PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# PCS-2302 / PCS-2024
# Lab. de Fundamentos de Eng. de Computação

## Aula 05

## Fundamentos da Programação Orientada a Objetos (Parte 3)

**Professores:**
Anarosa Alves Franco Brandão (PCS2302)
Jaime Simão Sichman (PCS 2302)
Reginaldo Arakaki (PCS 2024)
Ricardo Luís de Azevedo da Rocha (PCS 2024)

**Monitores:** Diego Queiroz e Tiago Matos

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Roteiro

1. Elementos básicos da POO – Parte 3
   - Tipos e hierarquia de tipos
   - Atribuição e despacho
   - Definição de hierarquias de tipos
   - Subtipos
2. Parte Experimental
   - Implementar o simulador MVN (Parte 3)
     - Classe **`GerenciadorDispositivos`**
     - Interface **`Dispositivo`**
     - Classes **`Teclado`**, **`Monitor`** e **`Disco`**

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Tipos e hierarquia de tipos (1)

- A abstração de dados, base da OO, é associada ao conceito de tipo. (aula 3 – slides 18 e 19)
  - Abstração de dados permite introduzir novos **tipos** de dados;
  - Um **tipo** pode ser caracterizado por uma *representação de dados* e por *operações* sobre esta representação.
- Tipos podem ser usados para classificar indivíduos (ou entidades) pertencentes a algum domínio. Em OO:
  - Tipos descrevem as propriedades que indivíduos (ou entidades) classificáveis através deles irão compartilhar;
  - Tipos especificam restrições de comportamento dos indivíduos (ou entidades) que eles classificam;
  - Indivíduos (ou entidades) classificados por um tipo "pertencem" ao tipo.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

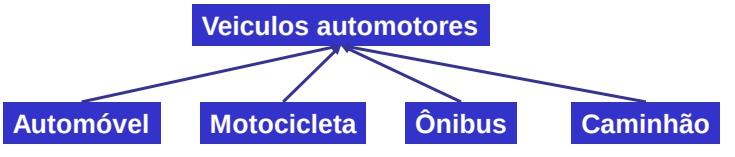
Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Tipos e hierarquia de tipos (2)

- Relacionando tipos:
  - Há um tipo mais geral que abranja tipos como Automóvel, Motocicleta, Ônibus e Caminhão?
  - Pode-se ter o tipo Veículos Automotores, o qual pode ser abrangido pelo tipo Meio de Transporte , etc.
- A generalização permite examinar se tipos têm algo em comum.



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

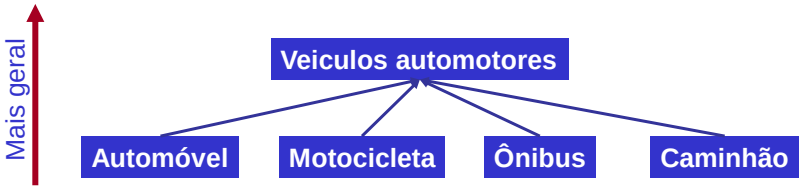
Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009

5

# Tipos e hierarquia de tipos (3)

- A generalização permite dizer que todas as instâncias de um determinado tipo são também instâncias de um tipo mais geral.
  - Um Fiat Idea é instância de Automóvel, mas também é instância de Veículo Automotor
- Generalização permite definir hierarquias de tipos, formando tipos mais gerais.




PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

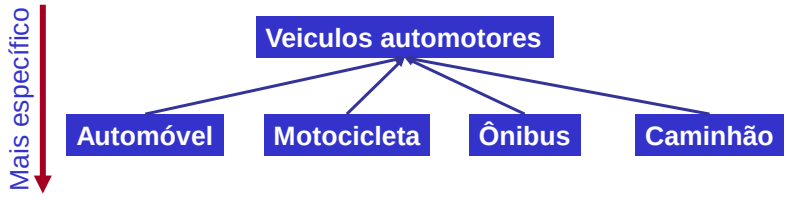
Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009

6

# Tipos e hierarquia de tipos (4)

- A especialização é o oposto da generalização
  - Automóvel e Motocicleta são especializações de Veículos Automotores.
  - Tipos especializados são denominados subtipos.



