Insertion

Sort

InsertionSort

```
InsertionSort(A[1..n], n)
    Para i = 2 até n
        atual = A[i]
        j = i - 1
        Enquanto j > 0 e A[j] > atual
            A[j+1] = A[j]
            j = j - 1
        A[j+1] = atual
```

InsertionSort - Análise

```
InsertionSort(A[1..n], n)
    Para i = 2 até n
        atual = A[i]
        j = i - 1
        Enquanto j > 0 e A[j] > atual
            A[j+1] = A[j]
            j = j - 1
        A[j+1] = atual
```

InsertionSort - Análise

InsertionSort(A[1..n], n)

Para $i = 2$ até n | $O(n)$ A

  atual = A[i]

  $j = i - 1$ | $\Theta(1) \cdot \Theta(n)$ B

  Enquanto $j > 0$ e $A[j] >$ atual | $\overbrace{O(n) \cdot \Theta(n)}^{O(n^2)}$ C

    $A[j+1] = A[j]$

    $j = j - 1$ | $\Theta(1) \cdot O(n) \cdot \Theta(n) = O(n^2)$ D

  $A[j+1] =$ atual | $\Theta(1) \cdot \Theta(n)$ E

$$T(n) = A + B + C + D + E = O(n) + \Theta(n) + O(n^2) + O(n^2) + O(n) = O(n^2)$$

InsertionSort - Análise

- O laço da linha 1 executa $O(n)$ vezes
- As linhas 2, 3, 7 levam tempo constante e executam $O(n)$ vezes. Assim, esse trecho leva $O(n)$ para executar no total.

```
InsertionSort(A[1..n], n)
1    Para i = 2 até n
2        atual = A[i]
3        j = i - 1
4        Enquanto j > 0 e A[j] > atual
5            A[j+1] = A[j]
6            j = j - 1
7        A[j+1] = atual
```

InsertionSort - Análise

- Para uma iteração fixa do laço da linha 1, temos que o laço da linha 4 executa $O(n)$ vezes. O custo de uma iteração do laço da linha 4 leva tempo constante. Assim, o custo com o trecho das linhas 4-6 para uma iteração fixa do laço da linha 1 é $O(n)$. Como o laço da linha 1 executa $O(n)$ vezes, temos que o custo total com as linhas 4-6 ao longo da execução do algoritmo é $O(n^2)$

```
InsertionSort(A[1..n], n)
1   Para i = 2 até n
2       atual = A[i]
3       j = i-1
4       Enquanto j>0 e A[j] > atual
5           A[j+1] = A[j]
6           j = j-1
7       A[j+1] = atual
```

InsertionSort - Análise

- Portanto, o custo total do algoritmo é $O(n^2)$

```
InsertionSort(A[1..n], n)
    Para i = 2 até n
        atual = A[i]
        j = i - 1
        Enquanto j > 0 e A[j] > atual
            A[j+1] = A[j]
            j = j - 1
        A[j+1] = atual
```

Correção

InsertionSort

```
InsertionSort(A[1..n], n)
    Para i = 2 até n
        atual = A[i]
        j = i - 1
        Enquanto j > 0 e A[j] > atual
            A[j+1] = A[j]
            j = j - 1
        A[j+1] = atual
```

- Seja C uma cópia do vetor A antes do enquanto da linha 2 começar

$Q(t) =$ " antes da t-ésima iteração começar, vale que

(i) $i = t+1$

(ii) $A[1..i-1]$ está ordenado

(iii) A contém os mesmos elementos de C "

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
C	3	10	2	7	8	9	1	6	11	4	12	5	14	13
A	1	2	3	7	8	9	10	6	11	4	12	5	14	13

$\uparrow$
i

- Seja B uma cópia do vetor A antes do enquanto da linha 4 começar

$P(\ell) =$ " antes da ℓ-ésima iteração começar, vale que

(i) $j = i - \ell$

(ii) $A[j+2..i] > \text{atual}$

(iii) $A[j+2..i] = B[j+1..i-1]$

(iv) $A[1..j] = B[1..j]$

(v) $A[i+1..n] = B[i+1..n]$ "

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
B	1	2	3	7	8	9	10	6	11	4	12	5	14	13
A	1	2	3	7	7 $\cancel{8}$	8 $\cancel{9}$	9 $\cancel{10}$	10 $\cancel{6}$	11	4	12	5	14	13
			j					i						

Atual = 6

Teo. O Algoritmo InsertionSort resolve o problema da ordenação

Demo

Por $Q(t)$, temos que $i = t+1$. Portanto o laço da linha 1 termina após n iterações. Por $Q(n+1)$, temos que $A[1..n]$ está ordenado e contém os mesmos elementos do vetor C $\square$

