Estruturas de Dados e Algoritmos Expert

Nosso guia "à prova de perdidos" sobre o treinamento :)

**Objetivo:**

Domine as principais estruturas de dados e técnicas de elaboração de algoritmos, desde o básico até o avançado, e adquira a base sólida para atuar profissionalmente na área e se sentir confiante nos processos seletivos.

**Para quem é:**

* Estudantes de programação
* Quem precisa repassar conteúdos da faculdade
* Profissionais e estudantes que precisam aprofundar em estruturas de dados e técnicas de algoritmos avançadas

**Pré-requisitos:**

* Lógica de programação em qualquer linguagem
* Conhecimento básico de orientação a objetos em qualquer linguagem

**O que você vai aprender no treinamento:**

* Tópicos iniciais
* Strings
* Arrays
* Recursividade
* Complexidade de algoritmos
* Busca e ordenação
* Listas
* Pilhas e filas
* Conjuntos e dicionários
* Árvores
* Grafos
* Algoritmos gulosos
* Programação dinâmica

**Recursos do treinamento:**

* Trilha de aulas gravadas (assista quantas vezes quiser)
* Materiais de apoio (slides, resumos, guias passo a passo, etc.)
* Suporte 1 a 1 dos instrutores (tenha suas dúvidas respondidas)
* Exercícios práticos (aprenda na prática)
* Desafios para entregar com correção e feedback (valide seu conhecimento)
* Certificado 160h (incluindo tempo de vídeos e de estudo)

**SUMÁRIO**

[**Onde eu acesso o treinamento? 3**](#_kwtxhi4e7grt)

[**Quais são os canais de contato e suporte? 3**](#_xcdi6cg0v4)

[1. Suporte dos professores a dúvidas sobre as aulas 3](#_5sdm8w7gsyiu)

[2. Suporte administrativo (matrícula, acesso, plataforma, etc.) 3](#_ar3dp0gobtrg)

[Observações: 3](#_8c77m4b0jsex)

[**Quem são os responsáveis pelo treinamento? 3**](#_tourkcrnslcb)

[**Onde acesso o material de apoio? 4**](#_omt4ujl914s9)

[Como funciona o envio de desafios? 4](#_szl1565rvuxe)

[Tem prazo para entregar os desafios? 4](#_a9gzu5ujpok5)

[**Como funciona a emissão de certificado? 4**](#_jt40fml5024r)

[**Em qual sequência devo estudar? 4**](#_inf9ijw0j3vh)

[Vocês vão mesmo ensinar "pegando na minha mão"? 5](#_ck75kqxt00s2)

[**Dicas comportamentais de estudo 5**](#_s5q66kwja8qk)

[**Conteúdo curricular do treinamento (ementa) 5**](#_onm5qq80xrop)

[Tópicos iniciais 5](#_v0brtlawys9f)

[Strings 6](#_votcsaygjk93)

[Arrays 7](#_2pith4tfgp3k)

[Recursividade 7](#_h3r4qcaq80r1)

[Complexidade de algoritmos 7](#_unqx3urkul4r)

[Busca e ordenação 8](#_oxe3rktl6m27)

[Listas 8](#_qlk0o1eeu9v3)

[Pilhas e filas 9](#_790r9ovy7a99)

[Conjuntos e dicionários 10](#_9pgvflli5656)

[Árvores 10](#_cie6uxq5sk4)

[Grafos 11](#_lwukkjiu6hz7)

[Algoritmos gulosos 12](#_k29ebsoz0xix)

[Programação dinâmica 13](#_qgdkh6ui0pr9)

# 

# Onde eu acesso o treinamento?

Link da plataforma de ensino:

<https://devsuperior.club>

Usuário: (seu email de inscrição)

Senha: (seu email de inscrição, se for o primeiro acesso)

*\* Para mudar a senha, use o recurso de “Recuperação de senha”*

*\* Se mesmo assim tiver algum problema, nos envie um email:* [*contato@devsuperior.com*](mailto:contato@devsuperior.com)

# Quais são os canais de contato e suporte?

## 1. Suporte dos professores a dúvidas sobre as aulas

Seção "Perguntas e respostas" abaixo da área de cada vídeo aula.

