## Aluno: Leonardo Folle de Souza Dias - RA: 242650-1

## **ETAPA 1 - TEÓRICA**

1 - O que é orientação a objetos e quais são seus princípios fundamentais?

**R:** A Programação orientada a objetos (POO) é um conceito aplicado na programação onde haverá a interação entre diversas unidades chamadas objetos. Usamos ela para nos basear na vida real e resolver problemas de SOFTWARE, a orientação é algo atemporal, ou seja, não está ligado a uma linguagem específica.

* Princípios fundamentais:
  + **Encapsulamento:** Protege o estado interno do objeto.
  + **Abstração:** Esconde complexidade e mostra apenas o necessário.
  + **Herança:** Permite a criação de novas classes a partir de classes existentes.
  + **Polimorfismo:** Permite que objetos sejam tratados como instâncias de suas classes pai.

2 - Quais são as vantagens da orientação a objetos em comparação com paradigmas de programação procedural?

**R:** As vantagens da orientação a objetos sobre a programação procedural incluem:

* **Reutilização de código** através da herança.
* **Modularidade** para separar preocupações.
* Facilidade de **manutenção** e **atualização** de código.
* Melhor **gestão de complexidade** do software.

3 - Explique o conceito de encapsulamento e como ele é implementado em linguagens de programação orientadas a objetos.

**R: Encapsulamento** é um conceito que envolve a restrição do acesso direto aos componentes do objeto. É implementado através de modificadores de acesso como private, protected, e public.

4 - Explique o conceito de classe e objeto em programação orientada a objetos, e como eles estão relacionados.

**R:** Uma **classe** é um blueprint para criar objetos (uma instância de uma classe). Um **objeto** é uma instância de uma classe. A classe define as propriedades e comportamentos, enquanto o objeto é uma entidade concreta baseada na classe.

5 - Qual é a importância dos construtores e métodos em uma classe em programação orientada a objetos?

**R: Construtores** são métodos especiais usados para inicializar novos objetos. **Métodos** são funções definidas dentro de uma classe que descreve os comportamentos dos objetos. Eles são importantes para criar instâncias e interagir com os objetos.

6 - Quais são as diferenças entre métodos estáticos e métodos de instância em uma classe?

**R:** **Métodos estáticos** pertencem à classe e não a uma instância específica, enquanto **métodos de instância** requerem uma instância específica de uma classe para serem chamados.

7 - Como são tratados os conceitos de acoplamento e coesão na orientação a objetos e por que são importantes?

**R:** **Acoplamento** refere-se à interdependência entre classes; **coesão** refere-se à unidade dentro de uma classe. Baixo acoplamento e alta coesão são importantes para um design de software robusto, pois facilitam a manutenção e a escalabilidade.

8 - Qual é o objetivo do método main?

**R:** O método **main** é o ponto de entrada de muitos programas e linguagens de programação orientadas a objetos. Ele é responsável por iniciar a execução do programa.

9 - Como ocorre a passagem de parâmetros em métodos de uma classe em programação orientada a objetos?

**R:** A **passagem de parâmetros** em métodos ocorre quando você chama um método e passa valores para ele, que são usados dentro do método para realizar operações ou cálculos.

10 - Qual é a diferença entre os modificadores de acesso "public", "private" e "protected" em uma classe?

**R:**

* **Public**: Torna os membros acessíveis de qualquer lugar, permitindo que outras classes acessem e modifiquem diretamente esses membros. É útil para criar uma interface pública para a classe.
* **Private**: Restringe o acesso aos membros apenas dentro da própria classe. Isso ajuda a encapsular o comportamento interno da classe, ocultando detalhes de implementação e protegendo seus dados.
* **Protected**: Permite o acesso aos membros dentro da própria classe, suas subclasses e classes do mesmo pacote. É útil quando se deseja fornecer acesso aos membros para subclasses, mas não para outras classes fora do pacote.

11 - Qual é a importância da palavra-chave "this" em programação orientada a objetos e como ela é utilizada?

**R:** A palavra-chave “this” **é** uma referência ao objeto atual. É usada para acessar membros da própria classe, diferenciar entre parâmetros de método e campos de classe, e para passar o objeto atual como parâmetro.