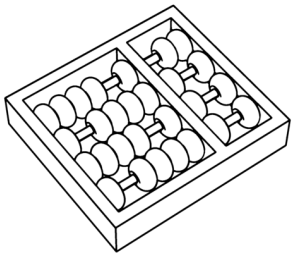


MC302 - Programação Orientada a Objetos
Instituto de Computação - Unicamp
Primeiro Semestre de 2017
Profa. Esther Colombini `esther@ic.unicamp.br`
PEDs: Elisangela Santos (ra149781@students.ic.unicamp.br)
Lucas Faloni (lucasfaloni@gmail.com)
Lucas David (lucasolivdavid@gmail.com)
Wellington Moura (wellington.tylon@hotmail.com)
PAD: Igor Torrente (igortorrente@hotmail.com)
`http://www.ic.unicamp.br/~esther/teaching/2017s1/mc302`

Lista de Exercícios

Tópicos:

- Princípios de Orientação a Objetos
- Linguagem Java
- Entrada e saída em java
- Propriedades e métodos
- Propriedades e métodos de instância
- Visibilidade
- Sobrecarga
- Herança e sobrescrita de métodos
- Relacionamentos de associação, agregação e composição: 1..1 e 1..*

---

1 Explique a diferença entre classes e objetos e dê um exemplo.

2 Explique o que é programação orientada a objetos e descreva um de seus benefícios.

3 [4] Escreva um modelo para representar uma lâmpada que está à venda em um supermercado. Que atributos devem ser representados por este modelo?

4 Crie uma classe para representar uma pessoa, com os atributos privados de nome, idade e altura. Crie os métodos públicos necessários para sets e gets e também um método para imprimir os dados de uma pessoa.

5 Crie uma classe denominada Elevador para armazenar as informações de um elevador dentro de um prédio. A classe deve armazenar o andar atual (térreo = 0), total de andares no prédio, excluindo o térreo, capacidade do elevador, e quantas pessoas estão presentes nele. A classe deve também disponibilizar os seguintes métodos:

- Inicializar: que deve receber como parâmetros a capacidade do elevador e o total de andares no prédio (os elevadores sempre começam no térreo e vazio);
- Entrar: para acrescentar uma pessoa no elevador (sé deve acrescentar se ainda houver espaço);
- Sair: para remover uma pessoa do elevador (só deve remover se houver alguém dentro dele);
- Subir: para subir um andar (não deve subir se já estiver no último andar);
- Descer: para descer um andar (não deve descer se já estiver no térreo);

Encapsule todos as propriedades usando os métodos set e get.

6 [5] As variáveis em Java são classificadas em: atributos de instância, atributos de classe, variáveis locais e parâmetros. Explique e dê um exemplo para cada classificação.

7 [5] Explique as visbilidades possíveis para classes, métodos e atributos em Java. Como o *princípio do menor privilégio* se adequa na definição de uma visibilidade? Como o conceito de encapsulamento se relaciona com a visibilidade daos atributos, métodos e classes?

8 [5] Escreva um programa em Java que imprime a mensagem Olá Mundo!. Descreva brevemente as palavras-chaves e instruções utilizadas.

9 [4] Identifique e explique o(s) erro(s) na classe abaixo.

```
public class Media {

        public int Media(int a,int b) {
                return (a+b)/2;
        }

        public double Media(int a,int b)
        {
                return (a+b)/2;
        }
}
```

10 [4] Identifique e explique o(s) erro(s) na classe abaixo.

```
public class DemoConstrutor {
        private int a,b;

        public DemoConstrutor() {
                System.out.println("No␣construtor␣sem␣argumentos...");
                DemoConstrutor(0,0);
        }

        public DemoConstrutor(int xa,int xb) {
                System.out.println("No␣construtor␣com␣argumentos...");
                a = xa;
                b = xb;
        }
}
```

11 Explique os conceitos de encapsulamento e herança de objetos de software.

12 Escreva o que você entende pelas relações *tem um* entre objetos: associação, agregação e composição. Dê um exemplo simples para cada tipo.

13 O que são métodos e atributos estáticos? Explique porque métodos estáticos não podem chamar métodos e atributos de instância diretamente. Por que o método main é declarado estático?

