**Bubblesort** (Classificação por bolhas).

O Bubble Sort, ou ordenação por flutuação, é um [algoritmo de ordenação](https://pt.wikipedia.org/wiki/Algoritmo_de_ordena%C3%A7%C3%A3o) dos mais simples que pode ser aplicado em arrays e listas dinâmicas. A ideia é percorrer o [vetor](https://pt.wikipedia.org/wiki/Vector) diversas vezes, e a cada passagem fazer flutuar para o topo o maior elemento da sequência.

O algoritmo bubblesort funciona executando duas tarefas principais, que são executadas em loop até que os dados estejam totalmente ordenados. São elas:

* Comparação de itens adjacentes;
* Troca de posição dos itens, quando for necessário.

Em comparação com outros métodos existentes, a classificação Bubblesort  é, no geral, lenta, mas é o mais simples dos algoritmos de classificação.

Ordenar vetores é uma tarefa muito importante, frequentemente realizada durante o processo de desenvolvimento de sistemas, pois arrays ordenados são, em muitos casos, imprescindíveis para que determinados algoritmos possam funcionar de maneira adequada.

Se o objetivo é ordenar os valores em forma decrescente, então, a posição atual é comparada com a próxima posição e, se a posição atual for menor que a posição seguinte, é realizada a troca dos valores nessa posição. Caso contrário, não é realizada a troca, apenas passa-se para o próximo par de comparações.

Se o objetivo é ordenar os valores em forma crescente, então, a posição atual é comparada com a próxima posição e, se a posição atual for maior que a posição seguinte, é realizada a troca. Caso contrário, a troca não é feita e passa-se para o próximo par de comparação.

No melhor caso, o algoritmo executa n {\displaystyle n}noperações relevantes, onde n representa o número de elementos do vetor. No pior caso, são feitas {\displaystyle n^{2}}n^2 operações. A complexidade desse algoritmo é de ordem quadrática. Por isso, ele não é recomendado para programas que precisem de velocidade e operem com quantidade elevada de dados.

Um array ou lista dinâmica pode estar já ordenado no momento em que se solicita a ordenação, dessa forma, esta situação tem de ser considerada na implementação do algoritmo.

Exemplo:

Para que não ocorra repetição desnecessária, utilizaremos uma FLAG, que nos avisará quando o vetor estará ordenado e, assim, a execução será terminada. Essa FLAG sempre será setada com 1 quando houver uma troca. Dessa forma, enquanto **n**for menor ou igual ao tamanho do vetor e, ao mesmo tempo a FLAG for igual a 1, então, deve-se continuar ordenando, caso contrário, o vetor estará ordenado.