Session 01:

Tổng quan về Java, biến, kiểu dữ liệu

NỘI DUNG

1. Tổng quan về ngôn ngữ lập trình Java
2. Khái niệm JDK, JRE, JVM
3. Biến, vùng nhớ Stack, Heap
4. Các kiểu dữ liệu trong Java
5. Các toán tử số học, so sánh, logic
6. Thao tác nhập xuất trong Java

1. Tổng quan về ngôn ngữ lập trình Java

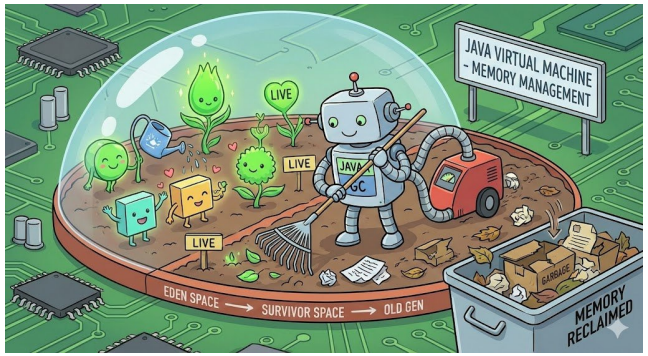
- Ngôn ngữ lập trình hướng đối tượng mã nguồn mở
- Khởi đầu bởi **James Gosling** và đồng nghiệp ở Sun MicroSystem năm 1991, phát hành năm 1994, đến năm 2010 được Oracle mua lại
- Độc lập nền tảng - chạy trên mọi nền tảng (**platform**) khác nhau

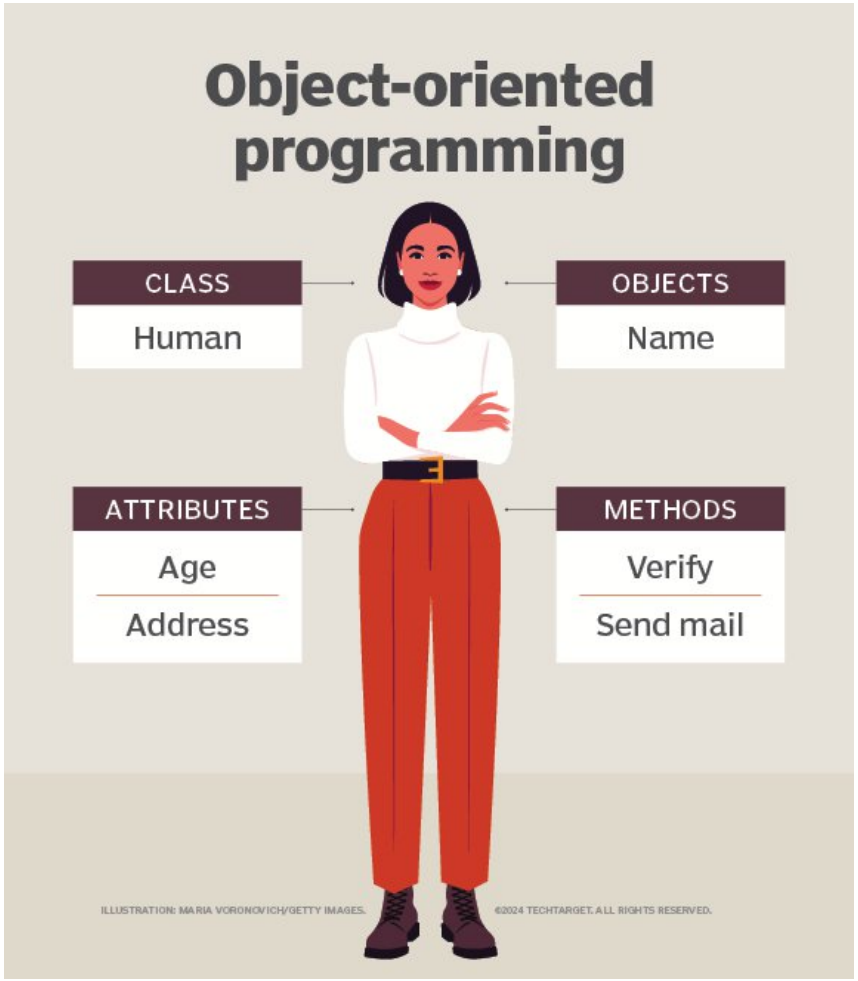
"Write Once, Run Anywhere"

1. Tổng quan về ngôn ngữ lập trình Java

Đặc điểm:

- Hướng đối tượng
- Độc lập phần cứng và hệ điều hành
- Ngôn ngữ vừa biên dịch vừa thông dịch
- Cơ chế thu gom rác tự động
- Đa luồng
- Bảo mật cao