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

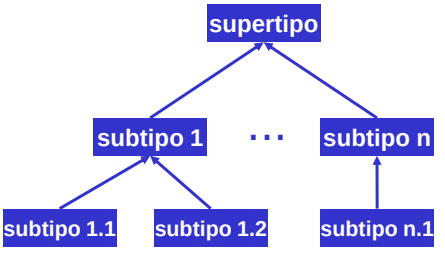
Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Tipos e hierarquia de tipos (5)

- A hierarquia de tipos é utilizada para definir famílias de tipos, consistindo de um supertipo e de seus subtipos.
  - um tipo é um subtipo de si próprio.
  - subtipos estendem o comportamento de seu supertipo, por exemplo, adicionando ou alterando operações.




PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Tipos e hierarquia de tipos (6)

- Em uma hierarquia de tipos, os subtipos devem estar em conformidade com a especificação de seu supertipo.
  - Os membros da hierarquia devem ter comportamento relacionado.
- Numa hierarquia de tipos vale o Princípio da Substituição:
  - O comportamento do supertipo deve ter suporte nos subtipos, i.e., um subtipo pode ser usado no lugar do supertipo.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

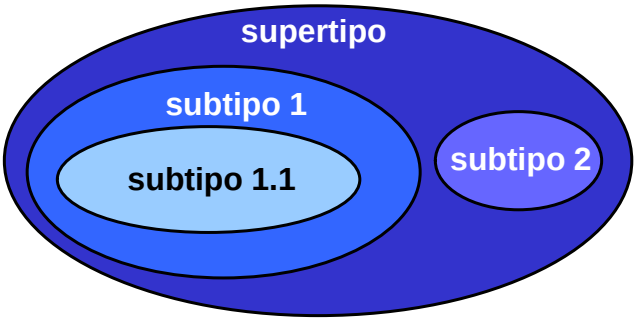
Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Tipos e hierarquia de tipos (7)

- A generalização pode ser imaginada como conjunto de objetos dentro de conjuntos de objetos.
  - Os conjuntos dos subtipos devem estar totalmente contidos no conjunto do supertipo, i.e., todas as instâncias dos subtipos devem ser instâncias do supertipo.




PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Tipos e hierarquia de tipos (8)

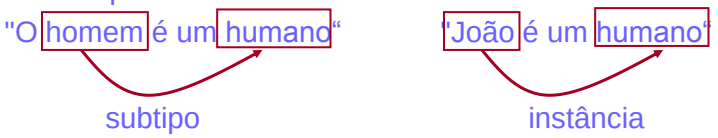
- Somente tipos de objetos podem ser subtipos (de objetos).

- Uma instância de um tipo não é um subtipo.
  - Exemplo:

"O homem é um humano“ "João é um humano“

# Tipos e hierarquia de tipos (8)

- Somente tipos de objetos podem ser subtipos (de objetos).
- Uma instância de um tipo não é um subtipo.
  - Exemplo:








# Tipos e hierarquia de tipos (9)

- Quando se classifica um objeto, declara-se que uma determinada abstração, ou tipo, é aplicável ao objeto classificado.
  - a classificação permite relacionar tipos com objetos.
  - um objeto deve ser uma instância de pelo menos um tipo.
  - objetos são instâncias pois foram classificados.
  - tipos de objetos são subtipos, pois foram especializados.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Tipos e hierarquia de tipos (10)

- Classificação e generalização (especialização) são fenômenos distintos e têm hierarquias distintas.
- Classificação é uma relação entre tipo e objeto
- Generalização (especialização) é uma relação apenas entre tipos.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Tipos e hierarquia de tipos (11)

- A classe é um mecanismo que especifica e gera uma coleção de instâncias da classe, ou objetos.
  - Tipo ≠ classe:
    - O tipo ajuda a definir uma descrição consistente de um sistema, enquanto a classe é um mecanismo para definir e criar objetos.
  - Na OO, a iteração entre abstração e decomposição realiza-se sobre tipos.
    - Uma solução computacional é, portanto, um sistema de tipos.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Tipos e hierarquia de tipos (12)

- Existem duas grandes categorias de linguagens OO:
  - baseadas em classe
    - a classe especifica a estrutura de criação (implementação) de todos os objetos
    - o objetivo é disciplinar a estrutura dos objetos criados, i.e., da implementação do tipo como especificado na classe
    - Ex: C++, Java, C#, etc. (maioria das linguagens comerciais)
  - baseadas em objeto
    - existem outros constructos (protótipos e clones, etc.) para a criação de objetos individuais
    - o objetivo principal aqui é a flexibilidade, pois o objeto pode variar de tipo ao longo de seu ciclo de vida.
    - como não há classes, objetos são implementações de tipos
    - Ex: CLOS, Self, etc.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Atribuição e despacho (1)

- A hierarquia de tipos afrouxa as regras que governam a atribuição, a passagem de argumentos e a chamada de métodos, apresentadas na aula 4.

