How do refactor (Tái cấu trúc thế nào?)

- Được thực hiện như 1 loạt các thay đổi nhỏ
- Mỗi thay đổi sẽ làm cho code hiện tại tốt hơn 1 chút
- Vẫn đảm bảo được chương trình hoạt động bình thường

Checklist of refactoring done right way (Danh sách kiểm tra tái cấu trúc đúng cách)

Mã sẽ trở nên sạch sẽ hơn

- Sau khi refactor mà code vẫn không sạch. Hãy tìm hiểu lý do tại sao điều này lại xảy ra.

- Điều này thường xảy ra khi bạn chuyển từ refactor những thay đổi nhỏ -> trộn 1 loạt refactor thay đổi lớn
- Rất dễ sót code, đặc biệt khi bạn bị giới hạn thời gian

- Nhưng nó cũng xảy ra khi làm việc với code bẩn. Bất kể bạn cải thiện điều gì, toàn bộ code vẫn là thảm hoạ.

- Trong trường hợp này, nên cân nhắc viết loại hoàn toàn các phần của code.
- Trước đó, nên viết test và dành ra 1 khoảng thời gian kha khá
- Nếu không, nó sẽ như kết quả đầu tiên, vẫn không sạch.

Không nên tạo chức năng mới trong quá trình tái cấu trúc

- Không nên vừa refactor code, vừa phát triển thêm feature mới.
- Cố gắng tách biệt các quy trình này ít nhất là phạm vi của từng commit, tốt hơn thì chia branches ra.

Tất cả các bài test hiện có phải vượt qua sau khi refactor code

Thông thường, có 2 trường hợp mà các bài test fail sau khi refactor code

- Bạn mắc lỗi trong quá trình refactor code. Hãy tiếp tục và sửa lỗi
- Các bài test của bạn ở mức quá thấp. Ví dụ, bạn đang kiểm tra private method của classes

- Trong trường hợp này, lỗi nằm ở các bài test
- Có thể refactor lại các bài test hoặc viết một bộ các bài test mới cấp cao hơn
- 1 cách tuyệt vời để tránh tình huống này là viết các bài kiểm tra theo kiểu TDD (google nhé)