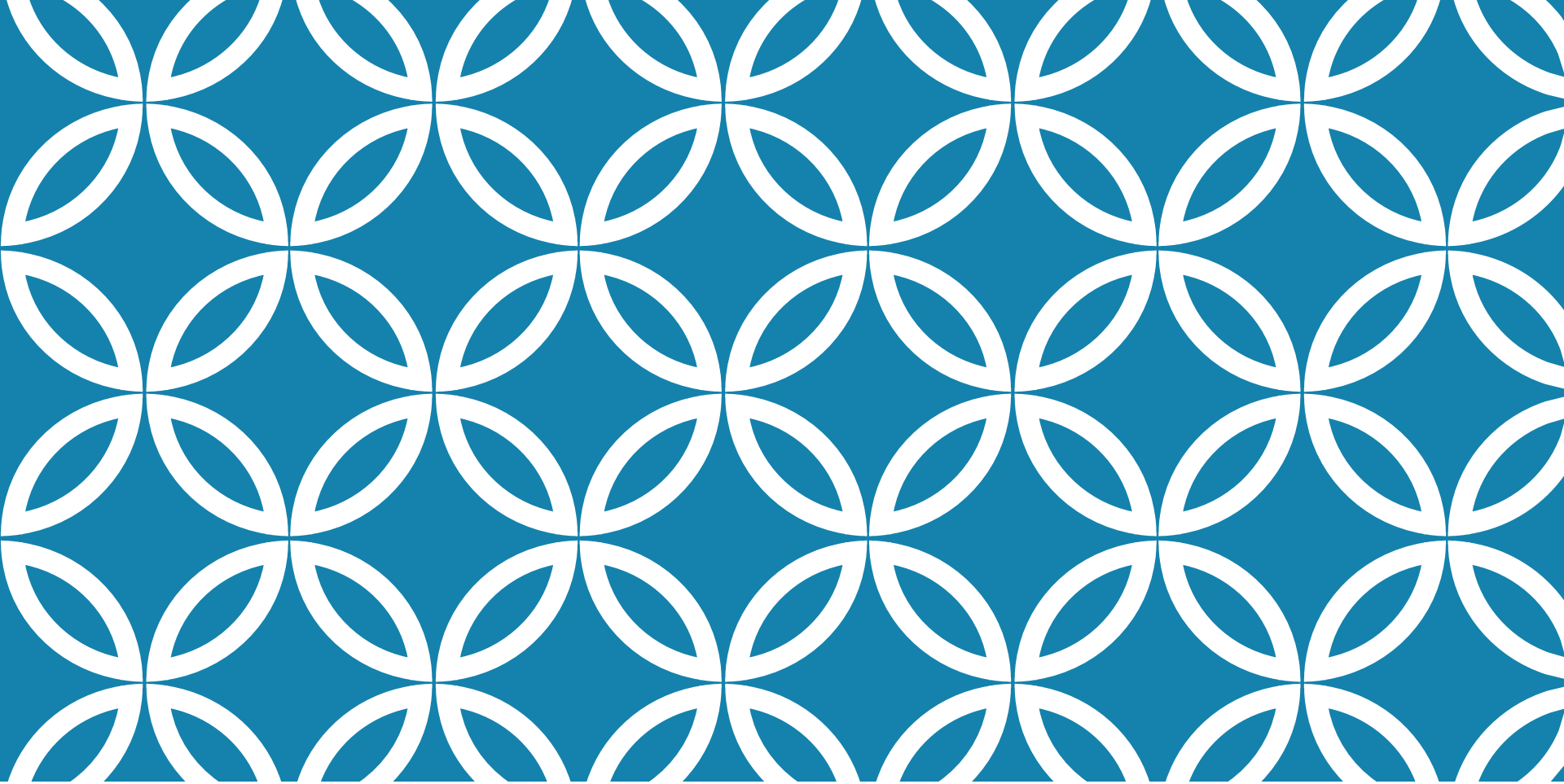

INSERÇÃO, REMOÇÃO E ORDENAÇÃO

Prof. André Vignatti

REMOVENDO ELEMENTOS

porque passar a **posição**, ao invés do **elemento** a ser removido?

```
procedure remove_vetor_ordenado (pos: integer; var v: vetor_r; var n: integer);
var i: integer;

begin
      for i:= pos to n–1 do
            v[i]:= v[i+1];
      n:= n – 1;
end;
```

Ex: removendo elemento 12

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	2	7	12	15	18	19	21	23	27	?	?	?	?	?	...	?	?	?	?

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	2	7	15	18	19	21	23	27	27	?	?	?	?	?	...	?	?	?	?

INSERINDO EM VETOR ORDENADO

```
procedure insere_vetor_ordenado (x: real; var v: vetor_r; var n: integer);
var i: integer;

begin
      v[0]:= x;
      i:= n;
      while x < v[i] do
      begin
            v[i+1]:= v[i];
            i:= i - 1;
      end;
      v[i+1]:= x;
      n:= n + 1;
end;
```

ex: inserindo elemento 17

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	2	7	12	15	18	19	21	23	27	?	?	?	?	?	...	?	?	?	?