## 2. Suporte administrativo (matrícula, acesso, plataforma, etc.)

Por email: contato@devsuperior.com

## Observações:

* Tempo de resposta: 1 dia **útil** (não inclui finais de semana e feriados).
* Mídias sociais não são canais de suporte (lá trabalham pessoas de marketing, e não professores).
* O suporte é somente sobre o conteúdo das aulas. O suporte não inclui consultoria a outros projetos ou conteúdos fora do treinamento.

# Quem são os responsáveis pelo treinamento?

Várias pessoas/empresas estão trabalhando para te atender. Entenda abaixo:

| Quem | Papel |
| --- | --- |
| Prof. Nelio Alves | Criador do treinamento. |
| Educandoweb Cursos | Escola responsável pela disponibilização e suporte dos cursos. |
| Devsuperior | É nossa marca dos cursos premium. Pertence à Educandoweb. |
| Neme Digital | Nossa agência, responsável pelo marketing e vendas. |
| Eduzz, Pagar.me | Plataforma parceira responsável pelo financeiro. |

| **Nelio Alves**  Doutor em Engenharia de Software, mais de 360 mil alunos online. Ficou mundialmente conhecido por ser autor dos cursos online de Java e C# em Língua Portuguesa mais vendidos do mundo pela plataforma Udemy. Possui mais de 20 anos de carreira, e vasta experiência como professor e coordenador de cursos superiores, técnicos e de pós-graduação. |  |
| --- | --- |

# Onde acesso o material de apoio?

Assista a vídeo aula “Material de apoio” localizada no primeiro capítulo.

# Como funciona o envio de desafios?

Na própria aula onde o desafio é apresentado, na aba “Conteúdo” haverá um formulário onde você deverá enviar o link do seu desafio. Todas orientações sobre a elaboração do desafio serão dadas na aula.

Você receberá o retorno do seu desafio em 1 a 2 dias úteis.

## Tem prazo para entregar os desafios?

Não. Você pode fazer o treinamento no seu tempo e também entregar os desafios no seu tempo, sem problemas. Apenas fique atento(a) ao prazo de validade do seu acesso ao treinamento, ok?

# Como funciona a emissão de certificado?

O certificado é obtido por meio da entrega dos desafios com sucesso, e não somente pela mera visualização das aulas.

Assim que você receber o retorno com sucesso do último desafio do treinamento, seu certificado estará disponível na seção “Meus certificados” da plataforma de ensino.

# Em qual sequência devo estudar?

O treinamento foi cuidadosamente preparado para que você tenha a melhor experiência de aprendizado e a melhor didática. Basta seguir as aulas e capítulos em sequência.

# Vocês vão mesmo ensinar "pegando na minha mão"?

Vamos sim! Vamos "pegar na sua" mão por meio:

1. das aulas gravadas e materiais de apoio, pois é tudo passo a passo, em sequência;
2. do suporte às dúvidas;
3. da correção de tarefas.

Mas para isso funcionar o aluno também tem que PARTICIPAR das atividades do treinamento, que são elas:

1. Assistir às aulas e fazer os exercícios.
2. Enviar dúvidas no canal de suporte.
3. Enviar as tarefas.
4. Nos procurar caso tenha alguma dúvida sobre o funcionamento do treinamento.

Em outras palavras: cada um, professor e aluno, tem que fazer a sua parte. Se estiver com algum problema, tome a iniciativa de entrar em contato, que te ajudamos, pois estamos aqui para isso. Este é um curso assíncrono, onde o aluno faz no seu tempo, então quem dita o ritmo do aluno aqui é o próprio aluno.

# Dicas comportamentais de estudo

* Infelizmente existe uma cultura de "ensino fake", de fabricação de números, de venda de certificados. Não aceite isso para sua vida. Não seja mais uma estatística.
* Não avance para o próximo tópico sem entender e fazer os exercícios indicados com entendimento.
* Precisou olhar na solução de um exercício? Ok, mas tente refazer novamente sem olhar.
* Não tenha pressa! Não se compare com os colegas. Respeite seu ritmo e seu contexto.
* Crie um horário de estudo e siga-o.
* Não copie desafios. Receber ajudas pontuais é natural, mas cuidado para não sabotar seu aprendizado.
* Quando for ajudar alguém, não envie o projeto completo. Apenas ajude pontualmente.

