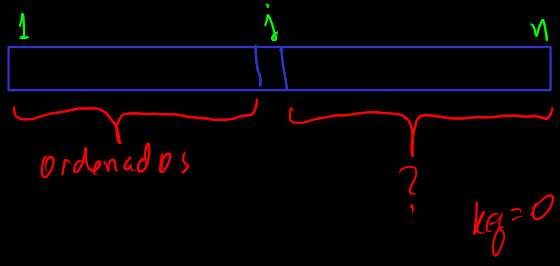
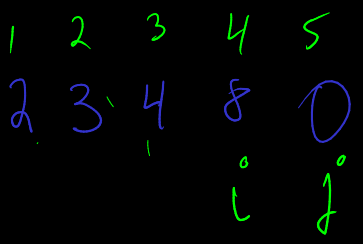
Prova de Corretude da Ordenação por Inserção

```
INSERTIONSORT(A)
1. FOR j = 2 TO A.size DO
2.     KEY = A[j]
3.     i = j - 1
4.     WHILE i > 0 AND A[i] > KEY DO
5.         A[i+1] = A[i]
6.         i = i - 1
7.     END WHILE
8.     A[i+1] = KEY
9. END FOR
```

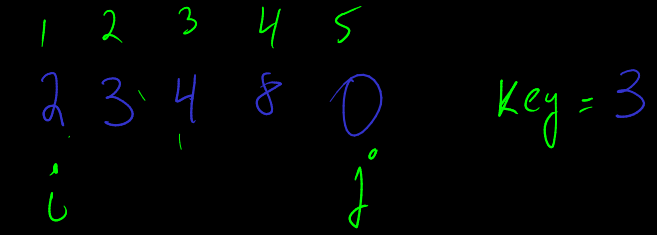
INVARIANTE:

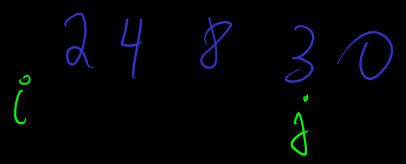
No início de cada iteração do laço for das linhas 1-9, o subvetor $A[1..j-1]$ consiste nos elementos que originalmente estavam em $A[1..j-1]$, mas em ordem crescente.





Inicialização Antes da primeira execução do laço FOR, $j=2$. Assim, $A[1..j-1] = A[1..(2)-1] = A[1..1]$. Como não houve permuta de qualquer elemento antes do laço, o elemento $A[1]$ está em sua posição original. Como o subvetor $A[1..1]$ é unitário, ele está trivialmente ORDENADO. Portanto, a invariante é verdadeira logo antes da primeira iteração do laço.

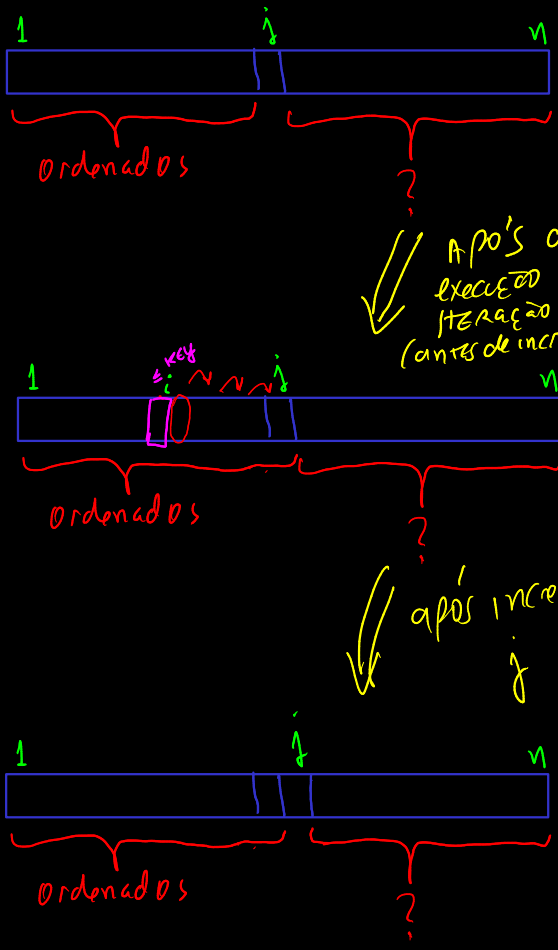




```
INSERTIONSORT(A)
1. FOR j = 2 TO A.size DO
2.     KEY = A[j]
3.     i = j - 1
4.     WHILE i > 0 AND A[i] > KEY DO
5.         A[i+1] = A[i]
6.         i = i - 1
7.     END WHILE
8.     A[i+1] = KEY
9. END FOR
```

INVARIANTE:
No início de cada iteração do laço for das linhas 1-9, o subvetor $A[1..j-1]$ consiste nos elementos que originalmente estavam em $A[1..j-1]$, mas em ordem crescente.

Encontra a posição p/ inserir "key" empurrando os elementos maiores que "key" uma posição p/ frente.



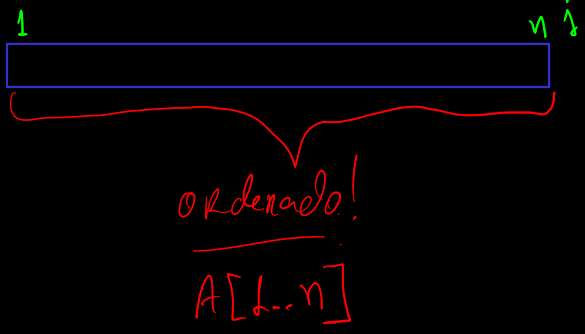
Manutenção Pela invariante de laço $A[1..j-1]$ consiste nos elementos originalmente nestas posições, mas em ordem crescente. Considerando que o laço while está correto, ele copia os elementos maiores que "key" (que era o elemento na posição "j" no início da iteração) uma posição p/ frente e pára com i na posição imediatamente anterior ao ponto de inserção da key. Na linha 8, "key" é inserida entre os elementos menores ($A[1..i]$) e os elementos maiores ($A[i+2..j]$), formando um vetor $A[1..j]$ ordenado. Ao incrementar j, a invariante é reestabelecida p/ a próxima iteração.

```
INSERTIONSORT(A)
1. FOR j = 2 TO A.size DO
2.     KEY = A[j]
3.     i = j - 1
4.     WHILE i > 0 AND A[i] > KEY DO
5.         A[i+1] = A[i]
6.         i = i - 1
7.     END WHILE
8.     A[i+1] = KEY
9. END FOR
```

n

INVARIANTE:

No início de cada iteração do laço for das linhas 1-9, o subvetor $A[1..j-1]$ consiste nos elementos que originalmente estavam em $A[1..j-1]$, mas em ordem crescente.



TÉRMINO A condição que TERMINA o LAÇO é $j > n$. Como o passo do for é 1, após a última ITERAÇÃO, $j = n+1$. Substituindo j na INVARIANTE, temos que o subvetor $A[1..j-1] = A[1..(n+1)-1]$ $= A[1..n]$ consiste nos elementos originalmente em $A[1..n]$, mas em ordem crescente. Como este subvetor é o vetor todo, conclui-se que o VETOR todo está ordenado. Portanto, o algoritmo está correto!