Programação Orientada a Objeto

INSTITUTO FEDERAL
Triângulo Mineiro
Campus Paracatu

Herança

Lista 03

Professor: Edwar Saliba Júnior

Exercício de Codificação:

1. Crie um *software* para controle de Veículos. Este *software* deverá possuir as seguintes classes: Veículo, com os seguintes atributos: marca (*String*), modelo (*String*), anoFabricacao (*int*) e anoModelo (*int*) e preço (*double*). Deverá ter também as classes: Carro (que herda de Veículo), com seguintes atributos: quantidadePortas (int) e a classe Caminhão (que também herda de Veículo) com o seguinte atributo capacidadeDeCarga (*float*). Crie um CRUD[1] para cadastro dos carros e caminhões em ArrayList's separados. Faça uso da classe de gerenciamento de objetos (Fichário).

2. Necessita-se de *software* para controle de funcionários e clientes. Este *software* deverá possuir as seguintes classes: Pessoa, com os seguintes atributos: cpf (*String*), nome (*String*), identidade (*String*). Funcionário (que herda de Pessoa), com seguintes atributos: salario (*float*), e ctps[2] (String). Também deverá possuir a classe Cliente (que herda de Pessoa), com os seguintes atributos: limiteDeCredito (*float*), valorDaCompra (float), dataDaCompra (LocalDate), dataLimiteParaPagamento (LocalDate), dataDePagamento (LocalDate). Crie um CRUD para cadastro dos clientes e funcionários em ArrayList's separados. Faça uso da classe de gerenciamento de objetos (Fichário). Na classe Cliente, além dos métodos get's e set's, crie também um método "calculaJuros" terá o tipo de retorno *float*. Este método deverá fazer o seguinte cálculo e retorná-lo:

   valorFinal = (("data de pagamento" – "data da compra") * 0.03) * valorDaCompra;

   Nos menus que você criar para o CRUD de cliente, coloque também uma opção para a impressão do valor final a ser pago pelo cliente baseado nas informações contidas no cadastro do mesmo. Ao imprimir o valor final a ser pago, imprima também a o "valor da compra", a "data da compra", a "data limite para pagamento" e a "data de pagamento" para que se possa averiguar se o cálculo ficou correto.

3. Em um Sistema de Gestão Hospitalar, considere as classes Médico e Paciente. As classes possuem características em comum, tais como: CPF, identidade, nome, endereço e telefone. As seguintes operações são comuns a ambas classes: cadastro, alteração e recuperação de dados. Entretanto,

1 CRUD – são as 4 operações básicas de um banco de dados (CREATE, RETRIEVE, UPDATE e DELETE), porém aqui se refere as operações de: inclusão, exclusão, alteração, consulta e relatório dos objetos que serão armazenados em um objeto do tipo Collection de Java.

2 CTPS – Carteira de Trabalho e Previdência Social

Médico e Paciente têm características diferentes. Por exemplo, Médico possui CRM e Especialidade, e paciente está associado a um convênio (Figura 1).

Implemente as classes do modelo proposto.

Na classe principal:

- crie uma estrutura de *menus* capaz de dar suporte, as operações de cadastro, alteração, exclusão e consulta, para toda esta estrutura que você criou;
- seu *software* deverá ser capaz de manipular quantos registros o usuário desejar cadastrar, para isto use *ArrayList*;
- não se esqueça de testar seu *software* para certificar-se de que tudo está funcionando.

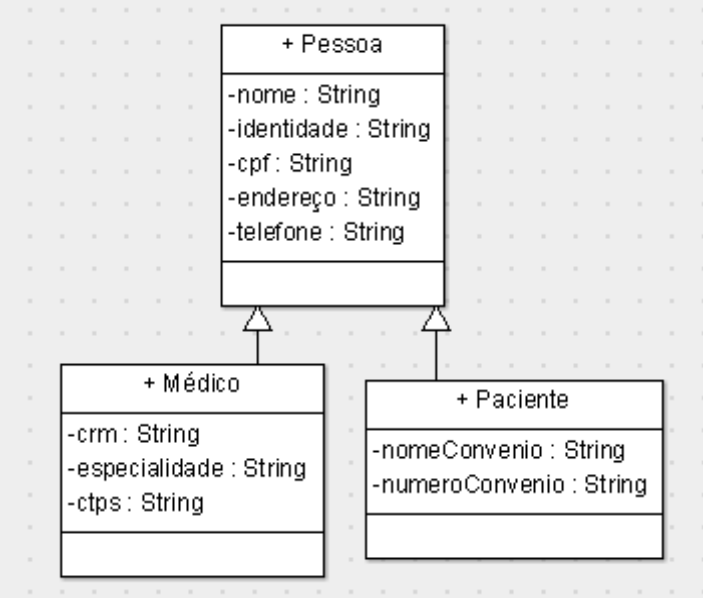

Figura 1: Gestão Hospitalar - Herança

Exercício de Depuração:

Mostre os 5 valores que serão impressos, na tela do computador, pelo código-fonte a seguir:

====[CalculoSimples]==========

```
package poo_lista_03_depuracao;

public class CalculoSimples {
   private float x;
   protected float y;
   protected float z;

   public CalculoSimples(float x, float y, float z) {
      this.x = x;
      this.y = y;
      this.z = z;
   }

   public float calcSimples01(){
      return x - 4 * y + z;
   }

   public float calcSimples02(){
      return (x - 2) - 2 * y;
   }

   public float calcSimples03(){
      return x - 2 * y + z;
   }

   public float getX() {
      return x;
   }

   public void setX(float x) {
      this.x = x;
   }

   public float getY() {
      return y;
   }

   public void setY(float y) {
      this.y = y;
   }

   public float getZ() {
      return z;
   }

   public void setZ(float z) {
      this.z = z;
   }
}
```

====[CalculoComplexo]==================================

```java
package poo_lista_03_depuracao;

public class CalculoComplexo extends CalculoSimples {

    private float num1;
    private float num2;
    private float total;
    private float valor00;
    private int valor01;
    private int valor02;
    private int valor03;

    public CalculoComplexo(int ad, int am, int aa){
        super(aa, ad, am);

        num1 = 0;
        num2 = 0;
        total = 0;
        valor01 = ad;
        valor02 = am;
        valor03 = aa;
        valor00 = valor03 * valor02 - super.calcSimples02() - 2850;
    }

    public float getTotal(){
        return(total);
    }

    public void setNum1(float v1){
        num1 = v1;
    }

    public void setNum2(float v1){
        num2 = v1;
    }

    public float calc01(float v1, float v2){
        float res;
        res = v1 - y + valor00;
        return res - 5;
    }

    public float calc02(){
        float n1 = super.calcSimples03();
        total = 0;
        for(int i = 1; i <= 2; i++){
            num1++;
            z--;
            num2 -= 2;
            if(super.getY() > 2)
                total += num1 + z - valor00;
            else
                total += z;
        }
        total += n1;

        return(total);
    }

    public void calc03(){
```

```
        float n2 = super.calcSimples03();
        if(num2 % num1 == 3)
            total = valor01 - n2;
        else
            total = valor00 + y - super.getX();
    }
}
```

====[POO_Lista_03_Depuracao]==========

```
package poo_lista_03_depuracao;

public class POO_Lista_03_Depuracao {

    public static void main(String[] args) {
        CalculoComplexo ca;

        float valor1 = 15,
              valor2 = 2,
              resultado = 0;

        int i = 1, v1 = 18, v2 = 5, v3 = 1234;
        ca = new CalculoComplexo(v1, v2, v3);

        while(i < 3){
            resultado += ca.calc01(valor1, i);
            i++;
        }

        System.out.println("\n\nCálculo 01: " + (i % 2 == 0 ? resultado + 1 : resultado + 2));

        ca.setNum1(valor1);
        ca.setNum2(valor2);
        resultado = ca.calc02();

        System.out.println("\n\nCálculo 02: " + resultado);

        ca.setNum1(valor1);
        ca.setNum2(valor2);
        ca.calc03();
        resultado = ca.getTotal();

        System.out.println("\n\nCálculo 03: " + resultado);

        ca.setNum1(resultado);
        ca.setNum2(18);
        ca.calc03();
        resultado = ca.getTotal();

        System.out.println("\n\nCálculo 04: " + resultado);

        ca.setNum1(10);
        ca.setNum2(20);
        ca.calc02();
        resultado = ca.getTotal();

        System.out.println("\n\nCálculo 05: " + resultado + "\n\n");
    }
}
```