14 Explique para que serve o método super().

15 Descreva em UML a classe Produto, tendo como atributos, por exemplo, nome e valor. Coloque todos os atributos que julgar necessários. Em seguida, crie operações para a sua manipulação.

16 [5] Em java, crie um programa que leia do terminal uma frase e imprima no terminal a frase invertida. Ex.: Frase – esarF

17 Quais são as diferenças entre métodos de instância e métodos de classe?

18 Qual a diferença entre parâmetros e argumentos?

19 Para que serve a palavra-chave this?

20 Explique o funcionamento da classe java.util.ArrayList. Por que a usamos para tratar relacionamentos de associação 1..*? Quais seriam outras alternativas?

21 [1] Discuta como o conceito de herança promove o reuso de software, economizando tempo de desenvolvimento e ajudando a prevenir erros.

22 Alguns programadores preferem não usar o modificador de acesso protected, pois acredita- se que isso atrapalha o encapsulamento da superclasse. Discuta as vantagens e desvantagens relativas ao uso do acesso protegido em comparação ao uso do acesso privado de atributos nas superclasses.

23 Por que o método toString() foi definido na classe Object?

24 O que é sobrescrita? Quando acontece e quais as vantagens?

25 O que é sobrecarga? Dê exemplo.

26 Crie uma classe pública chamada Matematica, com um construtor privado e funções como soma e subtracao. Essas funções devem aceitar como argumento variáveis do tipo int ou double. Instancie sua classe e faça testes.

27 [2] Considere as seguintes classes e responda:

- Qual método de ClassB sobrescreve (override) um método de ClassA?
- Qual método de ClassB esconde (hide) um método de ClassA?
- Quais métodos de ClassB são inválidos devido à herança de ClassA?

```
public class ClassA {
        public void methodOne(int i) {}
        public void methodTwo(int i) {}
        public static void methodThree(int i){}
        public static void methodFour(int i) {}
}

public class ClassB extends ClassA {
        public static void methodOne(int i){}
        public void methodTwo(int i){}
        public void methodThree(int i){}
        public static void methodFour(int i){}
}
```

28 Considere os métodos abaixo. Indique se está sendo utilizada sobrecarga de método ou sobrescrita. Explique e diga com que outro método esta sendo realizada a sobrecarga ou sobrescrita (se for o caso).

```
public class A {
        public A() { ... }
        public A( int x ) { ... }
        public void m1() { ... }
        public void m1( int h ) { ... }
        public void m3( ) { ... }
}

public class B extends A {
        public B() { ... }
        public void m1() { ... }
        public void m1( double x, double y ) { ... }
        public void m2() { ... }
        public void m3(int h) { ... }
}
```

- Quais métodos realizam sobrescrita? Por quê?
- Quais métodos realizam sobrecarga? Por quê?
- Qual método (de qual classe) será executado caso se instancie um objeto da classe B (b) e na sequência seja realizada a seguinte chamada: b.m3(10)? Justifique.
- Dado que um objeto a da classe A seja instanciado, é possível realizar a seguinte chamada: a.m3(10)? Justifique sua resposta. Caso a mesma tenha sido negativa, qual seria a forma correta de invocar este método?

29 [3] Considere o seguinte código:

```
Class Zing { Hmpf h; }
Class Woop extends Zing{}
Class Hmpf {Zing z; }
```

Para estas classe, podemos afirmar que:

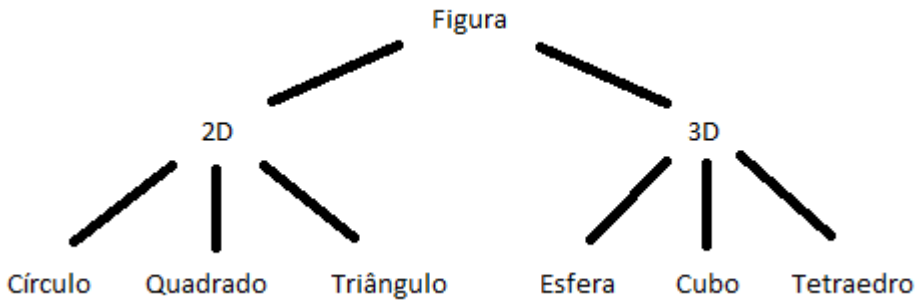
- Woop é Hmpf e é um Zing
- Zing é um Woop e tem um Hmp
- Hmpf tem um Woop e Woop tem um Zing
- Woop tem um Hmpf e Woop tem um Zing
- Zing tem um Hmpf e Woop é um Zing
- Hmpf tem um Zing e Woop tem um Hmp

Indique as alternativas corretas.