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	2	7	12	15	18	18	19	21	23	27	?	?	?	?	...	?	?	?	?

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	2	7	12	15	17	18	19	21	23	27	?	?	?	?	...	?	?	?	?

ORDENAÇÃO POR SELEÇÃO (SELECTION SORT)

ideia: seleciona o 1° menor elemento, troca com 1ª posição, seleciona o 2° menor elemento, troca com a 2ª posição, ...

vetor original:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
15	12	27	23	7	2	0	18	19	21	?	?	?	?	?	...	?	?	?	?

execução:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	12	27	23	7	2	**15**	18	19	21	?	?	?	?	?	...	?	?	?	?

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	197	198	199	200
0	**2**	27	23	7	**12**	15	18	19	21	?	?	?	?	?	...	?	?	?	?
0	2	**7**	23	**27**	12	15	18	19	21	?	?	?	?	?	...	?	?	?	?
0	2	7	**12**	27	**23**	15	18	19	21	?	?	?	?	?	...	?	?	?	?
0	2	7	12	**15**	23	**27**	18	19	21	?	?	?	?	?	...	?	?	?	?
0	2	7	12	15	**18**	27	**23**	19	21	?	?	?	?	?	...	?	?	?	?
0	2	7	12	15	18	**19**	23	**27**	21	?	?	?	?	?	...	?	?	?	?
0	2	7	12	15	18	19	**21**	27	**23**	?	?	?	?	?	...	?	?	?	?
0	2	7	12	15	18	19	21	**23**	**27**	?	?	?	?	?	...	?	?	?	?

ORDENAÇÃO POR SELEÇÃO (SELECTION SORT)

```
procedure selecao (var v: vetor_r; n: integer);
var i, j, pos_menor: integer; aux: real;

begin
      for i:= 1 to n-1 do
      begin
            (* acha a posicao do menor a partir de i *)
            pos_menor:= i;
            for j:= i+1 to n do (* inicia a partir de i+1 *)
                  if v[j] < v[pos_menor] then
                        pos_menor:= j;

            aux:= v[pos_menor]; (* troca os elementos *)
            v[pos_menor]:= v[i];
            v[i]:= aux;
      end;
end;
```

ANÁLISE – SELECTION SORT

análise simples: contar uma operação feita pelo algoritmo

- vamos contar as *comparações*

```
for i:= 1 to n–1 do
begin

      pos_menor:= i;
      for j:= i+1 to n do
            if v[j] < v[pos_menor] then
                  pos_menor:= j;

      aux:= v[pos_menor];
      v[pos_menor]:= v[i];
      v[i]:= aux;
end;
```

quando $i = 1$:
- j vai de 2 até n
- `if` é executado $n - 1$ vezes

quando $i = 2$:
- j vai de 3 até n
- `if` é executado $n - 2$ vezes

quando $i = 3$:
- j vai de 4 até n
- `if` é executado $n - 3$ vezes

⋮

quando $i = n - 2$:
- j vai de $n - 1$ até n
- `if` é executado 2 vezes

quando $i = n - 1$:
- j vai de n até n
- `if` é executado 1 vez

ANÁLISE – SELECTION SORT

contando as comparações do selection sort:

quando $i = 1$:
- `if` é executado $n - 1$ vezes

quando $i = 2$:
- `if` é executado $n - 2$ vezes

quando $i = 3$:
- `if` é executado $n - 3$ vezes

$\vdots$

quando $i = n - 2$:
- `if` é executado 2 vezes

quando $i = n - 1$:
- `if` é executado 1 vez

$$S = (n-1) + (n-2) + (n-3) + \cdots + 2 + 1$$

ANÁLISE – SELECTION SORT

contando as comparações do selection sort:

$$S = (n-1) + (n-2) + (n-3) + \cdots + 2 + 1$$

$$= \sum_{i=1}^{n-1} i$$

Carl Friedrich Gauss
(1777 - 1855)

$$= \frac{n(n-1)}{2}$$

$$\approx \frac{n^2}{2} \approx \boxed{n^2}$$

TEMPO DE EXECUÇÃO - ORDENAÇÃO

algoritmos "ingênuos": n^2

algoritmos "espertos": $n \log_2 n$

n	n^2	$n \log_2 n$
10	100	320
100	10000	650
1000	1 milhão	10000
1 milhão	1 trilhão	20 milhões
1 bilhão	10^9 bilhões	30 bilhões

ABRE PARÊNTESES: VELOCIDADE

medida popular, porém imprecisa:

núm. de instruções por segundo

atualmente: alguns bilhões de instruções por segundo

ordenação com 1 bilhão instr/seg

n	ingênuo	esperto
10	-	-
100	0,00001 s	-
1000	0,001 s	0,00001s
1 milhão	16,6 min	0,2 s
1 bilhão	31,7 anos	30 s