

Lista de exercícios:

1. Implemente a função da busca sequencial com a seguinte declaração:

```
int buscaSequencial(int elem, int *vet, int n);
```

2. Implemente a função da busca binária com as seguintes declarações:

```
a) int buscaBinaria(int elem, int *vet, int n);
```

```
b) int buscaBinariaRecursiva(int elem, int* vet, int a, int b);
```

3. Compare a busca binária e sequencial em relação ao número de comparações com um vetor ordenado de tamanho 100 e depois com um de tamanho 1000.

4. Implemente novamente a função de Fibonacci com a seguinte modificação: a função agora recebe um ponteiro para um inteiro chamado contador cujo valor para o qual ele aponta é inicializado com 0 no início do main. Em cada chamada da função de Fibonacci, incremente o valor para o qual o ponteiro aponta. Imprima o valor do ponteiro após a execução para saber quantas vezes a função de Fibonacci foi chamada.

5. Implemente a função de Fibonacci com um algoritmo de complexidade $O(n)$.

6. Implemente uma função recursiva que inverta um vetor:

```
void inverteVetor(int *vet, int inicio, int fim);
```

7. Implemente uma função que calcula o produto de duas matrizes $n \times n$. Quantas vezes a operação de multiplicação entre dois números é feita em função de n ?

8. Implemente uma função recursiva que, dado um número inteiro $x = d_1 d_2 d_3 \dots d_n$ retorne a soma de todos os seus dígitos. Por exemplo, se $x = 123$, o resultado é $1 + 2 + 3 = 6$.

9. Considere a seguinte definição de máximo divisor comum dada pelo algoritmo de euclides:

$MDC(A,B) = A$ se $B = 0$

$MDC(A,B) = B$ se $A = 0$

$MDC(A,B) = MDC(B,R)$ caso contrário, onde R é o resto da divisão de A por B .

Implemente então o algoritmo para encontrar o MDC(A,B) de forma recursiva.

Desafio: Determinante de uma matriz nxn

Implemente o cálculo do determinante usando o teorema de Laplace, onde o determinante é dado como uma combinação dos cofatores C_{ij} de uma linha i qualquer:

$$C_{i,j} = (-1)^{i+j} \det(A_{ij}),$$

$$\det(A) = a_{i,1}C_{i,1} + a_{i,2}C_{i,2} + \dots + a_{i,n}C_{i,n}.$$

Dica: Implemente uma função que calcule A_{ij} , que é a matriz obtida pela remoção da i -ésima linha e j -ésima coluna da matriz original. Note que a nova matriz tem dimensão $(n-1) \times (n-1)$. Em algum ponto será necessário calcular a determinante de uma matriz 1×1 , que é simplesmente um escalar.