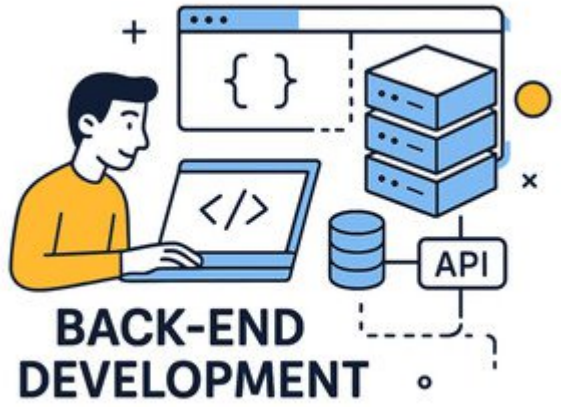


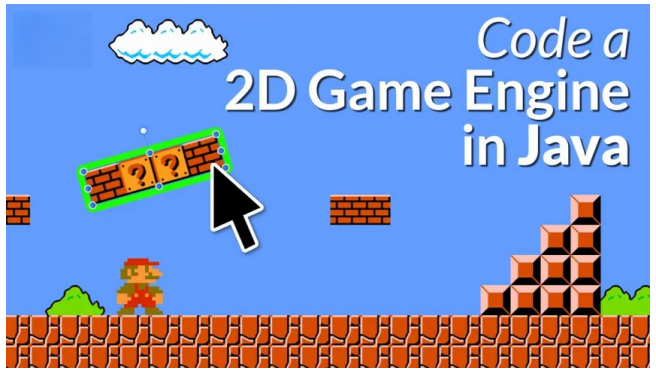


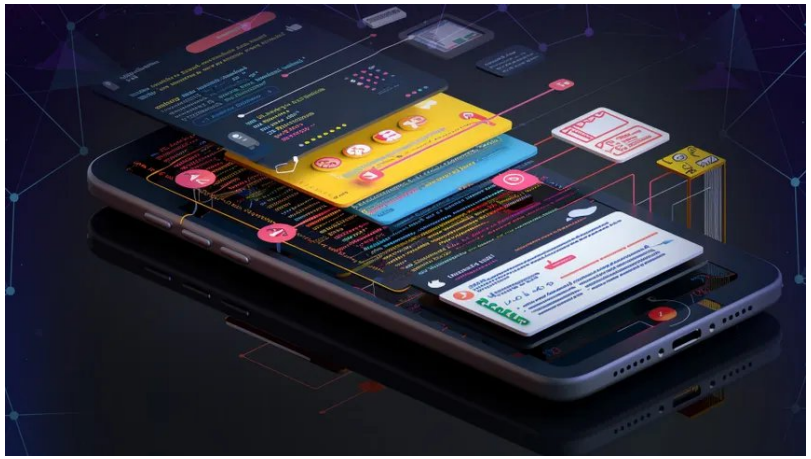
1. Tổng quan về ngôn ngữ lập trình Java

Java được dùng để làm:

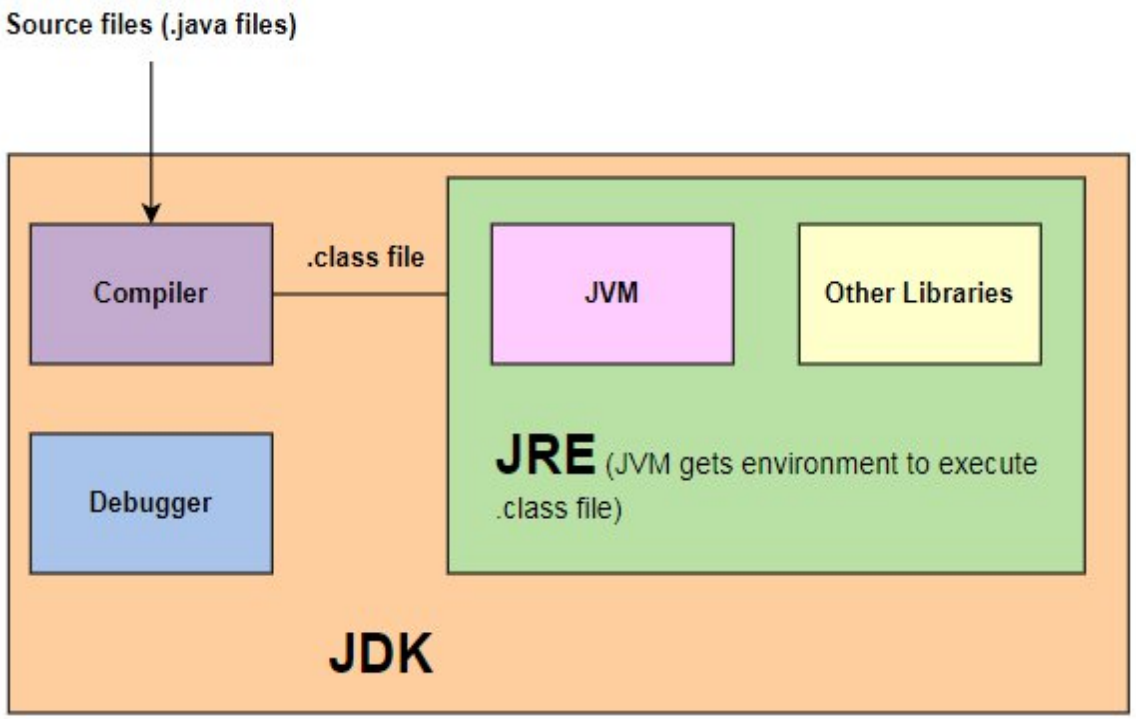
- Phát triển ứng dụng Desktop
- Phát triển Website và Backend
- Phát triển ứng dụng Android
- Phát triển phần mềm doanh nghiệp (Enterprise)
- Phát triển Game và ứng dụng nhúng
- Học lập trình và tư duy thuật toán







