

A importância da ordenação

Os n mais barato

Na prática, será que nós só estamos interessados em saber o valor mais barato ou o mais caro? Ou também estamos interessados em descobrir o **grupo** dos produtos mais baratos, o **grupo** dos mais caros, o **grupo** dos que fizeram mais pontos. Às vezes queremos saber duas informações, por exemplo: quem ganhou com **mais** pontos e quem perdeu com **menos** pontos. Porém, é possível que uma busca nos traga inúmeros resultados, 1 milhão de resultados... Como quando queremos descobrir quais são os 10 carros mais baratos ou os cinco primeiros times que irão para a próxima fase da competição. Ou quando queremos informações sobre um grupo com vários produtos.

Como nós podemos alterar o nosso processo para marcar quais são os dois carros mais baratos? Dê uma olha na nossa lista de carros novamente:

 Lamborghini R\$ 1.000.000 0	 Jipe R\$ 46.000 1	 Brasília R\$ 16.000 2	 Smart R\$ 46.000 3	 Fusca R\$ 17.000 4
--	--	--	--	---

Lista de produtos

Ao observarmos a lista, logo iremos concluir que a Brasília e o Fusca são os carros mais baratos. Qual foi o processo que fizemos mentalmente para descobrir que os dois eram os mais baratos? O mesmo aconteceria se quiséssemos detectar quais são os três carros mais baratos... A Brasília, o Fusca e o Smart (ou o Jipe) são os carros mais baratos. Em que estávamos pensando à medida que analisávamos cada produto?

Observe a lista de carros e tente imaginar qual foi o processo que realizamos ao analisar cada um dos itens.

Algoritmo básico (e ruim?) para encontrar o n mais barato

Qual é o processo que faremos ao analisar cada um dos elementos, para descobrir quais dos produtos são o `maisBarato` e o `segundoMaisBarato`? Para descobrir qual é o `maisBarato`, nós vamos precisar de uma variável que armazene o produto com o menor preço naquele momento... Iremos precisar também ter registrado (mentalmente ou anotado num papel) qual é o `segundoMaisBarato`. Isto significa que vou precisar de outra variável e assim, anotar o primeiro e o segundo mais barato.

Analisaremos cada elemento e identificaremos "esse é o mais barato" ou "esse é o segundo mais barato agora". Para isso, teremos que usar o `if` várias vezes.

 Lamborghini R\$ 1.000.000 0	 Jipe R\$ 46.000 1	 Brasília R\$ 16.000 2	 Smart R\$ 46.000 3	 Fusca R\$ 17.000 4
--	--	--	--	---

Lista de produtos

atual	maisBarato	segundoMaisBarato
-------	------------	-------------------

No exemplo dos carros, se desejarmos encontrar quais são os três carros mais baratos, teremos que usar três "ifs" e o código terá mais uma variável: `terceiroMaisBarato`.

atual	maisBarato	segundoMaisBarato	terceiroMaisBarato
-------	------------	-------------------	--------------------

Sempre que quisermos saber se a Lamborghini é o carro mais barato... Ou o segundo, talvez, o terceiro carro mais barato, teremos sempre que passar pelo `for`. Em consequência, será preciso utilizar o `if` várias vezes se quisermos criar um grupo dos produtos mais baratos.

Por exemplo, se desejarmos identificar o grupo dos cinco carros com o menor preço, vamos precisar utilizar `if` cinco vezes. O mesmo irá acontecer se quisermos descobrir quais são os 10 carros mais baratos. Para encontrar a resposta, vamos precisar usar 10 `if`s. Imagine o tamanho do nosso código se quisermos descobrir os 50 produtos mais baratos de uma lista...

Não parece ser a melhor solução colocar inúmeras variáveis para verificar se o produto é o primeiro, o segundo, o terceiro, quarto ou quinto mais barato. Não parece a melhor solução usar infinitos `if`s para identificar os elementos com o menor valor.

A Importância da ordenação

Continuamos com a questão de como descobrir quais são os produtos mais baratos e os mais caros, os times ou competidores que irão para a próxima fase da competição, os n elementos **mais** ou **menos**, **maiores** ou **menores**. Temos um

tipo de classificação, quando determinamos que "isso é menor do que aquilo". Como responder a pergunta sem colocar infinitas variáveis e ifs no meu código?

Com 500 variáveis no código, escrevemos e escrevemos... Em algum momento, iremos terminar. Porém, desejamos aproveitar melhor nosso tempo e não queremos inserir inúmeras vezes o `if`. O que podemos fazer?

