Classes e Objetos. Soluções aos exercícios

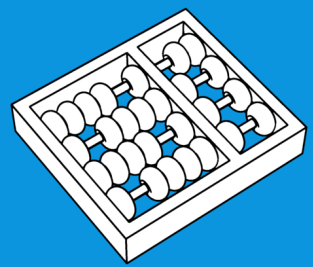
MC102 - Algoritmos e Programação de Computadores

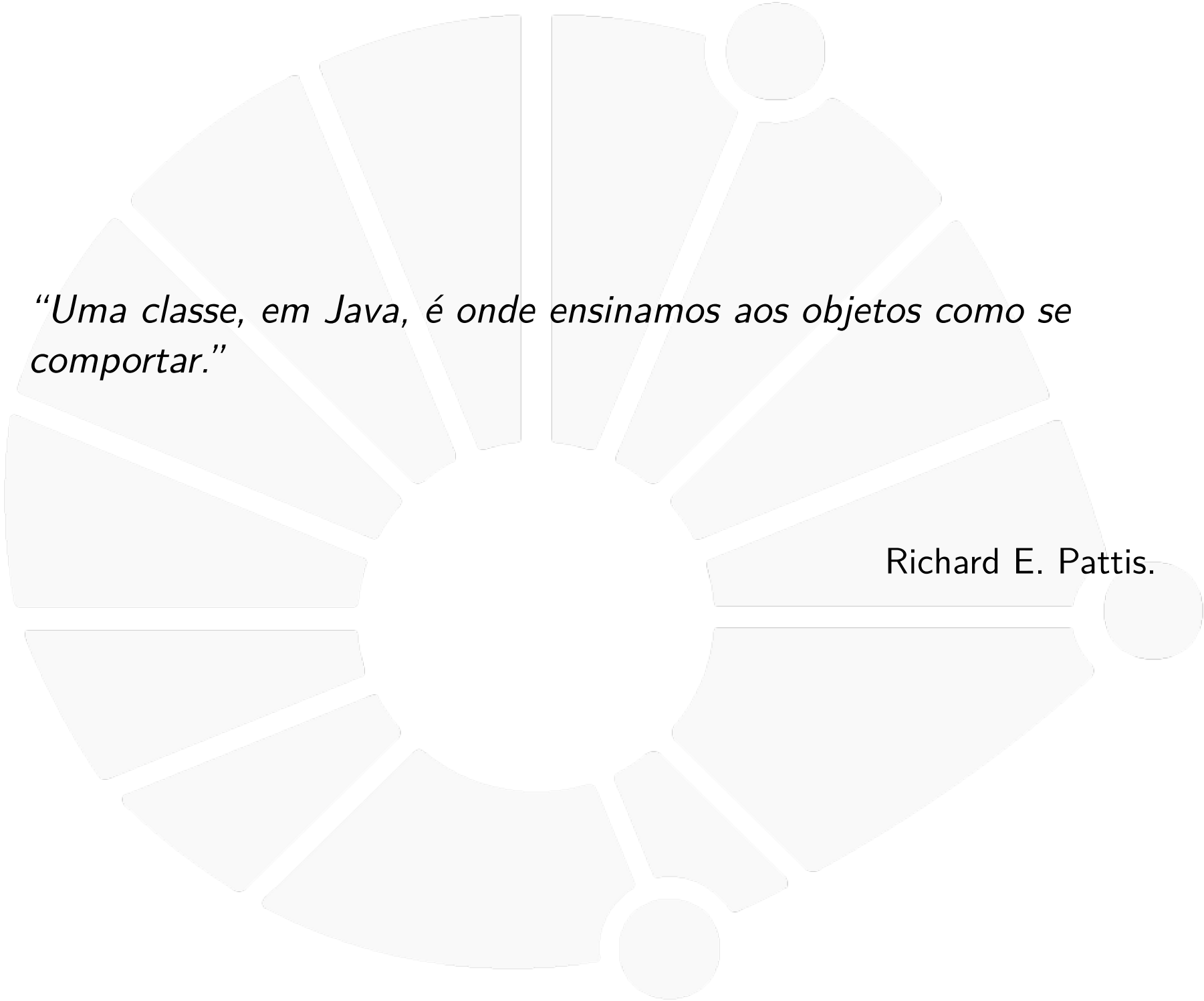
05/25

18

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP

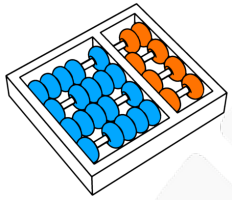




"Uma classe, em Java, é onde ensinamos aos objetos como se comportar."

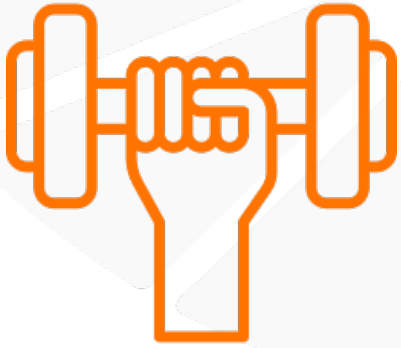
Richard E. Pattis.

