Compartilhar o seu link com: luciorocha@professores.utfpr.edu.br

Nome: Link:
Guilherme Ramalho Cópia de POCO4A - Aula 6 - Exercícios
Felipe Lorusso <Aula 06>
Luis Mendes Cópia de POCO4A - Aula 6 - Exercícios
Pedro Reis e Thiago Tieghi Cópia de POCO4A - Aula 6 - Exercícios
Angélica < Cópia de Aula 6 POCO4A Exercícios >
Deivid da Silva Galvão < POCO4A - Aula 6 - Exercícios >
Daniel Martins de Carvalho < Cópia de POCO4A - Aula 6 - Exercícios >
Marcos Tadao Shoji < Cópia de POCO4A - Aula 6 - Exercícios >
Julio Farias: Cópia de POCO4A - Aula 6 - Exercícios
Rafael Kendy Naramoto Lopes < Cópia de POCO4A - Aula 6 - Exercícios >
Thales Alves < POCO4A - Aula 6 - Exercícios >
Henrique Cois < Cópia de POCO4A - Aula 6 - Exercícios >
Vitor Luiz de Castro Viana: Cópia de POCO4A - Aula 6 - Exercícios
Filipe Augusto Parreira Almeida < Cópia de POCO4A - Aula 6 - Exercícios >
Plinio / João Pedro Cavani Meireles < Cópia de POCO4A - Aula 6 - Exercícios >
Gabriel Takeshi < Cópia de POCO4A - Aula 6 - Exercícios >
Vitor Hugo Leite Arruda de Oliveira < Cópia de POCO4A - Aula 6 - Exercícios >
Rodrigo Leandro Benedito: Cópia de POCO4A - Aula 6 - Exercícios
Roberto Furlani Neto < Cópia de POCO4A - Aula 6 - Exercícios >
Bruno Keller Margaritelli < Bruno - POCO4A - Aula 6 - Exercícios >
Matheus Hirata Vanzela && Thiago Cristovão < Cópia de POCO4A - Aula 6 - Exercícios >
Matheus Mazali Maeda and Alexandre Aparecido and Mabylly <
Cópia de POCO4A - Aula 6 - Exercícios >
João Vitor N. Yoshida < Cópia de POCO4A - Aula 6 - Exercícios >
Felipe Antonio Magro: Cópia de POCO4A - Aula 6 - Exercícios
Raphael Uematsu Cópia de POCO4A - Aula 6 - Exercícios

Atividade Quizz:

```
//Exemplo com palavra-reservada ‘static’
//Exemplo com palavra-reservada ‘this’

public class Calculadora {
```

```
    //Variavel de classe
    public static int QUANTIDADE_BOTOES = 20;
    //Variavel de instancia
    public int COR;
    private int MARCA;

    public int getMarca(){  return this.MARCA; }

}
public class Principal {

       public Principal(){

            Calculadora calculadora = new Calculadora();
            System.out.println(calculadora.COR); //Objeto
            System.out.println(calculadora.getMarca()); //Objeto

            System.out.println(Calculadora.QUANTIDADE_BOTOES); //Classe

      }

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

}
```

1) (Online) Acesse o link: https://codeboard.io/projects/343077
   a) Realize as atividades do link.

```
//Julio Farias

/**
 * DONE1: Classe Principal: Renomeie a classe Main para ter o mesmo nome do arquivo.
 * DONE2: Classe Principal: Instancie 3 (três) objetos da classe Estudante.
 *        Cada objeto deve ser instanciado com um construtor diferente do anterior.
 * DONE3: Classe Estudante: crie um método 'imprimir' que retorne a idade do Estudante.
 * DONE4: Classe Estudante: crie um método sobrecarregado 'imprimir' que retorne
 *        a idade e o nome do Estudante.
 * DONE5: Classe Estudante: crie um método sobrecarregado 'imprimir' que retorne
 *        a idade e o nome e o CPF do Estudante.
 */



public class Principal {

   public static void main(String[] args) {
      //System.out.println("Hello World");
      Estudante joao = new Estudante("joao");

      Estudante maria = new Estudante("maria",20);

      Estudante fulano = new Estudante("fulano", 19, "545646531");

      System.out.println(maria.imprimir());
      System.out.println(fulano.imprimir());
      System.out.println(fulano.imprimir());

   }
}

—----------------------------------------------------------

public class Estudante {
   private String nome;
   private int idade;
   private String CPF;

   public Estudante(String nome){
      this.nome=nome;
      this.idade=0;
```

```
      this.CPF="";
   }

   public Estudante(String nome, int idade){
      this.nome=nome;
      this.idade=idade;
      this.CPF="";
   }
   public Estudante(String nome, int idade, String CPF){
      this.nome=nome;
      this.idade=idade;
      this.CPF=CPF;
   }
   public int imprimir(){
      return this.idade;
   }

   public String imprimir(String nome, int idade){
      return this.nome + "" + this.idade;
   }

   public String imprimir(String nome, int idade, String CPF){
      return this.nome + "" + this.idade + "" + this.CPF;
   }

}
```

_____ Séfora e Carlos _____

```
   public String toString(int idade){
      return this.idade + "";

   }
```

Vitor Hugo:

```
public imprimir(String nome) { //pelo menos 1 ou 2 parâmetro(s)?
      return this.nome+" "+this.idade;
   }
```

