

## GUIÃO 04 – ANÁLISE DA COMPLEXIDADE DE ALGORITMOS

A documentação da linguagem C pode ser consultada em <https://en.cppreference.com/w/c>

1 – Considere uma **sequência de n elementos inteiros e não ordenada**. Pretende-se determinar quantos elementos dessa sequência respeitam a seguinte propriedade:

$$\text{array}[i] = \text{array}[i - 1] + \text{array}[i + 1], \text{ para } 0 < i < (n - 1)$$

- Implemente uma **função eficiente** que determine quantos elementos (resultado da função) de uma sequência com n elementos ( $n > 2$ ) respeitam esta propriedade.
- Pretende-se determinar experimentalmente a **ordem de complexidade do número de comparações** efetuadas pelo algoritmo e envolvendo elementos da sequência.  
**A complexidade é de ordem linear  $O(n)$**
- Considere as seguintes **sequências de 10 elementos inteiros**, que cobrem **algumas situações possíveis** de execução do algoritmo.

Determine, para cada uma delas, o número de elementos que obedecem à condição e o número de comparações efetuadas envolvendo elementos da sequência.

1	2	3	4	5	6	7	8	9	10
1	2	1	4	5	6	7	8	9	10
1	2	1	3	2	6	7	8	9	10
0	2	2	0	3	3	0	4	4	0
0	0	0	0	0	0	0	0	0	0

Resultado	0
Resultado	1
Resultado	2
Resultado	6
Resultado	8

Nº de comparações	8
Nº de comparações	8
Nº de comparações	8
Nº de comparações	8
Nº de comparações	8

Depois da execução do algoritmo responda às seguintes questões:

- Em termos do número de comparações efetuadas, podemos distinguir alguma variação na execução do algoritmo? Ou seja, existe a situação de melhor caso e de pior caso, ou estamos perante um algoritmo sistemático? **Independentemente do conteúdo do array, ele sempre realiza exatamente  $n-2$  comparações, onde  $n$  é o tamanho do array. Portanto, trata-se de um algoritmo sistemático, sem melhor ou pior caso, com complexidade de tempo linear  $O(n)$ .**
- Com base nos resultados experimentais, qual é a ordem de complexidade do algoritmo?  
**A ordem de complexidade é linear**
- **Determine formalmente a ordem de complexidade do algoritmo.** Tenha em atenção que deve obter uma expressão matemática exata e simplificada.  
$$NCOMPS = \sum_{i=1}^{size-2} 1 = SIZE - 2$$
- Calcule o valor dessa expressão para  $n = 10$  e compare-o com os resultados obtidos experimentalmente.  **$n-2=10-2=8$**

2 – Considere uma **sequência de n valores reais**. Pretende-se determinar se os elementos da sequência são sucessivos termos de uma **progressão geométrica**:

$$r = a[1] / a[0] \text{ e } a[i] = r \times a[i - 1], i > 1$$

- Implemente uma função **eficiente** (utilize um algoritmo em lógica negativa) que verifique se os n elementos ( $n > 2$ ) de uma sequência de valores reais são sucessivos termos de uma progressão geométrica. A função deverá devolver 1 ou 0, consoante a sequência verificar ou não essa propriedade.
- Pretende-se determinar experimentalmente a **ordem de complexidade do número de multiplicações e divisões** efetuadas pelo algoritmo e envolvendo elementos da sequência.

- Considere as seguintes sequências de 10 elementos, que cobrem as distintas situações possíveis de execução do algoritmo. Determine, para cada uma delas, se satisfazem a propriedade e qual o número de operações de multiplicação e de divisão efetuadas pelo algoritmo.

1	2	3	4	5	6	7	8	9	10
1	2	4	4	5	6	7	8	9	10
1	2	4	8	5	6	7	8	9	10
1	2	4	8	16	6	7	8	9	10
1	2	4	8	16	32	7	8	9	10
1	2	4	8	16	32	64	8	9	10
1	2	4	8	16	32	64	128	9	10
1	2	4	8	16	32	64	128	256	10
1	2	4	8	16	32	64	128	256	512

Resultado	0
Resultado	0
Resultado	0
Resultado	0
Resultado	0
Resultado	0
Resultado	0
Resultado	0
Resultado	1

Nº de operações	2
Nº de operações	3
Nº de operações	4
Nº de operações	5
Nº de operações	6
Nº de operações	7
Nº de operações	8
Nº de operações	9
Nº de operações	10

Depois da execução do algoritmo responda às seguintes questões:

- Qual é a sequência (ou as sequências) que corresponde(m) ao melhor caso do algoritmo?  
**array={1,2,4,8,16,32,64,128,256,512} onde o numero de operações foi n.**
- Qual é a sequência (ou as sequências) que corresponde(m) ao pior caso do algoritmo?  
**array= {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} onde o numero de operações foi n-8**
- Para as sequências dadas (**n = 10**) qual foi o número médio de operações efetuadas?  
**Numero medio de operações foi 6.**
- Qual é a ordem de complexidade do algoritmo?
- **Determine formalmente a ordem de complexidade do algoritmo nas situações do melhor caso, do pior caso e do caso médio, considerando uma sequência de tamanho n.** Deve obter expressões matemáticas exatas e simplificadas.
- Calcule o valor dessas expressões para **n = 10** e **compare-os com os resultados obtidos experimentalmente.**

