Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Programação de Computadores II - CSI032

# Lista de Exercicio VI

Aluno: Samira Resende Freitas
Matrícula: 17.1.8237
Professor: Rafael Alexandre
Curso: Engenharia de Computação

João Monlevade
Setembro
2020

# Conteúdo

# 1 Classes Abstratas

1. Explique com suas palavras por que uma classe abstrata não pode ser instanciada.
   Classes abstratas, basicamente, são classes que servem de modelo genérico para outras classes. Devido a isso elas não podem ser instanciadas e são utilizadas apenas para que classes filhas herdem os seus atributos/propriedades e métodos.Elas podem ter métodos abstratos ou não abstratos.Os métodos abstratos não podem ter corpo, ou seja, deve-se declarar apenas a assinatura do método e eles obrigatoriamente terão que ser implementados na classe herdeira.

2. Explique, com suas palavras, por que interfaces não podem ter construtores.
   Por definição, a interface é um "contrato", um compromisso comportamental que um determinado objeto deve ter. Por isso, não faz sentido atribuir um construtor à uma interface, pois seria equivalente a colocar no contrato uma série de definições de criação,um aspecto não comportamental, e extremamente dependente da entidade sendo criada.Logo, o ideal é que a interface possua apenas assinaturas, pois ao criar uma classe implementada de uma interface temos um "contrato" que deve ser seguido.

3. Explique com suas palavras por que não podemos ter construtores declarados com a palavrachave abstract.
   Se uma classe é abstrata, logo, ela não pode ser instanciada. Então, não faz sentido criar um construtor abstrato se é impossível criar construtores em classes abstratas.

# 2 Classe Produto

1. Identifique e explique o(s) erro(s) na classe abaixo:

```
1  public class Produto {
2      private String identificacao;
3      private double custoDeFabricacao;
4      Produto(String i, double c) {
5          identificacao = i;
6          custoDeFabricacao = c;
7      }
8      abstract public String toString();
9      abstract public void novoCusto(double nc);
10 }
```

Figura 1: Classe Produto

Primeiramente, como a classe produto não é generica, ou seja, não é abstrata não se pode ter metodos abstract. Além disso, no construtor da classe seria uma boa pratica de programação utilizar o this para referenciar os atributos do objeto e atribuir valores a eles. Além disso os metodos estão com ordem trocadas, o certo seria :

- public abstract String
- public abstract void

# 3 Classe Dispositivo

1. Identifique e explique o(s) erro(s) na classe abaixo:

```
1  public abstract class Dispositivo {
2      private String nome;
3      private long capacidadeEmBytes;
4      public Dispositivo(String n, long c) {
5          nome = n;
6          capacidadeEmBytes = c;
7      }
8      abstract public String toString();
9      abstract public double capacidadeEmMegabytes();
10 }
```

Figura 2: Classe Dispositivo

As classes abstratas não precisam ter construtores, mas elas podem ter. Realmente não vai ser possível instanciar as classes abstratas, mas as classes que herdarem essa primeira terão que chamar algum dos contrutores da classe abstrata.Por isso, pode-se dizer que, por Dispositivo ser uma classe abstrata ela não precisa de ter um construtor, ja que toda classe abstrata não pode ser instanciada não faz sentido ela ter um construtor com uma serie de regras para que o objeto seja construído. Além disso os metodos estão com ordem trocadas, o certo seria :

- public abstract String
- public abstract double

```
1  public class DiscoOtico extends Dispositivo {
2      public DiscoOtico(long c){
3          super("Disco  tico ", 241172480L);
4      }
5      public String toString() {
6          return "Dispositivo:" + nome + " Capacidade:" + c;
7      }
8  }
```

Figura 3: Classe DiscoOtico

Como DiscoOtico é uma classe extends de Dispositivo e a classe pai é uma classe abstrata, discoOtico pode ter um construtor. Então, ja que nao precisa de ter construtor na classe abstrata mas a classe acima tem a referencia super(..) parece estar correta. Por fim, acho que não encontrei nenhum erro nessa classe.