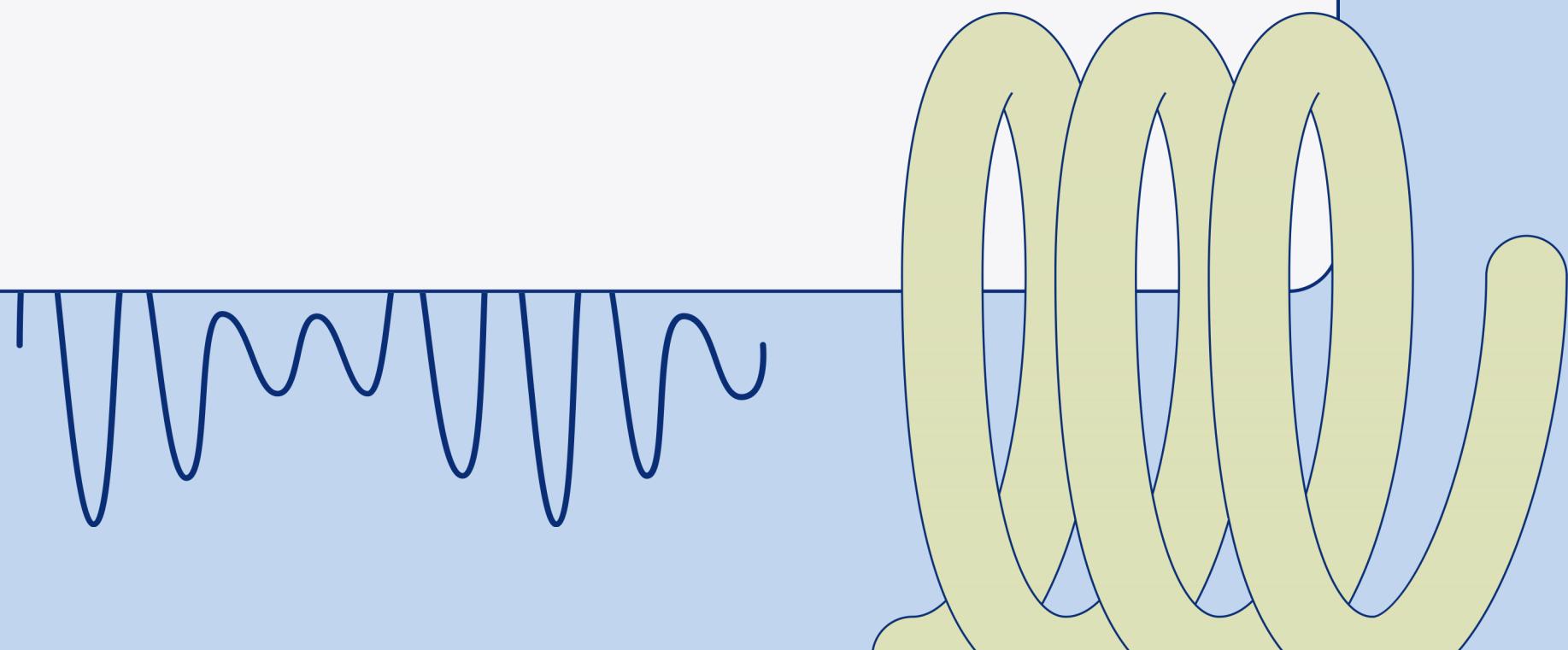
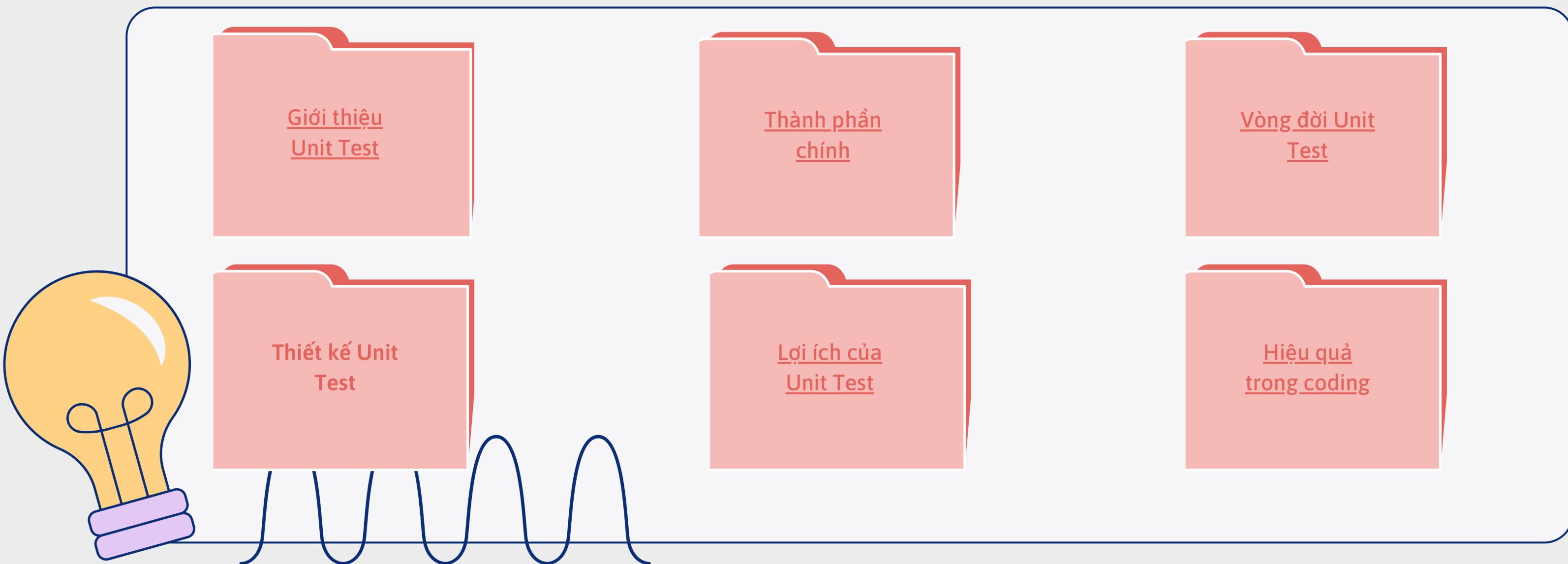


