AULA 6 - ANÁLISE DA COMPLEXIDADE DE ALGORITMOS RECURSIVOS (NÚMEROS DE MOTZKIN)

***** Entregue, num ficheiro ZIP, este guião preenchido e o código desenvolvido *****

- Os números de Motzkin

1, 1, 2, 4, 9, 21, 51,... (https://oeis.org/A001006)

são definidos pela seguinte relação de recorrência:

$$Motzkin(n) = \begin{cases} 1, & se\ n = 0\ e\ n = 1 \\ Motzkin(n-1) + \sum_{k=0}^{n-2} Motzkin(k) \times Motzkin(n-2-k), & se\ n > 1 \end{cases}$$

Função Recursiva

- Implemente uma **função recursiva Motzkin(n)** que use diretamente a relação de recorrência acima, **sem qualquer simplificação**.
- Construa um programa para executar a função **Motzkin(n)** para **sucessivos valores de n** e que permita **contar o número total de multiplicações efetuadas** para cada valor de n.
- **Preencha a as primeiras colunas tabela seguinte** com o resultado da função recursiva e o número de multiplicações efetuadas para os sucessivos valores de n.

n	Motzkin(n) – Versão Recursiva	Nº de Multiplicações	Motzkin(n) – Versão de Programação Dinâmica	Nº de Multiplicações
0	1	0	1	0
1	1	0	1	0
2	2	1	2	1
3	4	3	4	3
4	9	8	9	6
5	21	20	21	0
6	51	49	51	15
7	127	119	127	21
8	323	288	323	28
9	835	696	835	36
10	2188	1681	2188	45
11	5798	4059	5798	55
12	15511	9800	15511	66
13	41835	2366	41835	78
14	113634	57121	113634	91
15	310572	137903	310572	105

- Analisando os dados da tabela, estabeleça uma **ordem de complexidade** para a **função recursiva**.

Ordem de complexidade exponencial → $O(2.14^n)$. Obtida pela divisão de termos sucessivos.
(Ver ficheiro "Folha1.html" na pasta Gráficos)

Programação Dinâmica

- Uma forma alternativa de resolver alguns problemas recursivos, para evitar o cálculo repetido de valores, consiste em efetuar esse cálculo de baixo para cima (*"bottom-up"*), ou seja, de **Motzkin(0)** para **Motzkin(n)**, e utilizar um *array* para manter os valores entretanto calculados. Este método designa-se por **programação dinâmica** e reduz o tempo de cálculo à custa da utilização de mais memória para armazenar os valores intermédios.
- Usando **programação dinâmica**, implemente uma **função iterativa** para calcular Motzkin(n). **Não utilize um array global.**
- Construa um programa para executar a função iterativa que desenvolveu para **sucessivos valores de n** e que permita **contar o número de multiplicações efetuadas** para cada valor de n.
- **Preencha as últimas colunas tabela anterior** com o resultado da função iterativa e o número de multiplicações efetuadas para os sucessivos valores de n.
- Analisando os dados da tabela, estabeleça uma **ordem de complexidade** para a **função iterativa**.

Ordem de complexidade polinomial → $O(n^2)$ .Obtida pela divisão dos logaritmos da divisão de termos sucessivos com índices sucessivos.
(Ver ficheiro "Folha1.html" na pasta Gráficos)

Função Recursiva – Análise Formal da Complexidade

- Escreva uma **expressão recorrente** (direta) para o **número de multiplicações** efetuadas pela função recursiva Motzkin(n). Obtenha, depois, uma **expressão recorrente simplificada**. Note que $\sum_{k=0}^{n-2} Mult(k) = \sum_{k=0}^{n-2} Mult(n-2-k)$. **Sugestão:** efetue a subtração $Mult(n) - Mult(n-1)$.

Mult(0) = 0; Mult(1) = 0 → Casos iniciais

Mult(n) = $\sum_{k=0}^{n-2}(k) + \sum_{k=0}^{n-2}(n-k-2) + \sum_{k=0}^{n-2}(1) = 2\sum_{k=0}^{n-2}(k) + n - 1$

Mult(n) - Mult(n-1) = $[2\sum_{k=0}^{n-2}(k) + n - 1] - [2\sum_{k=0}^{n-3}(k) + n - 2] = 2\sum_{k=0}^{n-2}(k) + n - 1 - 2\sum_{k=0}^{n-3}(k) - n + 2 =$

2*Mult(n-1)+Mult(n-2)+1

- A equação de recorrência obtida é uma **equação de recorrência linear não homogénea**. Considere a correspondente **equação de recorrência linear homogênea**. Determine as raízes do seu **polinómio característico**. Sem determinar as constantes associadas, escreva a **solução da equação de recorrência linear não homogénea**.

Sabe-se que mult(n) é da forma $an = an^{(1)} + an^{(2)}$.

Equação característica:
$x^n = 2 * x^{n-1} + x^{n-2} \Leftrightarrow x^n - 2 * x^{n-1} - x^{n-2} = 0 \Leftrightarrow x^{n-2}(x^2 - 2x - 1) = 0 \Leftrightarrow x^2 - 2x - 1 = 0$

Resolvendo a equação:
$x = 1 + \sqrt{2} \vee x = 1 - \sqrt{2}$

$an^{(1)} = C1 * (1 + \sqrt{2})^n + C2 * (1 - \sqrt{2})^n$ sendo que $C1 * (1 + \sqrt{2})^n$ tem maior valor absoluto consideramos apenas esse.

$an^{(2)}$ *não tem grau zero e* 1 *não é raiz logo considerando* $an^{(2)} = A_0$ *temos que* :
$A_0 = 2A_0 + A_0 + 1 \Leftrightarrow A_0 - 2A_0 - A_0 - 1 = 0 \Leftrightarrow A_0 = -\frac{1}{2}$

A solução da equação característica será então:
$an = C1 * (1 + \sqrt{2})^n - \frac{1}{2}$

- Usando a solução da equação de recorrência obtida acima, determine a **ordem de complexidade do número de multiplicações** efetuadas pela função recursiva. **Compare** a ordem de complexidade que acabou de obter com o resultado da **análise experimental**.

Analisando os dados obtidos a ordem de complexidade é $O\,(1+\sqrt{2})^{n} \approx O(2.14^{\,n})$, que é igual ao resultado obtido experimentalmente.

Programação Dinâmica – Análise Formal da Complexidade

- Considerando o número de multiplicações efetuadas pela função iterativa, efetue a análise formal da sua complexidade. Obtenha uma **expressão exata e simplificada para o número de multiplicações** efetuadas.

$$C(n) = \sum_{i=2}^{n}\left(\sum_{k=0}^{i-2} 1\right) = \sum_{i=2}^{n}(i-1) = \frac{n^2-n}{2}$$

- Usando a expressão obtida acima, determine a **ordem de complexidade do número de multiplicações** efetuadas pela função iterativa. **Compare** a ordem de complexidade que acabou de obter com o resultado da **análise experimental**.

Analisando os dados obtidos a ordem de complexidade é $O(n^2)$, que é igual ao resultado obtido experimentalmente.