# Educação Profissional Paulista

## Técnico em **Ciência de Dados**

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO

# Lógica de programação e algoritmos

## Prática de busca e ordenação

Aula 3 - Comparação detalhada dos métodos de busca e ordenação

Código da aula: [DADOS]ANO1C3B4S25A3

Secretaria da Educação
SÃO PAULO
GOVERNO DO ESTADO

# Mapa da Unidade 1 Componente 4

**Lógica de programação e algoritmos**

semana 25

**Você está aqui!**
Prática de busca e ordenação

semana 27

Algoritmos de contagem e acumulação

semana 28

Pilhas e filas

Lógica de programação e algoritmos

Mapa da Unidade 1 Componente 4

# Você está aqui!

25

Prática de busca e ordenação

**Aula 3**

Código da aula:
[DADOS]ANO1C3B4S25A3

## Objetivos da aula

- Introduzir e praticar conceitos sobre algoritmos de busca e ordenação.

## Recursos didáticos

- Recurso audiovisual para exibição de vídeos e imagens;
- Acesso ao laboratório de informática e/ou internet.

## Duração da aula

50 minutos.

## Competências técnicas

- Identificar e resolver problemas relacionados a dados e análises;
- Compreender e dominar técnicas de manipulação de dados.

## Competências socioemocionais

- Adaptar-se a novas tecnologias, técnicas e tendências sem perder o foco, as metas e os objetivos da organização;
- Colaborar efetivamente com outros profissionais, como cientistas de dados e engenheiros de dados; trabalhar em equipes multifuncionais colaborando com colegas, gestores e clientes.

Construindo o **conceito**

# Revisão das aulas anteriores

Nas aulas 1 e 2, exploramos os fundamentos dos algoritmos de busca e ordenação.

- Na aula 1, introduzimos conceitos teóricos de busca linear e binária, além de ordenação por seleção e inserção, explicando com exemplos detalhados. A prática foi centrada em exemplos de código em Python, ajudando a entender como implementar e onde cada método é aplicável.
- Na aula 2, avançamos para a aplicação prática, realizando exercícios de implementação desses algoritmos em Python, ordenando lista de números e palavras, e executando buscas específicas.
- Na sessão prática, consolidamos a compreensão dos algoritmos, ao ver o impacto direto das diferentes abordagens de busca e ordenação em termos de eficiência e simplicidade.

Construindo o **conceito**

# Função *timeit*

- A função ***timeit*** do Python é uma ferramenta importante que permite medir o tempo de execução de pequenos trechos de código, sendo extremamente útil para realizar *benchmarks* e otimizações de desempenho em funções e algoritmos;
- A função ***timeit*** executa um código várias vezes (um *loop*) para obter uma medição de tempo mais precisa, eliminando variações casuais que possam ocorrer em uma única execução. Isso ajuda a garantir que o tempo medido seja representativo do desempenho real do código.

**Dica**

Ao utilizar *timeit*, é possível obter uma visão mais clara e precisa do desempenho do código, especialmente útil para comparar a eficiência de diferentes abordagens ou algoritmos.

Colocando em **prática**

# Exercício: Comparação de algoritmos de busca e ordenação

**Objetivo:**
Explorar, implementar e comparar diferentes algoritmos de busca e ordenação em termos de eficiência e aplicabilidade.

**Descrição do exercício:**
Aplique os algoritmos de busca linear, busca binária, ordenação por seleção e ordenação por inserção com a lista desordenada de números e a lista de palavras indicadas abaixo. Adicionalmente, meça e compare o tempo de execução de cada algoritmo de ordenação usando o módulo *timeit* do Python.

**Lista de números:** [45, 22, 11, 89, 76, 54, 33, 21, 10, 77].

**Lista de palavras**: ["zebra", "maçã", "cachorro", "banana", "elefante", "gato", "abacate"].

**35 minutos**

**Duplas**

**Código de programação**

# Colocando em **prática**

## Exercício: Comparação de algoritmos de busca e ordenação

**35 minutos**

**Duplas**

**Código de programação**

**1** **Ordenar a lista de números usando:**
- ordenação por seleção;
- ordenação por inserção.

**2** **Ordenar a lista de palavras usando:**
- ordenação por seleção;
- ordenação por inserção.

**3** **Encontrar um número na lista de números usando:**
- busca linear;
- busca binária (após ordenar a lista usando o método de sua escolha).

**4** **Encontrar uma palavra na lista de palavras usando:**
- busca linear;
- busca binária (após ordenar a lista usando o método de sua escolha).

Colocando em **prática**

## Exercício: Comparação de algoritmos de busca e ordenação

**Especificações da entrega:**

- Inserir o código-fonte completo para todas as tarefas;
- Especificar a escolha do método de ordenação para a busca binária;
- Registrar as diferenças observadas nos tempos de execução entre os algoritmos de ordenação;
- Responder qual algoritmo de ordenação se mostrou mais eficiente para cada tipo de lista e por quê.