2. Khái niệm JDK, JRE, JVM



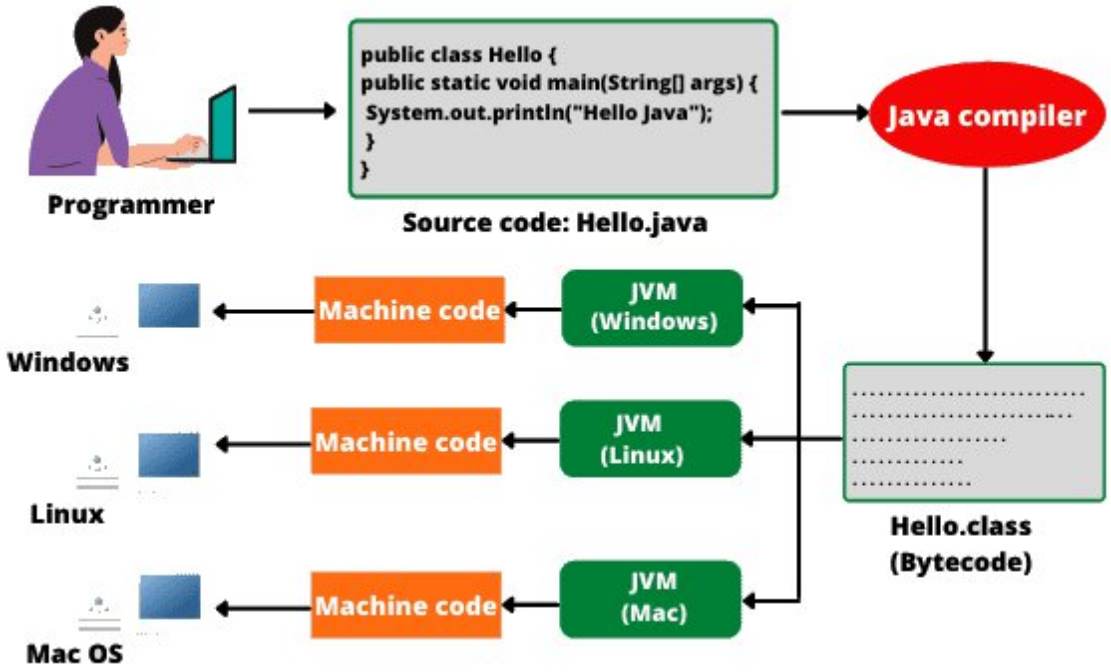
JDK **Java Development Kit**

JRE **Java Runtime Environment**

JVM **Java Virtual Machine**

2. Khái niệm JDK, JRE, JVM

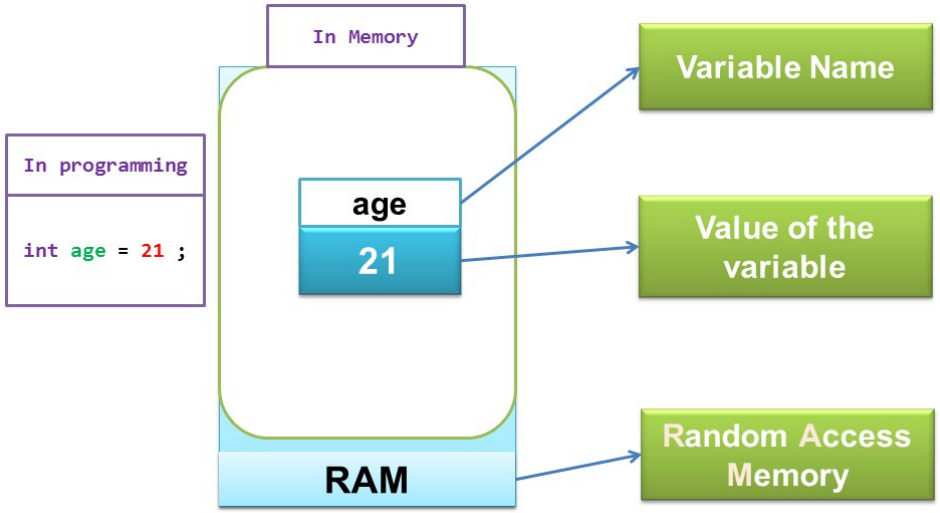
- **JDK -** Bộ công cụ phát triển Java
  - **Compiler**
  - **Debugger**
- **JRE -** Môi trường chạy các ứng dụng Java
  - **JIT**
  - **JVM**
  - **Class Loader**
  - **Java Libraries**
- **JVM -** Máy ảo Java thực thi các mã bytecode
  - **Class Loader**
  - **Execution Engine**
  - **Garbage Collector**



