Para realizar a **contagem de tempo de execução** do algoritmo `INSERTION-SORT`, devemos analisar **linha por linha** e identificar a **quantidade de vezes que cada passo básico (tempo *tt*) é executado**, com base nas instruções e critérios definidos:

Critérios:

Cada instrução abaixo vale um tempo *tt*:

- Atribuição de valores
- Operação lógica
- Operação aritmética
- Operação de acesso
- Operação de retorno

Algoritmo `INSERTION-SORT(A)`

```
1 for j ← 2 to length[A]
2     key ← A[j]
3     ▷ Comentário
4     i ← j - 1
5     while i > 0 and A[i] > key
6         A[i + 1] ← A[i]
7         i ← i - 1
8     A[i + 1] ← key
```

Análise de Complexidade e Contagem de Tempo:

Linha 1: `for j ← 2 to length[A]`

- Executa $n-1n$ - 1 vezes (de 2 até nn).
- Contém:
    - Atribuição: tt
    - Comparação de laço: tt
    - Incremento: tt

Custo total:

→ 1x inicialização + (n−1)x comparação + (n−2)x incremento

→ $t+(n-1)t+(n-2)t=(2n-2)tt + (n-1)t + (n-2)t = (2n - 2)t$

Linha 2: `key ← A[j]`

- Executa $n-1n - 1$ vezes
- Operações:
  - Acesso a A[j]: tt
  - Atribuição à key: tt

Total: $2(n-1)t2(n-1)t$

Linha 4: `i ← j - 1`

- Executa $n-1n - 1$ vezes
- Operações:
  - Aritmética (j - 1): tt
  - Atribuição: tt

Total: $2(n-1)t2(n-1)t$

Linha 5: `while i > 0 and A[i] > key`

- No **pior caso** (vetor reversamente ordenado), o `while` executa até $j-1j-1$ vezes a cada iteração do `for`.
- Cada teste `i > 0 and A[i] >` key envolve:
  - Comparação lógica: tt
  - Acesso A[i]: tt
  - Comparação A[i] > key: tt
  - Comparação final do `while`: tt

Total por iteração j: $4(j-1)t4(j-1)t$

Somando para todos os $j∈[2,n]j \in [2,n]$:

$\sum_{j=2}^{n} 4(j-1)t = 4t \sum_{j=1}^{n-1} j = 4t \cdot \frac{(n-1)n}{2} = 2n(n-1)t$

Linha 6: `A[i + 1] ← A[i]`

- Executa no pior caso até $j-1$ vezes para cada j.
- Operações:
  - Acesso A[i]: t
  - Aritmética i + 1: t
  - Atribuição: t

Total por j: $3(j-1)t$

Somando:

$\sum_{j=2}^{n} 3(j-1)t = 3t \cdot \frac{(n-1)n}{2} = \frac{3n(n-1)}{2}t$

Linha 7: `i ← i - 1`

- Executa o mesmo número de vezes da linha 6: $j-1$
- Aritmética: t
- Atribuição: t

Total por j: $2(j-1)t$

Somando:

$\sum_{j=2}^{n} 2(j-1)t = 2t \cdot \frac{(n-1)n}{2} = n(n-1)t$

Linha 8: `A[i + 1] ← key`

- Executa $n-1$ vezes (após o fim de cada `while`)
- Aritmética: t
- Atribuição: t

Total: $2(n-1)t$

Soma Final (Pior Caso)

Linha	Tempo Total
Linha 1	$(2n-2)t(2n - 2)t$
Linha 2	$2(n-1)t2(n-1)t$
Linha 4	$2(n-1)t2(n-1)t$
Linha 5	$2n(n-1)t2n(n - 1)t$
Linha 6	$3n(n-1)2t\|frac{3n(n - 1)}{2}t$
Linha 7	$n(n-1)tn(n - 1)t$
Linha 8	$2(n-1)t2(n-1)t$

Total geral (T(n)):

$T(n)=(2n-2+2n-2+2n(n-1)+3n(n-1)2+n(n-1)+2(n-1))tT(n) = (2n - 2 + 2n - 2 + 2n(n - 1) + \|frac{3n(n - 1)}{2} + n(n - 1) + 2(n-1))t$

Simplificando:

$T(n)=[4n-4+2n(n-1)+3n(n-1)2+n(n-1)]tT(n) = \|left[4n - 4 + 2n(n-1) + \|frac{3n(n-1)}{2} + n(n-1) \|right]t$
$T(n)=[4n-4+(2n(n-1)+3n(n-1)2+n(n-1))]tT(n) = \|left[4n - 4 + \|left(2n(n-1) + \|frac{3n(n-1)}{2} + n(n-1)\|right)\|right]t$
$T(n)=[4n-4+(4n(n-1)2+3n(n-1)2+2n(n-1)2)]tT(n) = \|left[4n - 4 + \|left(\|frac{4n(n-1)}{2} + \|frac{3n(n-1)}{2} + \|frac{2n(n-1)}{2}\|right)\|right]t$
$T(n)=[4n-4+9n(n-1)2]tT(n) = \|left[4n - 4 + \|frac{9n(n-1)}{2} \|right]t$

Resultado:

Tempo de execução (pior caso):

$T(n)=(9n(n-1)2+4n-4)tT(n) = \|left(\|frac{9n(n-1)}{2} + 4n - 4 \|right)t$