# Conteúdo curricular do treinamento (ementa)

## Tópicos iniciais

Algoritmos e Lógica de Programação

Estruturas de dados é sobre o quê

Precisa saber OO antes de ED

Esse curso também é para outras linguagens

Vamos falar sobre objetos e funções

Tipos estruturados em JavaScript PARTE 1

Tipos estruturados em JavaScript PARTE 2

Funções em Javascript sem OO

Funções em Javascript com OO

Tipos estruturados em Java PARTE 1

Tipos estruturados em Java PARTE 2

Funções em Java sem OO

Funções em Java com OO

Tipos estruturados em CSharp

Funções em CSharp sem OO

Funções em CSharp com OO

Tipos estruturados em Python

Funções em Python sem OO

Funções em Python com OO

Vamos falar sobre comportamento de memória

Tipos referência e tipos valor em Java

Tipos referência e tipos valor em CSharp

Tipos referência e tipos valor em Javascript

Tipos referência e tipos valor em Python

Desalocação de memória garbage collector e escopo local

Visao geral ler arquivo JSON e manipular objetos

Leitura arquivo JSON em JavaScript

Leitura arquivo JSON em Python (Parte 1)

Leitura arquivo JSON e tratamento campo tipo timestamp (Parte 2)

Criacao projeto Java, mapeamento entidades

Inclusao dependencia leitura JSON

Leitura arquivo JSON em Java

Ajustes no projeto java para lidar com atributo timestamp

Criacao projeto C# e mapeamento entidades

Leitura arquivo JSON em C#

## Strings

Literais e expressões em Javascript PARTE 1

Literais e expressões em Javascript PARTE 2

Imutabilidade de strings em Javascript

Funções de string em Javascript PARTE 1

Funções de string em Javascript PARTE 2

Funções de string em Javascript PARTE 3

Expressões regulares

Dica de ChatGPT para outras linguagens

Exemplo limpar CPF

Exemplo testar domínio br

Exemplo encontrar emails em um texto

Referência de expressões regulares PARTE 1

Referência de expressões regulares PARTE 2

Apresentando os exercícios

Solução do problema cpf

Solução alternativa do problema cpf usando for

Solução do problema dominio-email

Solução do problema data1

Solução alternativa do problema data1 usando substring

Solução do problema data2

Solução do problema senha

Solução alternativa do problema senha usando regex

Solução do problema anagram (1)

Solução do problema anagram (2)

Solução alternativa do problema anagram usando array

Solução do problema prefixo-comum

Solução do problema transacoes PARTE 1

Solução do problema transacoes PARTE 2

## Arrays

Exercicio maximo 1s consecutivos

Exercicio produto escalar dois arrays

Exercicio numeros par de digitos

Exercicio encontrar vendedor com maior valor de venda

Exercício quadrado de um array ordenado

Exercicio duplicar zeros

Exercicio merge arrays

Exercicio contem valores duplicados

## Recursividade

Recursividade introdução e motivação

Solução do problema soma-naturais

Solução do problema fatorial

Vantagens e desvantagens da recursividade

Casos base e casos recursivos

Pilha de chamadas

Recursividade de cauda

Problema fatorial com recursividade de cauda

Solução ineficiente do problema fibonacci

Solução fibonacci com recursividade de cauda

Conceito de cabeça e cauda de uma lista

Solução do problema reverse

## Complexidade de algoritmos

O que é complexidade de algoritmos

Exemplo busca sequencial

Análise da complexidade de tempo

Análise da complexidade de espaço

Notação assíntótica

Big O, Big Omega, Big Theta

Exemplo de algoritmo de ordem linear

Discutindo as complexidades mais comuns

Discutindo a complexidade exponencial

Exemplo de algoritmo de ordem quadrática

Exemplo de algoritmo de ordem cúbica

Exemplo de algoritmo de complexidade exponencial

Exemplo de algoritmo de coplexidade logarítmica

## Busca e ordenação

Definição e tipos de busca

Busca sequencial

Busca binaria

Implementacao busca binaria iterativa

Implementacao busca binaria recursiva

Definição e exemplos de ordenação

Bubble sort

Implementacao bubble sort

Melhorias bubble sort e complexidade

Selection sort

Implementacao selection sort

Insertion sort

Implementacao insertion sort

Merge sort

Implementacao Merge sort

Complexidade merge-sort

Quick sort

Implementando quick sort

Complexidade quick sort

## Listas

Visao geral capítulo

Definicao lista encadeada e representação na memória

Analogia, vantagens e desvantagens

Implementacao estrutura no

Adicionar elemento ao final da lista

Imprimir elementos da lista

Obter tamanho da lista

Verificar lista vazia

Limpar lista

Adicionar elemento no inicio da lista

Obter elemento em uma posicao especifica

Adicionar elemento em uma posicao especifica

Obter posicao de um elemento

Verificar se elemento existe na lista

Remover elemento de uma posicao especifica (Parte 1)