**Dicas:**

- Utilize comentários no código para explicar as decisões tomadas;
- Use o módulo ***timeit*** para medir o tempo de execução de cada algoritmo e inclua esses dados no relatório.

**35 minutos**

**Duplas**

**Código de programação**

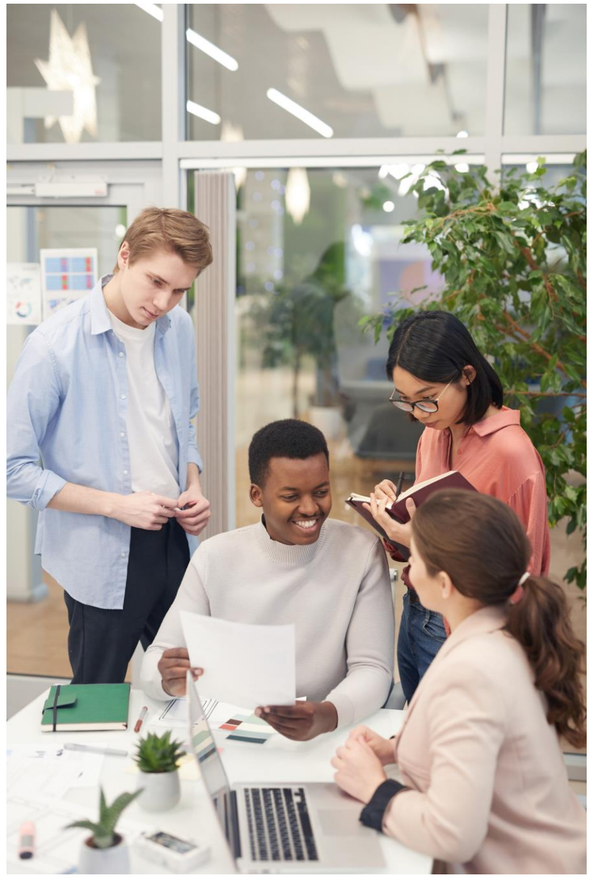
Ser
**sempre +**

# Situação

Imagine que você e seus colegas de turma foram designados para trabalhar em um projeto de final de curso.

O projeto envolve o desenvolvimento de um aplicativo que ajuda os estudantes a organizar seus horários de estudo.

A equipe é diversa, com membros tendo diferentes habilidades e níveis de experiência em programação.



© Getty Images

Situação fictícia elaborada especialmente para o curso.

Ser
**sempre +**

# Ação

1. Você percebe que dois membros da equipe têm dificuldades em se comunicar, o que resulta em mal-entendidos frequentes sobre os requisitos do projeto. Como você poderia facilitar uma comunicação mais clara e eficiente entre eles?

2. Algumas tarefas do projeto requerem habilidades avançadas de programação, e alguns membros da equipe não se sentem confiantes para executá-las. Como você pode ajudar a equilibrar as tarefas de modo que todos se sintam envolvidos e capazes de contribuir?

3. Você nota que a motivação da equipe está diminuindo à medida que o projeto avança, especialmente quando enfrentam desafios técnicos. Como você pode manter a equipe motivada e engajada até a conclusão do projeto?

Situação fictícia elaborada especialmente para o curso.

Diálogo

© Getty Images

O que nós **aprendemos hoje?**

# Nuvem de palavras



O que nós **aprendemos hoje?**

# Então ficamos assim...

**1** Focamos em uma atividade prática que envolveu a aplicação e comparação de diversos algoritmos de busca e ordenação, utilizando listas com 100 elementos.

**2** Implementamos e medimos o desempenho de algoritmos como busca linear, busca binária, ordenação por seleção e ordenação por inserção, utilizando o módulo *timeit* para análise precisa.

**3** O exercício realizado proporcionou uma compreensão aprofundada das diferenças entre os métodos, permitindo identificar as situações mais adequadas para aplicar cada algoritmo. Vimos, também, a importância da escolha de algoritmos em contextos de desenvolvimento de software.

# Saiba mais

Está pronto para ir além das listas e tuplas e dominar as estruturas de dados mais eficientes em Python?

Neste curso, você aprenderá a trabalhar com conjuntos e dicionários, aprimorando suas habilidades para manipular dados de forma organizada e otimizada.

ALURA. Programação: Phyton: Python Collections parte 2: conjuntos e dicionários, [s.d.]. Disponível em: https://www.alura.com.br/curso-online-python-collections-conjuntos-e-dicionarios/. Acesso em: 11 jul. 2024.

# Referências da aula

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. *Lógica de programação*: a construção de algoritmos e estruturas de dados com aplicações em Python. São Paulo: Pearson; Porto Alegre: Bookman, 2022.

# Educação Profissional Paulista

## Técnico em **Ciência de Dados**

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO