UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
CURSO DE CIÊNCIA DA COMPUTAÇÃO
ESTRUTURA DE DADOS II

LABORATÓRIO 1

Complexidade de Algoritmos

EXERCÍCIOS DE REVISÃO

1. Expresse as funções abaixo em notação O:
    a. $T = n^3/1000 - 100n^2 - 100n + 3$
    b. $T = n \lg n + n$
    c. $T = 3n \log n + 2^n$
    d. $T = n^2 + 2\log n$
    e. $T = 302$

2. Ache a complexidade do algoritmo:

```
ALGORITMO
  se A[i] > A[i+1] então
  |   temp = A[i]
  |   A[i] = A[i+1]
  |   A[i+1] = temp
```

3. Ache a complexidade do algoritmo a seguir:

```
ALGORITMO
  para i = 0 até comprimento(A) - 1 faça
  |  se A[i] > A[i+1] então
  |  |   temp = A[i]
  |  |   A[i] = A[i+1]
  |  |   A[i+1] = temp
```

4. Ache a complexidade do algoritmo de ordenação Bubble Sort:

```
BUBBLE-SORT(A)
  faça
  |  troca = falso
  |  para i = 0 até comprimento(A) - 1 faça
  |  |  se A[i] > A[i+1] então
  |  |  |   trocar(A[i], A[i+1])
  |  |  |   troca = verdadeiro
  enquanto troca
```

EXERCÍCIOS DE PROGRAMAÇÃO

1. Implemente o algoritmo de ordenação por inserção (Insertion Sort). O algoritmo deve ser implementado em uma função que recebe um vetor de inteiros como argumento.

```
INSERTION-SORT(A)
  para j = 1 até comprimento(A) faça
  |     chave = A[j]
  |     i = j-1
  |     enquanto i >= 0 e A[i] > chave faça
  |     |     A[i+1] = A[i]
  |     |     i = i – 1
  |     A[i+1] = chave
```

2. Implemente o algoritmo de ordenação por flutuação (Bubble Sort). Para isso implemente também a função trocar, que deve receber dois valores inteiros e trocar os valores das variáveis na função chamadora.

```
BUBBLE-SORT(A)
  faça
  |  troca = falso
  |  para i = 0 até comprimento(A) - 1 faça
  |  |  se A[i] > A[i+1] então
  |  |  |   trocar(A[i], A[i+1])
  |  |  |   troca = verdadeiro
  enquanto troca
```

3. O C++ possui uma biblioteca chamada **chrono** que permite calcular o tempo de execução de uma tarefa utilizando um contador de alta precisão da CPU. O Exemplo abaixo mostra como usar este contador para obter o tempo de execução de um trecho de código.

```
#include <iostream>
#include <chrono>
using namespace std;
using namespace std::chrono;
int main()
{
    auto t0 = high_resolution_clock::now();

    // insira atividade a ser cronometrada aqui

    auto t1 = high_resolution_clock::now();

    cout << "Tempo da atividade: "
         << duration_cast<nanoseconds>(t1-t0).count()
         << " nanosegundos" << endl;

    system("pause");
    return 0;
}
```

Compare o tempo de execução do Insertion Sort com o Bubble Sort.