$Q(t)$ = " antes da t-ésima iteração começar, vale que

(i) $i = t+1$

(ii) $A[1..i-1]$ está ordenado

(iii) A contém os mesmos elementos de C "

```
InsertionSort(A[1..n], n)
1 Para i = 2 até n
2     atual = A[i]
3     j = i - 1
4     Enquanto j > 0
            e A[j] > atual
5         A[j+1] = A[j]
6         j = j - 1
7     A[j+1] = atual
```

$Q(t) =$ " antes da t-ésima iteração começar, vale que (i) $i = t+1$ (ii) $A[1..i-1]$ está ordenado (iii) A contém os mesmos elementos de C "

1º passo é reescrever o enunciado usando t sempre que possível

Variável de indução

$Q(t) =$ " antes da t-ésima iteração começar, vale que (i) $i = t+1$ (ii) $A[1..t]$ está ordenado (iii) A contém os mesmos elementos de C "

$Q(t) =$ " antes da t-ésima iteração começar, vale que (i) $i = t+1$ (ii) $A[1..t]$ está ordenado (iii) A contém os mesmos elementos de C "

Base $t = 1$

- antes da primeira iteração, temos que $i = 2 = t+1$, então (i) vale
- $A[1..t] = A[1..1] = A[1]$, portanto está trivialmente ordenado e (ii) tbm vale
- $A = C$, pela definição de C. Logo (iii) tbm vale.

```
InsertionSort(A[1..n], n)
Para i = 2 até n
    atual = A[i]
    j = i - 1
    Enquanto j > 0
          e A[j] > atual
        A[j+1] = A[j]
        j = j - 1
    A[j+1] = atual
```

$Q(t) =$ " antes da t-ésima iteração começar, vale que (i) $i = t+1$ (ii) $A[1..t]$ está ordenado (iii) A contém os mesmos elementos de C "

Passo $Q(t-1) \Rightarrow Q(t)$

① Suponha que $Q(t-1)$ seja válida, i.e., antes do início da $(t-1)$-ésima iteração, temos que ⓐ $i = t$; ⓑ $A[1..t-1]$ está ordenado; ⓒ A contém os mesmos elementos de C.

② Vemos mostrar que a invariante continua válida no início da iteração t. Para isso, suponha que a iteração $t-1$ ocorre. Portanto $n \geq i = t$

③ Na linha 2 fazemos $Atual = A[t]$

④ Na linha 3 fazemos $j = i - 1 = t - 1$

⑤ Então atingimos a linha 4. A invariante do laço da linha 4 diz que

```
InsertionSort(A[1..n], n)
1 Para i = 2 até n
2     atual = A[i]
3     j = i - 1
4     Enquanto j > 0
          e A[j] > atual
5             A[j+1] = A[j]
6             j = j - 1
7     A[j+1] = atual
```

$P(\ell) =$ "antes da ℓ-ésima iteração começar, vale que (A) $j = i-\ell$

(B) $A[j+2..i] >$ atual; (C) $A[j+2..i] = B[j+1..i-1]$ (D) $A[1..j] = B[1..j]$

(E) $A[i+1..n] = B[i+1..n]$"

① Vou fazer por passos, você poderia fazer direto

② Primeiro, vamos trocar tudo o que for possível pela variável de indução

$P(\ell) =$ "antes da ℓ-ésima iteração começar, vale que (A) $j = i-\ell$

(B) $A[i-\ell+2..i] >$ atual; (C) $A[i-\ell+2..i] = B[i-\ell+1..i-1]$ (D) $A[1..i-\ell] = B[1..i-\ell]$

(E) $A[i-\ell+1..n] = B[i-\ell+1..n]$"

③ note que sabemos que $i = t$ e t é a variável de indução da invariante externa. Então vamos trocar tbm

$P(\ell) =$ "antes da ℓ-ésima iteração começar, vale que (A) $j = i-\ell$

(B) $A[i-\ell+2 \,..\, i] >$ atual; (C) $A[i-\ell+2 \,..\, i] = B[i-\ell+1 \,..\, i-1]$ (D) $A[1 \,..\, i-\ell] = B[1 \,..\, i-\ell]$

(E) $A[i-\ell+1 \,..\, n] = B[i-\ell+1 \,..\, n]$"

③ note que sabemos que $i = t$ e t é a variável de indução da invariante externa. Então vamos trocar tbm

