Single-responsibility Principle

S.R.P for short – this principle states that:

A class should have one and only one reason to change, meaning that a class should have only one job.

Cohesion

low Cohesion- a class is designed from the point of view of multiple tasks. This is bad because it has two drawbacks

a) high maintenance

b) no reusability

high cohesion- a class is designed from the point of view of single task. This is good because it gives u two advantages:

a) low maintenance

b) reusability

high cohesion is also called as "SRP" i.e. Single Responsibility Principle.

High cohesion is when you have a class that does a well defined job. Low cohesion is when a class does multiple jobs that don't have much in common.

Let's take this example:

You have a class that adds two numbers, but the same class creates a window displaying the result. This is a low cohesive class because the window and the adding operation don't have much in common. The window is the visual part of the program and the adding function is the logic behind it.

To create a high cohesive solution, you would have to create a class Window and a class Sum. The window will call Sum's method to get the result and display it. This way you will develop separately the logic and the GUI of your application.