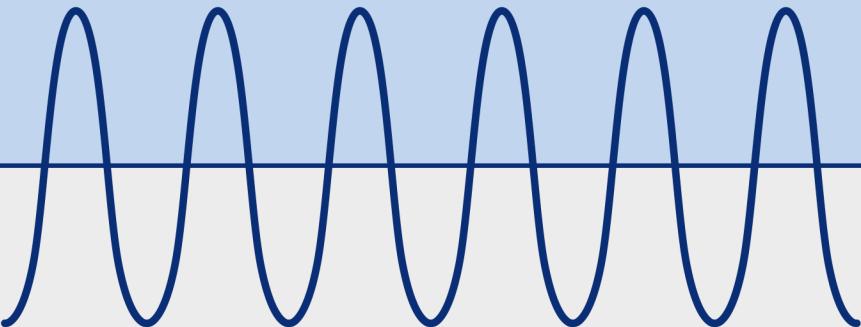
Unit Test



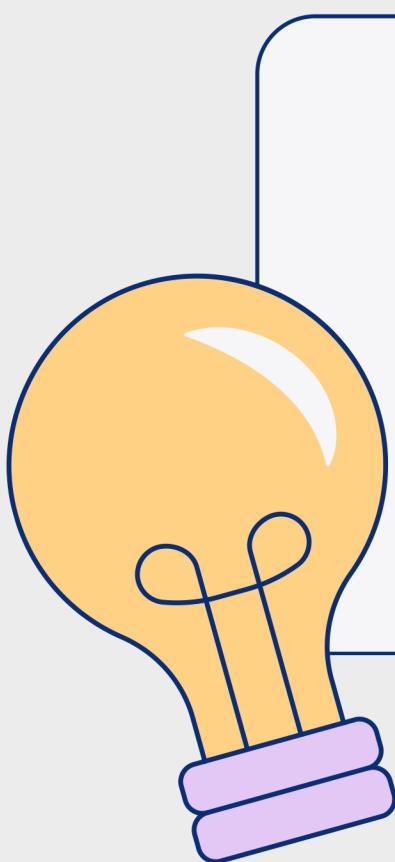
Nội dung



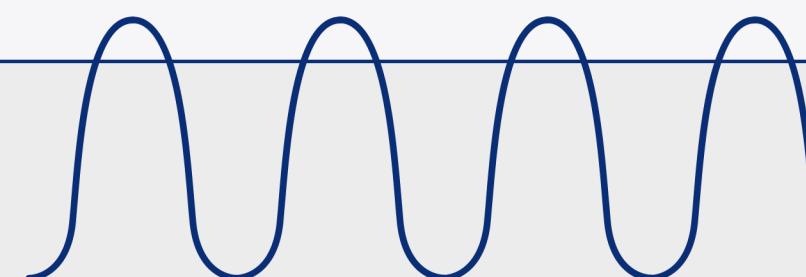
Giới thiệu Unit Test



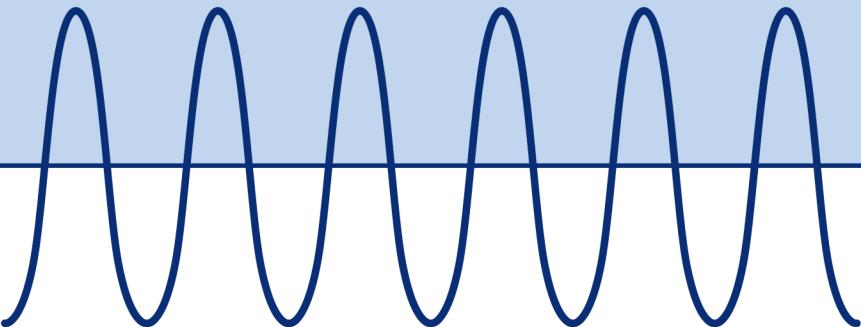
1. Định nghĩa



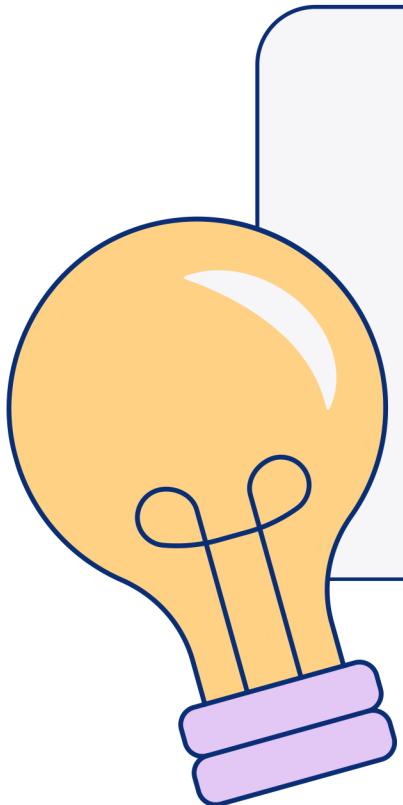
Unit test là một phương pháp kiểm thử phần mềm tập trung vào việc kiểm tra các đơn vị nhỏ nhất của mã nguồn, thường là các hàm, phương thức, hoặc lớp, để đảm bảo rằng chúng hoạt động đúng cách.



