Java để tính lũy thừa mũ n của cơ số a.
2. Sử dụng kiến thức của lớp Math trong Java để tìm số lớn nhất trong 3 số nguyên.
3. Sử dụng kiến thức của lớp Math trong Java để tính thương của 2 số nguyên, kết quả làm tròn tới 2 chữ số thập phân.