$P(\ell) =$ "antes da ℓ-ésima iteração começar, vale que (A) $j = t-\ell$

(B) $A[t-\ell+2 \,..\, t] >$ atual; (C) $A[t-\ell+2 \,..\, t] = B[t-\ell+1 \,..\, t-1]$ (D) $A[1 \,..\, t-\ell] = B[1 \,..\, t-\ell]$

(E) $A[t-\ell+1 \,..\, n] = B[t-\ell+1 \,..\, n]$"

④ Note que os temos $t, \ell, n, 1$ são constantes em uma dada iteração. Isso facilita na hora da escrita porque os valores ñ são atualizados

$P(\ell)$ = "antes da ℓ-ésima iteração começar, vale que (A) $j = t-\ell$

(B) $A[t-\ell+2..t] > atual$; (C) $A[t-\ell+2..t] = B[t-\ell+1..t-1]$ (D) $A[1..t-\ell] = B[1..t-\ell]$

(E) $A[t-\ell+1..n] = B[t-\ell+1..n]$"

⑥ Por (A), $j = t-\ell$. Assim o laço da linha 4 termina após $t-1$ iterações (no início da t-ésima iteração $j = t-t$)

```
InsertionSort(A[1..n], n)
1 Para i = 2 até n
2      atual = A[i]
3      j = i - 1
4      Enquanto j > 0
              e A[j] > atual
5           A[j+1] = A[j]
6           j = j - 1
7      A[j+1] = atual
```

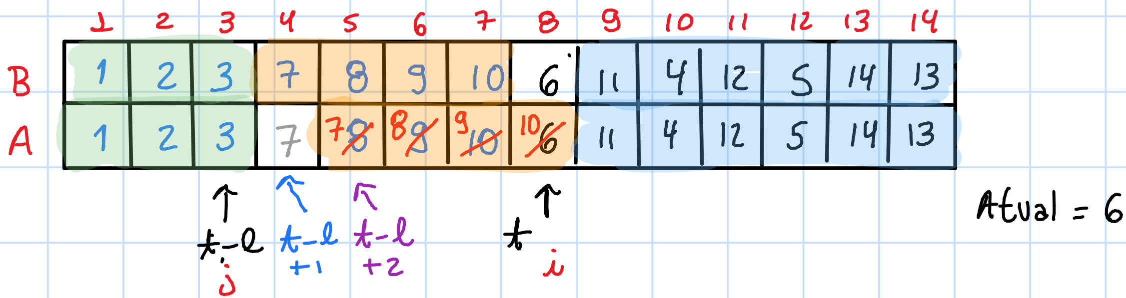
$P(\ell)$ = "antes da ℓ-ésima iteração começar, vale que (A) $j = t - \ell$

(B) $A[t-\ell+2..t] > atual$; (C) $A[t-\ell+2..t] = B[t-\ell+1..t-1]$ (D) $A[1..t-\ell] = B[1..t-\ell]$

(E) $A[t-\ell+1..n] = B[t-\ell+1..n]$"

(6) Por (A), $j = t - \ell$. Assim o laço da linha 4 termina após $t-1$ iterações (no início da t-ésima iteração $j = t - t$)

(7) Então o algoritmo executa a linha 7 fazendo $A[j+1] = A[t-\ell] = atual$.



(8) Por (b), (D), (C), temos que $A[1..t-\ell]$ e $A[t-\ell+2..t]$ estão ordenados e que os elementos de $A[t-\ell+2..t]$ são maiores ou iguais aos elementos de $A[1..t-\ell]$.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
B	1	2	3	7	8	9	10	6	11	4	12	5	14	13
A	1	2	3	7	7 ~~8~~	8 ~~9~~	9 ~~10~~	10 ~~6~~	11	4	12	5	14	13

$t-\ell$ (j) $\quad t-\ell+1 \quad t-\ell+2 \quad t$ (i) $\qquad$ Atual = 6

(9) Por (B), (7) e (8), temos que $A[1..t]$ está ordenado e, consequentemente, (ii) é satisfeito.

(10) Por (D), (E), (C), (3) e (7), temos que os elementos de A são os mesmos de C, logo, (iii) vale.

(11) Após executar a linha 7, o laço finaliza fazendo

$$i = i + 1 = t + 1$$

Assim, (i) também é satisfeito.

Por (9), (10) e (11), temos que o passo vale $\square$

$P(\ell)$ = "antes da ℓ-ésima iteração começar, vale que

(i) $j = i - \ell$

(ii) $A[j+2..i] > atual$

(iii) $A[j+2..i] = B[j+1..i-1]$

(iv) $A[1..j] = B[1..j]$

(v) $A[i+1..n] = B[i+1..n]$ "

Exercício: Provar que $P(\ell)$ é uma invariante de laço do laço da linha 4.