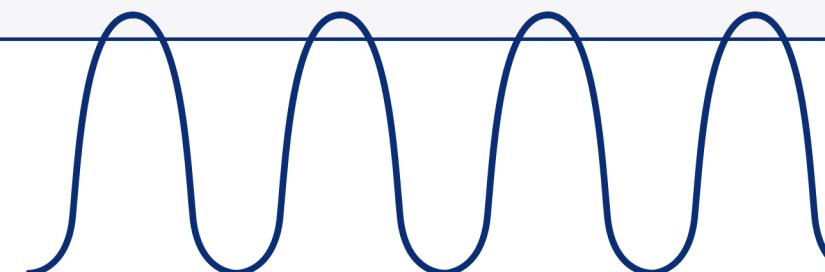
Giới thiệu Unit Test



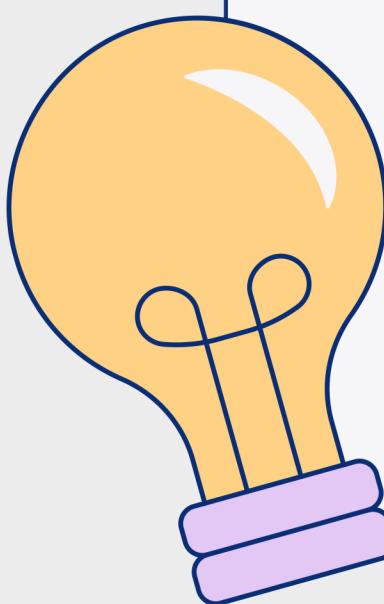
2. Mục tiêu



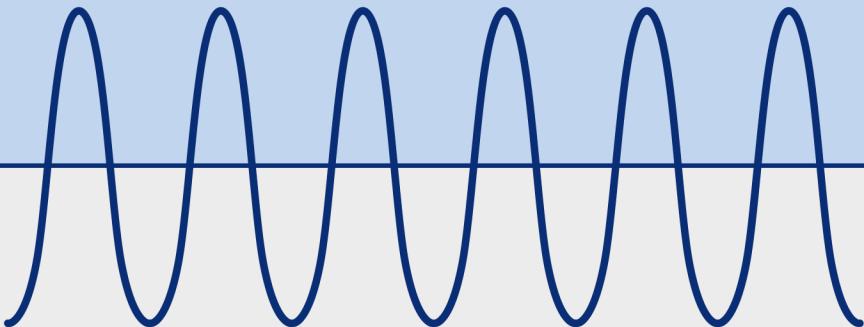
Mục tiêu của unit test là phát hiện và khắc phục lỗi ở giai đoạn phát triển sớm nhất có thể, giúp tiết kiệm thời gian và công sức về sau.



Thành phần chính



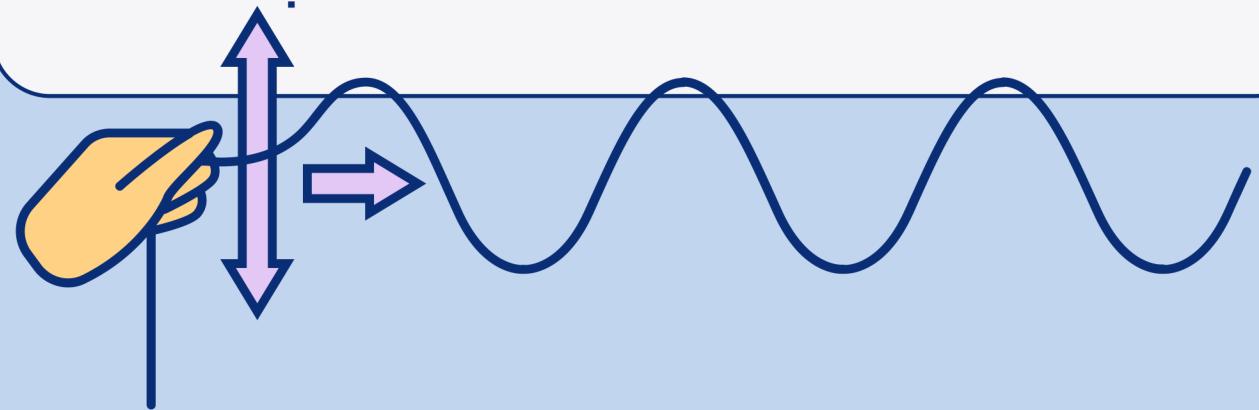
- Assertion: Phát biểu mô tả kiểm tra tính đúng đắn của mã.
- Test Case & Test Point: Tập hợp kiểm thử cho mỗi tính năng cụ thể.
- Test Suite: Nhóm các test case cho mỗi module.
- Regression Testing: Kiểm thử tự động ngăn lỗi cũ tái phát



Vòng đời Unit Test

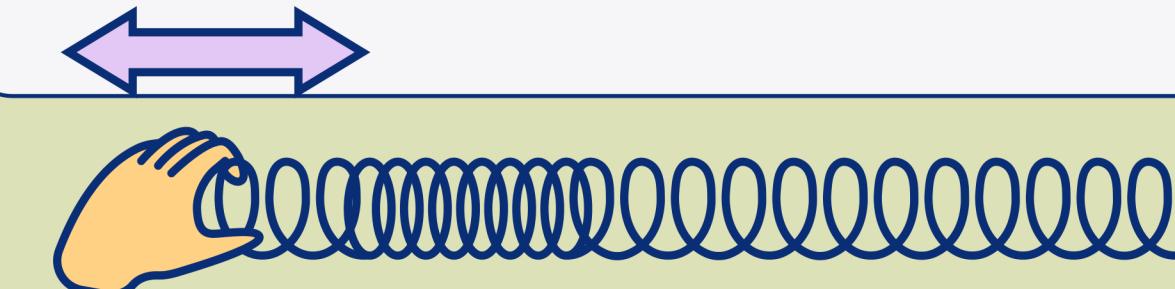
Trạng thái

- -Fail (trạng thái lỗi)
- -Ignore (tạm ngừng thực hiện)
- -Pass (trạng thái làm việc)
- Toàn bộ Unit Test được vận hành trong một hệ thống tách biệt.