30 [3] Qual é a saída do programa a seguir?

```
class Alpha { int over = 1; }
class Beta extends Alpha { int over = 2; }
class Gamma extends Beta {
        int over = 3;
        public static void main( String [ ] args ) {
               new Gamma().go();
        }
        void go() {
                 Beta b = new Gamma();
                 Alpha a = new Gamma();
                 System.out.println(super.over + "␣" b.over + "␣" + a.over);
        }
}
```

31 [1] Implemente a seguinte hierarquia descrita na figura abaixo. Cada figura bidimensional contém o método `getArea` para calcular a área, enquanto cada figura tridimensional contém os métodos `getArea` e `getVolume` para calcular, respectivamente, a área de superfície e o volume. Crie um vetor, contendo objetos do tipo `Figura`. Para cada elemento do vetor, determine se é uma figura bidimensional ou tridimensional e imprima seus respectivos atributos.



32 [4] Escreva uma classe ConversaoDeUnidadesDeTemperatura que contenha métodos estáticos para calcular a conversão entre diferentes escalas de temperatura. Considere as fórmulas de conversão abaixo:

- De graus Celsius(C) para graus Fahrenheit(F): $F = (9 * C/5) + 32)$
- De graus Fahrenheit (F) para graus Celsius (C): $C = (F - 32) * 5/9$
- De graus Celsius (C) para graus Kelvin (K): $K = C + 273.15$
- De graus Kelvin (K) para graus Celsius (C): $C = K - 273.15$
- De graus Celsius (C) para graus Réaumur (Re): $Re = C * 4/5$
- De graus Réaumur (Re) para graus Celsius (C): $C = Re * 5/4$
- De graus Kelvin (K) para graus Rankine (R): $R = K * 1.8$
- De graus Rankine (R) para graus Kelvin (K): $K = R/1.8$

Veja que já que existem cinco sistemas de medidas de temperatura, devem haver 20 diferentes métodos de conversão de temperatura. Alguns podem ser escritos indiretamente, por exemplo, para converter de Celsius para Rankine, podemos converter de Celsius para Kelvin e converter esse resultado para Rankine.

33 Crie uma classe Televisão e uma classe ControleRemoto capaz de controlar o volume e trocar os canais da televisão. Apresente a estrutura de classes em UML e o código correspondente em Java.

- O controle de volume permite aumentar ou diminuir a potência do volume de som em uma unidade de cada vez;
- O controle de canal também permite aumentar e diminuir o número do canal em uma unidade, porém, também possibilita trocar para um canal indicado;
- Também devem existir métodos para consultar o valor do volume de som e o canal atualmente selecionado.

34 [4] É possível evitar completamente a necessidade de sobreposição de métodos criando métodos em classes descendentes que tenham assinaturas diferentes. Por exemplo, a classe Pessoa poderia ter o método imprimePessoa para imprimir seus atributos, e a classe Aluno que estende a classe Pessoa poderia ter o método imprimeAluno para imprimir os atributos de Aluno. Que vantagens e desvantagens essa abordagem teria sobre a sobreposição de métodos?

Referências

[1] Paul Deitel, Harvey Deitel, *Java: How to Program.* Deitel, 9a Edição.

[2] *Java: The java tutorials.* `http://docs.oracle.com/javase/tutorial/java/javaOO/QandE/nested-questions.html`

[3] *Blog: Questões Comentadas para Certificação Java SCJP.* `http://scjpquestoes.blogspot.com.br/`

[4] *Primeira Lista de Exercícios* `http://www.lac.inpe.br/~rafael.santos/Docs/IntroPOOJava/`

[5] *Lista de Exercícios de POO* `http://www.ic.unicamp.br/~fusberti/`