- Uma variável de um tipo pode referenciar um objeto de um subtipo deste tipo. Pelo princípio de substituição, se **`S`** é um subtipo de **`T`**, então os objetos **`S`** podem ser atribuídos a variáveis do tipo **`T`**, podendo também serem passados como argumentos ou resultados em que se esperava **`T`**.
  - Exemplo usando o tipo **`Conta`** e seus subtipos **`ContaCorrente`** e **`ContaPoupanca`**.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Atribuição e despacho (2)

- Supondo que **`ContaCorrente`** e **`ContaPoupanca`** sejam subtipos de **`Conta`**, pode-se ter:

```
Conta c1324 = new ContaCorrente();
```

tipo

subtipo

```
Conta c5345 = new ContaPoupanca();
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Atribuição e despacho (3)

- Hierarquia de tipos X atribuição e despacho:
  - Tipo aparente: o que o compilador pode deduzir da informação das declarações.
    - O objeto **`c1324`** tem tipo aparente **`Conta`**
  - Tipo real: o tipo que o objeto realmente tem. O tipo real sempre será um subtipo de seu tipo aparente (um tipo é um subtipo de si próprio).
    - O objeto **`c1324`** tem tipo real **`ContaCorrente`**.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Atribuição e despacho (4)

- O compilador faz a verificação de tipos baseada nos tipos aparentes. Assim, a verificação da chamadas de métodos baseia-se nos tipos aparentes.

```
Conta c1324 = new ContaCorrente();

double oSaldo = c1324.obterSaldo();
```

  - OK, pois **`Conta`** tem um método **`obterSaldo()`**, sem argumentos e retornando um **`double`**.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Atribuição e despacho (5)

- O objetivo da verificação:
  - é assegurar que, na chamada do método, o objeto efetivamente tenha o método com a assinatura apropriada.
- O objeto referenciado por **`c1324`** deve ter todos os métodos indicados pelo supertipo com as assinaturas esperadas.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Atribuição e despacho (6)

- O compilador pode não ser capaz de determinar qual código deve ser executado na chamada de um método, pois só conhece o tipo aparente.

