MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS APUCARANA
Bacharelado em Engenharia de Computação

**Disciplina:** Programação Orientada a Objetos **Turma:** POCO4A – 2022/2

**Professor:** Lucio Agostinho Rocha

## Lista de Exercícios 2 (DUPLA)

---

Observe o seguinte Diagrama de classes:

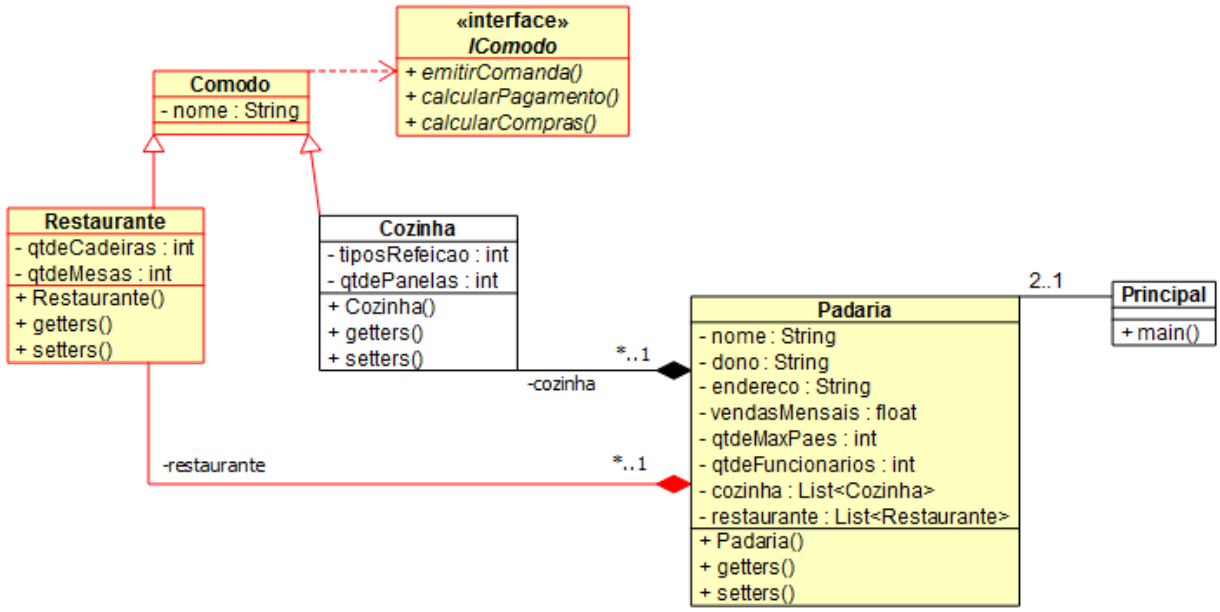


Figura 1 - Diagrama de Classes.

Desenvolva um programa orientado a objeto em linguagem de programação Java conforme segue:

**1) (1,0 ponto)** Utilize Polimorfismo para implementar na classe Principal uma lista dinâmica de objetos das classes Restaurante e Cozinha. A classe Principal deve exibir o estado desses objetos com a saída de todos os métodos acessores. Utilize o trecho a seguir.

```
//Classe Principal.java

List <Comodo> lista = new ArrayList<>();
lista.add(cozinha1);
lista.add(restaurante1);
lista.add(cozinha2);
lista.add(restaurante2);
```

**2) (1,0 ponto)** A partir do diagrama de classes da Figura 1, exiba na classe Principal a saída dos métodos da interface IComodo, da seguinte forma:

| | Restaurante | Cozinha |
|---|---|---|
| emitirComanda() | Recebe um pedido | Recebe um pedido do Restaurante e faz uma refeição. |
| calcularPagamento() | Recebe um valor da Cozinha e retorna este valor acrescido de 10% do valor do pedido. | Recebe um pedido do Restaurante e retorna o valor da refeição. |
| calcularCompras() | Emite aviso de compras se foram feitos pelo menos 10 pedidos. | Emite aviso de compras se foram feitas pelo menos 15 refeições. |

**3) (1,0 ponto)** A partir do diagrama de classes da Figura 1, defina a classe Comodo como uma classe abstrata. Acrescente nesta classe Comodo 5 (cinco) métodos abstratos e imprima a saída dos objetos Restaurante e Cozinha na classe Principal.

**4) (1,0 ponto)** Acrescente ao programa a forma de pagamento, da seguinte forma:

a) Crie 3 (três) classes não relacionadas por herança: BitCoin, Euro e Real. Crie a interface IMoeda com um método abstrato. Cada classe deste item deve implementar essa interface.

b) Na classe Principal crie uma lista dinâmica com objetos de cada uma das classes do item a). Exiba polimorficamente a lista invocando o método do item b) de cada objeto da lista.

**5) (1,0 ponto)** Modifique o programa dos itens 1 até o item 4 para que todas as classes sejam classes internas da classe Principal.

