**CLEAN CODE**

1. Đặt tên
   1. Sử dụng tên thể hiện rõ ý đồ
   2. Sử dụng tên có thể phát âm được, tìm kiếm được
   3. Tránh mã hoá: mã hoá tên phổ biến trước đây là Hungarian Notation ( đúng với Java, là một ngôn ngữ có quy định chặt chẽ về kiểu )
   4. Tránh mental mapping
   5. Tên Class
      1. Class và đối tượng nên có tên là danh từ hoặc cụm danh ( ex: Customer, WikiPage, Account)
      2. Hạn chế các từ như Manager, Processor, Data hoặc Info khi đặt tên class và đối tượng.
   6. Tên hàm
      1. Tền hàm nên được đặt bằng động từ hoặc cụm động

( ex: postPayment, deletePage hoặc save.)

* + 1. Khi overload constructor:

( ex: Complex fulcrumPoint = Complex.FromRealNumber(23.0);)

1. Hàm
   1. Ngắn gọn và làm một việc duy nhất
      1. Một hàm lý tưởng viết không quá 20 dòng
   2. Tham số
      1. Số lý tưởng của một hàm không nên quá 2

( ex: Circle makeCircle(Point center, double radius);)

* 1. Không có tác dụng phụ
  2. Tách biệt giữa các hành động truy vấn

1. Comment
   1. Comment không thể chửa code xấu
   2. Comment code không dùng đến ( remove nó đi …, sd Git đã xem history )
2. Định dạng
   1. Độ mở theo chiều dọc giữa các nhóm code
      1. Mỗi dòng thể hiện một diễn đạt nhất định
      2. Mỗi nhóm các dòng thế hiện một suy nghĩ, một sự liên quan nhất định đến nhau.
      3. Các suy nghĩ đó nên được tách biệt với nhau bằng một khoảng trắng.
   2. Khoảng cách giữa các thành phần liên quan: Các thành phần có mối liên quan mật thiết với nhau nên được đặt gần nhau
      1. Khai báo biến:
         1. Biến nên được khai báo gần vị trí sử dụng hết mức có thể.
         2. Các thuộc tính của class nên được khai báo ở đầu class để đảm bảo nhất quán
      2. Không có phụ thuộc:
         1. Nếu một hàm gọi một hàm khác, chúng nên nằm gần nhau.
         2. Hàm được gọi nằm ngay dưới hàm gọi.
3. Statement
   1. Tổ chức code theo 1 đường thẳng
   2. Mệnh đề if ( Tổ chức sao cho trường hợp bình thường được xử lý trước, sau đó mới đến các trường hợp ngoại lệ )
   3. Mệnh đề case
      1. Sắp xếp theo thứ tự bảng chữ cái: Nếu vai trò của mỗi case là như nhau, sắp theo thứ tự này làm code dễ đọc hơn và việc tìm một case cụ thể dễ hơn.
      2. Để các case thường dùng lên đầu tiên: Người đọc có thể tìm các case thường được thực thi nhất một cách nhanh chóng.
      3. Để các case bình thường lên trước: Trong trường hợp bạn có một số case xử lý flow thông thường và một số case xử lý ngoại lệ, để case thông thường lên trước. Thêm comment chỉ rõ case nào là bình thường, case nào xử lý ngoại lệ.