2) (Netbeans) Composição: Implemente o diagrama UML de classes da Figura 1:

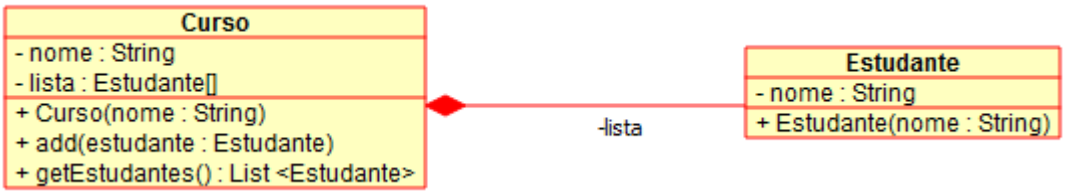

Figura 1 - Diagrama UML de Classes.

```
//Curso TEM UMA lista de Estudante

import java.util.ArrayList;

public class Curso {
  private String nome;
  private ArrayList<Estudante>  lista1;   //Lista dinamica
  private Estudante [ ] lista2;     //Lista estatica

  public Curso(String nome) {
        this.nome = nome;
        lista1 = new ArrayList<>();  //Lista dinamica
        lista2 = new Estudante[10];  //Lista estatica
  }

 public void add(Estudante estudante){
        lista1.add( estudante );

        //lista2[ 0 ] = estudante;

  }
  public ArrayList<Estudante> getEstudantes() {
        return this.lista1;
  }

}

//
public class Estudante {
    private String nome;

    public Estudante(String nome){ this.nome = nome; }
}
```

```
public class Principal {

   public Principal(){

         Curso poo = new Curso(“POO”);
         Estudante   joao   = new Estudante(“JOAO”);
         Estudante   carlos = new Estudante(“CARLOS”);

         poo.add( joao );
         poo.add( carlos );

         ArrayList<Estudante>  lista = poo.getEstudantes();
         System.out.println( lista.size() );

         Iterator item = lista.iterator();
         while( item.hasNext() )
               System.out.println( item.next() );

  }


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

   }


}
```

3) (NetBeans) Composição: Implemente o diagrama UML de classes da Figura 2:

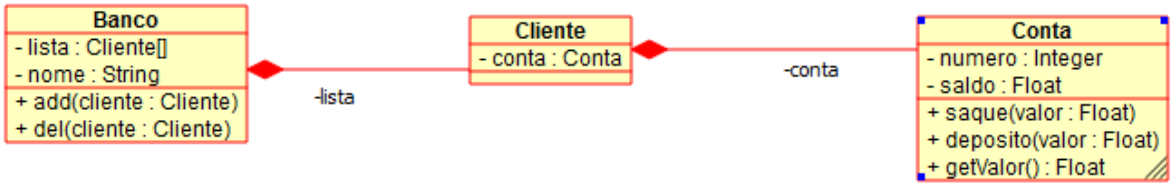

Figura 2 - Diagrama UML de Classes.

```
public class Banco {
    private ArrayList<Cliente>  lista;
    private String nome;

    public void add(Cliente cliente){
          this.lista.add( cliente );
    }
    public void del( Cliente cliente ){
        this.lista.remove( cliente );
    }
}
public class Cliente {
   private Conta conta;
}
public class Conta {
   private Integer numero;          //Classe encapsuladora de tipo (wrapper)
   private Float saldo;             //Integer → int
                                    //Float → float
                                    //Float.parseFloat(String str)
                                    //Integer.parseInt(String str)

   public void saque(Float saque){

   }



}
```

4) (NetBeans) Composição: Implemente o diagrama UML de classes da Figura 3:

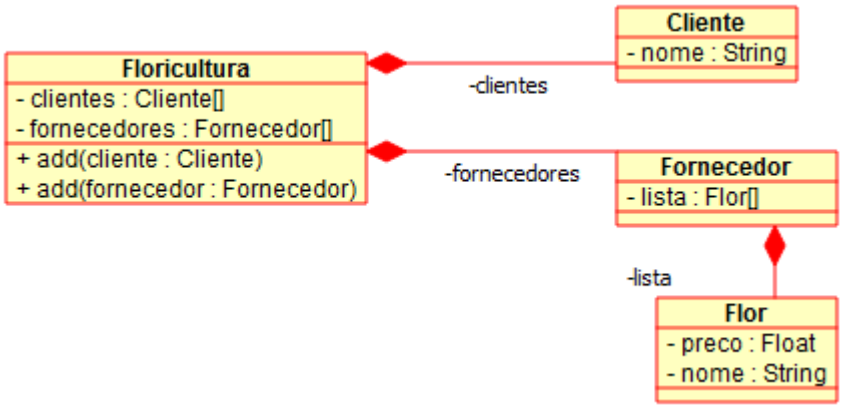

Figura 3 - Diagrama UML de Classes.

5) (NetBeans) A partir do enunciado, identifique:

	Uma transportadora possui veículos de dois tipos: van e caminhão. A transportadora faz entregas de dois tipos de encomendas: normal e expresso.
Classes	
Atributos	
Comportamentos	

6) (NetBeans) A partir do enunciado, identifique:

	Uma estrutura de dados do tipo pilha possui uma lista de dados do tipo inteiro. A pilha é do tipo FILO (First-In, Last-Out). As operações da pilha são: inserir (push) e remover (pop).
Classes	
Atributos	
Comportamentos	