



Compartilhar o seu link com: [luciorocha @ professores.utfpr.edu.br](mailto:luciorocha@professores.utfpr.edu.br)

Nome / Link:

Michael Pariz Pereira: [Cópia de Aula 12 - POCO4A - 20/09/2022 - Exercícios propostos](#)

Fernando Rafael:

Matheus Henrique de A. Pereira:

[Cópia de Aula 12 - POCO4A - 20/09/2022 - Exercícios propostos](#)

Arnald: [Aula 12 - 20/09](#)

Ruan Perondi Urbanjos:

Ruan Mateus Trizotti:

Ale

Mariana Pedroso Naves: [Aula 12 - POO](#)

Exercícios propostos:

- 1) (Online) Acesse o link da atividade e faça as tarefas solicitadas:

<https://codeboard.io/projects/345870>

```
import java.util.Scanner;

public abstract class AVendas {

    protected float valor;

    public abstract void imprimir();

    public void ler(){
        Scanner entrada = new Scanner(System.in);
        this.valor = entrada.nextFloat();
    }
}

public class Vendas extends AVendas {

    public void imprimir(){
        System.out.println(this.valor);
    }
}
```

```

}

/**
 * TODO 1: Classe Principal: Instancie um objeto Floricultura
 * do tipo da interface IFloricultura no construtor da classe Principal.
 * TODO 2: Classe Principal: Acesse os metodos 'ler' e 'imprimir'
 * do objeto do tipo da interface IFloricultura.
 * TODO 3: Classe Principal: Instancie um objeto Vendas
 * do tipo da classe abstrata AVendas no construtor da classe Principal.
 * TODO 4: Classe Principal: Acesse os metodos 'ler' e 'imprimir'
 * do objeto do tipo da classe abstrata AVendas.
 */

public class Principal {

    public Principal(){

        //TODO1
        IFloricultura floricultura = new Floricultura();
        //TODO2
        System.out.println("LER:");
        floricultura.ler();

        System.out.println("IMPRIMIR:");
        floricultura.imprimir();

        //TODO3
        //Vendas vendas = new Vendas();
        AVendas vendas = new Vendas();
        //TODO4
        System.out.println("LER:");
        vendas.ler();

        System.out.println("IMPRIMIR:");
        vendas.imprimir();

    }

    public static void main(String[] args) {
        Principal p = new Principal();
    }
}

public interface IFloricultura {

    public abstract void imprimir();
    public abstract void ler();
}

```

```
}  
  
import java.util.Scanner;  
  
public class Floricultura implements IFloricultura {  
    private float valor;  
  
    public void imprimir(){  
        System.out.println(this.valor);  
    }  
  
    public void ler(){  
        Scanner entrada = new Scanner(System.in);  
        this.valor = entrada.nextFloat();  
    }  
}
```

2) Marque verdadeiro ou falso:

- a) (V) Uma interface é um tipo de dado abstrato com métodos abstratos.
- b) (V) Uma interface não possui implementação de seus métodos.
- c) (V) Uma classe concreta pode ter uma ou mais implementações de interfaces.
- d) (Vr) A classe concreta que implementa uma interface deve implementar todos os métodos da interface.
- e) (V) Uma classe abstrata é um tipo abstrato com pelo menos um método abstrato.
- f) (V) A classe deve ser explicitamente declarada abstract caso possua um método abstract.
- g) (V) Um objeto pode ter o tipo da classe abstrata, mas deve ser instanciado com uma classe concreta.

3) Observe a Figura 1 a seguir:

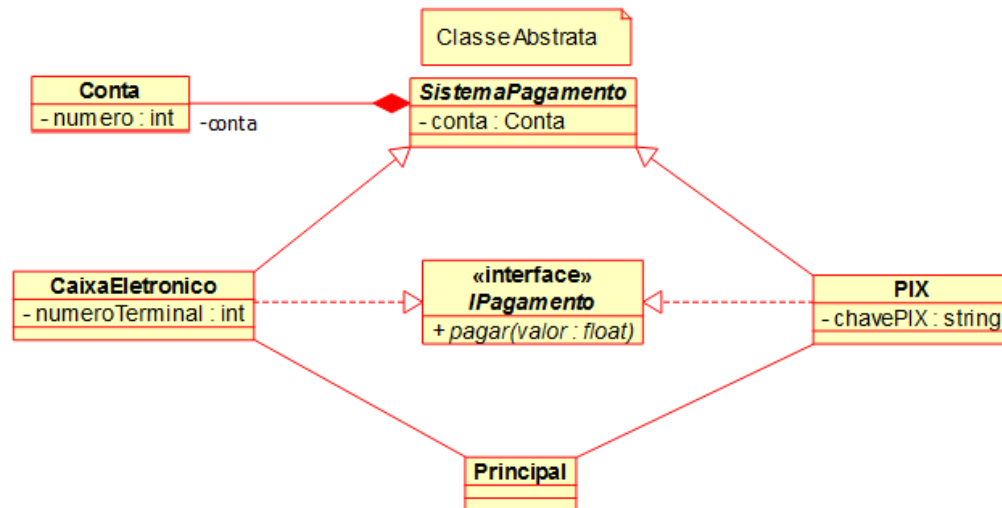


Figura: Diagrama de Classes.