3. Biến, vùng nhớ Stack, Heap

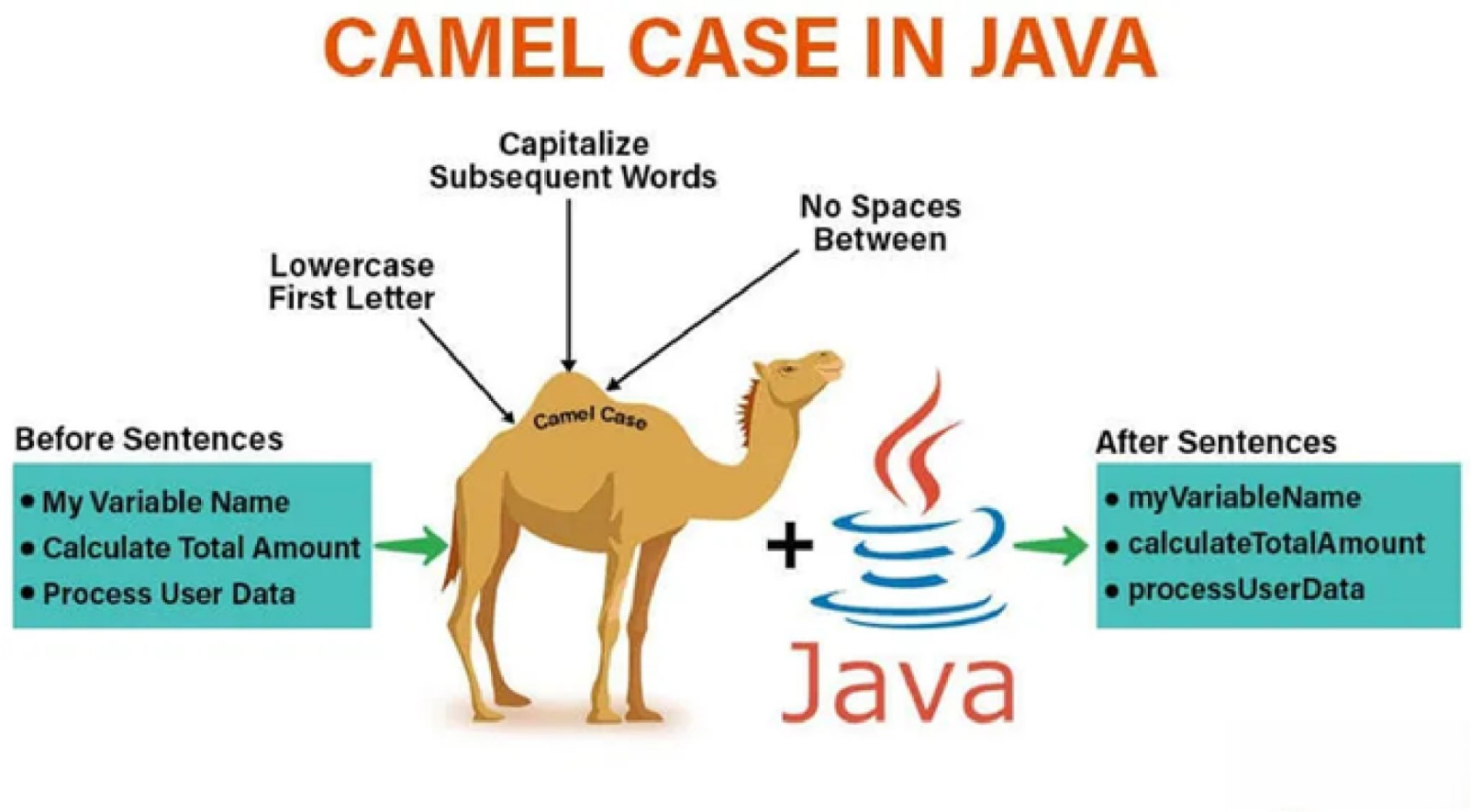
Biến: là một vùng nhớ được dùng để lưu trữ dữ liệu trong quá trình thực thi chương trình

Mỗi biến có một tên (**identifier**) và một kiểu dữ liệu (**data type**) gắn liền với nó



3. Biến, vùng nhớ Stack, Heap

Quy tắc đặt tên



3. Biến, vùng nhớ Stack, Heap

Quy tắc đặt tên

3. Biến, vùng nhớ Stack, Heap

Quy tắc đặt tên

3. Biến, vùng nhớ Stack, Heap

- Biến có giá trị không thay đổi trong quá trình chạy chương trình
- Cú pháp:
  - ***final*** data_type CONSTANT_NAME = constant_value;
- Trong đó:
  - **data_type:** kiểu dữ liệu của biến hằng số
  - **CONSTANT_NAME:** Tên biến hằng số
  - **constant_value:** Giá trị của biến hằng số
- Quy tắc định danh hằng số: các ký tự phải được viết hoa

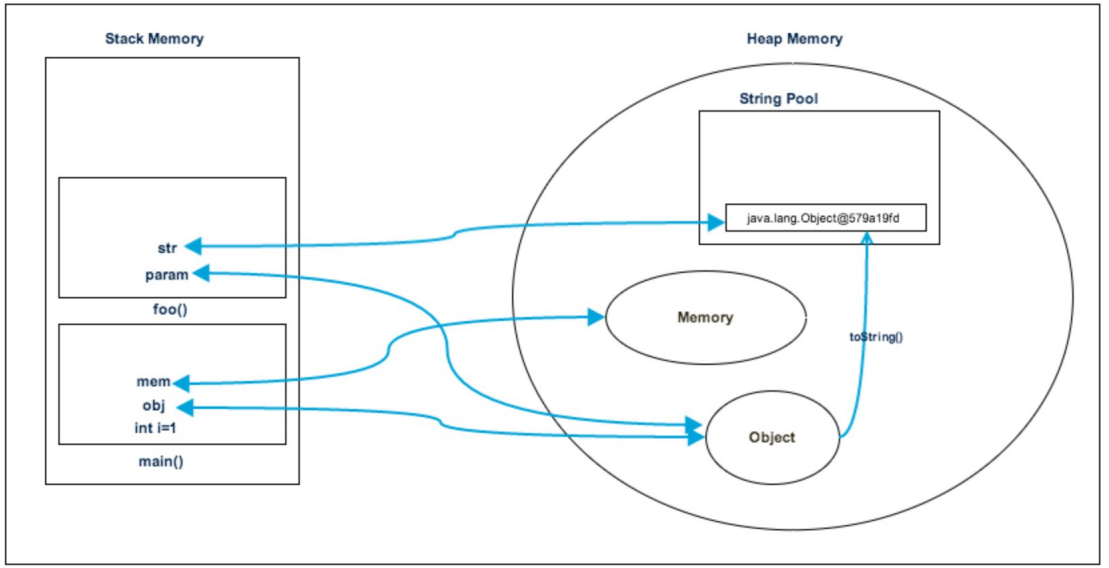
3. Biến, vùng nhớ Stack, Heap

Stack trong Java

- Lưu **biến cục bộ** và **tham số phương thức**
- Quản lý theo **LIFO** (vào trước ra sau)
- Vào method → cấp phát
- Thoát method → giải phóng
- Truy cập **nhanh**

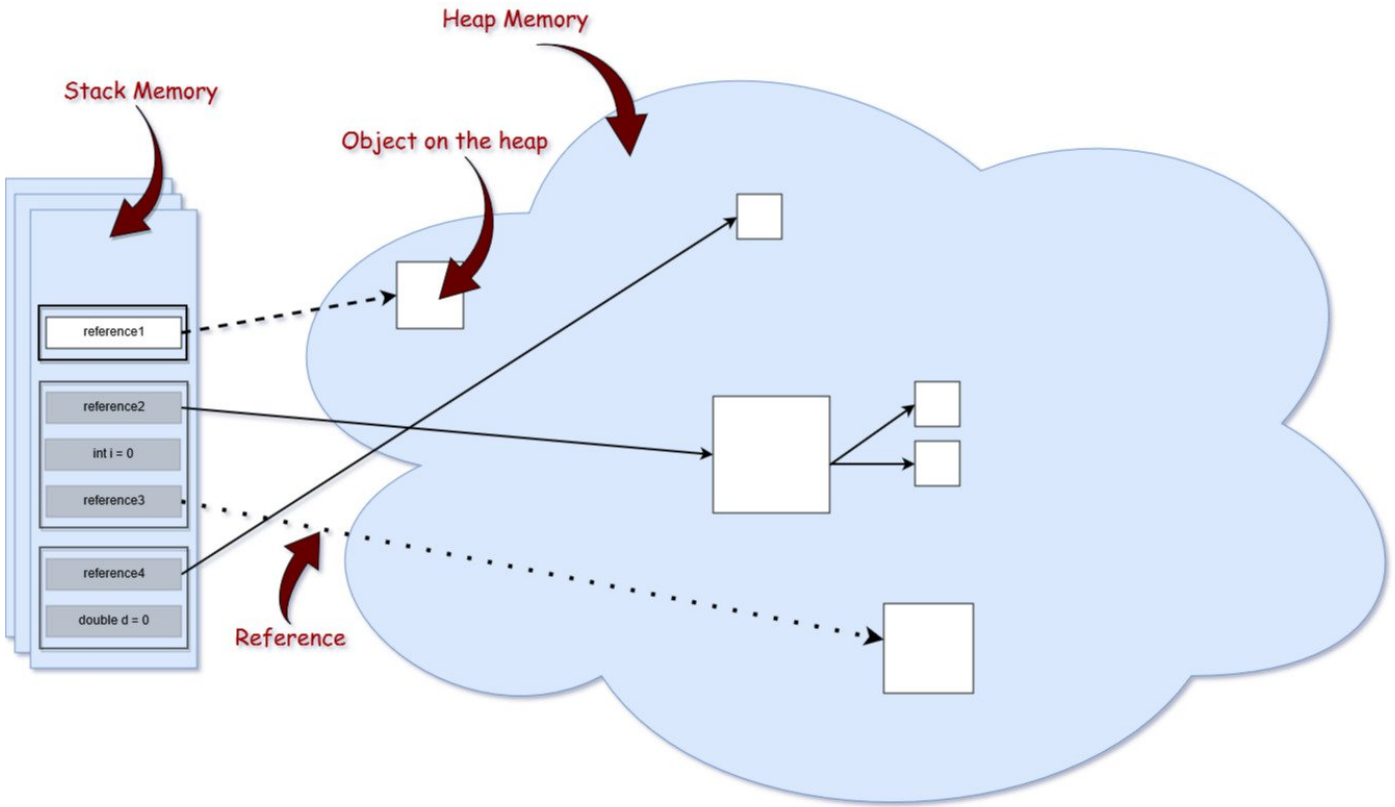
Heap trong Java

- Lưu **đối tượng (object)** và **mảng**
- Cấp phát khi dùng **new**
- Quản lý bởi **Garbage Collector**
- Kích thước lớn hơn Stack