**6) (1,0 ponto)** A partir do diagrama da Figura 2 a seguir, implemente como segue:

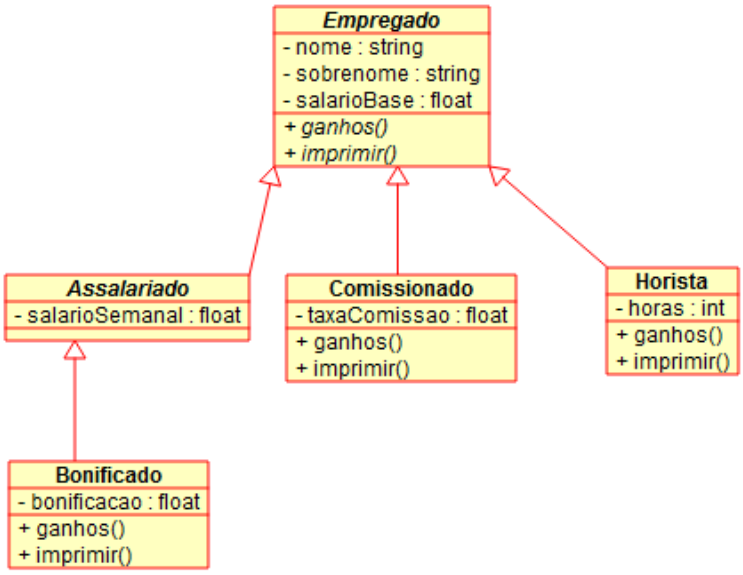


Figura 2: Diagrama de Classes: Folha de Pagamento.

a) As Classes Empregado e a Classe Assalariado são classes abstratas que possuem apenas métodos abstratos. Crie uma Classe que instancie objetos das classes

derivadas da Classe raiz da hierarquia apresentada. Utilize o trecho de código a seguir:

```
Bonificado b1 = new Bonificado("Joao","Silva", salarioBase, bonificacao, salarioSemanal);

Comissionado c1 = new Comissionado("Maria","Soares", salarioBase, taxaComissao);

Horista h1 = new Horista("Jomar","Silva Soares", salarioBase, horas);
```

b) Utilize polimorfismo para exibir a saída dos métodos públicos das classes derivadas que podem ser instanciadas. Utilize o trecho de código a seguir:

```
for ( Empregado emp : lista ) {
     empregado.imprimir();
     empregado.ganhos();
}
```

**7) (1,0 ponto)** Um aluno de Engenharia de Computação deseja implementar uma calculadora simples com polimorfismo de Interface. Para isso, cada operação é implementada por uma Classe: Soma, Subtração, Divisão e Multiplicação. A Interface 'IOperacoes' é implementada por essas Classes e possui os seguintes métodos abstratos:

```
void setOperando1(float operando1);   //Define o valor do primeiro operando
void setOperando2(float operando2);   //Define o valor do segundo operando
float getResultado();                 //Retorna o resultado da operação
String getNome();                     //Retorna o nome da operação
int getQuantidade();                  //Retorna a quantidade de instâncias da classe
```

Utilize polimorfismo para exibir a saída dos métodos acessores dos objetos instanciados na Classe Principal.

**8) (1,0 ponto)** Utilize herança para criar uma superclasse de exceção (chamada ExceptionA) e subclasses de exceção ExceptionB e ExceptionC, em que ExceptionB herda de ExceptionA e ExceptionC herda de ExceptionB. Escreva um programa para demonstrar que o bloco catch para tipo ExceptionA captura exceções de tipos ExceptionB e ExceptionC.

**9) (1,0 ponto)** Escreva um programa que demonstra como várias exceções são capturadas com
catch (Exception exception). Desta vez, defina as classes ExceptionA (que herda da classe Exception) e ExceptionB (que herda da classe ExceptionA). Em seu programa, crie blocos try que lançam exceções de tipos ExceptionA, ExceptionB, NullPointerException e IOException. Todas as exceções devem ser capturadas com blocos catch para especificar o tipo Exception.

**10) (1,0 ponto)** Na Plataforma de Ensino, Acesse o Link do Exercício 10 desta lista:

a) (0,1 ponto) Responda à postagem anterior da seguinte forma: informe, no início da postagem e antes do código-fonte, em um comentário de bloco, o nome completo dos membros da sua equipe.
b) (0,2 ponto) Informe a seguir, antes do código, em um comentário de bloco, se o exemplo de entrada e a saída informados na postagem anterior pela outra equipe está correta. Caso não esteja, informe o motivo.
c) (0,5 ponto) Informe a seguir, antes do código, em um comentário de bloco, a nova INTERFACE acrescida pela sua equipe. Acrescente a nova INTERFACE ao programa da postagem anterior.
d) (0,2 ponto) Informe a seguir, antes do código-fonte, em um comentário de bloco, um exemplo de entrada e a saída do seu programa.