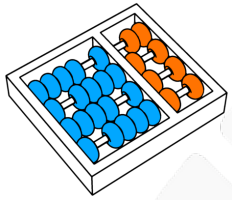
Exercícios



Classes e objetos

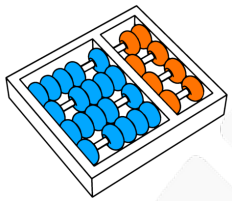
Soluções para os exercícios!





Exercícios

1. Crie uma classe **Turma** que:
   - Armazena estudantes.
   - Permite adicionar estudantes.
   - Permite imprimir os estudantes.
   - Permite imprimir os estudantes aprovados.
   - Permite imprimir os estudantes reprovados.
2. Refaça a classe Turma usando encapsulamento.
3. Refaça a classe Estudante usando encapsulamento de forma que a nota do estudante é a média aritmética de três notas.
4. Refaça a classe Turma usando dataclass.Dica para não ter problemas:
   - **from dataclasses import dataclass, field**.
   - Defina a lista de estudantes na turma da seguinte forma: **estudantes: list = field(default_factory=list)**

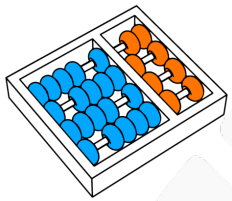


Soluções

Turma

```
1  class Turma:
2      def __init__(self):
3          self.estudantes = []
4
5      def adiciona(self, estudante):
6          self.estudantes.append(estudante)
7
8      def imprime(self):
9          for estudante in self.estudantes:
10             print(estudante)
11
12     def imprime_aprovados(self):
13         for estudante in self.estudantes:
14             if estudante.aprovado():
15                 print(estudante)
16
17     def imprime_reprovados(self):
18         for estudante in self.estudantes:
19             if not estudante.aprovado():
20                 print(estudante)
21
```

Note que nessa solução há muita repetição de código nas funções imprime.

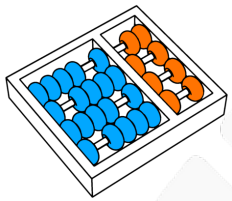


Soluções. Continuação

Turma com encapsulamento:

```
1  class Turma:
2      def __init__(self):
3          self._estudantes = []
4
5      def adiciona(self, estudante):
6          self._estudantes.append(estudante)
7
8      def _imprime(self, filtro):
9          for estudante in self.estudantes:
10             if filtro(estudante):
11                 print(estudante)
12
13     def imprime(self):
14         self._imprime(lambda x: True)
15
16     def imprime_aprovados(self):
17         self._imprime(lambda x: x.aprovado())
18
19     def imprime_reprovados(self):
20         self._imprime(lambda x: not x.aprovado())
21
```

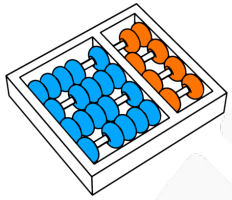
Ao usar encapsulamento, podemos ter uma função imprime filtrando os elementos por alguma regra e que não é para ser usada fora da classe. Ela é chamada dentro da classe, para imprimir todos (regra que deixa passar qualquer estudante da lista), para imprimir os aprovados (filtro que só é verdadeiro nos aprovados), etc.



Soluções. Continuação I

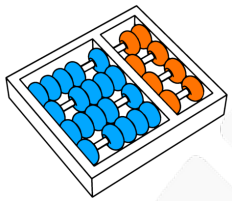
Estudantes com três notas:

```
1  class Estudante:
2
3
4      def __init__(self, nome, RA, curso, nota1, nota2, nota3):
5          self.nome = nome
6          self.RA = RA
7          self.curso = curso
8          self._notas = [nota1, nota2, nota3]
9
10     def _atribui(self, nota, i):
11         if nota < 0 or nota > 10:
12             raise ValueError("Nota inválida!")
13         self._notas[i] = nota
14
15     @property
16     def nota1(self):
17         return self._notas[0]
18
19     @nota1.setter
20     def nota1(self, nota):
21         self._atribui(nota, 0)
22
23     @property
24     def nota2(self):
25         return self._notas[1]
26
```



Soluções. Continuação II

```
27      @nota2.setter
28      def nota2(self, nota):
29          self._atribui(nota, 1)
30
31      @property
32      def nota3(self):
33          return self._notas[2]
34
35      @nota3.setter
36      def nota3(self, nota):
37          self._atribui(nota, 2)
38
39      @property
40      def media(self):
41          return sum(self._notas) / 3
```



Soluções. Continuação

Turma com dataclass:

```
1  from dataclasses import dataclass, field
2
3  class Turma:
4      _estudantes: list = field(default_factory=list)
5
6      def adiciona(self, estudante):
7          self._estudantes.append(estudante)
8
9      def _imprime(self, filtro):
10         for estudante in self.estudantes:
11             if filtro(estudante):
12                 print(estudante)
13
14     def imprime(self):
15         self._imprime(lambda x: True)
16
17     def imprime_aprovados(self):
18         self._imprime(lambda x: x.aprovado())
19
20     def imprime_reprovados(self):
21         self._imprime(lambda x: not x.aprovado())
22
```

Classes e Objetos. Soluções aos exercícios

MC102 - Algoritmos e Programação de Computadores

05/25

18

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP