EXERCÍCIO DE PROGRAMAÇÃO ORIENTADA A OBJETOS– DATA: _____/____/____
PROFESSORA: Ayla Rebouças

ALUNO: _____

Considere alguns trechos de classes referentes a um sistema comercial e o diagrama apresentado a seguir:

```
public abstract class Cliente {
	private String nome;
	private String endereco;
	private String email;

	public abstract String getId();

	public Cliente(String nome, String endereco, String email) {
		this.nome = nome;
		this.endereco = endereco;
		this.email = email;
	}	//...
```

```
public class ClientePF extends Cliente {
	private String CPF;

	public ClientePF(String nome, String endereco, String email, String CPF) {
		super(nome, endereco, email);
		this.CPF = CPF;
	}

	@Override
	public String getId() {
		return this.CPF;
	}
}
```

```
import java.util.Collection;

public interface SistemaComercial {
    boolean existeProduto(Produto produto);

    Produto pesquisaProduto(String codigoProduto)
            throws ProdutoNaoExisteException;

    boolean cadastraProduto(Produto produto);

    boolean existeCliente(Cliente cliente);

    Cliente pesquisaCliente(String id) throws ClienteNaoExisteException;

    Collection<Produto> pesquisaProdutosDaCategoria(CategoriaProduto categoria);
}
```

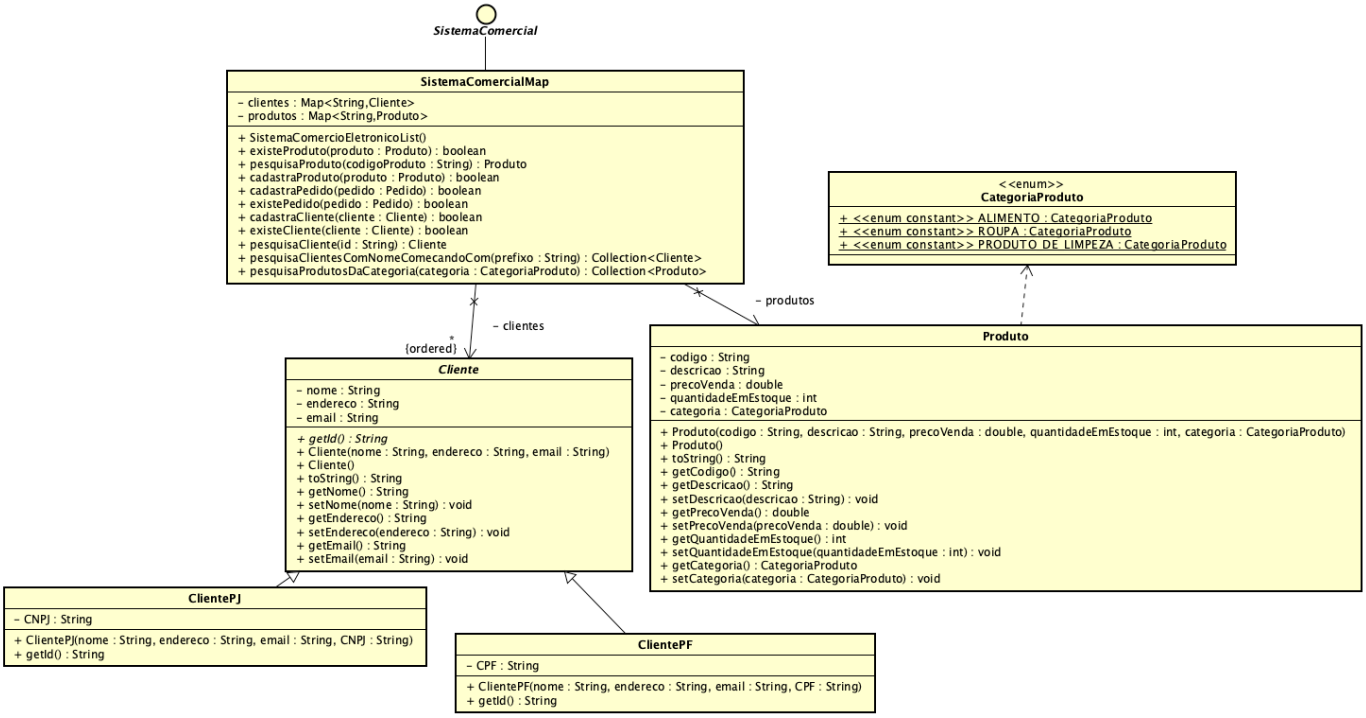
```
public class SistemaComercialMap implements SistemaComercial {
	private Map<String, Cliente> clientes;
	private Map<String, Produto> produtos;

	public SistemaComercialMap () {
		this.clientes = new HashMap<String, Cliente>();
		this.produtos = new HashMap<String, Produto>();
	}

	@Override
	public boolean existeProduto(Produto produto) {
		if (this.produtos.containsKey(produto.getCodigo())){
			return true;
		} else {
			return false;
		}
	}
```

```java
@Override
public Produto pesquisaProduto(String codigoProduto)
        throws ProdutoNaoExisteException {
        if (this.produtos.containsKey(codigoProduto)) {
                return this.produtos.get(codigoProduto);
        }
        throw new ProdutoNaoExisteException("Não foi encontrado produto"
                        +" com o código "+codigoProduto);
}


@Override
public boolean cadastraProduto(Produto produto) {
        if (existeProduto(produto)) {
                return false;
        } else {
                this.produtos.put(produto.getCodigo(),produto);
                return true;
        }
}
```



1. (2.0) Crie a classe `ClientePJ`, que herda da classe `Cliente`, sabendo que seu método `getId`() deve retornar o CNPJ do cliente pessoa jurídica.

2. (3.0) Implemente os métodos `existeCliente`, `pesquisaCliente e pesquisaProdutosDaCategoria` da classe `SistemaComercialMap` sabendo que a chave do Map de clients é o identificador do cliente, obtido por `getId`(). O método `pesquisaCliente` lança a exceção `ClienteNaoExisteException` caso não exista cliente com o Id indicado.

3. (3.0) Complemente o teste abaixo para a classe `SistemaComercialMap` que cadastre um produto da categoria `CategoriaProduto.ALIMENTO` e depois verifique que esse produto cadastrado existe e também deve verificar que a quantidade de produtos da categoria alimento é 1.

```
public class SistemaComercialMapTest {
	@Test
	public void testaCadastroProdutos() {
		SistemaComercialMap sistema = new SistemaComercialMap();
		Collection<Produto> alimentos =
			sistema.pesquisaProdutosDaCategoria(CategoriaProduto.ALIMENTO);
		assertTrue(alimentos.size()==0);
		//TODO
	}

}
```

4. (2.0) Você identifica algum método polimórfico no sistema comercial apresentado? Se sim, qual? Justifique sua resposta.