Quy trình

- -Được vận hành lặp lại nhiều lần
- -Tự động hoàn toàn
- -Độc lập với các Unit Test khác.

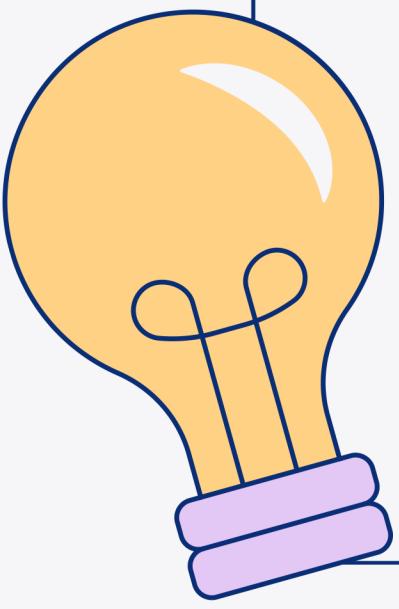


Thiết kế Unit Test

Thiết lập các điều kiện cần thiết: khởi tạo các đối tượng, xác định tài nguyên cần thiết, xây dựng các dữ liệu giả...Triệu gọi các phương thức cần kiểm tra.Kiểm tra sự hoạt động đúng đắn của các phương thức.Dọn dẹp tài nguyên sau khi kết thúc kiểm tra.



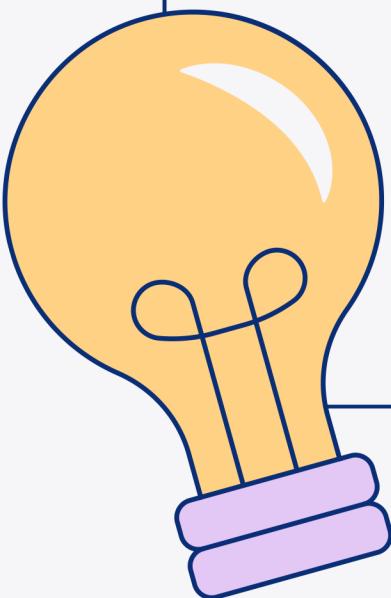
Ứng dụng Unit Test



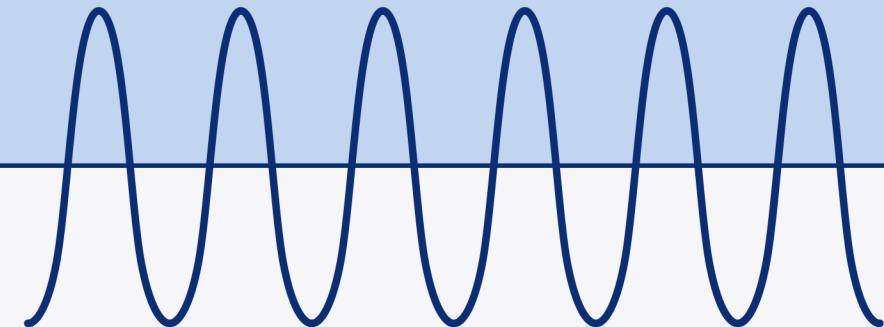
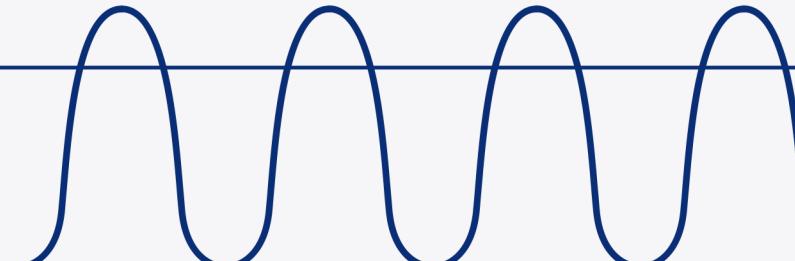
A yellow lightbulb with a purple base is shown on the left side of the slide. A thin blue line connects the top of the lightbulb to the start of a wavy blue line that runs horizontally across the bottom of the slide.

- Kiểm tra mọi đơn vị nhỏ nhất là các thuộc tính, sự kiện, thủ tục và hàm.
- Kiểm tra các trạng thái và ràng buộc của đối tượng ở các mức sâu hơn mà thông thường chúng ta không thể truy cập được.
- Kiểm tra các quy trình (process) và mở rộng hơn là các khung làm việc(workflow – tập hợp của nhiều quy trình)