```
public int obterPontosCliente(Conta umaConta){
    Conta c = umaConta.obterSaldo();
    // ...
}
```

- Quando o método é compilado, o compilador não sabe o tipo real de **`umaConta`**.
  - Porém, o código do tipo real do objeto passado como argumento deve ser adequadamente invocado.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Atribuição e despacho (7)

- A chamada do código apropriado do método é possibilitada por um mecanismo em tempo de execução denominado despacho.

- Através do mecanismo de despacho o compilador gera o código para encontrar o método adequado e desviar para ele, ao invés de chamar diretamente o método.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

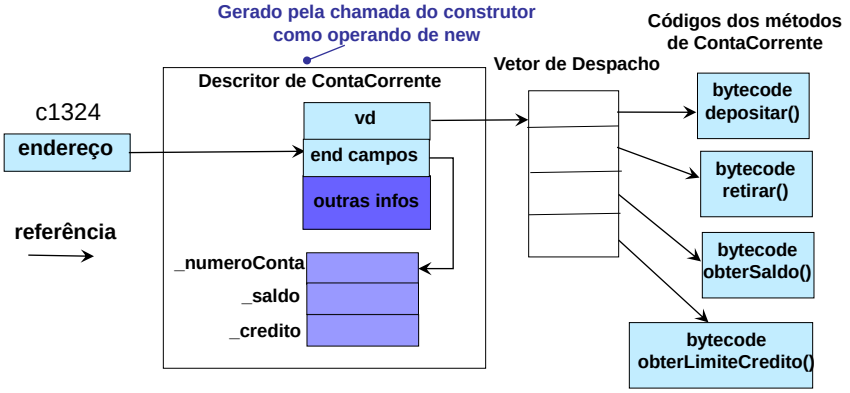
Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Atribuição e despacho (8)

- Uma possível implementação do despacho é ter um vetor de despacho. A chamada de um método obtém a referência para o código do método do tipo real e desvia para o seu ponto de entrada.




PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Definição de hierarquias de tipos (1)

- Como realizar uma hierarquia de tipos nas linguagens OO comerciais?
  - Em uma linguagem OO típica, um supertipo é definido por uma classe ou uma interface, as quais fornecem a sua especificação. No caso da classe, ela fornece uma implementação parcial ou completa.
  - Uma classe abstrata fornece uma implementação parcial. Ela não pode ser diretamente instanciada, i.e. não tem objetos.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



## Definição de hierarquias de tipos (2)

- Como realizar uma hierarquia de tipos nas linguagens OO comerciais?
  - Uma subclasse pode herdar a implementação de métodos de sua superclasse. Pode também sobrepor (*override*) tais implementações (Java: para métodos não finais), i.e., prover implementações específicas.
  - Os campos de uma subclasse consistem de suas próprias variáveis de instância e as de sua superclasse. O acesso a estes últimos só é possível se forem declarados **`protected`**.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009

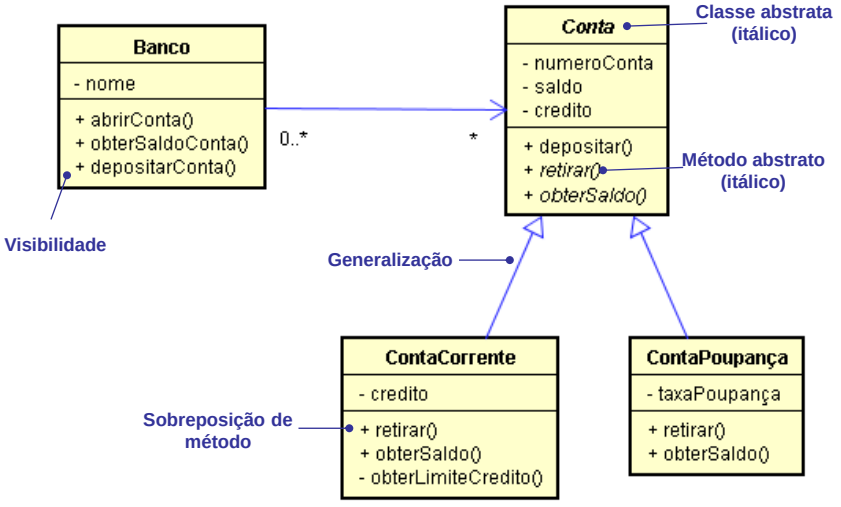


## Definição de hierarquias de tipos (3)

- Classe abstrata
  - Fornece uma implementação parcial de um tipo.
  - Pode ter variáveis de instância e construtores. Estes não podem ser chamados pelos usuários, mas podem ser usados para iniciar a parte da superclasse de uma instância de subclasse.
  - Normalmente, contém métodos abstratos e não-abstratos, fornecendo a implementação destes últimos. As implementações dos primeiros são diferidas para subclasses.
  - A implementação de métodos não-abstratos pode ser interessante, pois são implementados apenas uma vez.

# Definição de hierarquias de tipos (4)

- Exemplo didático parcial em UML.






PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009

# Definição de hierarquias de tipos (5)

```
public abstract class Conta{
   // Atributos (variáveis de instância)
   // Não é necessário apresentar construtor(es). Quando é interessante tê-lo(s)?
   // Métodos: retirar() e obterSaldo() não fornecem código.
   public double obterSaldo();
   // ...
}

public class ContaCorrente extends Conta{
   // Atributos (variáveis de instância)
   // Construtor(es)
   // Métodos: fornecer a implementação dos métodos da classe.
}

