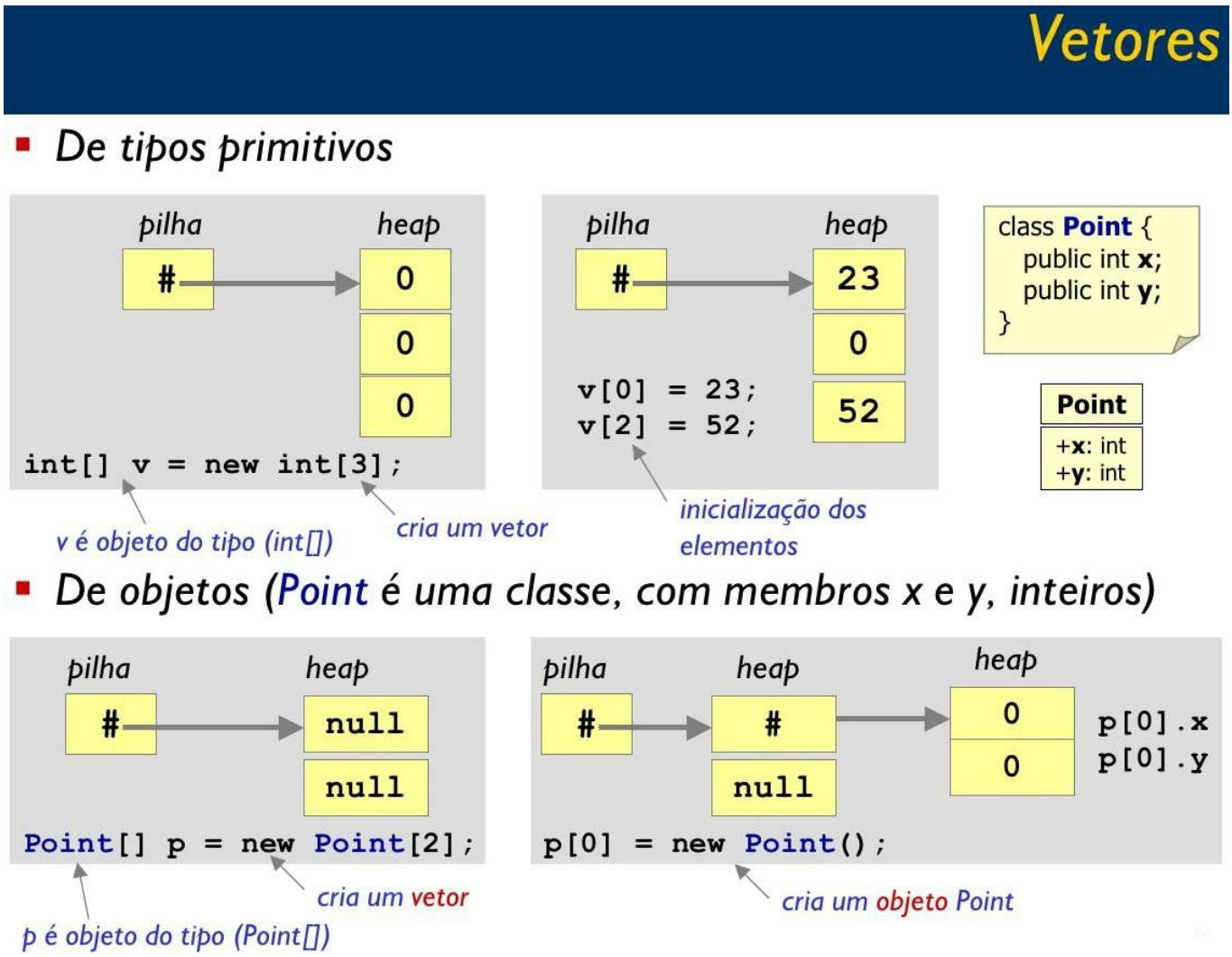
MC302 – Programação Orientada a Objetos 15/03/2016
Instituto de Computação – UNICAMP
Prof.: Cecília Rubira

LAB03: Herança

Coleções de Objetos ou tipos primitivos em Java:

Vamos tomar emprestado uma transparência do Prof. Helder da Rocha (www.argonavis.com.br) para entendermos as diferenças na criação de vetores de tipos primitivos e vetores de objetos em java:



Conhecendo um pouco mais sobre a linguagem Java:

Laços **`for`** em Java também podem iterar sobre arrays e coleções. A sintaxe:

```
for (Object obj : colecao ) { … }
```

pode ser lida como: "repita o bloco para cada objeto obj de colecao. Exemplo (usando arrays):

```
int[] a = {1,2,3,4,5,6,7,8,9,10};
int  sum = 0;

// Usando for convencional
for (int i = 0; i < a.length; i++)
      sum += a[i];
```

```
System.out.println(sum);


// Usando enhanced for
for (int i : a)
      sum += i;
System.out.println(sum);
```