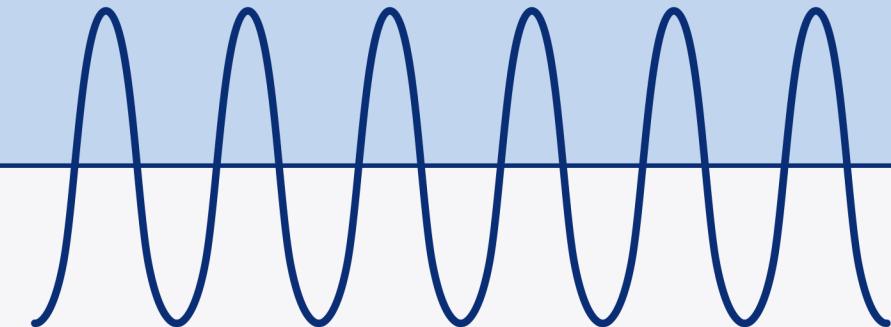
Lợi ích của việc áp dụng Unit Test



- -Tạo ra môi trường lý tưởng để kiểm tra bất kỳ đoạn code nào, có khả năng thăm dò và phát hiện lỗi chính xác, duy trì sự ổn định của toàn bộ PM và giúp tiết kiệm thời gian so với công việc gỡ rối truyền thống.
- -Phát hiện các thuật toán thực thi không hiệu quả, các thủ tục chạy vượt quá giới hạn thời gian.
- -Phát hiện các vấn đề về thiết kế, xử lý hệ thống, thậm chí các mô hình thiết kế.
- -Phát hiện các lỗi nghiêm trọng có thể xảy ra trong những tình huống rất hẹp.
- -Tạo hàng rào an toàn cho các khối mã: Bất kỳ sự thay đổi nào cũng có thể tác động đến hàng rào này và thông báo những nguy hiểm tiềm tàng.
 -



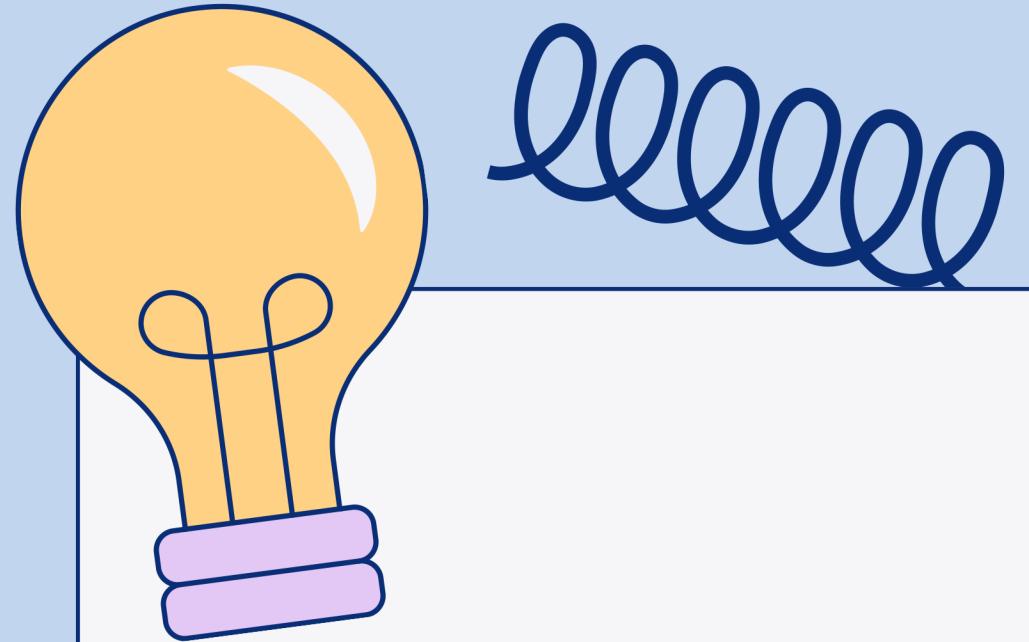
Hiệu quả trong coding



A yellow lightbulb icon with a purple base, connected by a line to a bulleted list of coding best practices.

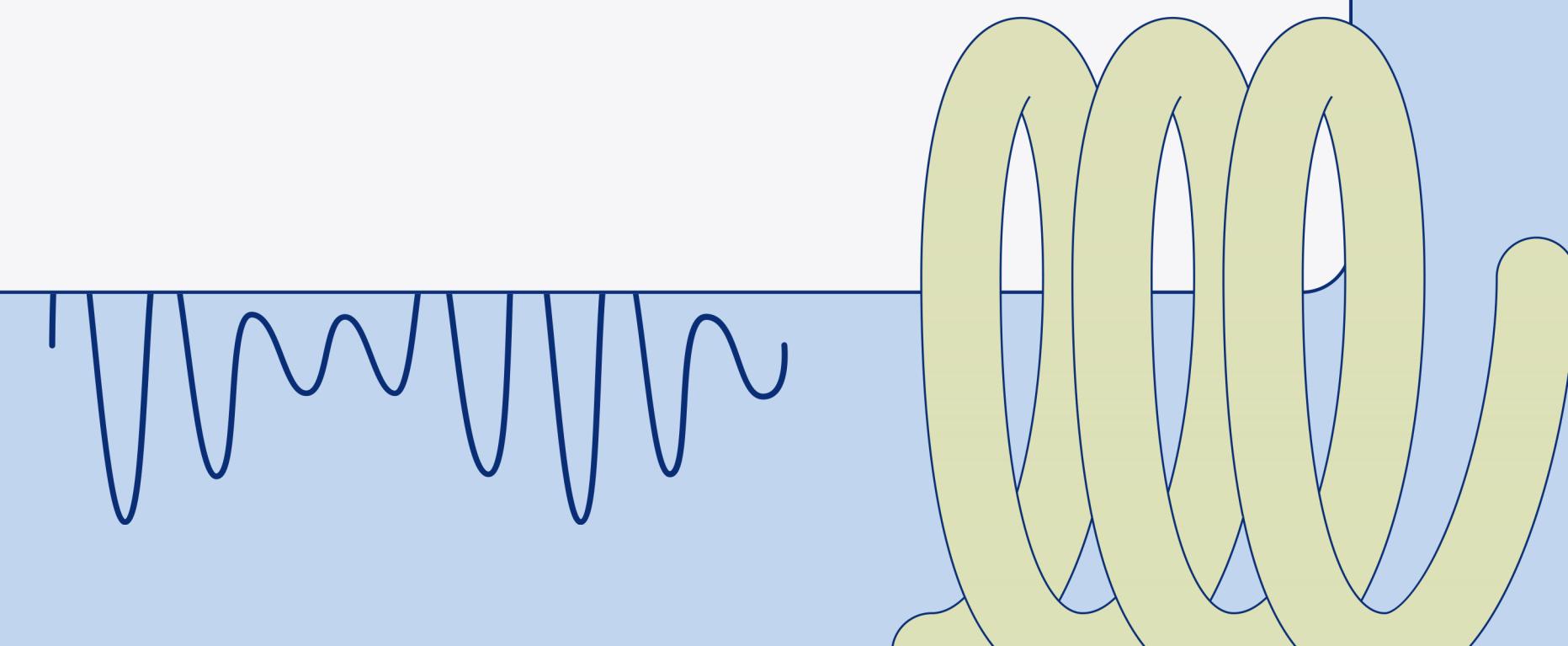
- Chuẩn bị cho các tình huống xấu: Xử lý ngoại lệ, lỗi kết nối.
- Phương châm: Khởi đầu bằng Fail, kết thúc bằng Pass.
- Thực thi thường xuyên: Tự động hóa để phát hiện sớm lỗi.