public class ContaPoupanca extends Conta{
   // Atributos (variáveis de instância)
   // Construtor(es)
   // Métodos: fornecer a implementação dos métodos da classe.
}
```

Indica o relacionamento de especialização

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Definição de hierarquias de tipos (6)

- Interface
  - Define apenas as restrições relativas a um tipo, mas não se destina a criar objetos com estas características.
  - Contém apenas métodos públicos e não-estáticos. Todos os métodos são abstratos.
  - Não fornece nenhuma implementação.
  - Permite uma separação efetiva entre a especificação do comportamento requerido (métodos da interface) e de suas implementações
  - As implementações não têm restrições estruturais como em uma derivação de uma (super-)classe abstrata.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

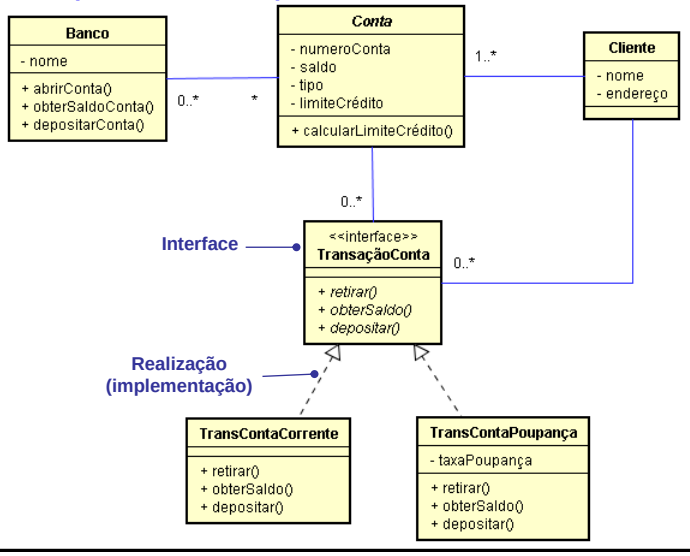
Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Definição de hierarquias de tipos (7)

- Exemplo didático parcial em UML.



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Definição de hierarquias de tipos (8)

```
public interface TransacaoConta {
   // Métodos: retirar(), obterSaldo() e depositar() não fornecem código.
   public void retirar(double valor);
   public double obterSaldo();
   // ...
}

public class TransContaCorrente implements TransacaoConta {
   // Atributos (variáveis de instância)
   // Construtor(es)
   // Métodos: fornecer a implementação dos métodos da classe.
}

