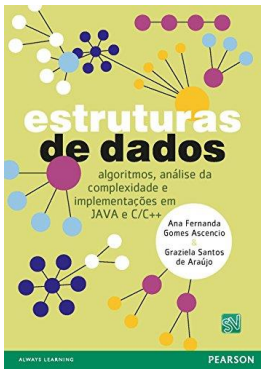
FACULDADE ENGENHEIRO SALVADOR ARENA
ENGENHARIA DA COMPUTAÇÃO
ESTRUTURA DE DADOS – Aula 2 – 1º SEMESTRE/2024

TEORIA: ALGORITMOS DE ORDENAÇÃO (PARTE 1)

Nossos **objetivos** nesta aula são:

- Conhecer o algoritmo de ordenação bubble sort
- Praticar o uso do algoritmo de ordenação bubble sort
- conhecer geração de números aleatórios inteiros em C/C++
- praticar o uso de números aleatórios inteiros em C/C++
- conhecer geração de números aleatórios reais em C/C++
- praticar o uso de números aleatórios reais em C/C++



Para esta aula, foram utilizados como referência o **Capítulo 2 (Algoritmos de Ordenação e Busca)** do nosso livro-texto:

ASCENCIO, Ana Fernanda Gomes; ARAÚJO, Graziela Santos de. Estruturas de Dados: algoritmos, análise da complexidade e implementações em JAVA e C/C++. **São Paulo: Pearson Prentice Halt**, v. 3, 2010.

Não deixem de ler depois desta aula!

ALGORITMOS

- algoritmos de ordenação são procedimentos computacionais que organizam um conjunto de elementos em uma ordem específica.
- reorganizar os elementos de forma crescente ou decrescente, facilitando a busca, recuperação e manipulação de dados.
- existem diversos algoritmos de ordenação com diferentes abordagens e eficiências.
- a escolha de um algoritmo depende do tamanho do conjunto de dados e dos recursos disponíveis.
- algoritmos a serem estudados na disciplina são Bubble Sort, Selection Sort, Insertion Sort, Merge Sort e Quick Sort.

BUBBLE SORT

- é um de ordenação simples e intuitivo.
- realiza a ordenação por flutuação.
- o algoritmo percorre repetidamente a lista a ser ordenada.
- compara cada elemento com a posição adjacente realizando a troca caso estejam fora de ordem.
- o processo é repetido continuamente até que a lista ser completamente ordenada.

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Realize a análise exploratória visual do funcionamento do algoritmo Bubble sort

https://www.toptal.com/developers/sorting-algorithms/bubble-sort

https://www.cs.usfca.edu/~galles/visualization/ComparisonSort.html

ANÁLISE BUBBLE SORT

- é um algoritmo de ordenação ineficiente para grandes arranjos.
- significa que seu tempo de execução aumenta rapidamente à medida que o tamanho da lista aumenta.
- o pior caso ocorre quando a lista está completamente desordenada.
  - o número total de comparações e trocas é dado por uma soma aritmética.
  - uma lista com n elementos, o número de comparações será aproximadamente:

    $(n-1) + (n-2) + ... + 1$, o que é equivalente a $(n^2 - n) / 2$.
  - o número de trocas é o mesmo que o número de comparações.
  - o pior caso de tempo de execução é $O(n^2)$.

EXERCÍCIO TUTORIADO

Qual é o número máximo de comparações e trocas possíveis para uma lista de 10 posições completamente desordenada utilizando o algoritmo Bubble Sort?

o número total de comparações e trocas no pior caso é dado por uma soma aritmética dos primeiros 9 posições. Assim, o algoritmo realizará 45 comparações e 45 trocas no pior caso.

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Qual é a versão do C++ utilizada em sua estação de trabalho ?

```cpp
#include <iostream>

int main() {
    std::cout << "Versão do C++: " << __cplusplus << std::endl;
    return 0;

}
```

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Defina, atribua valores e imprima um arranjo contendo 10 posições do tipo inteiro ?

```cpp
#include <iostream>

int main() {
    // Definição do vetor de 10 posições
    int vetor[10];

    // Atribuindo valores ao vetor
    for (int i = 0; i < 10; i++) {
        vetor[i] = i + 1;
    }

    // Imprimindo os valores do vetor
    for (int i = 0; i < 10; i++) {
        std::cout << "vetor[" << i << "] = " << vetor[i] << std::endl;
    }

    return 0;
}
```

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Defina um arranjo e atribua os valores em sua inicialização. Em seguida, imprima o conteúdo do arranjo.

```cpp
#include <iostream>

int main() {
    // Definição e atribuição de valores no vetor de 10 posições
    int vetor[] = {4, 13, 17, 22, 25, 12, 22, 11, 41, 90};

    // Imprimindo os valores do vetor
    for (int i = 0; i < 10; i++) {
        std::cout << "vetor[" << i << "] = " << vetor[i] << std::endl;
    }

    return 0;
}
```

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Como obter o tamanho do arranjo na linguagem C++ ?

```cpp
int main() {

    int arr[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
    int tamanhoArranjo = sizeof(arr) / sizeof(arr[0]);
    std::cout << tamanhoArranjo << std::endl;
    return 0;

}
```

ALGORITMO BUBBLE SORT

1. percorra a lista a ser ordenada, comparando cada elemento com o próximo.
2. se o elemento atual for maior que o próximo, realize a troca.
3. continue percorrendo a lista até o final.
4. repita os passos 1 a 3 até que nenhum elemento precise mais ser trocado.
5. a lista estará ordenada após completar todas as iterações.

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Implemente um método do algoritmo Bubble Sort na linguagem C++ ?

```cpp
#include <iostream>

void bubbleSort(int arranjo[], int tamanho) {

    for (int i = 0; i < tamanho - 1; i++) {

        for (int j = 0; j < tamanho - i - 1; j++) {
            if (arranjo[j] > arranjo[j + 1]) {
                // Troca os elementos
                int temp = arranjo[j];
                arranjo[j] = arranjo[j + 1];
                arranjo[j + 1] = temp;
            }
        }
    }

}
```

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Implemente o método principal com a chamada ao método bubbleSort construído previamente passando um arranjo de inteiros.

```
1   int main() {
2       int arranjo[] = {64, 34, 25, 12, 22, 11, 90};
3       int tamanho = sizeof(arranjo) / sizeof(arranjo[0]);
4
5       std::cout << "Arranjo original: ";
6       for (int i = 0; i < tamanho; i++) {
7           std::cout << arranjo[i] << " ";
8       }
9       std::cout << std::endl;
10
11      bubbleSort(arranjo, tamanho);
12
13      std::cout << "Arranjo ordenado: ";
14      for (int i = 0; i < tamanho; i++) {
15          std::cout << arranjo[i] << " ";
16      }
17      std::cout << std::endl;
18
19      return 0;
20  }
```

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Gere dois números aleatórios entre 0 e 100, sendo um número inteiro e outro real na linguagem C++.

EXERCÍCIO COM DISCUSSÃO EM DUPLAS

Implemente um arranjo de inteiros com 100 posições, contendo valores aleatórios entre 0 e 1000. Em seguida, imprima o conteúdo do arranjo não ordenado e, após ordene-o e imprima o seu conteúdo ordenado registrando as métricas de tempo de execução.