A horizontal blue sine wave oscillating from left to right below the lightbulb icon.



GIỚI THIỆU VỀ

Code Style



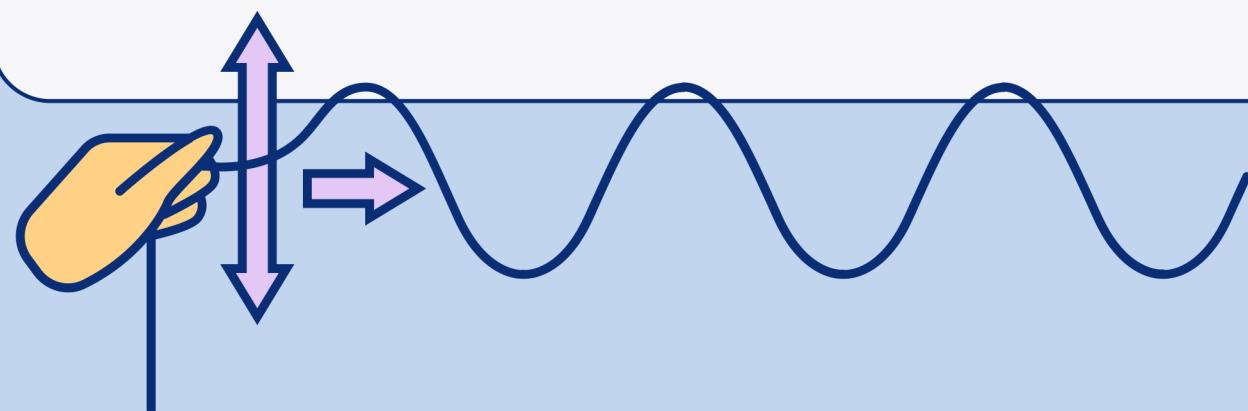
Nội dung



White Space

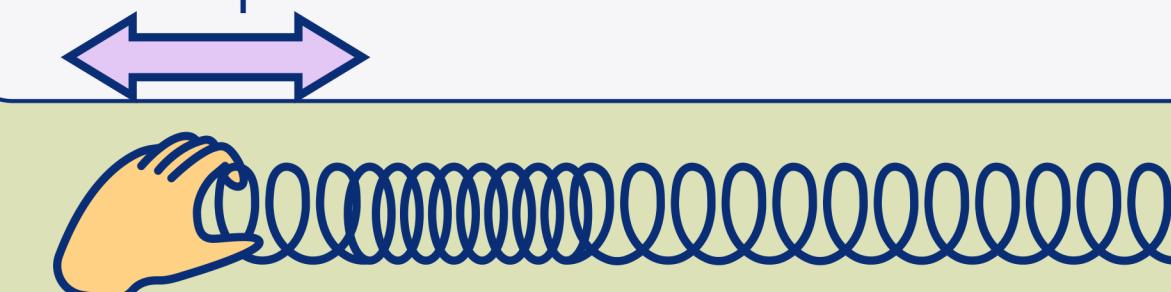
Thụt đầu dòng

- 1 tab = 1 đơn vị thụt đầu dòng.
- Dòng code cùng cấp cách nhau 1 tab.

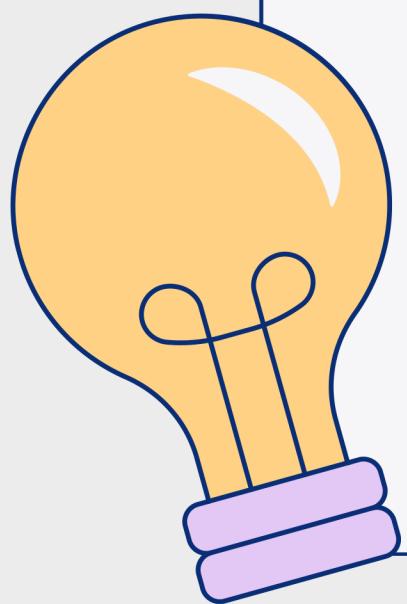


Dòng trống

- Gom dòng code liên quan thành block.
- Giữa hai block cách nhau 1 dòng trống.
- Khoảng trắng sau dấu phẩy, chấm phẩy và xung quanh toán tử.