Remover elemento de uma posicao especifica (Parte 2)

Remover elemento especifico

Apresentacao exemplo pratico to-do List

Organizando projeto, implementando classe Task

Classe TaskList

Adicionar tarefas

Exibindo lista de tarefas

Obter lista com tarefas de um dado tipo

Buscar tarefa por id

Deletar tarefa por id

Marcar tarefa como concluida por id

Editar descrição, tipo, status de uma tarefa por id

Apresentacao algoritmo reposicionar tarefa

Reposicionar tarefa

Lista duplamente encadeada, vantagens e desvantagens

Estrutura no

Estrutura lista duplamente encadeada

Adicionar elemento ao final da lista duplamente encadeada

Exibir elementos da lista duplamente encadeada

Limpar lista, obter tamanho, verifica se esta vazia

Adicionar elemento no inicio da lista duplamente encadeada

Obter elemento em uma posicao especifica

Adicionar elemento em uma posicao especifica da lista duplamente encadeada

Obter posicao elemento, verifica se elemento existe na lista

Remover primeiro elemento lista duplamente encad

Remove ultimo elemento lista duplamente encad

Remover elemento de uma posicao especifica lista duplamente encad

Remover elemento especifico lista duplamente encad

Reverter lista duplamente encadeada

## Pilhas e filas

Visão geral do capítulo

Pilha - definição e aplicações

Problema exemplo is-balanced

Operações de uma pilha

Implementação de pilha com array PARTE 1

Implementação de pilha com array PARTE 2

Implementação de pilha com lista

Solução do problema is-balanced

Usando a pilha da própria linguagem

Solução do problema valid-parentheses

Solução do problema remove-duplicates

Fila - definição e aplicações

Operações de uma fila

Implementação de fila com lista PARTE 1

Implementação de fila com lista PARTE 2

Solução do problema tickets PARTE 1

Solução do problema tickets PARTE 2

Solução do problema sandwich PARTE 1

Solução do problema sandwich PARTE 2

## Conjuntos e dicionários

Visão geral do capítulo

Conjunto - definição e aplicações

Implementações de conjunto

Operações de um conjunto

Testando as operações de conjunto

Solução do problema visitantes

Solução do problema alunos

Solução do problema intersection

Dicionário - definição e aplicações

Implementações de dicionários

Operações de um dicionário

Testando operações de dicionário

Solução do problema votacao

Solução do problema word-count PARTE 1

Solução do problema word-count PARTE 2

Solução do problema two-sum PARTE 1

Solução do problema two-sum PARTE 2

Solução do problema transacoes PARTE 1

Solução do problema transacoes PARTE 2

## Árvores

Visão geral e aviso sobre conteúdo avançado

Árvores - definição e aplicações

Raiz, filho, pai, irmão, nós externos e internos, arestas

Caminho, ancestral, descendente, subárvore, árvore binária

Profundidade, nível, altura

Árvores genéricas

Projeto da árvore genérica

Começando a classe Node

Finalizando a classe Node

Convenção underline para métodos protegidos

Começando GenericTree e método add

Primeiro teste da árvore genérica

Função validade e children atualizada

Implementando o método add completo

Melhorando o print e instanciando toda árvore

DFS - Busca em profundidade

Funções elements e positions

Função find

Funções isExternal, isRoot e parent

Funções replace, size, isEmpty

Função remove PARTE 1

Função remove PARTE 2

BFS - Busca em largura

## Grafos

Por que estudar grafos

História dos grafos

Definição Grafos

Conceitos Básicos - Parte 1

Conceitos Básicos - Parte 2

Conceitos Básicos - Parte 3

Resolução Exercícios - Conceitos Básicos

Tipos de Grafos

Resolução Exercícios - Tipos de Grafos

Caminhos em Grafos - Parte 1

Caminhos em Grafos - Parte 2

Resolução Exercícios - Caminhos em Grafos - Parte 1

Resolução Exercícios - Caminhos em Grafos - Parte 2

Resolução Exercícios - Caminhos em Grafos - Parte 3

Resolução Exercícios - Caminhos em Grafos - Parte 4

Estruturas em Grafos - Parte 1

Estruturas em Grafos - Parte 2

Resolução Exercícios - Estrutura em Grafos - Parte 1

Resolução Exercícios - Estrutura em Grafos - Parte 2

Representação de Grafos em Memória - Lista de arestas

Representação de Grafos em Memória - Matriz de Adjacência

Implementação Matriz de Adjacência - Parte 1

Implementação Matriz de Adjacência - Parte 2

Representação de Grafos em Memória - Lista de Adjacências

Implementação Lista de Adjacências - Parte 1

Implementação Lista de Adjacências - Parte 2

Resolução Exercícios - Representação de Grafos em Memória - Parte 1

Resolução Exercícios - Representação de Grafos em Memória - Parte 2

Percorrendo um grafo

Busca em Largura - Intuição

Busca em Largura - Algoritmo

Busca em Largura - Implementação

Exercício Busca em Largura 1

Exercício Busca em Largura 2

Busca em Profundidade - Intuição

Busca em Profundidade - Algoritmo Recursivo

Busca em Profundidade - Algoritmo Iterativo

Busca em Profundidade - Implementação Iterativa

Exercício Busca em Profundidade

Busca em Profundidade - Tempos de entrada e saída

Ordenação Topológica - Parte 1

Ordenação Topológica - Parte 2

O Problema do Menor Caminho

Conceitos fundamentais menores caminhos

Algoritmo de Dijkstra

Resolução exercício - Algoritmo de Dijkstra

Implementação Algoritmo de Dijkstra

Algoritmo de Bellman-Ford - Parte 1

Algoritmo de Bellman-Ford - Parte 2

Algoritmo de Bellman-Ford - Parte 3

Implementação Algoritmo de Bellman-Ford

Algoritmo de Floyd-Warshall

Implementação Algoritmo de Floyd-Warshall

Union-Find (DSU) - Parte 1

Union-Find (DSU) - Parte 2

Union-Find (DSU) - Implementação

Árvores Mínimas

Algoritmo de Prim

Algoritmo de Prim - Implementação

Algoritmo de Kruskal

Algoritmo de Kruskal - Implementação

Apresentação Lista Exercícios

## Algoritmos gulosos

Visão geral do capítulo

O que é um algoritmo guloso

Problemas clássicos guloso - Problema da moeda

Implementação do problema da moeda

Problemas clássicos guloso - Agendamento de intervalos

Implementação do problema agendamento de intervalos

Problemas clássicos guloso - Colocando feras na jaula

Implementação o problema colocando feras na jaula

Algoritmo guloso - Breve revisão e panorama

Apresentação da lista de exercícios guloso

Solução Garrafas

Solução Scarecrow

Solução Cookies

Solução Maior número possível

Solução Lemonade

Solução Minimum rooms

Solução Boats

Solução Tarefas e prazos

Solução Police and thieves

Solução polícia e ladrão

## Programação dinâmica

Visão geral do capítulo

Por que estudar Programação Dinâmica

Introdução à Programação Dinâmica Parte 1

Introdução à Programação Dinâmica Parte 2

Programação Dinâmica - Consolidando conceitos

Problema do Troco - Parte 1

Problema do Troco - Parte 2

Problema Caminhos no Grid - Parte 1

Problema Caminhos no Grid - Parte 2

Problema Soma Contígua Máxima - Parte 1

Problema Soma Contígua Máxima - Parte 2

Problema Maior Subsequência Crescente - Parte 1

Problema Maior Subsequência Crescente - Parte 2

Problema Corte de Hastes - Parte 1

Problema Corte de Hastes - Parte 2

Problema Corte de Hastes - Parte 3

Problema da Mochila - Parte 1

Problema da Mochila - Parte 2

Problema da Mochila - Parte 3

Problema Maior Subsequência Comum - Parte 1

Problema Maior Subsequência Comum - Parte 2

Problema Maior Subsequência Comum - Parte 3

Problema Maior Subsequência Comum - Parte 4

Apresentação da lista de exercícios

Solução do problema climbing\_stairs

Solução do problema mincost\_climbing\_stairs

Solução do problema frog\_jumps

Solução do problema coins

Solução do problema minimum\_path\_sum

Solução do problema precious\_stones

Solução do problema jump\_game

Solução do problema min\_falling\_path\_sum

Solução do problema house\_robber