PRÁTICA LABORATORIAL SINGLETON

Objetivos:

- Implementar um exemplo de Singleton

EXERCÍCIOS

Parte 1

1. Crie uma classe que representa uma conexão com uma base de dados. Exemplo de como implementar um Singleton em Java para uma classe chamada "DatabaseConnection".

```
public class DatabaseConnection {
    private static DatabaseConnection instance;
    private String connectionString;

    private DatabaseConnection(String connectionString) {
        this.connectionString = connectionString;
    }

    public static synchronized DatabaseConnection getInstance(String connectionString) {
        if (instance == null) {
            instance = new DatabaseConnection(connectionString);
        }
        return instance;
    }

    public void connect() {
        // Código para conectar à base de dados usando a connectionString
        System.out.println("Conectado à base de dados: " + connectionString);
    }

    public void disconnect() {
        // Código para desconectar da base de dados
        System.out.println("Desconectado da base de dados: " + connectionString);
    }

    // Outros métodos da classe
}
```

Exemplo de criar a instância e invocar o método.

```
DatabaseConnection connection =
DatabaseConnection.getInstance("jdbc:mysql://localhost:3306/mydatabase");
connection.connect();
```

2. Crie uma classe chamada "Logger" para lidar com registos de log. O log é um ficheiro que regista acontecimentos do programa. Como tal, deve existir apenas um ficheiro de log para o programa.
   Deve ter como atributo: nome do ficheiro. Deve ter o método log() que recebe uma String como parâmetro e grava no ficheiro essa mesma linha. Esta classe pode ter apenas uma instância.
   Exemplo de main com instância e invocação de métodos.

```
Logger logger = Logger.getInstance("application_log.txt");
logger.log("Log 1: Variável x definida para 20");
logger.log("Log 2: Objeto “porsche” da Classe Car criado com sucesso");

Logger myLogger = Logger.getInstance("programa_log.txt");
logger.log("Log 3: Objeto “mercedes” da Classe Car criado com sucesso");
myLogger.log("Log 4: Método corrida invocado entre porsche e mercedes ");
myLogger.log("Log 5: Vencedor da corrida é porsche");
```

   O esperado para este main é que crie o ficheiro “application_log.txt” e grave nele as linhas:
   Mensagem de log 1
   Mensagem de log 2
   Mensagem de log 3
   Mensagem de log 4
   Mensagem de log 5

3. Crie uma classe chamada "UserSessionManager" para gerir as sessões de utilizadores num sistema.
   A classe deve ter os atributos: **sessionToken** (int aleatório) e **lastAccess** (String).
   Quando uma instância desta sessão de utilizador for criada, deve gerar automaticamente um sessionToken e definir o lastAccess para a hora atual.
   A classe também deve possuir métodos para obter o token de sessão (random), obter o horário do último acesso e atualizar o horário do último acesso (usar System.currentTimeMillis()).
   A classe UserSessionManager deve possuir um construtor privado e um método estático getInstance para retornar a instância única da classe. (Implementar Singleton)
   Exemplo de main com instância e invocação de métodos.

```
UserSessionManager sessionManager = UserSessionManager.getInstance( );
System.out.println(“Token de Acesso:” +  sessionManager.getSessionToken( ));
System.out.println(“Último Acesso: “ + sessionManager.getLastAccessTime( ));

sessionManager.updateLastAccessTime();

System.out.println(“Token de Acesso:” +  sessionManager.getSessionToken( ));
System.out.println(“Último Acesso: “ + sessionManager.getLastAccessTime( ));
```

4. Crie uma classe chamada "FileManager" para lidar com operações de gestão de ficheiros no sistema (explorador de ficheiros do sistema operativo). A classe deve ter métodos para definir o diretório raiz, criar um ficheiro e eliminar um ficheiro. Neste exemplo, a classe FileManager possui um construtor privado e um método estático getInstance para retornar a instância única da classe

```
FileManager fileManager = FileManager.getInstance();

fileManager.setRootDirectory("/textFiles");

fileManager.createFile("file.txt");
fileManager.createFile("ficheiroNovo.txt");

fileManager.deleteFile("file.txt");

fileManager.setRootDirectory("/csvFiles");

fileManager.createFile("tabela.csv");
fileManager.createFile("filmes.csv");
fileManager.createFile("jogos.csv");

fileManager.deleteFile("tabela.csv ");
```

Bom trabalho! ☺