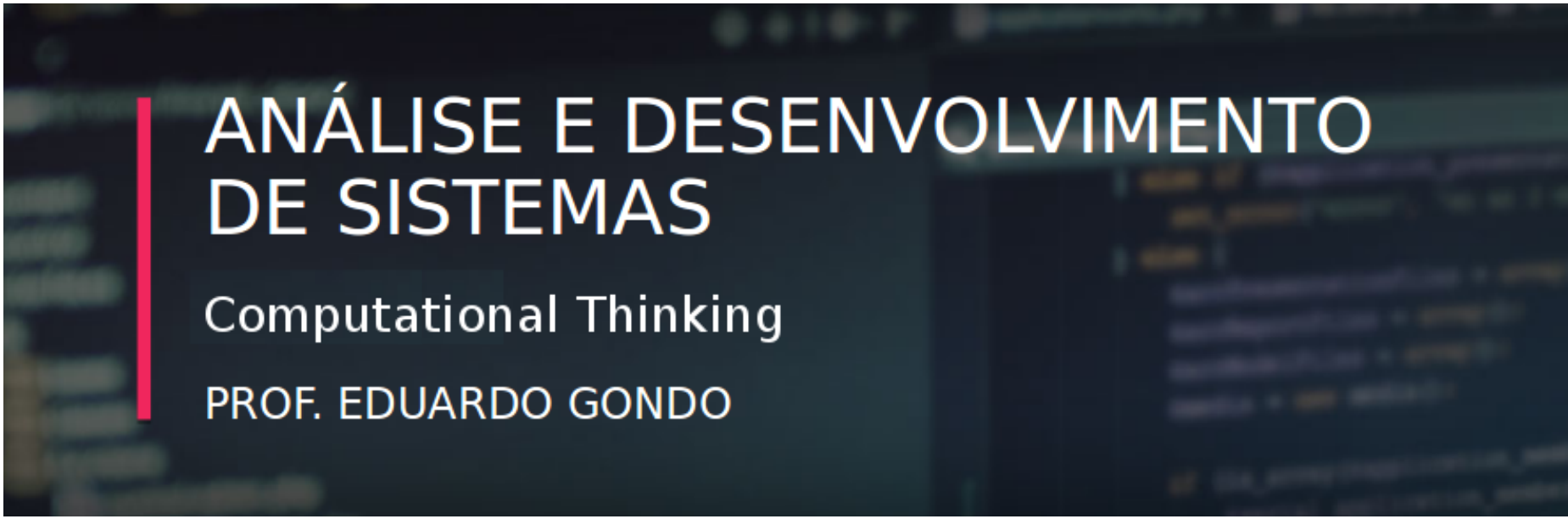
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
Computational Thinking
PROF. EDUARDO GONDO

Ordenação — BubbleSort

- imagine um vetor desenhado na vertical cujos elementos são bolhas em um tanque de água com densidade proporcional ao seu valor
- a tendência é que as bolhas mais "leves" (com valor menor) vão subir
- o algoritmo de ordenação bubble sort baseia-se nessa idéia
- pegamos o último elemento do vetor e comparamos com o anterior, se o elemento de baixo é menor que o elemento de cima trocamos ele de lugar
- repetimos esse processo até a primeira posição do vetor
- nessa situação na primeira posição temos o menor elemento do vetor na primeira posição

Ordenação — Simulação Bubble Sort

Vamos imaginar a situação de bolhas com densidades diferentes e movimentar os elementos do vetor:

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	12	84	5	10	**17**

Ordenação — Simulação Bubble Sort

Vamos imaginar a situação de bolhas com densidades diferentes e movimentar os elementos do vetor:

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	12	84	5	10	**17**

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	12	84	5	**10**	17

Simulação Bubble Sort

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	12	84	**5**	10	17

Simulação Bubble Sort

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	12	84	**5**	10	17

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	12	**5**	84	10	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	**5**	12	84	10	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	**5**	12	84	10	17

0	1	2	3	4	5	6	7	8	9
10	15	8	19	**5**	30	12	84	10	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
10	15	8	19	30	**5**	12	84	10	17

0	1	2	3	4	5	6	7	8	9
10	15	8	19	**5**	30	12	84	10	17

0	1	2	3	4	5	6	7	8	9
10	15	8	**5**	19	30	12	84	10	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
10	15	**5**	8	19	30	12	84	10	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
10	15	**5**	8	19	30	12	84	10	17

0	1	2	3	4	5	6	7	8	9
10	**5**	15	8	19	30	12	84	10	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
10	15	**5**	8	19	30	12	84	10	17

0	1	2	3	4	5	6	7	8	9
10	**5**	15	8	19	30	12	84	10	17

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	84	10	17

Ordenação — Simulação

Vamos repetir o processo já que a "bolha"mais leve está na 1ª posição do vetor:

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	84	10	**17**

Ordenação — Simulação

Vamos repetir o processo já que a "bolha"mais leve está na 1ª posição do vetor:

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	84	10	**17**

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	84	**10**	17

Ordenação — Simulação

Vamos repetir o processo já que a "bolha"mais leve está na 1ª posição do vetor:

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	84	10	**17**

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	84	**10**	17

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	12	**10**	84	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	**10**	12	84	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	**10**	12	84	17

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	**10**	30	12	84	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	30	**10**	12	84	17

0	1	2	3	4	5	6	7	8	9
5	10	15	8	19	**10**	30	12	84	17

0	1	2	3	4	5	6	7	8	9
5	10	15	8	**10**	19	30	12	84	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
5	10	15	**8**	10	19	30	12	84	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
5	10	15	**8**	10	19	30	12	84	17

0	1	2	3	4	5	6	7	8	9
5	10	**8**	15	10	19	30	12	84	17

Ordenação — Simulação

0	1	2	3	4	5	6	7	8	9
5	10	15	**8**	10	19	30	12	84	17

0	1	2	3	4	5	6	7	8	9
5	10	**8**	15	10	19	30	12	84	17

0	1	2	3	4	5	6	7	8	9
5	**8**	10	15	10	19	30	12	84	17

BubbleSort - implementação

- implemente o algoritmo descrito acima no seguinte método:

```
1       void subir(int[] v)
```

- agora modifique o método subir para que ele não suba o valor sempre até a posição 0, mas sim até a posição i passada como parâmetro:

```
1       void subir(int[] v, int i)
```

- após fazer o método subir, o algoritmo bubble sort fica:

```
1        public void bubbleSort(int[] v) {
2            for(int i = 0; i < v.length; i++) {
3                 this.subir(v, i);
4            }
5        }
```

Referência Bibliográfica

- **Fundamentos da Programação de Computadores: algoritmos, Pascal, C/C++ e Java**, Ascencio e Campos - 2ª ed., Pearson 2007
- **Lógica de Programação e Estrutura de Dados**, Puga e Rissetti - 2ª ed., Pearson Prentice Hall, 2008.
- **Algoritmos em linguagem C**, Feofiloff - Campus/Elsevier, 2009 (http://www.ime.usp.br/~pf/algoritmos-livro/index.html)
- **Construção de Algoritmos e Estruturas de Dados**, Forbellone e Eberspacher - Pearson Prentice Hall, 2010.
- **Projeto de Algoritmos com Implementações com Java e C++**, Ziviani - Thompson, 2006
- **Java como Programar**, Deitel e Deitel - 8ª ed., Pearson, 2010
- **Algoritmos**, Cormen, Leiserson, Rivest e Stein - Campus

Copyleft