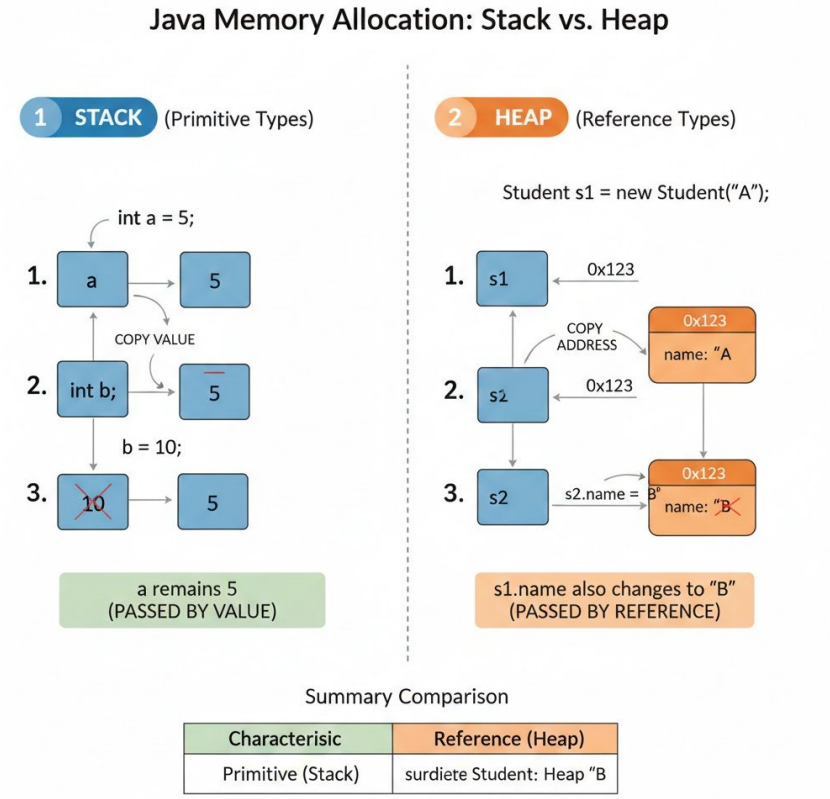
3. Biến, vùng nhớ Stack, Heap

Java Runtime Memory

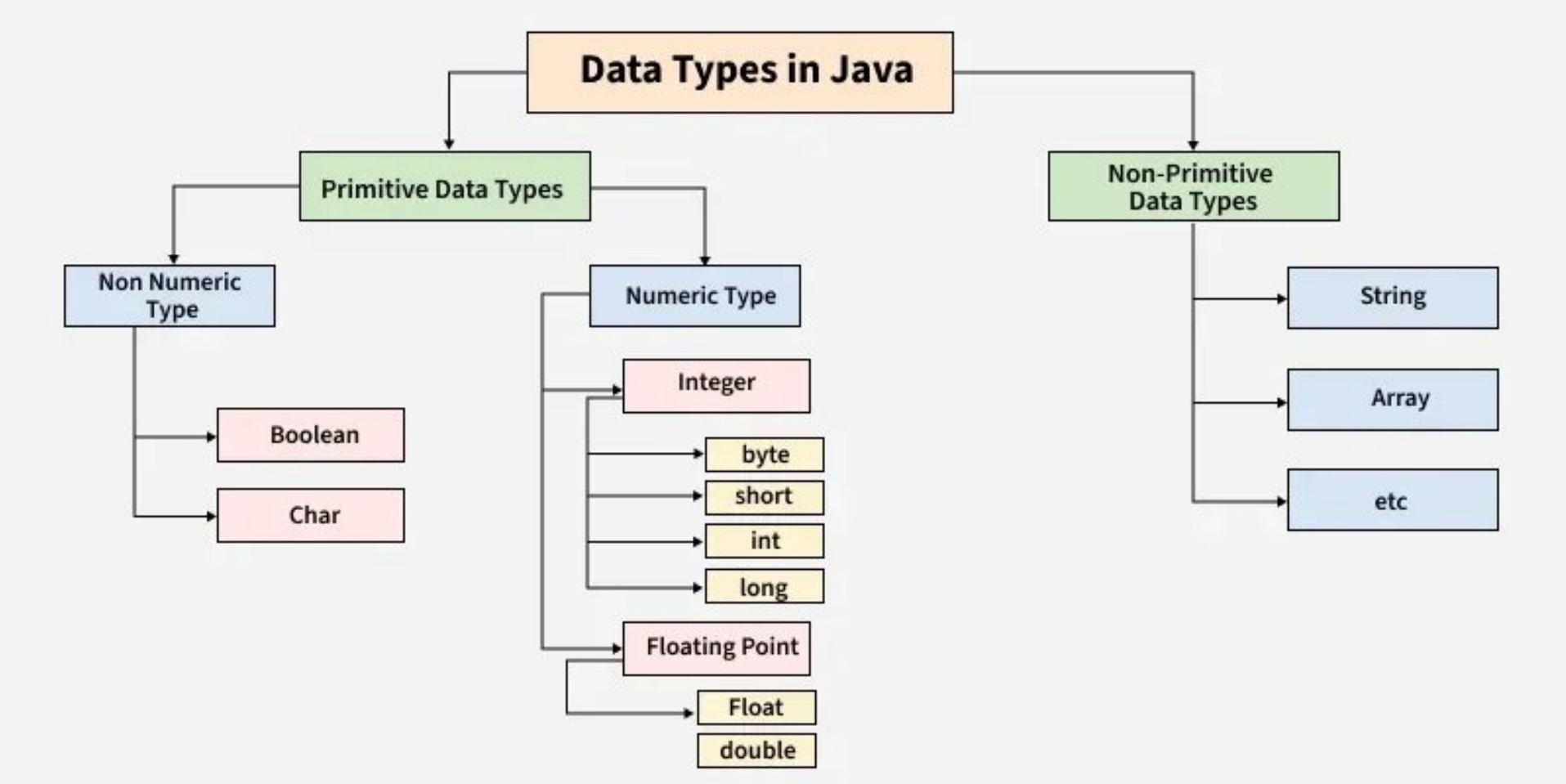


3. Biến, vùng nhớ Stack, Heap

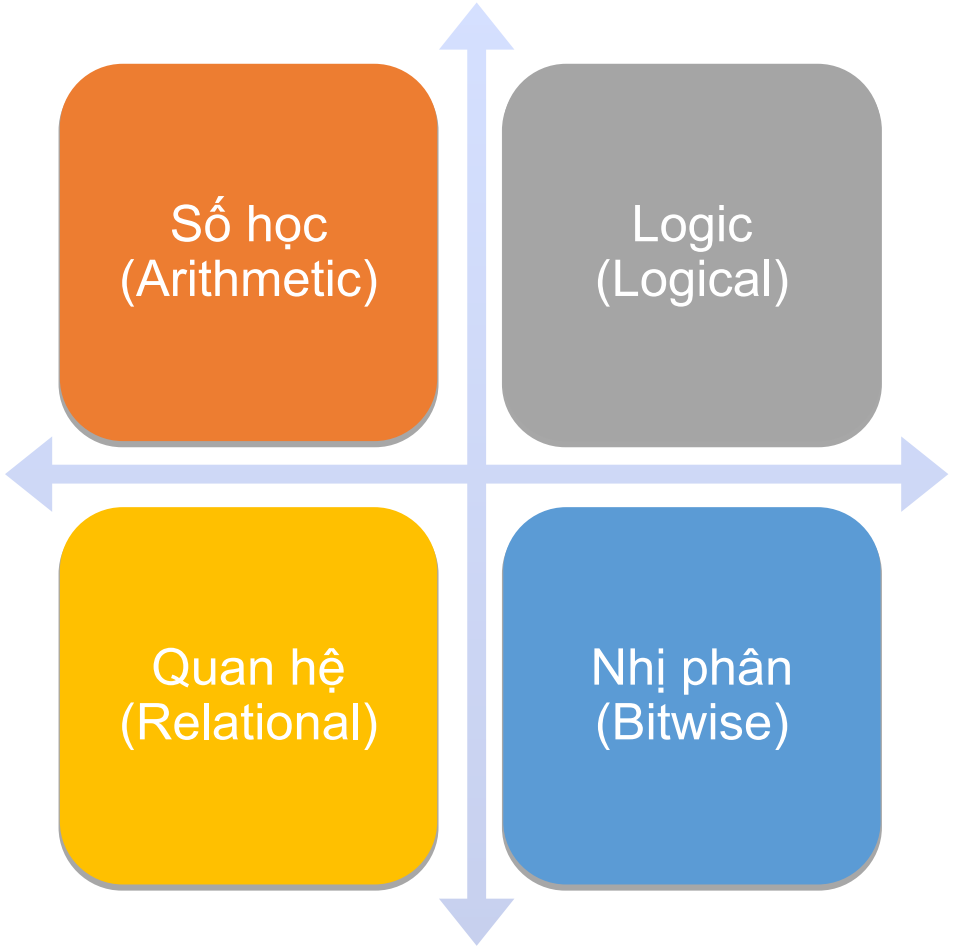
```java
int a = 5;
int b = a;
b = 10;
// a không đổi → tham trị

Student s1 = new Student("A");
Student s2 = s1;
s2.name = "B";
// s1.name cũng đổi → tham chiếu
```



4. Các kiểu dữ liệu trong Java



5. Các toán tử số học, so sánh, logic



CÁC LOẠI TOÁN TỬ

5. Các toán tử số học, so sánh, logic

TOÁN TỬ SỐ HỌC

Toán tử	Ý nghĩa	Ví dụ	Kết quả
+	Toán tử cộng	10 + 5	15
-	Toán tử trừ	10 – 5	5
*	Toán tử nhân	10 * 5	50
/	Toán tử chia	10 / 5	2
%	Toán tử chia lấy dư	10%5	0
++	Tăng 1 đơn vị	10++	11
--	Giảm 1 đơn vị	10--	9

5. Các toán tử số học, so sánh, logic

TOÁN TỬ QUAN HỆ

Toán tử	Ý nghĩa	Ví dụ	Kết quả
>	So sánh lớn hơn	10 > 5	true
>=	So sánh lớn hơn hoặc bằng	10 >= 10	true
<	So sánh nhỏ hơn	10 < 10	false
<=	So sánh nhỏ hơn hoặc bằng	10 <= 10	true
==	So sánh bằng	10 == 5	false
!=	So sánh khác	10 != 5	true

5. Các toán tử số học, so sánh, logic

TOÁN TỬ LOGIC

Toán tử	Ý nghĩa	Ví dụ	Kết quả
&&	Toán tử logic VÀ	true && true	true
||	Toán tử logic HOẶC	10 >= 10	true
!	Toán tử logic phủ định	10 < 10	false

5. Các toán tử số học, so sánh, logic

TOÁN TỬ BITWISE

Toán tử	Ý nghĩa	Ví dụ	Kết quả
Bitwise AND (x & y)	Kết quả là 1 nếu 2 toán hạng là 1	1101 & 0110	0100
Bitwise OR (x | y)	Kết quả là 1 nếu 1 trong 2 toán hạng là 1	1100 | 0110	1110
Bitwise NOT (~x)	Đảo ngược giá trị của toán hạng	~1010	0101
Bitwise XOR(x ^ y)	Kết quả là 1 khi chỉ 1 trong 2 toán hạng là 1	1101 ^ 0110	10111

