EXERCÍCIOS DO MÓDULO 4

Exercício 1 – Importância de se escrever código limpo

Explique, com suas palavras, por que os seguintes princípios de código limpo são importantes:

1 - Usar nomenclatura adequada

Dar nomes coerente para pacotes, classes, métodos e variáveis é muito importante na legibilidade do código. Isso faz com que o leitor se situe e absorva o mais rápido possível a lógica do sistema em questão e ainda demonstra organização e cuidado no momento da codificação. Existe um ditado que diz: “dar nomes aos bois”, isso é crucial na programação, ou seja, uma boa nomenclatura deve ser capaz de descrever o que aquilo é e o que aquilo faz de maneira muito clara seguindo as convenções como a Java que utiliza camel-case em suas terminologias.

2 - Resolver os problemas na causa raiz

Encontrar soluções que vá de encontro às origens dos problemas diretamente é imprescindível. Não adianta protelar ou fazer gambiarras escondendo os bugs debaixo do tapete. Penso que essa ação faz com que nosso trabalho seja mais eficiente, não permitindo que, posteriormente, essas questões retornem muitas vezes mais complexas.

3 - Seguir a política do escoteiro

Essa política pode ser resumida em fazer a diferença. Buscar sempre melhorar o que estiver na sua frente. Isso é de grande importância porque isso pode refletir na sua identidade e na forma como as pessoas ao seu redor te verão, tipo passou essa frase: “Esse código está diferente (no sentido de bom), com toda certeza passou pelas mãos de fulano”. Isso mostra competência e excelência no que faz.

Exercício 2 – Princípios de código limpo

Para cada uma das assinaturas de método abaixo, explique qual o princípio de código limpo que eles estão ferindo:

1- *private void somaNumeros(int a, int b, int c, int d, int e, int f)*:

Esse método está passando vários argumentos além de está retornando valor algum (void).

2- *private void oPaiTaOn()*

A assinatura de método apresenta um mal uso de terminologia, pois não descreve de forma adequada o que é e nem o que esse método faz.

3- *private double checaSaldoEAtualiza(long userId, double value)*

Esse método quebra o princípio da responsabilidade única, isto é, faz mais de uma coisa. Ele deveria ou só checar ou atualizar o saldo.