

## AULA 2 - ANÁLISE DA COMPLEXIDADE DE ALGORITMOS

Simule a execução de cada uma das seguintes funções (por exemplo, para  $n = 5$ ) e determine o seu resultado.

De seguida implemente-as no computador e confirme os resultados obtidos.

Para cada uma das funções, determine experimentalmente, em função do valor ( $n$ ) da entrada, o número de vezes que a instrução mais interna é executada.

```
unsigned int f1 (unsigned int n)
{
    unsigned int i, j, r = 0;
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
            r += 1;
    return r;
}
```

```
unsigned int f2 (unsigned int n)
{
    unsigned int i, j, r = 0;
    for (i = 1; i <= n; i++)
        for (j = 1; j <= i; j++)
            r += 1;
    return r;
}
```

```
unsigned int f3 (unsigned int n)
{
    unsigned int i, j, r = 0;
    for (i = 1; i <= n; i++)
        for (j = i; j <= n; j++)
            r += j;
    return r;
}
```

```
unsigned int f4 (unsigned int n)
{
    unsigned int i, j, r = 0;
    for (i = 1; i <= n; i++)
        for (j = i; j >= 1; j /= 10)
            r += i;
    return r;
}
```

Preencha a tabela com o resultado de cada função e o número de iterações realizadas, para os sucessivos valores de entrada.

n	f1 (n)	Nº de Iterações	f2 (n)	Nº de Iterações	f3 (n)	Nº de Iterações	f4 (n)	Nº de Iterações
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
<b>O(n)</b>								

NOME: \_\_\_\_\_

Nº MEC: \_\_\_\_\_

Após os testes computacionais, responda às seguintes questões:

- 1 Analisando os dados da tabela qual é a **ordem de complexidade** de cada algoritmo?
- 2 Determine formalmente a ordem de complexidade de cada algoritmo, obtendo uma **expressão** que corresponda aos resultados experimentais.

### Exercícios Adicionais:

- 3 Um **factoriã**, para uma dada base, é um número inteiro positivo  $n$  que é igual à soma do factorial de cada um dos seus algarismos.  
Escreva um programa eficiente que lhe permita listar, para a base 10, todos os factoriões menores que  $10^6$ .  
Determine experimentalmente o **número de multiplicações** e o **número de divisões** efetuadas pelo seu algoritmo.

ATENÇÃO:  $0! = 1$

SUGESTÃO: Armazene num *array* os factoriais dos sucessivos algarismos.

Verifique os seus resultados consultando a sequência A014080 na OEIS <https://oeis.org/A014080>

- 4 Um **número de Armstrong**, para uma dada base, é um número inteiro positivo de  $n$  algarismos que é igual à soma de cada um dos seus algarismos levantado à  $n$ -ésima potência.  
Escreva um programa eficiente que lhe permita listar, para a base 10, todos os números de Armstrong de 3 algarismos.  
Determine experimentalmente o **número de multiplicações** e o **número de divisões** efetuadas pelo seu algoritmo.

SUGESTÃO: Armazene num *array* as potências dos sucessivos algarismos.

OPCIONAL: Generalize o seu algoritmo, de modo a poder listar números de Armstrong com um maior número de algarismos.

Verifique os seus resultados consultando a sequência A005188 na OEIS <https://oeis.org/A005188>