Análise de Complexidade de Algoritmos

Monitora: *Estela Miranda Batista*

Sobre a Complexidade de Algoritmos...

O objetivo da análise de algoritmos é verificar qual algoritmo é melhor para cada situação, por exemplo, em problemas simples podemos usar complexidades altas, como os quadráticos, mas em problemas que o tempo é essencial seria melhor algoritmos logarítmos.

Na análise do **Melhor Caso**, iremos considerar então que o mínimo de coisas será executado, por exemplo, tendo uma comparação em um if, e possuindo um else if, iremos considerar que apenas o if será executado sempre.

Já no **Pior Caso**, no mesmo exemplo iremos considerar que os dois são executados sempre.

```
float exemplo(float *a, float *b, int n){
   float soma = 0.0;

   if(b[0] < 0) soma -= b[0]; // Comparação 1

   for(int i=1; i<n; i++){
      if(a[i] > 600) soma += a[i]*b[i]; //Comparação 2
      else if(a[i] > =0) soma += b[i]; // Comparação 3
      else if(a[i] < 0){ // Comparação 4
         for(int j=0; j<n; j+=2)
            if(b[j] < 0) soma -= b[j]; // Comparação 5
      }
   }

   return soma;
}
```

Exercício 2...

No **Melhor Caso**, temos que iremos considerar das comparações presentes no laço de repetição, apenas a comparação 1 irá ocorrer, obtemos então:

#Comparação2

$$\underbrace{1}_{\#Comparação1} + \sum_{1}^{n-1} 1 \rightarrow 1 + ((n-1) - 1 + 1) \rightarrow n \rightarrow O(n)$$

#Comparação1

Para o **Pior Caso**, considerando que TODAS as comparações serão feitas, teremos:

#Comparação2,3,4

$$1 + (\sum_{1}^{n-1} 2) + (\sum_{1}^{n-1} \sum_{1}^{\frac{n}{2}}) \rightarrow 1 + (2n-2) + ((n-1)(\frac{n}{2} - 1)) \rightarrow$$

#Comparação1

#Comparação5

$$2n + \frac{n^2}{2} - \frac{3n}{2} \rightarrow \frac{n^2}{2} + \frac{n}{2} \rightarrow O(n^2)$$

Exercício 3...

```c
float exemplo(float *a, float *b, int n){
    float soma = 0.0;

    for(int i=0; i<n; i++){
        if(a[i] > 600) // Soma 1
            for(int j=n-1; j>=0; j--) soma += a[i]*b[i];

        else if(a[i] < 300) // Soma 2
            for(int j=n; j<n*n; j+=5) soma += a[i]*j;

        else soma += b[i]/a[i];
    }

    return soma;
}
```

No **Melhor Caso**, temos que iremos considerar das somas presentes no laço de repetição, apenas a soma 1 irá ocorrer, obtemos então:

$$\sum_{0}^{n-1}\sum_{0}^{n-1} 1 \rightarrow \sum_{0}^{n-1}((n-1)-0+1) \rightarrow n((n-1)-0+1) \rightarrow n^2 \rightarrow O(n^2)$$

Para o **Pior Caso**, considerando que soma 1 e soma 2 serão feitas, teremos:

$$\sum_{0}^{n-1}\sum_{n}^{\frac{n^2}{5}} 1 \rightarrow \sum_{0}^{n-1}(\frac{n^2}{5} - n + 1) \rightarrow n(\frac{n^2}{5} - n + 1)$$

$$-n^2 + \frac{n^3}{5} - n^2 + n \rightarrow O(n^3)$$

#Soma1 **#Soma2**

```c
int moda(int *vetor, int tam){
    int moda[tam];

    // preenchendo vetor com zero
    for(int i=0; i<tam; i++) moda[i] = 0;

    // somando as modas do vetor
    for(int i=0; i<tam; i++) moda[vetor[i]] += 1;

    // verificando maior moda
    int maiorModa = moda[0], pos = 0;
    for(int i=0; i<tam; i++)
        if(moda[i] > maiorModa){
            maiorModa = moda[i];
            pos = i;
        }

    // retornando maior moda
    return maiorModa;
}
```

Exercício 4...

No **Melhor Caso**, temos que considerar que das atribuições feitas ocorrem apenas uma vez, dessa forma, no último if elas não irão acontecer.

$$(\sum_{0}^{n-1} 1) + (\sum_{0}^{n-1} 1) + 2 \rightarrow n + n + 2 \rightarrow 2n + 2 \rightarrow O(n)$$

Para o **Pior Caso**, considerando que TODAS as atribuições serão feitas, temos:

$$(\sum_{0}^{n-1} 1) + (\sum_{0}^{n-1} 1) + 2 + (\sum_{0}^{n-1} 2) \rightarrow$$

$$n + n + 2 + 2n \rightarrow 4n + 2 \rightarrow O(n)$$