

AULA 5 - ANÁLISE DA COMPLEXIDADE DE ALGORITMOS

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

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

$$r = a[1] / a[0] \quad e \quad a[i] = r \times a[i-1], i > 1.$$

- Implemente uma função **eficiente** (utilize um algoritmo em lógica negativa) e **eficaz** 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.

Depois de validar o algoritmo apresente a função no verso da folha.

- 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	Resultado	0	Nº de operações	2
1	2	4	4	5	6	7	8	9	10	Resultado	0	Nº de operações	3
1	2	4	8	5	6	7	8	9	10	Resultado	0	Nº de operações	4
1	2	4	8	16	6	7	8	9	10	Resultado	0	Nº de operações	5
1	2	4	8	16	32	7	8	9	10	Resultado	0	Nº de operações	6
1	2	4	8	16	32	64	8	9	10	Resultado	0	Nº de operações	7
1	2	4	8	16	32	64	128	9	10	Resultado	0	Nº de operações	8
1	2	4	8	16	32	64	128	256	10	Resultado	0	Nº de operações	9
1	2	4	8	16	32	64	128	256	512	Resultado	1	Nº de operações	9

Depois da execução do algoritmo responda às seguintes questões:

- Qual é a sequência (ou as sequências) que corresponde(m) ao melhor caso do algoritmo?

Melhor caso acontece na primeira sequência, onde o número de operações é mínimo

- Qual é a sequência (ou as sequências) que corresponde(m) ao pior caso do algoritmo?

O pior caso, quando as operações são máximas é 9 operações, na penúltima e última sequências

- Determine o número de operações efetuadas no caso médio do algoritmo (**para $n = 10$**).

6

- Qual é a ordem de complexidade do algoritmo?

$O(n)$

- 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. **Faça essas análises no verso da folha.**

FUNÇÃO

```
int seq(int size, int *a){
    int res = 1;

    c++;
    int r = a[1] / a[0];

    for(int j = 2; j < size; j++){
        c++;
        if (a[j] != r*a[j-1]){
            i = 0;
            break;
        }
    }
    return res;
}
```

ANÁLISE FORMAL DO ALGORITMO

$$W(m) = m - 1 \quad B(m) = 2$$

$$A(m) = p(i) \times \text{cost}(i)$$

$$\sum_{i=2}^{m-1} \frac{1}{m-2} \times i = \frac{1}{m-2} \sum_{i=2}^{m-1} i$$

$$= \frac{1}{m-2} \times \frac{((m-1)+1-2)((m-1)+2)}{2}$$

$$= \frac{1}{m-2} \times \frac{(m-2)(m+1)}{2} =$$

$$= \frac{m+1}{2}$$

$$p(i) = \frac{1}{m-2}$$

$$\text{cost}(i) = i$$

$$\frac{i}{m-2}$$

$$\frac{\text{cost}(i)}{m-2}$$

$$W = O(m)$$

$$B = \text{Constante}(2)$$

$$A = O(m)$$

- Calcule o valor das expressões para $n = 10$ e compare-os com os resultados obtidos experimentalmente.

$$W(10) = 9$$

$$B(10) = 2$$

$$A(10) = 11/5$$

2 – Considere uma sequência (array), possivelmente não ordenada, de n elementos inteiros e positivos. Pretende-se **eliminar os elementos da sequência que sejam iguais ou múltiplos ou submúltiplos de algum dos seus predecessores**, sem fazer a sua ordenação e sem alterar a posição relativa dos elementos.

Por exemplo, 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 e eficaz** que elimina os elementos iguais ou múltiplos ou submúltiplos de algum dos seus predecessores numa sequência com n elementos ($n > 1$). A função deverá ser *void* e alterar o valor do parâmetro indicador do número de elementos efetivamente armazenados na sequência (que deve ser passado por referência).

Depois de validar o algoritmo apresente a função no verso da folha.

- 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 anteriormente indicadas de 10 elementos e 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? Justifique.

Inicial:	1	5	6	10	3	10	3	1	2	2	Nº de comparações	1
Final:	1	x	x	x	x	x	x	x	x	x	Nº de cópias	0

Visto que o 1 é um múltiplo universal, não existirá nenhum valor que não seja múltiplo ou submúltiplo de 1. A posição do um no melhor caso, poderá se na primeira ou segunda posição, uma vez que o primeiro elemento fará sempre parte do array final, o algoritmo inicia-se logo a partir da segunda posição.

- 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? Justifique.

Inicial:	2	3	5	7	11	13	17	19	23	29	Nº de comparações	109
Final:	2	3	5	7	11	13	17	19	23	29	Nº de cópias	10

Um array sem repetições de números primos apresenta o pior caso, pois cada elemento é apenas divisível por si e por 1, e como o array é sem repetições, cada elemento também só aparece uma única vez.

- 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. **Faça essas análises no verso da folha.**

Função

```
int comp = 0; // contador
int seq(int *array, int *size){
    int elem = 1;
    for(int i = 0; i < *size; i++){
        comp++;
        if(*(array+i) != 1){
            int dif = 1;
            for (int j = 0; j < i; j++){
                comp++;
                if( *(array+i) % *(array+j) == 0) {
                    dif = 0 ;
                    break;
                }else{
                    comp++;
                    if( *(array+j) % *(array+i) == 0) {
                        dif = 0 ;
                        break;
                    }
                }
            }
            comp++
            if(dif == 1){
                *(array+elem) = *(array+i);
                elem++;
            }
        }else{
            elem++;
            break;
        }
    }
    *(size) = elem;
}
```

ANÁLISE FORMAL DO ALGORITMO – COMPARAÇÕES – MELHOR CASO – PIOR CASO

$$B(m) = 1 \quad W(m) = m^2 - m + 1$$

Demonstração Worst case

$$B(m) = \text{constante}$$

$$W(m) = O(m^2)$$

$$\begin{aligned} \sum_{i=0}^{m-1} \left[\sum_{j=0}^{i-1} 2 + 1 \right] &= \sum_{i=0}^{m-1} \left[2 \times (i-0+1) + 1 \right] \\ &= \sum_{i=0}^{m-1} 2i + 1 = 2 \sum_{i=0}^{m-1} i + 1 = 2 \times \left[\frac{m(m-1)}{2} \right] + 1 \\ &= m^2 - m + 1 // \end{aligned}$$

ANÁLISE FORMAL DO ALGORITMO – DESLOCAMENTOS – MELHOR CASO – PIOR CASO

$B(m) = 1$ So é efetuada uma "cópia", mais
Constante concretamente, o size é alterado
para 1, isto só acontece quando o
primeiro elemento do array é 1

$W(m) = m$ Acontece quando todos os elementos
 $O(m)$ são copiados, acontece quando todos
eles são diferentes entre si e não
múltiplos, um exemplo é um array de
números primos todos diferentes