Nós já vimos vários exemplos onde é uma onde é uma boa ideia trabalhar com a lista ordenada.

Mas, para que a lista esteja ordenada primeiro é preciso ordená-la, não é?

Bom, agora nós vamos ver um algoritmo que coloca a lista em ordem.

Provavelmente, a ideia mais natural para ordenar uma lista é

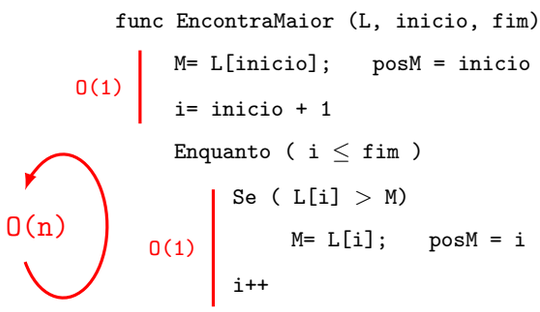
- colocar o maior elemento na última posição
- colocar o 2o maior elemento na penúltima posição
- colocar o 3o maior elemento na antepenúltima posição
- e por aí vai ...

Para fazer isso, é preciso encontrar o maior elemento.

E depois que o maior elemento está na última posição, o segundo maior elemento é o maior da porção `V[1..N − 1]` da lista.

Então aqui, é uma boa ideia escrever uma função que encontra o maior elemento em uma porção da lista — (*e retorna a sua posição*)

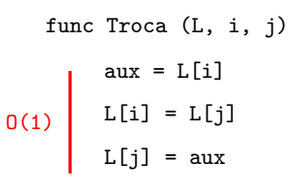
A coisa fica assim



E é fácil ver que a função executa em tempo $O(n)$ — (*no pior caso*)

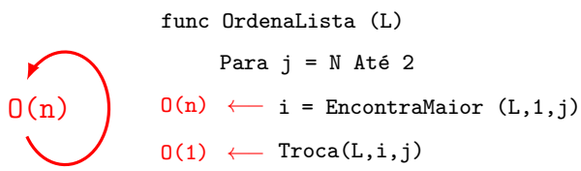
Certo.

Já que a gente está aqui, não custa nada escrever a função que troca dois elementos de lugar



que executa em tempo $O(1)$.

E agora, é bem fácil escrever o algoritmo que ordena a lista



Agora basta examinar as anotações para obter o tempo de execução do algoritmo

$$O(n) \cdot \Big[O(n) \quad + \quad O(1) \Big] \qquad = \qquad O(n^2)$$

Legal, não é?

◇