Universidade Federal do Rio Grande do Norte
Estrutura de dados I
Lista de exercícios - 1ª Unidade

Questão 01 - Determine o termo dominante e a complexidade Big-O das equações abaixo:

Expression	Dominant term(s)	$O(\ldots)$
$5 + 0.001n^3 + 0.025n$		
$500n + 100n^{1.5} + 50n \log_{10} n$		
$0.3n + 5n^{1.5} + 2.5 \cdot n^{1.75}$		
$n^2 \log_2 n + n(\log_2 n)^2$		
$n \log_3 n + n \log_2 n$		
$3 \log_8 n + \log_2 \log_2 \log_2 n$		
$100n + 0.01n^2$		
$0.01n + 100n^2$		
$2n + n^{0.5} + 0.5n^{1.25}$		
$0.01n \log_2 n + n(\log_2 n)^2$		
$100n \log_3 n + n^3 + 100n$		
$0.003 \log_4 n + \log_2 \log_2 n$		

Questão 02 - Os algoritmos A e B gastam exatamente Ta(n) = 0,1n²logn e Tb(n) = 2,5n² unidades de tempo respectivamente, para um problema de tamanho n. Escolha o algoritmo que tem melhor desempenho na notação Big-O.

Questão 03 - Como a notação Big-O é usada para descrever a complexidade de tempo dos algoritmos?

Questão 04 - Descreva resumidamente as principais características e o funcionamento dos algoritmos de ordenação bubble sort, selection sort, insertion sort, merge sort e quick sort.

Questão 05 - Dado o vetor = {8, 9, 7, 9, 3, 2, 3, 4, 6, 1} explique o passo a passo executado pelo algoritmo bubble sort para ordenar de forma crescente (a resposta pode ser escrita ou através de diagramas).

Questão 06 - Descreva resumidamente quais as principais características e diferenças entre os algoritmos de busca binária e de busca linear.

Questão 07 - O programa abaixo foi escrito de forma iterativa. Escreva esse algoritmo de maneira recursiva de forma que o resultado final seja o mesmo.

```
int fiboIterativo(int n){
    int a = 0, b = 1, c;
    int i = 2;
    if(n == 1){
        return a;
    }else{
        if(n == 2){
          return b;
        }else{
            while(i < n){
                c = a + b;
                a = b;
                b = c;
                i++;
            }
            return c;
        }
    }
}
```

Questão 08 - Explique porque, para problemas muitos grandes, não é recomendável utilizar soluções recursivas.

Questão 09 - Defina o que é caso base (condição de parada) de um algoritmo recursivo.

Questão 10 - Dado o algoritmo abaixo, calcule a complexidade local e a complexidade assintótica (O).

```c
int buscaMenor(int **matriz, int linhas, int colunas){
    int menor = matriz[0][0];
    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            if (matriz[i][j] < menor) {
                menor = matriz[i][j];
            }
        }
    }
    return menor;
}
```