SOLID

Avoid code smells, extend, maintain and easily refactor code

**1. Single reponsibility principle**: 1 class chỉ nên có 1 nhiệm vụ mà thôi (1 nhiệm vụ không có nghĩa là 1 funciton)

**2. Open close principle**: mở rộng code thoải mái nhưng k dc chỉnh sửa (build ra 1 class thành chuẩn chung)

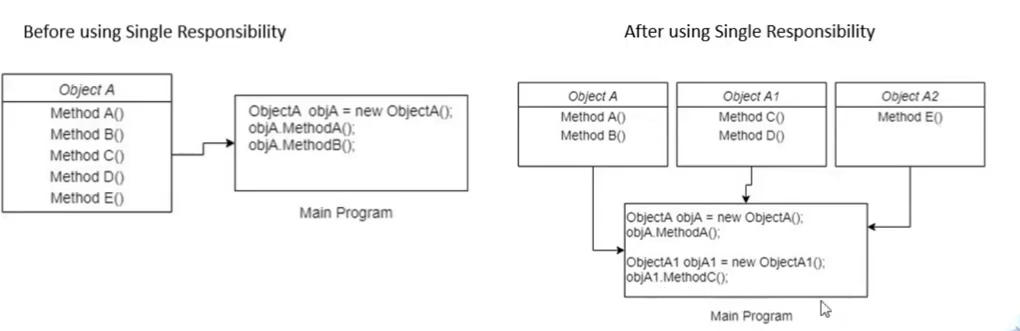
**3. Liskov substitution principle**: Giống như mở rộng của open và close nhưng focus trên behivior class cha và con của nó , dể hiểu là thể hiện của object con có thể thay thế object cha nhưng **không làm thay đổi tính đúng đắng** của chương trình.

**4. interface segregation principle**: Có interface bự thì chia nhỏ nó ra

**5. Dependency inversion principle**: Định nghĩa astract cho những obejct khác phụ thuộc (nhưng nó đúng 1 mình không ai phụ thuộc vào nó) Và high level không dc phụ thuộc vào low level (mà dùng DI hoặc interface)

=====================================================================================Details:

**1. Single reponsibility principle**: không apply dễ bị duplicate code, nhiệm vụ không rõ ràng



**2. Open close principle**:

