Disciplina: POO

Sistema de Gerenciamento de Repositório Genérico:

Você está desenvolvendo um sistema de gerenciamento de dados para diferentes tipos de objetos. O objetivo é criar uma interface genérica para um repositório que pode armazenar, buscar e remover qualquer tipo de dado.

Sua tarefa é implementar um repositório genérico que funcione com diferentes tipos de entidades (por exemplo, Produtos, Clientes), garantindo que o sistema seja flexível e reutilizável

Especificações:

1. Interface Repositorio<T>:

Crie uma interface genérica chamada Repositorio<T> que declare os seguintes métodos:

- void salvar(T entidade): salva uma entidade no repositório.
- T buscar(int id): retorna a entidade com base no ID.
- void remover(int id): remove uma entidade com base no ID.
- List<T> listar(): retorna uma lista com todas as entidades armazenadas.

2. Classe RepositorioMemoria<T>:

Implemente a interface Repositorio<T> em uma classe chamada RepositorioMemoria<T>. Essa classe deve armazenar as entidades em uma lista na memória (use Arra**yList<T>). O ID de cada entidade deve ser gerado automaticamente ao salvar.**

3. Criação de Entidades:

Crie duas classes de entidades que o repositório irá gerenciar:

- Produto:
    - Atributos: id (int), nome (String), preco (double).
- Cliente:
    - Atributos: id (int), nome (String), email (String).

Essas classes devem implementar os métodos toString() e ter os métodos getters e setters apropriados.

4. Classe Main para Testes:

Crie uma classe Main com o método main(String[] args) onde você irá testar o repositório com as entidades Produto e Cliente. A classe deve:

- Criar instâncias de RepositorioMemoria<Produto> e RepositorioMemoria<Cliente>.
- Adicionar alguns produtos e clientes.
- Exibir todos os produtos e clientes armazenados.
- Buscar um produto ou cliente pelo ID e exibir o resultado.
- Remover um produto ou cliente e exibir a lista atualizada.

Exemplo de Código:

Interface Repositorio<T>:

```
import java.util.List;


public interface Repositorio<T> {
    void salvar(T entidade);
    T buscar(int id);
    void remover(int id);
    List<T> listar();
}
```

Implementação RepositorioMemoria<T>:

```
import java.util.ArrayList;
import java.util.List;

public class RepositorioMemoria<T> implements Repositorio<T> {
    private List<T> entidades = new ArrayList<>();
    private int contadorId = 0;

    @Override
    public void salvar(T entidade) {
        entidades.add(entidade);
        contadorId++;
    }

    @Override
    public T buscar(int id) {
        return entidades.get(id);
    }

    @Override
    public void remover(int id) {
        entidades.remove(id);
    }

    @Override
    public List<T> listar() {
```

```
        return entidades;
    }
}
```

Classe Produto:

```
public class Produto {
    private int id;
    private String nome;
    private double preco;

    public Produto(int id, String nome, double preco) {
        this.id = id;
        this.nome = nome;
        this.preco = preco;
    }

    public int getId() { return id; }
    public String getNome() { return nome; }
    public double getPreco() { return preco; }

    @Override
    public String toString() {
        return "Produto [id=" + id + ", nome=" + nome + ", preco="
+ preco + "]";
    }
}
```

Classe Cliente:

```
public class Cliente {
    private int id;
    private String nome;
    private String email;

    public Cliente(int id, String nome, String email) {
        this.id = id;
        this.nome = nome;
        this.email = email;
    }

    public int getId() { return id; }
    public String getNome() { return nome; }
    public String getEmail() { return email; }

    @Override
    public String toString() {
        return "Cliente [id=" + id + ", nome=" + nome + ", email="
+ email + "]";
    }
}
```

Classe Main para Testes:

```
public class Main {
    public static void main(String[] args) {
        Repositorio<Produto> repositorioProduto = new
RepositorioMemoria<>();
        Repositorio<Cliente> repositorioCliente = new
RepositorioMemoria<>();

        // Adicionar Produtos
        Produto p1 = new Produto(1, "Notebook", 3000.00);
        Produto p2 = new Produto(2, "Smartphone", 1500.00);
        repositorioProduto.salvar(p1);
        repositorioProduto.salvar(p2);

        // Adicionar Clientes
        Cliente c1 = new Cliente(1, "João", "joao@email.com");
        Cliente c2 = new Cliente(2, "Maria", "maria@email.com");
        repositorioCliente.salvar(c1);
        repositorioCliente.salvar(c2);

        // Listar Produtos
        System.out.println("Produtos:");
        repositorioProduto.listar().forEach(System.out::println);

        // Listar Clientes
        System.out.println("\nClientes:");
        repositorioCliente.listar().forEach(System.out::println);
```

```
        // Buscar Produto
        System.out.println("\nBuscar Produto com ID 1:");
        Produto produto = repositorioProduto.buscar(0); // índice 0 é o primeiro produto
        System.out.println(produto);

        // Remover Cliente
        System.out.println("\nRemover Cliente com ID 1:");
        repositorioCliente.remover(0); // Remove o primeiro cliente
        repositorioCliente.listar().forEach(System.out::println);
    }
}
```