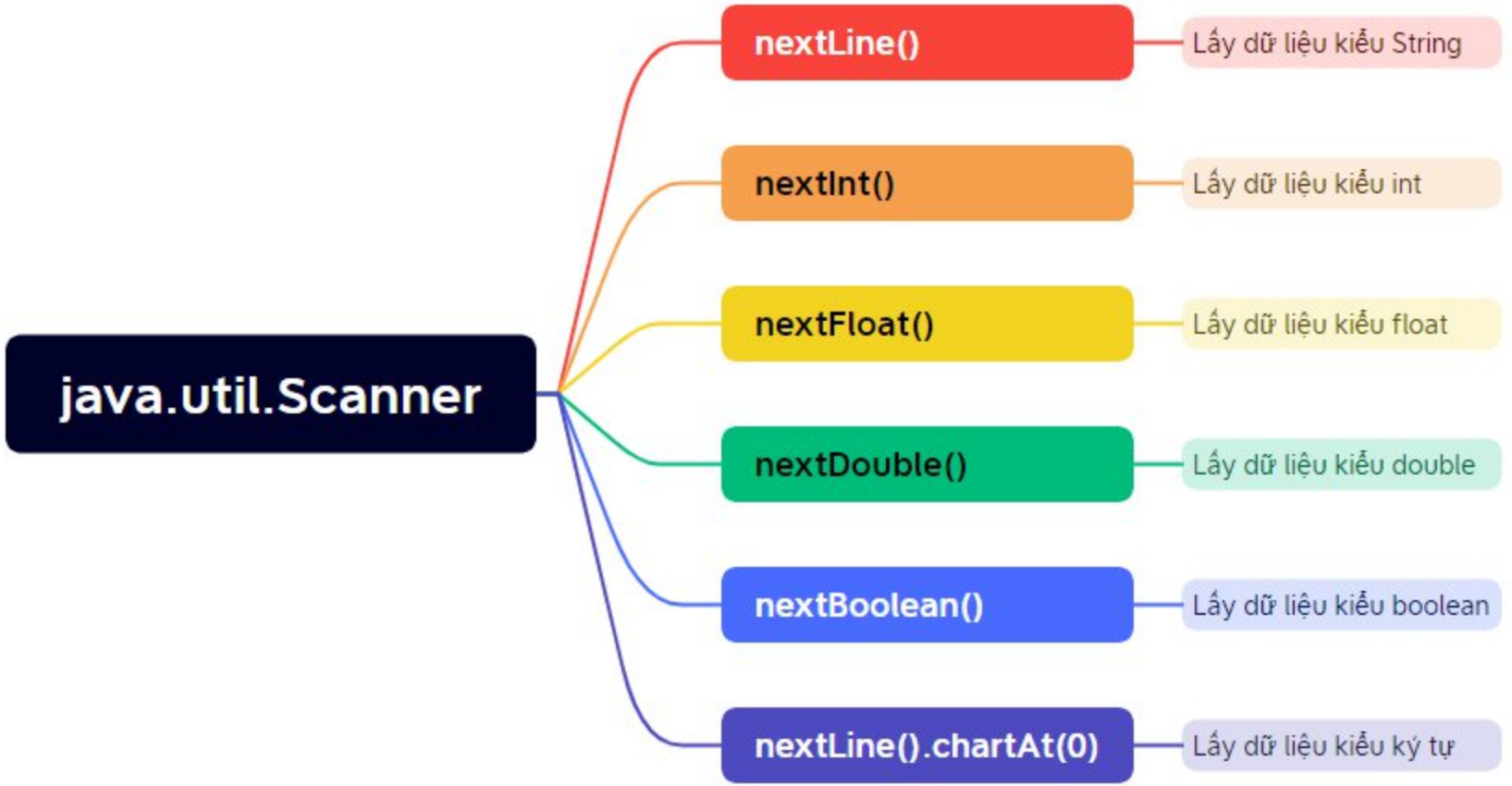
6. Thao tác nhập xuất trong Java

XUẤT DỮ LIỆU

- Cú pháp câu lệnh in dữ liệu ra màn hình console, in xong xuống dòng
  - **System.out.println(“String”)**
- Cú pháp câu lệnh in dữ liệu ra màn hình console, in xong không xuống dòng
  - **System.out.print(“String”)**
- Cú pháp câu lệnh in dữ liệu có định dạng ra màn hình console
  - **System.out.printf(“Control String”, value1, value2,….,valueN)**
- Định dạng dữ liệu
  - int - %d
  - float, double - %f
  - String - %s
  - char - %c
  - boolean - %b

6. Thao tác nhập xuất trong Java

NHẬP DỮ LIỆU



6. Thao tác nhập xuất trong Java

CÁC BƯỚC THỰC HIỆN NHẬP DỮ LIỆU

TỔNG KẾT

- ❑ Hiểu được sơ lược về ngôn ngữ lập trình JAVA
- ❑ Hiểu về khái niệm JDK, JRE và JVM
- ❑ Nắm được các khái niệm về biến
- ❑ Nắm được các khái niệm vùng nhớ
- ❑ Nắm được các kiến thức về các kiểu dữ liệu
- ❑ Nắm được các thao tác với các toán tử
- ❑ Nắm thao tác nhập xuất dữ liệu cơ bản

RIKEI Academy®
where the dream begins

KẾT THÚC

HỌC VIỆN ĐÀO TẠO LẬP TRÌNH CHẤT LƯỢNG NHẬT BẢN