# 

**Curso:** Ciência de Computação **Período:** 6º **Turno:** Diurno

**Disciplina:** Estrutura de Dados 2 **Professor:** Fermín Alfredo Tang Montané

**Atividade:** Semana 1-2  **Data:** 21/08/2021

**Aluno:** João Vítor Fernandes Dias **Matrícula:** 00119110377

## Verdadeiro ou Falso sobre notação Big-O

Responda com verdadeiro (V) ou falso (F), justificando a sua resposta:

1. Sempre é possível afirmar que um algoritmo A é melhor que outro Algoritmo B apenas comparando seus tempos de execução para um tamanho fixo de entrada .

(F) Caso analisemos os algoritmos em um tamanho fixo de entrada, apenas podemos afirmar quanto a este tamanho. Entretanto, diferentes algoritmos geralmente têm diferentes equações que definem aproximadamente a quantidade de procedimentos que são executados, assim, algumas funções podem ser maiores que outras durante um período inicial de tempo, porém, quando analisamos o seu comportamento quando a entrada de dados tende ao infinito, percebemos qual algoritmo tem melhor desempenho neste caso.

1. A função mede de forma precisa o número de operações realizadas por um algoritmo em função da entrada do algoritmo.

(F) A função T(n) mede de forma **aproximada** o número de operações realizadas por um algoritmo em função da entrada do algoritmo. Principalmente até porque é complicado calcular todo e qualquer procedimento executado e para números suficientemente grandes se torna irrisória a precisão calculada inicialmente.

1. A função mede de forma aproximada o número de posições de mémória realizadas por um algoritmo em função da entrada do algoritmo.

(V) A função T(n) mede aproximadamente o tempo, a função E(n) mede aproximadamente o espaço, ambos dependentes de uma quantidade (n) de valores de entrada.

1. A notação serve para classificar o comportamento de um algoritmo com função na categoria mas não para medir seu desempenho.

(F) A notação O(f(n)) de fato serve para classificar o comportamento de uma função tendendo ao infinito, isso auxilia na medição de seu desempenho, justamente por classificar em uma categoria, mesmo que seja ainda menos preciso do que a própria T(n), ainda assim pode medir o desempenho.

1. Na notação podemos descartar termos de menor grau e coeficientes multiplicativos, o que significa que eles não são necessários para medir o seu desempenho.

(F) O descarte é feito para se encontrar a notação O(f(n)), mas mesmo que sejam descartados, eles são sim necessário para se medir o desempenho como um todo, por isso utiliza-se a função T(n) para este fim, e esta função possui os tais coeficientes descartados.

## Uma iteração dos algoritmos de ordenação

Considerando o seguinte vetor:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 15 | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 2 |

Realize a **primeira iteração** do algoritmo de ordenação elementar indicado:

### Usando o algoritmo Bubble sort:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 15 | 5 | | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 2 |
| 15 | | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 2 | 26 |
| 15 | | 5 | 16 | 4 | 10 | 23 | 18 | 2 | 39 | 26 |
| 15 | | 5 | 16 | 4 | 10 | 23 | 2 | 18 | 39 | 26 |
| 15 | | 5 | 16 | 4 | 10 | 2 | 23 | 18 | 39 | 26 |
| 15 | | 5 | 16 | 4 | 2 | 10 | 23 | 18 | 39 | 26 |
| 15 | | 5 | 16 | 2 | 4 | 10 | 23 | 18 | 39 | 26 |
| 15 | | 5 | 2 | 16 | 4 | 10 | 23 | 18 | 39 | 26 |
| 15 | 2 | | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 |
| 2 | 15 | | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 |

Indique o número de comparações: 10

Indique o número de trocas: 10

### Usando o algoritmo Selection sort:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 15 | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 2 |
| 2 | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 15 |

Indique o número de comparações: 2

Indique o número de trocas: 1

### Usando o algoritmo Insertion sort:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 15 | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 2 |
| 15 | 5 | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 2 |
| 5 | 15 | 16 | 4 | 10 | 23 | 18 | 39 | 26 | 2 |

Indique o número de comparações: 1

Indique o número de deslocamentos: 1

## Sexta iteração do Bubble Sort:

Considerando o seguinte vetor:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 5 | 10 | 15 | 26 | 18 | 39 | 16 | 23 |

Realize a **sexta iteração** do algoritmo de ordenação Bubble Sort:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 5 | 10 | 15 | 26 | 18 | 39 | 16 | 23 |
| 2 | 4 | 5 | 10 | 15 | 26 | 18 | 39 | 16 | 23 |
| 2 | 4 | 5 | 10 | 15 | 26 | 18 | 16 | 39 | 23 |
| 2 | 4 | 5 | 10 | 15 | 26 | 16 | 18 | 39 | 23 |
| 2 | 4 | 5 | 10 | 15 | 16 | 26 | 18 | 39 | 23 |
| 2 | 4 | 5 | 10 | 15 | 16 | 26 | 18 | 39 | 23 |
|  |  |  |  |  |  |  |  |  |  |

Indique o número de comparações: 5

Indique o número de trocas: 3

## Sexta iteração do Selection Sort:

Considerando o seguinte vetor:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 5 | 10 | 15 | 26 | 18 | 39 | 16 | 23 |

Realize a **sexta iteração** do algoritmo de ordenação selection sort:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 5 | 10 | 15 | 26 | 18 | 39 | 16 | 23 |
| 2 | 4 | 5 | 10 | 15 | 16 | 18 | 39 | 26 | 23 |

Indique o número de comparações: 2

Indique o número de trocas: 2

## Análise das duas questões anteriores:

Nas duas questões anteriores, após 5 iterações tanto o algoritmo Bubble Sort quanto o algoritmo Selection Sort, colocaram 5 menores elementos nas primeiras 5 posições. Responda se acredita que isso acontece, sempre, no caso do algoritmo insertion sort. Justifique.

R: Embora o Insertion Sort possa vir a ter este comportamento, isso não é uma regra. Consideremos um vetor de 10 números em ordem decrescente da esquerda para a direita e queremos ordená-lo utilizando o Insertion Sort. Após a quinta iteração, teríamos:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | 7 | 8 | 9 | 10 | 5 | 4 | 3 | 2 | 1 |

Logo, ao utilizarmos o Selection Sort, nem sempre após n iterações os n menores elementos estariam ordenados nos n primeiros elementos.

## Quinta iteração do Selection Sort:

Considerando o seguinte vetor:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 5 | 26 | 39 | 10 | 18 | 15 | 16 | 23 |

Realize a **quinta iteração** do algoritmo de ordenação selection sort:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 5 | 26 | 39 | 10 | 18 | 15 | 16 | 23 |
| 2 | 4 | 5 | 26 | 10 | 39 | 18 | 15 | 16 | 23 |
| 2 | 4 | 5 | 10 | 26 | 39 | 18 | 15 | 16 | 23 |

Indique o número de comparações: 3

Indique o número de deslocamentos: 2

# Fim