

# Estrutura de Dados II

Prof. Me. Pietro M. de Oliveira



# Selectionsort

Ordenação por seleção



#### **Selectionsort**

# Também conhecido como técnica de **ordenação por seleção**

Simples implementação

Alto custo computacional

### **Estável**

Compara todos os elementos entre si

### Ineficiente, independentemente da distribuição

- Ordeando
- Parcialmente ordenado
- Não ordenado





### Algoritmo

```
Selectionsort(arranjo A)
```

- 1. Para  $i \leftarrow 0$  até comprimento [A] 1 faça
- 2.  $min \leftarrow i$
- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[i] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca  $A[i] \leftrightarrow A[min]$



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01





### Selectionsort(arranjo A)

- 1. Para  $i \leftarrow 0$  até comprimento [A] 1 faça
- 2. *min ← i*
- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se *A[j] < A[min]* então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca **A[i] ↔ A[min]**

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01

/1



```
Selectionsort(arranjo A)
```

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento [A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



### Selectionsort(arranjo A)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento [A] - 1 faça
```

- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca *A[i]* ↔ *A[min]*

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



```
Selectionsort(arranjo A)
```

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



```
Selectionsort(arranjo A)

1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



```
Selectionsort(arranjo A)
```

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01

n



```
Selectionsort(arranjo A)
```

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01



### Selectionsort(arranjo A)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento [A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01

δI



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
26	32	46	19	15	67	81	22	55	01

min



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento [A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9	
01	32	46	19	15	67	81	22	55	26	

min



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. 
$$troca_i A[i] \leftrightarrow A[min]$$

0		2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



### **Selectionsort**(arranjo **A**)

- 1. Para  $i \leftarrow 0$  até comprimento[A] 1 faça
- 2. *min ← i*
- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se *A[j] < A[min]* então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca  $A[i] \leftrightarrow A[min]$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



### Selectionsort(arranjo A)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



### **Selectionsort**(arranjo **A**)

- Para  $i \leftarrow 0$  até comprimento [A] 1 faça
- $min \leftarrow i$
- 3. Para  $i \leftarrow i+1$  até comprimento [A] faça
- Se **A[j] < A[min]** então
- 5.  $min \leftarrow i$
- 6. Se *i* ≠ *min* então
- troca **A[i]** ↔ **A[min]** 7.

0		2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



### Selectionsort(arranjo A)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



### **Selectionsort**(arranjo **A**)

- 1. Para  $i \leftarrow 0$  até comprimento[A] 1 faça
- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca  $A[i] \leftrightarrow A[min]$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



### Selectionsort(arranjo A)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento [A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



```
Selectionsort(arranjo A)
1. Para i ← 0 até comprimento[A] - 1 faça
2. min ← i
```

3. Para  $j \leftarrow i+1$  até comprimento[A] faça

4. Se *A[j] < A[min]* então

5.  $min \leftarrow j$ 

6. Se *i ≠ min* então

7. troca *A[i]* ↔ *A[min]* 

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26

mir



```
Selectionsort(arranjo A)
```

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26



```
Selectionsort(arranjo A)
```

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0		2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26

min



94

### Selectionsort(arranjo A)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
01	32	46	19	15	67	81	22	55	26

min



### Selectionsort(arranjo A)

- 1. Para  $i \leftarrow 0$  até comprimento[A] 1 faça
- 2. *min ← i*
- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i* ≠ *min* então
- 7. troca *A[i]* ↔ *A[min]*

0	(1)	2	3	4	5	6	7	8	9
01	15	46	19	32	67	81	22	55	26



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

7. troca 
$$A[i] \leftrightarrow A[min]$$

0	1	2	3	4	5	6	7	8	9
01	15	19	46	32	67	81	22	55	26



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca  $A[i] \leftrightarrow A[min]$

0	1	2	3	4	5	6	7	8	9	
01	15	19	22	32	67	81	46	55	26	



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

3. Para 
$$j \leftarrow i+1$$
 até comprimento[A] faça

5. 
$$min \leftarrow j$$

0	1	2	3	4	5	6	7	8	9
01	15	19	22	26	67	81	46	55	32



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca *A[i]* ↔ *A[min]*

0	1	2	3	4	5	6	7	8	9
01	15	19	22	26	32	81	46	55	67



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

- 3. Para  $j \leftarrow i+1$  até comprimento [A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca  $A[i] \leftrightarrow A[min]$

0	1	2	3	4	5	6	7	8	9	
01	15	19	22	26	32	46	81	55	67	1

min



### **Selectionsort**(arranjo **A**)

```
1. Para i \leftarrow 0 até comprimento[A] - 1 faça
```

```
2. min ← i
```

- 3. Para  $j \leftarrow i+1$  até comprimento[A] faça
- 4. Se **A[j] < A[min]** então
- 5.  $min \leftarrow j$
- 6. Se *i ≠ min* então
- 7. troca  $A[i] \leftrightarrow A[min]$

0	1	2	3	4	5	6	7	8	9	
01	15	19	22	26	32	46	55	81	67	101
								min		



### **Selectionsort**(arranjo **A**)

- Para  $i \leftarrow 0$  até comprimento [A] 1 faça
- $min \leftarrow i$
- 3. Para  $i \leftarrow i+1$  até comprimento [A] faça
- Se *A[i] < A[min]* então
- 5.  $min \leftarrow i$
- 6. Se *i ≠ min* então
- 7. troca *A[i]* ↔ *A[min]*

0	1	2	3	4	5	6	7	8	9	
01	15	19	22	26	32	46	55	67	81	102



# Considerações Preliminares

- Imagine que o arranjo a ser ordenado possui
   n elementos
- Bubblesort e Selectionsort irão fazer, obrigatoriamente, cerca de n² operações para completar a ordenação
- Conclusão: caso o arranjo não seja muito grande, não há grandes problemas em utilizá-los!



# Estrutura de Dados II

Prof. Me. Pietro M. de Oliveira