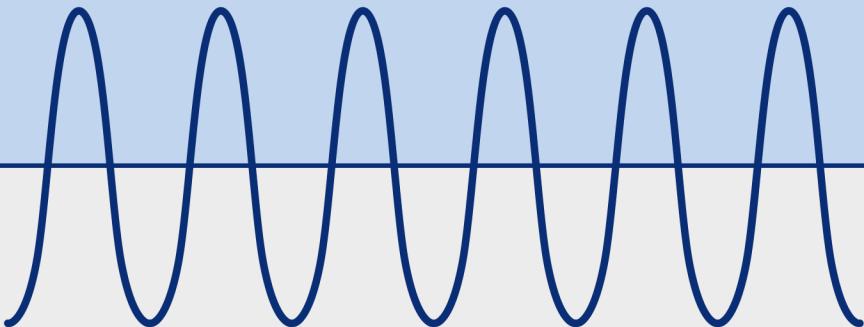
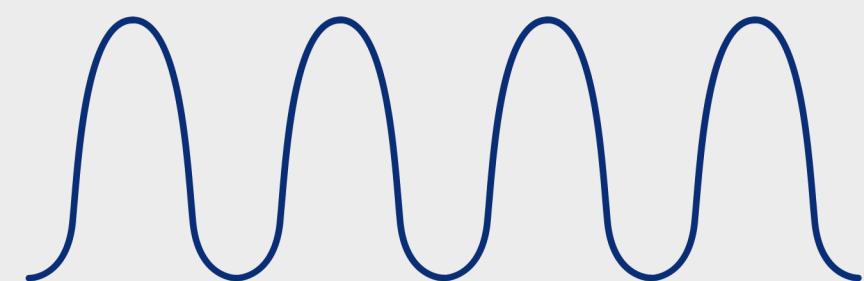
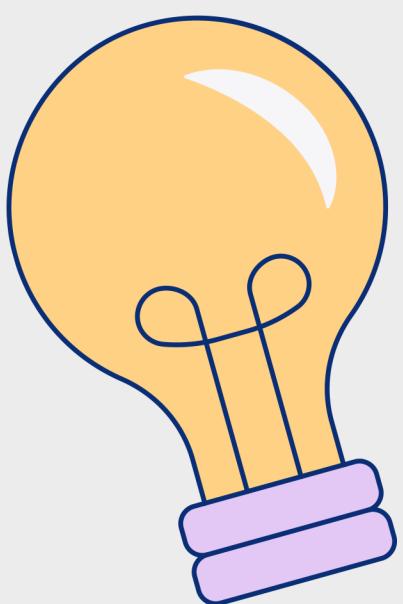


Comment

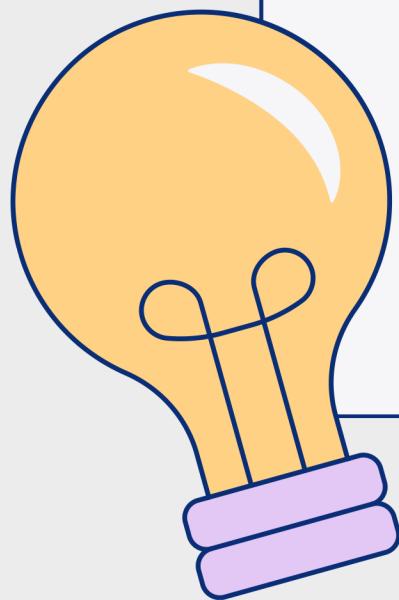


- Comment đơn giản, rõ ràng.
- Endline comment canh lề như nhau.
- Viết comment trong quá trình code.
- Chỉ viết comment khi code phức tạp.

Quy tắc đặt tên

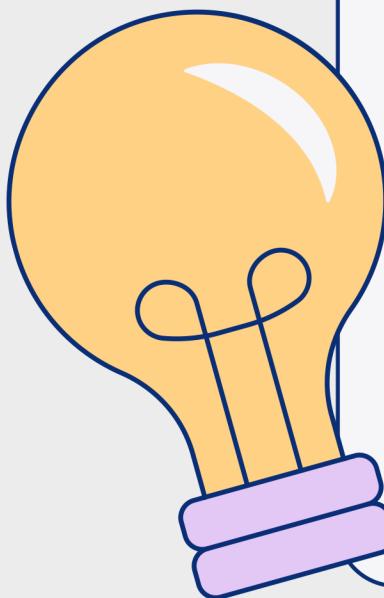


Quy tắc viết hoa



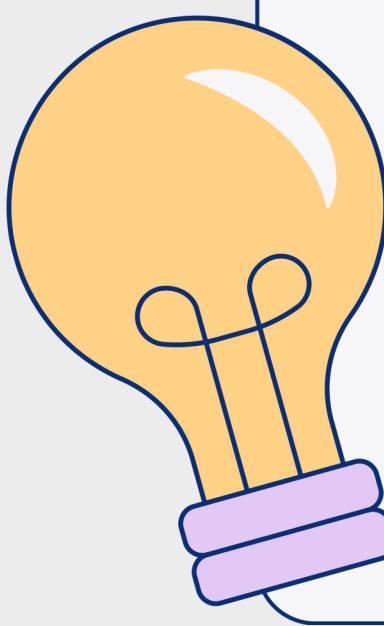
- Pascal Case: MyProvider, StringBuilder
- Camel Case: myProvider, stringBuilder

