Data: DD/MM/AAAA
Nome: Fulano de Tal
Assinatura: ######
Matrícula: XX/XXXX-X

EXEMPLO DE PROVA (GABARITO)
Disciplina: Arquitetura & Desenho de Software - Aula de Revisão - Profa. Milene Serrano - Monitores Artur Sousa e Hugo Sobral

****

Questão Única
****

Contexto:
Um Engenheiro de Software foi contratado recentemente em uma empresa.
Para ele, foi conferido um primeiro desafio.
Esse desafio consistiu na aplicação de padrões de projeto, visando solucionar algumas demandas específicas.

Nesta empresa, é mantido um sistema de controle de informação que busca informações de múltiplas bases de dados, portanto, são instanciadas várias conexões para vários bancos diferentes.
Entretanto, cada funcionalidade do sistema possui uma própria instância de vários bancos de dados, mesmo que a conexão do banco de dados já tenha sido feita anteriormente.
Segue abaixo as principais conexões à bancos de dados do projeto:

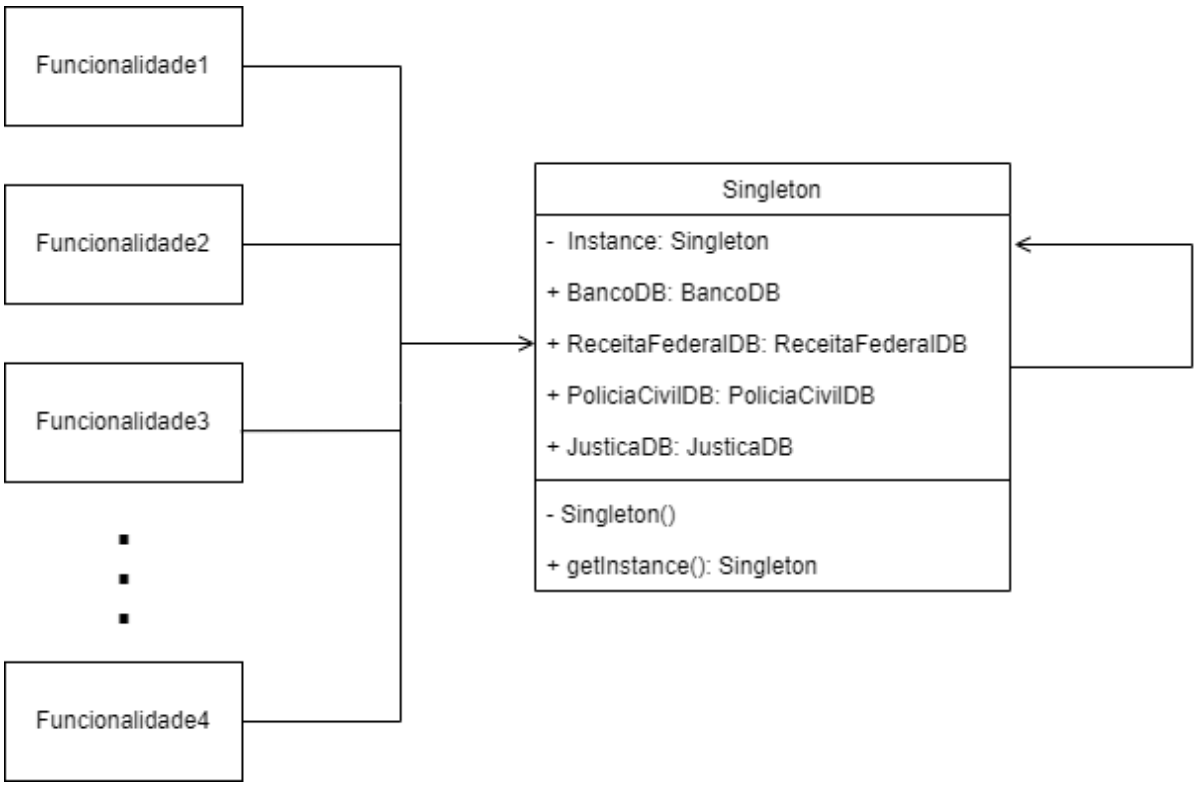
- historico_transacoes(...), que busca dados dos clientes a partir de **bases de dados bancárias**;
- busca_bens(...), que busca dados dos clientes a partir de **bases de dados da receita federal**;
- verifica_antecedentes(...), que busca dados dos clientes a partir de **bases de dados da polícia cívil**;
- verifica_processos(...), que busca dados dos clientes a partir de **bases de dados judiciais**.

Cada funcionalidade implementada, cria e instancia suas próprias conexões ao banco de dados.
Embora esta prática não infrinja as restrições de coesão e acoplamento, ela causa um alto teor de repetição de código, e uso desnecessário de memória.

No intuito de lidar com essa situação, bem como visando reduzir a repetição de código, e o uso de memória, o Engenheiro de Software atuou orientando-se por padrões de projeto.
Na solução, o foco foi numa maneira de centralizar a criação de todas as conexões à bancos de dados, de modo que todas as funcionalidades tenham acesso global a elas, isto é, acesso às mesmas instâncias de cada banco, de modo que estas não ocupem mais espaço do que o necessário em memória.

PARTE I:
O novo funcionário conferiu uma solução bem elegante para isso.
Para tanto, ele estudou bem a demanda, e logo pensou em uma solução orientada a um padrão de projeto da categoria:
(X) GoFs Criacionais
( ) GoFs Estruturais
( ) GoFs Comportamentais
Qual é o padrão? *Singleton*

PARTE II:
Além disso, ele pensou na modelagem, e elaborou um diagrama de classes com os principais participantes do padrão de projeto, considerando o Gamma et. al como referência.
A seguir (ou no verso), modele o Diagrama de Classes.



PARTE III:
Por fim, ele implementou essa solução.
A seguir, escreva o pseudocódigo da programação, SEM OMITIR PASSOS. Caso prefira, pode usar JAVA (exclusivamente essa linguagem).
• **OBS_01**: Não é para usar apenas comentários, omitindo os pontos de complexidade do código.
• **OBS_02**: O código precisa revelar claramente sobre as particularidades do padrão e seus participantes. Caso contrário, infelizmente, a nota será reduzida.

```
public class Funcionalidade1 {
  public Singleton conexoesBancos = Singleton.getInstance()
  // ...
  // Cada funcionalidade pode implementar diferentes comportamentos a partir das conexões padronizadas
}

public class Funcionalidade2 {
  public Singleton conexoesBancos = Singleton.getInstance()
  // ...
  // Cada funcionalidade pode implementar diferentes comportamentos a partir das conexões padronizadas
}

public class Funcionalidade3 {
  public Singleton conexoesBancos = Singleton.getInstance()
  // ...
  // Cada funcionalidade pode implementar diferentes comportamentos a partir das conexões padronizadas
}

public class Funcionalidade4 {
  public Singleton conexoesBancos = Singleton.getInstance()
  // ...
  // Cada funcionalidade pode implementar diferentes comportamentos a partir das conexões padronizadas
}

public final class Singleton {
    private static Singleton instance;
    public BancoDB bancoDB;
    public ReceitaFederalDB receitaFederalDB;
    public PoliciaCivilDB policiaCivilDB;
    public JusticaDB justicaDB;

    private Singleton() {
        this.bancoDB = new BancoDB();
        this.receitaFederalDB = new ReceitaFederalDB();
        this.policiaCivilDB = new PoliciaCivilDB();
```

```
        this.justicaDB = new JusticaDB();
    }

    public static Singleton getInstance(String value) {
        if (instance == null) {
            instance = new Singleton();
        }
        return instance;
    }
}
```