Vamos dar uma olhada na lista de carros:

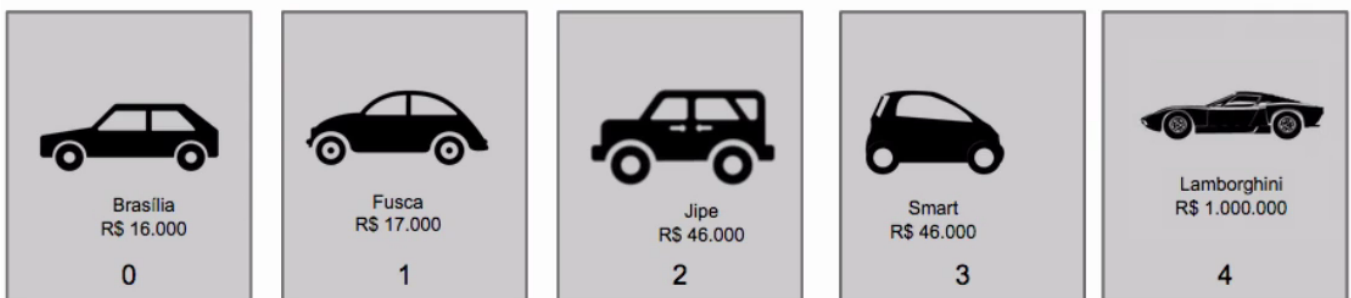
- Lamborghini custa R\$ 1.000.000
- Jipe custa R\$ 46.000
- Brasília custa R\$16.000
- Smart custa R\$46.000
- Fusca custa R\$17.000



Lista de produtos

É mais difícil encontrar o maior e o menor valor na nossa lista, porque os elementos estão misturados. Mas, e se tivéssemos uma lista diferente? Uma lista que fosse ordenada do menor para o maior preço...

- Brasília custa R\$16.000
- Fusca custa R\$17.000
- Jipe custa R\$ 46.000
- Smart custa R\$46.000
- Lamborghini custa R\$ 1.000.000



Com a nova lista, vamos responder a mesma pergunta de antes: qual é o carro mais barato? É possível identificá-lo com maior facilidade, afinal, os três primeiros elementos são os mais baratos...

Se queremos descobrir quais são os dois carros mais baratos, será o mesmo: os dois primeiros carros têm os menores preços. Caso nossa lista tenha 1 milhão de elementos e eles forem ordenados desta maneira, os 30 primeiros também serão os 30 mais baratos. Se quiséssemos os dois carros mais caros? Seriam os dois últimos da lista. Observe que a ordem faz toda diferença...

No momento em que os elementos de uma lista (carros, produtos, times, alunos) estão ordenados do menor para o maior, fica fácil responder a perguntas como:

- Quem são os dois menores?
- Quem são os dois maiores?
- Quem são os três menores?
- Quem são os cinco maiores?

Basta selecionarmos os três primeiros ou os cinco últimos. Ou os dois primeiros e os dois itens do fim da lista. Com a ordenação, você descobre os maiores e os menores elementos rapidamente.

Perguntas como:

- Quais são os 500 menores?
- Quais são os 500 maiores?

precisariam ser respondidas com 500 `if`s no nosso código. Porém, basta que os produtos estejam ordenados para que encontremos a resposta rapidamente...

Outro exemplo seria se quiséssemos encontrar o produto mais barato. Caso nossa lista esteja em ordem, sabemos que o elemento com o menor preço será aquele que estiver na primeira posição. Quando o *array* está ordenado (uma coleção de itens ordenados) é simples identificar quais são os elementos maiores e menores. O desafio está na ordenação dos itens. No exemplo dos carros, nós ordenamos os elementos manualmente... Movemos os produtos e conseguimos colocá-los em novas posições.

Como conseguimos ordenar um *array*? Este é o desafio que iremos resolver: como ordenar nossas listas de maneira inteligente e rápida?

Outras questões:

- Como criar uma ordem para cartas de baralho, provas de alunos ou resultados de uma eleição?
- Como podemos ordenar itens e, depois, aperfeiçoar a ordenação?
- Qual o mais barato?
- Qual carro está posicionado no meio da lista?

Quando as coisas estão ordenadas, o processo de busca se torna mais rápido. Essa é a grande sacada da ordenação: ela nos permite resolver tarefas do cotidiano quase imediatamente.

O nosso desafio será descobrir **como criar uma ordem para os produtos que estão desordenados...**

Como ordenar?

Nós queremos ordenar o nosso *array* de produtos. E como podemos fazer isto? Temos os elementos:

- Lamborghini
- Jipe
- Brasília
- Smart
- Fusca

 Lamborghini R\$ 1.000.000 0	 Jipe R\$ 46.000 1	 Brasília R\$ 16.000 2	 Smart R\$ 46.000 3	 Fusca R\$ 17.000 4
--	--	--	--	---

Lista de produtos

Escreva os cinco elementos num papel, na mesma ordem. Em seguida, tente ordená-los do produto mais barato para o mais caro. Como você ordenou os carros? Responda e eu irei mostrar o meu processo de ordenação, daqui a pouco.

Será que podemos encontrar uma outra solução para encontrar o produto mais barato?