- Implemente o diagrama de classes da figura com o construtor e métodos acessores e mutadores para as classes, com exceção da classe Principal.
- As classes Conta, CaixaEletronico e PIX não devem ser estendidas por nenhuma outra classe.
- A classe Principal deve ser capaz de instanciar 1 (um) objeto da Classe CaixaEletronico e 1 (um) objeto da Classe PIX.
- O método 'pagar(valor: float)' na Classe PIX deve receber o valor de entrada e acrescentar uma taxa de 10% sobre o valor de entrada.
- O método 'pagar(valor: float)' na Classe CaixaEletronico deve receber o valor de entrada e acrescentar uma taxa de 2% sobre o valor de entrada.
- Crie um menu que permita escolher o método de pagamento. O menu deve ser finalizado com a entrada 'X'.

```

public final class Conta {
    private int numero;
    public Conta(){
        this.numero = 0;
    }
    public int getNumero(){
        return this.numero;
    }
    public void setNumero(int numero){
        this.numero = numero;
    }
}
  
```

```
public abstract class SistemaPagamento {  
  
    private Conta conta;  
  
    public SistemaPagamento(){  
        this.conta = new Conta();  
    }  
    public Conta getConta(){  
        return this.conta;  
    }  
    public void setConta(Conta conta){  
        this.conta = conta;  
    }  
}
```

```
public interface IPagamento {  
  
    public abstract void pagar(float valor);  
}
```

```
public final class CaixaEletronico  
    extends SistemaPagamento  
    implements IPagamento {  
  
    private int numeroTerminal;  
  
    public CaixaEletronico(){  
        super(); //Invoca explicitamente o construtor da superclasse  
        this.numeroTerminal=0;  
    }  
    public CaixaEletronico(int numeroTerminal){  
        super();  
        setNumeroTerminal( numeroTerminal );  
    }  
  
    public void pagar(float valor){  
        valor += ( valor * 0.02);  
        System.out.println("PAGAR: " + valor);  
    }  
  
    public void setNumeroTerminal(int numeroTerminal){  
        this.numeroTerminal = numeroTerminal;  
    }  
    public int getNumeroTerminal(){  
        return this.numeroTerminal;  
    }  
}
```

```

public final class PIX
    extends SistemaPagamento
    implements IPagamento {

    private String chavePIX;

    public PIX(){
        super(); //Invoca explicitamente o construtor da superclasse
        this.numeroTerminal=0;
    }
    public void pagar(float valor){
        valor += (valor * 0.1f);
        System.out.println("PAGAR: " + valor);
    }

    public String getChavePIX(){
        return chavePIX;
    }
    public void setChavePIX(String chavePIX){
        this.chavePIX = chavePIX;
    }
}

```

```

import java.util.Scanner;
public class Principal {
    public static void main(String [ ] args){
        Principal principal = new Principal();
        principal.iniciar();
    }

    public void iniciar(){

        Scanner entrada = Scanner(System.in);

        int opcao=0;
        String continuar;
        float valor=0;
        do {
            System.out.println("Pagamento: 1) Caixa; 2) PIX");
            opcao = entrada.nextInt();
            switch( opcao ){
                case 1:
                    System.out.println("CaixaEletronico: pagar: ");
                    CaixaEletronico caixa = new CaixaEletronico(123);
                    valor = entrada.nextFloat();
                    caixa.pagar(valor);

```

```

        break;
    case 2:
        System.out.println("PIX: pagar: ");
        PIX pix = new PIX();
        pix.setChavePIX("CHAVEPIX");
        pix.pagar(valor);
        break;
    default:
        System.out.println("Opcao invalida.");
        break;
} //fim switch
System.out.println("Continuar? (X=Nao)");
continuar = entrada.next();

} while ( !continuar.equals("X") );

    entrada.close();
}
}

```

4) Implemente uma classe com múltiplas interfaces.

```

public interface IFloricultura {
    public abstract void imprimir();
}
public interface ILeitura {
    public abstract void ler();
}

public class Floricultura implements IFloricultura, ILeitura {

    public void imprimir(){
        System.out.println("FLORICULTURA");
    }
    public void ler(){
        Scanner entrada = new Scanner(System.in);
        System.out.println("Nome da floricultura: ");
        String nome = entrada.next();
        imprimir();
    }
}

public class Transportadora implements IFloricultura, ILeitura {

```

```
public void imprimir(){
    System.out.println(this.getClass().getSimpleName());
}
public void ler(){
    Scanner entrada = new Scanner(System.in);
    System.out.println("Nome da Transportadora: ");
    String nome = entrada.next();
    imprimir();
}
}
```

```
public final class Floricultura
    extends AFloricultura implements ILeitura {

    public void imprimir(){
        System.out.println("FLORICULTURA");
    }

    public void ler(){
        Scanner entrada = new Scanner(System.in);
        System.out.println("Nome da floricultura: ");
        String nome = entrada.next();
        imprimir();
    }
}
```

```
//Implementacao dos metodos com classes abstratas
public abstract class AFloricultura {
    public abstract void imprimir();
}
```