algoritmo de ordenação bubble sort

O algoritmo de ordenação bubble sort é um método simples de ordenação que funciona iterando através do vetor várias vezes. Em cada passagem, compara os elementos adjacentes e troca-os se estiverem na ordem errada. Esse processo é repetido até que o vetor esteja completamente ordenado.

Aqui está um resumo do funcionamento do bubble sort:

1. **Iteração pelo vetor**: O algoritmo começa no início do vetor e compara o primeiro par de elementos adjacentes.
2. **Troca se necessário**: Se o primeiro elemento for maior que o segundo, eles são trocados.
3. **Avanço**: Então, move-se para o próximo par de elementos adjacentes e repete-se o processo até que todo o vetor seja percorrido sem necessidade de trocas em uma passagem completa.
4. **Repetição**: Esse processo é repetido para cada elemento do vetor, reduzindo o número de elementos a serem comparados a cada iteração, pois os elementos maiores "borbulham" para a extremidade direita do vetor, daí o nome "bubble sort".

Apesar de ser fácil de entender e implementar, o bubble sort não é eficiente para grandes conjuntos de dados devido à sua complexidade de tempo O(n2)O(n^2)O(n2), onde nnn é o número de elementos no vetor.

3.5

gere um método de ordenação bubble sort em python

ChatGPT

Aqui está uma implementação simples do algoritmo de ordenação bubble sort em Python:

python

Copiar código

def bubble\_sort(arr):

n = len(arr)

# Percorre todos os elementos do array

for i in range(n):

# Últimos i elementos já estão na posição correta

for j in range(0, n-i-1):

# Troca se o elemento encontrado é maior do que o próximo elemento

if arr[j] > arr[j+1]:

arr[j], arr[j+1] = arr[j+1], arr[j]

return arr

# Exemplo de uso

arr = [64, 34, 25, 12, 22, 11, 90]

sorted\_arr = bubble\_sort(arr)

print("Array ordenado:", sorted\_arr)

Neste exemplo, a função bubble\_sort percorre a lista várias vezes, comparando pares adjacentes e trocando-os se estiverem na ordem errada. O loop externo garante que cada elemento seja colocado na posição correta, enquanto o loop interno realiza as comparações e trocas necessárias.

4o