Slide 07 – Membros estáticos e enumerações

Baseado nos slides oficiais do livro Java – Como programar – Deitel e Deitel – 10ª edição

Estado e comportamento

- Chamamos de **estado de um objeto** os valores atuais assumidos pelos seus atributos
- Chamamos de **comportamento de um objeto** o resultado esperado ou obtido quando uma operação é invocada por um objeto

```
Aluno jose = new Aluno(1.0, 4.0, 4.5);
Aluno maria = new Aluno (3.5, 2.8, 6.0);
if (jose.calculaMedia() == maria.calculaMedia() )
...
```

Membros de instância (de objetos)

- Atributos e operações de objetos diferentes tendem a ter estados e comportamentos diferentes
- **Membros de instância** representam características que todos objetos possuem cujo estado e comportamento podem variar. Ex: Notas de um aluno; Tamanho da janela
- Membros da classe são acessados a partir do nome do objeto. Ex: jose.getNota1(), maria.setNota2(4)

Membros estáticos (da classe)

- Em algumas situações, alguns atributos e métodos são os mesmos, independente dos objetos
- **Membros estáticos** representam características cujo estado e comportamento não variam. Ex: Nota minima para ser aprovado na disciplina; Resolução da tela, cálculo de uma média ponderada
- Membros estáticos são acessados a partir do nome da classe. Ex: Math.PI, Integer.parseInt

Declarando membros estáticos

- Um membro estático é declarado com a palavra reservada **static** após o tipo de encapsulamento
  - private static double mediaAprovacao;
  - public static double mediaPonderada (double valores[ ], double pesos [ ]) {...}
- Constantes geralmente são declaradas estáticas e com a grafia com todas as letras em maiúsculo
  - public static final int PESOS [ ] = {4, 5, 6};

```
1  // Fig. 6.3: MaximumFinder.java
2  // Programmer-declared method maximum with three double parameters.
3  import java.util.Scanner;
4
5  public class MaximumFinder
6  {
7     // obtain three floating-point values and locate the maximum value
8     public static void main(String[] args)
9     {
10       // create Scanner for input from command window
11       Scanner input = new Scanner(System.in);
12
13       // prompt for and input three floating-point values
14       System.out.print(
15          "Enter three floating-point values separated by spaces: ");
16       double number1 = input.nextDouble(); // read first double
17       double number2 = input.nextDouble(); // read second double
18       double number3 = input.nextDouble(); // read third double
19
20       // determine the maximum value
21       double result = maximum(number1, number2, number3);
22
```

Fig. 6.3 | Programmer-declared method `maximum` with three `double` parameters. (Part 1 of 3.)

```
23          // display maximum value
24          System.out.println("Maximum is: " + result);
25       }
26
27       // returns the maximum of its three double parameters
28       public static double maximum(double x, double y, double z)
29       {
30          double maximumValue = x; // assume x is the largest to start
31
32          // determine whether y is greater than maximumValue
33          if (y > maximumValue)
34             maximumValue = y;
35
36          // determine whether z is greater than maximumValue
37          if (z > maximumValue)
38             maximumValue = z;
39
40          return maximumValue;
41       }
42    } // end class MaximumFinder
```

Fig. 6.3 | Programmer-declared method `maximum` with three `double` parameters. (Part 2 of 3.)

```
Enter three floating-point values separated by spaces: 9.35 2.74 5.1
Maximum is: 9.35
```

```
Enter three floating-point values separated by spaces: 5.8 12.45 8.32
Maximum is: 12.45
```

```
Enter three floating-point values separated by spaces: 6.46 4.12 10.54
Maximum is: 10.54
```

Fig. 6.3 | Programmer-declared method `maximum` with three `double` parameters. (Part 3 of 3.)

Exemplo

- Criar o pacote para aula10 e copiar as classes Aluno e JanelaMedia do pacote aula06
- Modificar a classe Aluno e acrescentar constantes de pesos e médias mínimas para aprovação
- Criar um método estático em que dadas duas notas ele devolva a nota mínima que o aluno deve tirar na 3ª para ser aprovado por média, ou -1 se não for possível

Import estático

- Alguns métodos ou atributos estáticos são usados de forma recorrente em um código
- Para reduzir o tamanho do código, pode-se importar esses membros estáticos para não necessitar usar os nomes das classes
  - import **static** java.lang.Integer.parseInt;
  - import **static** java.lang.Math.*;
  - Ex: int x = pow(parseDouble(“20”), 2) ;

Enumerações

- Assim como em C/C++, Java dá suporte a enums
- Enumerações representam um conjunto de valores constantes que podem ser atribuídos a uma variável
- Por exemplo, imagina que queremos devolver a situação do aluno de acordo com a média
  - public final int MATRICULADO=1, APROVADO=2, QUARTA_PROVA=3, REPROVADO = 4;

Declarando enums

- Como novos tipos, enums são declarados de forma semelhante a uma classe, mas com a palavra enum ao invés de class
- Após as chaves, deve-se declarar a lista de constantes que a enum pode assumir

```
// Arquivo StatusAluno.java
pubic enum StatusAluno{
    APROVADO, REPROVADO, MATRICULADO;
}
```

Usando enums

- Cada valor da enum é considerado um objeto da classe enum definida
- Para comparar, usa-se o método equals como na comparação entre objetos

```
StatusAluno status =
StatusAluno.MATRICULADO;
if(status.equals(StatusAluno.APROVADO)
    System.out.println("Aluno aprovado");
```

Exemplo

- Criar a enum StatusAluno com os possíveis status dos alunos da uern
- Adicionar um método de instância na classe aluno que devolva um StatusAluno a partir das notas
- Modificar a classe JanelaMedia, para exibir o status do aluno, com uma cor diferente para cada status. Usar constantes da classe java.awt.Color e o método setForeground ou setBackground do JLabel

Enumerações com atributos

- Como as enums são comparadas a classes em java, elas também podem ter atributos, construtores, métodos get/set, …
- Contudo, os únicos objetos possíveis de serem usados são os que constam na lista definida no início da enum, declarados de acordo com um dos construtores

Exemplo de código

```
public enum ElementoQuimico{
    H (“Hidrogênio”, “H”, 1, 1.008),
    HE (“Hélio”, “He”, 2, 4.002);
     private String nome, símbolo;
     private int número;
     private double massaAtômica;
     ElementoQuimico (String nome, String
símbolo,
                int número, double massaAtômica) { ...
```

Percorrendo os valores

- Toda enumeração tem um método values () que devolve um vetor com todos os valores
- Pode-se usar o laço for each do java para acessar todos os valores

```
for (ElementoQuimico eq:
ElementoQuimico.values())
    System.out.println(eq.getNome());
```

Exemplo

- Criar uma enum Imposto, que tenha atributos para o nome, sigla e taxa. Ela deve conter três valores, IPI(30%), ICMS(25%) e PIS/COFINS (9.25%)
- Crie uma classe VideoGame com os atributos nome e preçoFinal, construtor e get/set
- Faça um método que devolva o preço do jogo sem o valor dos impostos. Sobrescreva o método toString para retornar algo semelhante.