INSTRUÇÕES INICIAIS:

1. Obtenha o arquivo **lab03.zip** (disponível no ensino aberto);
2. Descompacte esse arquivo no seu diretório de trabalho (workspace do Eclipse);
3. Crie um novo projeto Java no Eclipse. Clicar em File > New > Java Project, manter o lugar padrão e escrever como nome do projeto exatamente o nome da pasta descompactada.
4. Ao final do lab, compacte a pasta raiz do projeto, em um arquivo com extensão .**zip**, e suba no ensino aberto no seu portfólio, disponibilizando-o para os formadores. Suba apenas um arquivo. **Outros formatos não serão avaliados.**
5. Suba um arquivo de texto com a resposta para as questões 3, 6 e 7 **dentro** da pasta compactada. Serão aceitos os formatos .**txt** e **pdf**. Coloque seu nome e RA neste arquivo.

Nota: Os exercícios devem ser executados na ordem apresentada, pois o nível de dificuldade é crescente.

Pacote *br.unicamp.ic.mc302.documento2*:

Abra os arquivos Documento2.java, Carta2.java, Telegrama2.java e ExemploDocumento2.java compile-os, execute o último e observe o seu comportamento.

1. Defina uma classe CartaRegistrada, subclasse de Carta2. Essa classe deve ter dois atributos: a data e local de envio. Defina também uma classe Encomenda, subclasse de CartaRegistrada, com um único atributo correspondente ao conteúdo da encomenda (pode ser um String).

2. Crie a classe ExCartaEncomenda.java com um método main() que cria objetos dos tipos CartaRegistrada e Encomenda e chama a operação imprimir( ) de cada um desses objetos. Compile esse programa e execute-o.

Pacote *br.unicamp.ic.mc302.conta*:

3. Abra os arquivos Conta.Java, ContaDePoup.Java e ExemploConta.Java. Compile as três classes e execute a última. O que acontece? Você sabe explicar o porquê?

4. Modifique a visibilidade do atributo saldo da classe Conta para private. Recompile o arquivo Conta.java. Por que a compilação não funciona?

5. Modifique o código, de modo a retirar os erros existentes, mantendo a visibilidade do atributo como private.

6. Modifique seu programa de tal forma que Conta e ContaDePoup estejam em pacotes distintos – veja o item 12.3 do artigo disponível em http://www.vogella.com/articles/Eclipse/article.html

6.1. Modifique a visibilidade de saldo para protegida e altere as implementações da classe ContaDePoup para acessá-lo diretamente. O código funciona ou não?

6.2. Modifique novamente a visibilidade de saldo para privada e crie 2 operações protegidas (getSaldo() e setSaldo()) que serão usadas pela subclasse. Explique as vantagens dessa solução.

Pacote *br.unicamp.ic.mc302.listaInts*:

7. Este é o mesmo pacote apresentado no lab anterior, que cria um ArrayList para guardar coleções de Inteiros. Estude o arquivo ListaInts.java e comente as diferenças entre os dois métodos fornecidos para se encontrar o menor elemento da lista: minimum e minimum_version2. Qual método apresenta a idéia de um código mais limpo? Na sua opinião, quando devemos usar a primeira versão e quando temos que usar a segunda?

Herança

8. Implemente um programa Java que cria a classe Pessoa com atributos como nome, sexo, rg, cpf, nome do pai, nome da mãe, etc. Crie a subclasse EstudanteUniversitario a partir da classe Pessoa com atributos como, por exemplo, RA, nome do curso, créditos concluídos, etc. Implemente a classe Graduando que herda de EstudanteUniversitario, com operações para calcular o CR (coeficiente de rendimento) e o CP (coeficiente de Progressão). Crie um programa principal que instancia você na classe Graduando.

9. Implemente um programa Java que represente corpos celestes (estrelas, planetas, luas, etc). Inclua nas classes atributos como nome, massa, diâmetro, etc. Considere, por exemplo, 2 tipos de planetas: Planetas Grandes Gasosos e Planetas Terrestres. Implemente operações significativas para todas as suas classes e use a visibilidade protegida para os atributos como massa e diâmetro serem acessados diretamente pelas subclasses. Crie um programa principal que instancia o sistema solar.

Exercício Extra (O aluno que responder corretamente receberá uma pontuação extra)

1. Implemente uma classe Java chamada SistemaPlanetario que encapsula sistemas planetários – objetos não-estelares (planetas e satélites naturais) que orbitam uma estrela. Crie variáveis e métodos que achar pertinente. Crie uma classe chamada MainSistemaSolar e instancie um objeto que corresponde ao Sistema Solar no método main desta classe.

2. Considere a classe Object que contém, dentre outros, o método toString() na sua interface pública. Faça a classe SistemaPlanetario da sua solução herdar a classe Object. É possível "chamar" o método toString() para instâncias dessas classes? Por quê? (Inclua o "output" como parte da resposta)