3 – Considere uma **sequência de n elementos inteiros e não ordenada**. Pretende-se determinar quantos ternos (**i, j, k**) de índices da sequência respeitam a seguinte propriedade:

$$\text{array}[k] = \text{array}[i] + \text{array}[j], \text{ para } i < j < k$$

- Implemente uma **função eficiente** que determine quantos ternos (**i, j, k**) de índices (resultado da função) de uma sequência com n elementos (**n > 2**) respeitam essa propriedade.
- Pretende-se determinar experimentalmente a **ordem de complexidade do número de comparações** efetuadas pelo algoritmo e envolvendo elementos da sequência.  
tem uma complexidade de  $O(n^3)$  porque ele usa três loops aninhados, cada um iterando potencialmente sobre n elementos. O número de comparações feitas no algoritmo é proporcional ao número de iterações dos loops aninhados.
- Considere as sequências de 10 elementos indicadas no exercício 1 e outras sequências diferentes à sua escolha; **use sequências com 5, 10, 20, 30 e 40 elementos.**

Determine, para cada uma delas, quantos ternos (**i, j, k**) de índices respeitam propriedade e o número de comparações efetuadas.

Array 5: nternos = 4, ncomps = 10  
 Array 20: nternos = 70, ncomps = 1140  
 Array 30: nternos = 233, ncomps = 4060  
 Array 40: nternos = 169, ncomps = 9880

Depois da execução do algoritmo responda às seguintes questões:

- Em termos do número de comparações efetuadas, podemos distinguir alguma variação na execução do algoritmo? Ou seja, existe a situação de melhor caso e de pior caso, ou estamos perante um algoritmo sistemático?

**não há variação na execução do algoritmo; ele sempre realiza o mesmo número de comparações, independentemente da entrada. Portanto, estamos perante um algoritmo sistemático.**

- Com base nos resultados experimentais, qual é a ordem de complexidade do algoritmo?  
**A complexidade é  $O(n^3)$**
- Determine formalmente a ordem de complexidade do algoritmo.** Tenha em atenção que deve obter uma expressão matemática exata e simplificada.  **$T(n)=O(n^3)$**
- Calcule o valor da expressão para  **$n = 10$**  e  **$n = 20$** ; compare-o com os resultados obtidos experimentalmente.

$$T(n) = \frac{1}{2} \sum_{k=2}^{n-1} (k^2 - k)$$

$$T(10) = \frac{1}{2} \times 240 = 120 \quad \checkmark$$

$$T(20) = \frac{1}{2} \cdot 2280 = 1140 \quad \checkmark$$

### \*\* Exercício Adicional \*\*

4 – Considere uma **sequência**, possivelmente não ordenada, de **n elementos inteiros e positivos**. **Percorrendo a sequência a partir do seu primeiro elemento** (current\_value), pretende-se sucessivamente **eliminar os elementos seguintes** que sejam **iguais ou múltiplos ou submúltiplos desse elemento** (current\_value), sem ordenar a sequência e sem alterar a posição relativa dos elementos. Depois, pretende-se fazer o mesmo a partir do segundo elemento, do terceiro e assim sucessivamente.

Por exemplo, a sequência { 8, 2, 6 } com 3 elementos será transformada na sequência { 8, 6 } com 2 elementos; a sequência { 2, 2, 2, 3, 3, 4, 5, 8, 8, 9 } com 10 elementos será transformada na sequência { 2, 3, 5 } com 3 elementos; e a sequência { 7, 8, 2, 2, 3, 3, 3, 8, 8, 9 } com 10 elementos será transformada na sequência { 7, 8, 3, } com 3 elementos.

- Implemente uma função **eficiente** que, para uma sequência com n elementos ( **$n > 1$** ), elimina os elementos iguais ou múltiplos ou submúltiplos como indicado acima.

**A função deverá ser void e alterar o valor do parâmetro indicador do número de elementos armazenados na sequência (que deve ser passado por ponteiro).**

- Pretende-se determinar experimentalmente a **ordem de complexidade do número de comparações** e do **número de deslocamentos** (i.e., cópias) efetuados pelo algoritmo e envolvendo elementos da sequência.
- Considere as sequências de 10 elementos indicadas no exercício 2, bem como outras à sua escolha. Determine, para cada uma delas, a sua configuração final, bem como o número de comparações e de deslocamentos efetuados.

**Depois da execução do algoritmo responda às seguintes questões:**

- Indique uma sequência inicial com 10 elementos que conduza ao **melhor caso do número de comparações** efetuadas. Qual é a sequência final obtida? Qual é o número de comparações efetuadas? Qual é o número de deslocamentos (i.e., cópias) de elementos efetuados?

Inicial:											Nº de comparações	
Final:											Nº de cópias	

- Indique uma sequência inicial com 10 elementos que conduza ao **pior caso do número de comparações** efetuadas. Qual é a sequência final obtida? Qual é o número de comparações efetuadas? Qual é o número de deslocamentos (i.e., cópias) de elementos efetuados?

Inicial:											Nº de comparações	
Final:											Nº de cópias	

- Determine formalmente a ordem de complexidade do algoritmo** nas situações do **melhor caso** e do **pior caso**, considerando uma sequência de tamanho n. Deve obter expressões matemáticas exatas e simplificadas.