FTCE

Faculdade de Tecnologia e Ciências Exatas

USJT – 2016/2 - Programação de Computadores - Laboratório de Exercícios

Professores: Celso, Hamilton e Vaz

**Aula: 07**
Assunto: Herança

**Exemplo Resolvido:** Junto com seu professor, copie o código, cole no JGrasp; analise o código e entenda-o, depois execute-o para ver os resultados.

Desenvolver a hierarquia de classes a seguir (autor: Prof. Fulvio Prevot)

I. Escrever, em linguagem Java, a classe Ponto, cujos atributos únicos são coordenadaX (double) e coordenadaY (double). Escrever, nesta classe:

a) Um método construtor sem parâmetros, que armazena o valor 0.0 nos atributos;
b) Um método construtor que inicia os atributos com valores passados por meio de parâmetros;
c) Métodos de acesso e métodos modificadores para cada atributo;
d) Um método que retorna, em uma String, os valores armazenados nos atributos.

II. Escrever, em linguagem Java, a classe Circulo, como extensão da classe Ponto (Herança). Acrescentar à classe Circulo o atributo raio (double), que não pode receber valores negativos. Escrever, nesta classe:

a) Um método construtor sem parâmetros, que inicia o atributo raio com o valor 1.0;
b) Um método construtor que inicia os atributos (próprios e herdados) com valores passados por meio de parâmetros;
c) Método de acesso e método modificador para o atributo raio;
d) Um método que calcula e retorna o valor do diâmetro do Circulo;
e) Um método que calcula e retorna o perímetro do Circulo;
f) Um método que calcula e retorna a área do Circulo;
g) Um método que retorna, em uma String, os valores armazenados nos atributos (próprios e herdados).
Dados: perímetro do circulo = Math.PI * diâmetro
área do círculo = Math.PI * raio * raio diâmetro = 2 * raio

III. Escrever a classe Cilindro, como extensão da classe Circulo. Acrescentar à classe Cilindro, o atributo altura (double), que não pode receber valores negativos. Escrever, nesta classe:

a) Um método construtor sem parâmetros, que inicia o atributo altura com o valor 1.0;

b) Um método construtor que inicia os atributos (próprios e herdados) com valores passados por meio de parâmetros;
c) Método de acesso e método modificador para o atributo altura;
d) Um método que calcula e retorna o volume do Cilindro;
e) Um método que calcula e retorna a área da superfície externa do Cilindro;
f) Um método que retorna, em uma String, os valores armazenados nos atributos (próprios e herdados).
Dados: Volume do Cilindro = área da base * altura
Área da superfície do cilindro = 2 * área da base + área lateral

IV. Escrever uma classe com o método main(), com a finalidade de testar os métodos da hierarquia de classes criada nos itens anteriores. Os dados de entrada podem ser fixos no código e os de saída podem ser mostrados via System.out.println().

```
public class Ponto{
   private double coordenadaX, coordenadaY;

   public Ponto(){
      coordenadaX = 0.0;
      coordenadaY = 0.0;
   }
   public Ponto(double x, double y){
      coordenadaX = x;
      coordenadaY = y;
   }
   public double getCoordenadaX(){
      return coordenadaX;
   }
   public double getCoordenadaY(){
      return coordenadaY;
   }
   public void setCoordenadaX(double x){
      coordenadaX = x;
   }
   public void setCoordenadaY(double y){
      coordenadaY = y;
   }
   public String toString(){
      return "Ponto:[coordenadaX="+coordenadaX+
         "][coordenadaY="+coordenadaY+"]";
   }
}
```

```
public class Circulo extends Ponto{
   private double raio;

   public Circulo(){
      super();
      raio = 1.0;
   }
```

```
   public Circulo(double r, double x, double y){
      super(x,y);
      setRaio(r);
   }

   public void setRaio(double r){
      if(r >= 0){
         raio = r;
      }
   }

   public double getRaio(){
      return raio;
   }

   public double diametro(){
      return 2*raio;
   }

   public double perimetro(){
      return 2*Math.PI*raio;
   }

   public double area(){
      return Math.PI*raio*raio;
   }

   public String toString(){
      return "Circulo:[raio="+raio+"]"+super.toString();
   }
}
```

```
public class Cilindro extends Circulo{
   private double altura;

   public Cilindro(){
      altura = 1.0;
   }

   public Cilindro(double a, double r, double x, double y){
      super(r,x,y);
      setAltura(a);
   }

   public double getAltura(){
      return altura;
   }
```

```
    public void setAltura(double a){
        if(a >= 0){
            altura = a;
        }
    }

    public double volume(){
        return area()*altura;
    }

    public double areaDaSuperficieExterna(){
        return (2*area())+(perimetro()*altura);
    }

    public String toString(){
        return "Cilindro:[altura="+altura+"]"+super.toString();
    }
}
```

```
public class Teste{
    public static void main(String[] args){
        Ponto ponto = new Ponto();
        System.out.println(ponto.toString());
        Circulo circulo = new Circulo();
        System.out.println(circulo.toString());
        circulo = new Circulo(3.0, -1.0, 2.0);
        System.out.println(circulo.toString());
        Cilindro cilindro = new Cilindro(4.0, 3.0, -1.0, 2.0);
        System.out.println(cilindro.toString());
    }
}
```