Đặt tên class, interface, abstract class

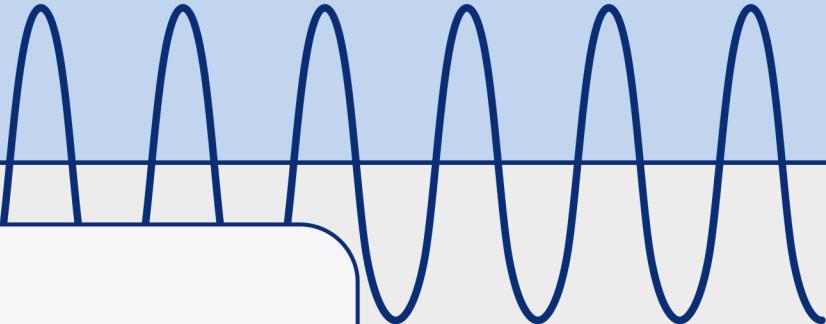
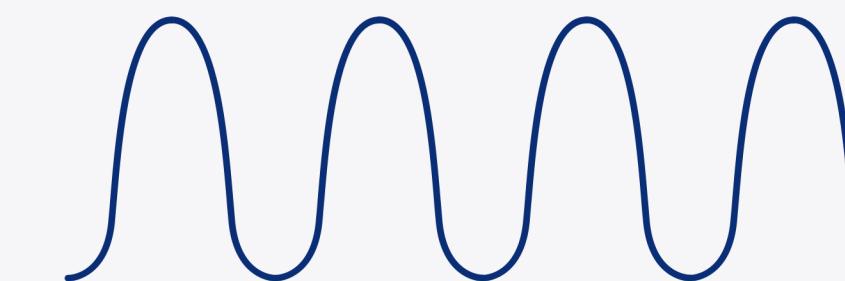


- Sử dụng danh từ hoặc cụm danh từ: SinhVien, FormSinhVien.
- Dùng Pascal Case: SinhVien, FormSinhVien.
- Hạn chế viết tắt gây khó hiểu:
 - Sai: FormSV
 - Đúng: FormSinhVien
- Không dùng tiền tố khi đặt tên lớp:
 - Sai: ISinhVien
 - Đúng: SinhVien

Viết phương thức hiệu quả

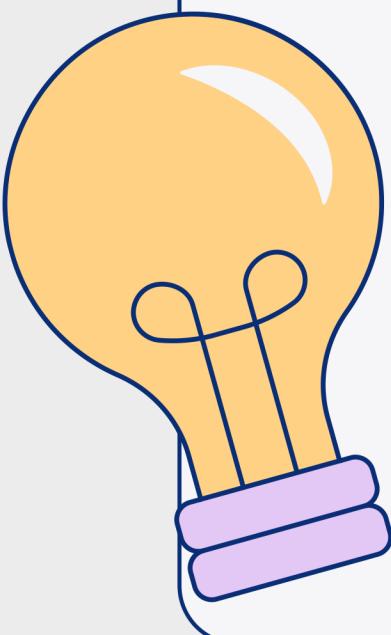


- Gom code lặp thành phương thức.
- Tách code phức tạp ra phương thức riêng.
- Khai báo tham số vừa đủ.
- Mỗi phương thức chỉ thực hiện 1 chức năng.
- Phương thức dài 50-150 dòng.

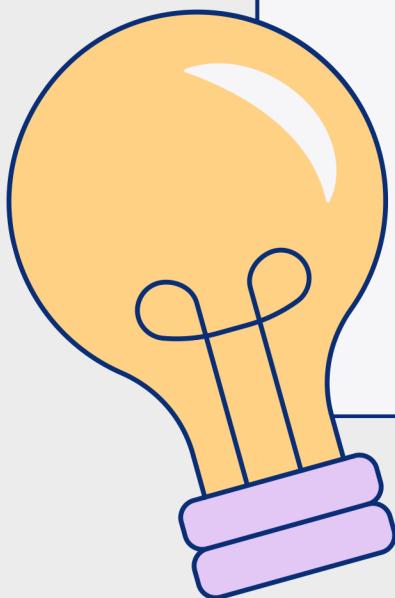
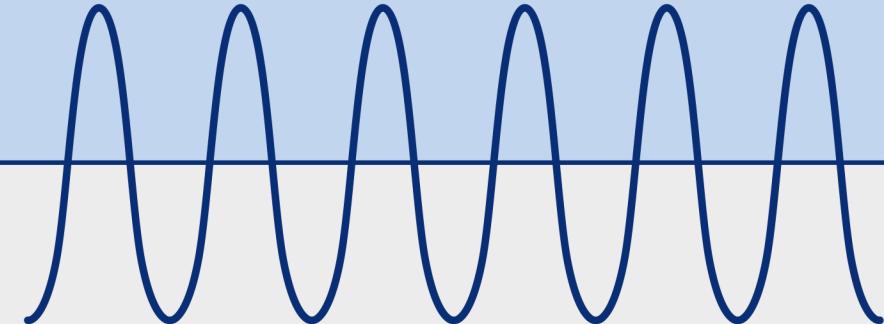


Biến

- Sử dụng Camel Case để đặt tên biến, ví dụ: int diemTrungBinh, String hoTen.
- Không dùng tiền tố, ví dụ:
- Đúng: String address
- Sai: String strAddress
- Tên biến gợi nhớ, tránh viết tắt gây khó hiểu, ví dụ:
- Đúng: String address
- Sai: `String

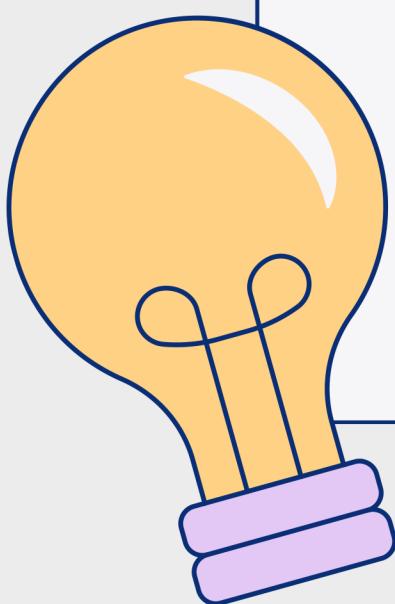


Sử dụng biến



- Tránh khai báo biến không dùng.
- If, while, for không lồng nhau quá 3 bậc.

Import thư viện



- Chỉ import cần thiết: `import java.util.List;` thay vì `import java.util.*;`



xin cảm ơn!