# JAVA CORE

Link học java core:

* <https://v1study.com/java-methods-of-the-patternsyntaxexception-class.html>
* <https://www.phongvanit.com/ky-nang/java-1001>

## Quản lý bộ nhớ trong java

<https://gochocit.com/ngon-ngu-lap-trinh/quan-ly-bo-nho-trong-java-bo-nho-stack-va-bo-nho-heap#1-1-kien-truc-bo-nho-trong-java>

## Khái niệm Static

<https://topdev.vn/blog/tu-khoa-static-va-final-trong-java/>

* Static được cấp phát bộ nhớ cố định trong class (method, class hoặc biến), lưu ở heap
* Từ khoá static biểu thị cho biến hoặc phương thức có thể được truy cập (sử dụng) mà không cần tạo ra thực thể của lớp chứa nó. Người dùng không thể override phương thức static trong Java, bởi vì kỹ thuật đè (overriding) phương thức được dựa trên quá trình gán (binding) động khi runtime (khi chương trình đang chạy) và những phương thức static  được gán tĩnh trong thời gian biên dịch. Phương thức tĩnh không ràng buộc với thực thể của lớp nên phương thức tĩnh sẽ không thể override.
* Static thì ko sử dụng được từ khóa this, super;
* Không thể truy cập biến non-static trong 1 trong nó, vì static được cấp phát cố định mà các instance Object( chứa static cũng đc) lại không được cấp phát cố định bộ nhớ.

## Kiểu Wrapper {Integer,String,Long,Byte,..}

Nó là final class { m ko thể tạo 1 lớp con kế thừa từ nó, nó là lớp ko có con}

## Từ khóa fetch trong java

LAZY bản chất nó sẽ lấy dữ liệu sau khi hàm getter đc gọi , nên là nếu sout thì sẽ ko nhìn thấy có dữ liệu trong Object cha chứa nó.

EAGER bản chất sẽ lấy dữ liệu luôn ngay khi gọi object cha chứa nó.

## Quan hệ HAS-A, IS-A.

Nếu class A **extends** class B: thì đó là quan hệ **is-a**. Vì sao? Is-a có nghĩa là cái này là cái kia. Khi A kế thừa B thì A sẽ có những tính chất của B, nói cách khác A là một bản sao có các tính chất của B.

Nếu class A **implements** interface B: thì đó là quan hệ **has-a**. Vì sao? Has-a có nghĩa là cái này có khả năng gì đó. Khi A implements (triển khai - từ này dịch hơi ngựa) B thì là lớp A có các khả năng được mô tả trong các hàm của interface B.

## . Collection và Collections khác gì nhau

* Collections là 1 class chứa các static method để thao tác vs tập dữ liệu
* colection thì là 1 interface trong collection framework

## Abtract class và Interface

<https://viblo.asia/p/interface-vs-abstract-class-ke-tam-lang-nguoi-nua-can-07LKX9JeZV4>

Abstract class: nên có ít nhất 1 abstract method, có đẩy đủ tính năng như 1 class bình thường, có các access modifier cho abstract method, chỉ extend đc 1 abstract class, có constructor

[10:05](https://techasiansjsc.slack.com/archives/D03JJ4P6TNE/p1658070309068449)

Interface: chỉ chứa biến static final, chứa các method trìu tượng, mặc định là public, có thể extends nhiều interface khác, ko có constructor, default method đc implement trong interface và override ở class con, static method đc implement nhưng không đc override, ko thể khai báo interface là private và protected

1 class extends 1 class và implement nhiều interface

## Bộ nhớ trong java

<https://gochocit.com/ngon-ngu-lap-trinh/quan-ly-bo-nho-trong-java-bo-nho-stack-va-bo-nho-heap>

lưu trữ các biến local và các method,

* Stack ( Lưu biến cục bộ)

stack có kích thước nhỏ,tự tăng kích thước khi method đc gọi là giảm khi method thực hiện xong,các biến local chỉ tồn tại khi method đang chạy, Truy cập nhanh hơn Heap

* Mỗi thread là 1 stack
* Lỗi là stackoverflowerror
* ưu trữ các đối tượng, biến static, biến toàn cục
* ko tự giải phóng bộ nhớ, phải dùng GC kích thước lớn, Truy cập lâu hơn stack
* chia sẻ cho tất cả các luồng Lỗi outofmemoryerror

Các object mới đc khởi tạo sẽ đc lưu ở heap và các tham chiếu đến chúng đc lưu ở stack

Stack: - lưu trữ các biến local và các method, -  stack có kích thước nhỏ, - tự tăng kích thước khi method đc gọi là giảm khi method thực hiện xong, -  các biến local chỉ tồn tại khi method đang chạy - Truy cập nhanh hơn Heap - Mỗi thread là 1 stack - Lỗi là stackoverflowerror Heap: - lưu trữ các đối tượng, biến static, biến toàn cục - ko tự giải phóng bộ nhớ, phải dùng GC -  kích thước lớn, - Truy cập lâu hơn stack -  chia sẻ cho tất cả các luồng - Lỗi outofmemoryerror Các object mới đc khởi tạo sẽ đc lưu ở heap và các tham chiếu đến chúng đc lưu ở stack

* HEAP

Lưu biến static

Lưu các các object đc khởi tạo với từ khóa new 🡪 biến global cũng sẽ đc khởi tạo theo sau

Bộ nhớ sẽ lớn hơn stack

Dung lượng sử dụng của heap sẽ tăng giảm phụ thuộc vào các object được tạo ra.

Thời gian tồn tại của object phụ thuộc vào Garbage Collector (GC) của Java. Khi một object bị **null**hoặc không còn được sử dụng thì GC sẽ xóa object khỏi bộ nhớ.

Dung lượng vùng **heap** thường lớn hơn vùng nhớ **stack**.

## Hướng đối tượng là gì

<https://topdev.vn/blog/oop-la-gi/>

**Định nghĩa OOP là gì?**

OOP (từ viết tắt của -Object Oriented Programming)\_lập trình hướng đối tượng là một phương pháp lập trình dựa trên các khái niệm về lớp và đối tượng. OOP thường tập trung vào các đối tượng thao tác hơn là logic để có thể thao tác chúng.

OOP là một nền tảng quen thuộc của các design pattern hiện nay. OOP đặt ra mục tiêu quản lý source code giúp gia tăng khả năng tái sử dụng và quan trọng hơn hết là có thể tóm gọn được các thủ tục đã biết trước tính chất thông qua quá trình sử dụng các đối tượng.

## Database là gì

Database là 1 tập hợp các dữ liệu có quan hệ với nhau là các bảng

Bảng là tập hợp dữ liệu có cùng định dạng với nhau, các bảng có thể có quan hệ với nhau 1-1,1-n,n-1,n-n;

## 11 Quy tắc viết Code ( CleanCode )

<https://techmaster.vn/posts/36509/13-quy-uoc-can-thiet-de-viet-clean-code>