**Problemas Propostos:**

**Exercícios iniciais: valor 0,5 ponto**

Resolva os exercícios desta seção para conquistar 0,5 ponto

1) Desenvolver a hierarquia de classes abaixo. (Prof. Fulvio Prevot)

a) Escrever, em linguagem Java, a classe Pagamento, cujos atributos únicos são nomeDoPagador (String), cpf (String) e valorASerPago (double). Escrever, nesta classe, métodos construtores, métodos de acesso e métodos modificadores para os atributos.

b) Escrever a classe CartaoDeCredito, como extensão da classe Pagamento, com o atributo próprio numeroDoCartao (String). Escrever, nesta classe, métodos construtores, métodos de acesso e métodos modificadores para o atributo próprio.

**Exercícios intermediários: valor 0,5 ponto**
Resolva os exercícios desta seção para conquistar mais 0,5 ponto

c) Escrever a classe Cheque, como extensão da classe Pagamento, com o atributo próprio numeroDoCheque (String). Escrever, nesta classe, métodos construtores, métodos de acesso e métodos modificadores para o atributo próprio.

d) Escrever a classe Boleto, como extensão da classe Pagamento, com os atributos próprio numeroDoBoleto (String), dia (int), mes (int) e ano (int) de vencimento. Escrever, nesta classe, métodos construtores, métodos de acesso e métodos modificadores para o atributo próprio.

**Exercícios complementares (para praticar)**
Resolva os exercícios desta seção para aprimorar seus conhecimentos

2 ) Com base nas classes desenvolvidas nas aulas de Teoria e no Dojo (Empregado, Mensalista, Comissionado, Horista e Tarefeiro), faça (Prof. Fulvio Prevot):

a) A superclasse PessoaFisica, que tem como atributos nome (String), sobrenome (String) e cpf (String). Escrever 2 métodos construtores, um deles sem parâmetro, que deve iniciar os atributos com valores padrões definidos pelo programador, e o outro, que inicia os atributos por meio de parâmetros. Escrever também métodos de acesso e métodos modificadores para os atributos desta classe. A seguir, escrever o método dados() que retorna, em ma String, os valores armazenados nos tributos.

b) A classe Desempregado como extensão (especialização) da superclasse PessoaFisica. A classe Desempregado deve ter o atributo seguroDesemprego (double), que armazenará o valor (em reais) do seguro-desemprego recebido. Escrever também, nesta classe, 2 métodos construtores, um deles sem parâmetro, que deve iniciar o atributo próprio da classe com o valor 0.0, e o outro, que inicia os atributos próprios e herdados por meio de parâmetros.

c) Escrever também métodos de acesso e métodos modificadores para os atributos próprios desta classe. A seguir, escrever o método dados() que retorna, em uma String, os valores armazenados nos atributos próprios e herdados, criado a partir da modificação do método dados() herdado da classe PessoaFisica (uso da sobrecarga).

d) Fazer as modificações necessárias no código-fonte da classe Empregado, de modo que ela passe a ser uma extensão da classe PessoaFisica, e as mudanças necessárias nas classes Mensalista, Horista, Tarefeiro e Vendedor, por causa das alterações feitas nos itens anteriores e feitas na classe Empregado.

e) Fazer as alterações e inclusões necessárias na classe que contém o método main(), de modo que se possa efetuar testes dos métodos das novas classes e das possíveis mudanças ocorridas nas classes que já faziam parte da hierarquia.

**Bibliografia**

LOPES, ANITA. GARCIA, GUTO. Introdução à Programação: 500 algoritmos resolvidos. Rio de Janeiro: Elsevier, 2002.

DEITEL, P. DEITEL, H. Java: como programar. 8 Ed. São Paulo: Prentice – Hall (Pearson), 2010.