public class TransContaPoupanca implements TransacaoConta {
   // Atributos (variáveis de instância)
   // Construtor(es)
   // Métodos: fornecer a implementação dos métodos da classe.
}
```

Indica o relacionamento de realização

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Exercícios

TYGXXA05E01_09

Esta terceira parte da implementação do simulador MVN tem por finalidade a sedimentação dos conceitos essenciais da programação OO em Java.

O simulador deve carregar programas em MVN que utilizem as instruções de entrada e saída: 0x000d e 0x000e. Os dispositivos utilizados são: teclado, monitor e disco. A implementação estende o código produzido para a Parte 2 da implementação da MVN.

Segue o diagrama de classes parcial, em UML, do simulador.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009

33

# Implementação da MVN - Parte 3 (1)

## Operações de Entrada e Saída

| *OP* | *Tipo* | *Dispositivo* |
|---|---|---|

**OP** D (entrada) ou E (saída)

**Tipo** Tipos de dispositivo:
- 0 = Teclado
- 1 = Monitor
- 2 = Impressora
- 3 = Disco

**Dispositivo** Identificação do dispositivo. Pode-se ter vários tipos de dispositivo, ou *unidades lógicas* (LU). No caso do disco, um arquivo é considerado uma unidade lógica.

**Pode-se ter, portanto, até 16 tipos de dispositivos e, cada um, pode ter até 256 unidades lógicas.**

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

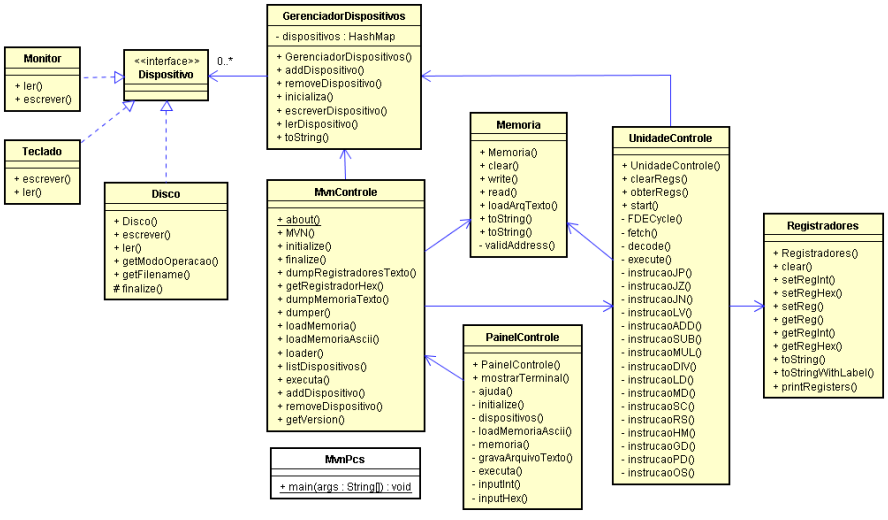
Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009

34

# Implementação da MVN - Parte 3 (2)



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

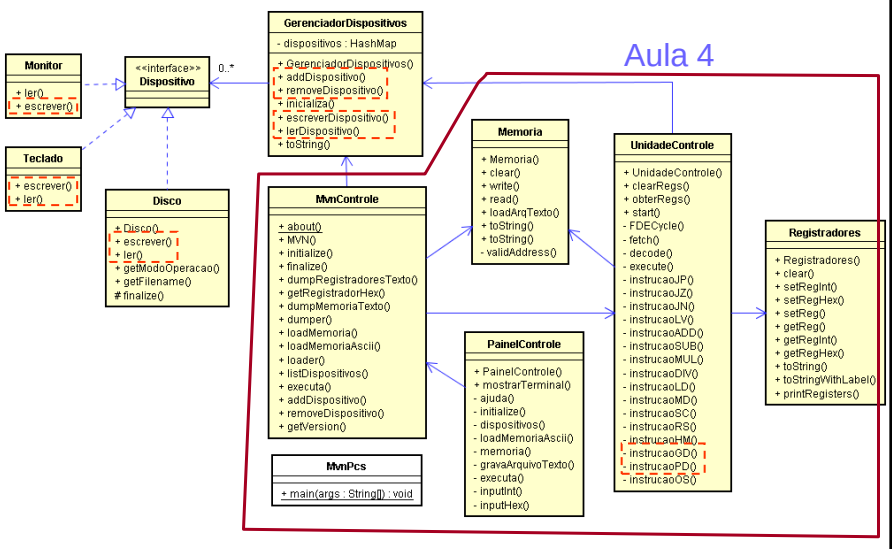
Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Implementação da MVN - Parte 3 (2)




PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha



# Implementação da MVN - Parte 3 (3)

- Vocês receberão o código fonte incompleto, com especificações parciais de alguns métodos e a implementação de alguns métodos.
  - Implementar o simulador da MVN de acordo com o diagrama de classes, acrescentando e codificando os atributos e métodos especificados na interface **`Dispositivo`** e nas classes **`UnidadeControle`**, **`GerenciadorDispositivos`**, **`Teclado`**, **`Monitor`** **`e`** **`Disco`**.
  - A implementação deverá **obrigatoriamente** estender o código da Parte II da MVN produzido anteriormente pelo grupo.
  - Especificar todos os métodos no estilo apresentado.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



## Implementação da MVN - Parte 3 (4)

- Todos os arquivos-fonte vêm comentados, incluindo as especificações dos métodos. No entanto, nem todos os métodos estão especificados no estilo da abstração procedimental. Colocá-los **todos** neste estilo.
- Instruções e esclarecimentos adicionais em sala de aula.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



## Algumas dicas

- Vejam a documentação das classes Java relacionadas com a escrita e leitura de arquivos (`FileOutputStream` e `FileInputStream`).
- Um detalhe importante da classe de escrita `FileOutputStream`: caso não se esteja em modo de anexar (*append*), apaga-se o arquivo de entrada no momento em que é criado o objeto (ou seja, na execução do construtor).
- A biblioteca do Java possui um conjunto de estruturas de dados **já implementadas**, entre elas o `HashMap` (usado pelo gerenciador de dispositivos). Assim, não é preciso alterar a classe `HashMap` do Java, apenas deve-se usá-la. Convenientemente. Consultem a documentação do Java para saber o seu funcionamento.
- A compilação da classe `GerenciadorDispositivos` poderá gerar alguns *warnings* caso a versão do compilador usado seja de 1.5 em diante (como no caso do laboratório). Não se preocupem com isso. Quem se interessar, veja a documentação sobre Generics (similares aos Templates em C++) em

  http://java.sun.com/j2se/1.5.0/docs/guide/language/generics.html.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F.Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 5:

Fundamentos Prog. Orientada Objetos (3)

Autores:
Anarosa A.F. Brandão
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

v. 1.4 set 2009



# Bibliografia Complementar

Budd, T. *An Introduction to Object-Oriented Programming*. 3a. Ed. Addison Wesley, 2001.

Liskov, B.; Guttag, J. *Program Development in Java*. Addison-Wesley, 2001.