Anotações do Hiring Coders

**Lógica de programação**

Algoritmo é uma sequência finita de passos (instruções/comandos) ordenados, cujo objetivo é a resolução de um problema através da transformação da informação (dados).

O computador não faz absolutamente nada com conhecimento prévio (ele não tem isso). Você deve ser o mais específico possível nas suas instruções (e saber o que cada uma faz).

Entender que as instruções podem ser executadas de forma repetitiva e também pode haver alternativas de instruções.

Entrada 🡪 Processamento 🡪 Saída

Entrada: Tudo o que a gente precisa informar ao programa

Processamento: A sequência de passos para resolver o problema

Saída: O resultado esperado

Levando em conta a sequência das instruções em um determinado fluxo, é válido afirmar que as instruções são sequenciais, porém podem sofrer desvios de execução (tais como as condicionais) ou mesmo serem executadas repetidas vezes.

**HTML**

HTML significa Hyper Text Markup Language, em outras palavras Linguagem de Marcação de Hipertexto. Trata-se de uma linguagem de marcação utilizada na construção de páginas Web.

**Estrutura de Dados**

Quando falamos em Tipos Abstratos de Dados, nos referimos a novos tipos de dados, oriundos da combinação de tipos existentes, manipuláveis através de operações.

Pilha, fila e lista

Pilha, fila e listas tratam de conjuntos, e conjuntos têm critérios para colocar os elementos em uma certa ordem.

Fila: first in first out  
 Pilha: last in first out   
 Lista: ordem alfabética, ordem crescente, ordem decrescente, etc.

Filas e Pilhas têm capacidades máximas, por isso elas podem ser implementadas através de listas. Listas são conceitos que nós temos para valores que a gente armazena em um conjunto, só que esse conjunto tem tamanho “infinito. As listas podem ser classificadas por outras ordens, como ordem alfabética, numérica, decrescente, crescente.

Busca de elementos

Busca sequencial: a partir de um determinado conjunto de valores, eu quero encontrar em qual posição está aquele valor, ou, se a posição não for encontrada, eu retorno um valor

Busca binária: exige que os valores estejam ordenados.

Tabelas hash: calcula a posição do vetor e acessa o elemento em um único passo.

**JavaScript**

“JavaScript é uma das linguagens de programação mais populares atualmente. É conhecida como a linguagem da internet porque os navegadores a entendem de modo nativo sem a instalação de qualquer plugin.”

Loiane Groner

**Estruturas de dados e algoritmos com JavaScript, 2018**

Javascript também é usado em ferramentas para:

* Apps Mobile (React Native)
* Games
* Interfaces gráficas de programas para desktop
* Robótica
* Internet das coisas
* E muito mais

Variáveis

Variáveis são usadas para armazenar dados que podem ser atualizados ou recuperados sempre que necessário.



A diferença entre elas é de escopo. Escopo é o lugar dentro do algoritmo ou dentro de uma função onde a nossa variável fica disponível.

Quando fazemos declarações de variáveis usando a palavra reservada *var*, estamos declarando um escopo global ou um escopo de funções, enquanto *let* e *const* são escopos bloqueados, não podem ser acessados fora daquele escopo. Além disso, *var* podem ser atualizadas ou reatribuídas dentro desses escopos, já as variáveis *let*, só podem ser atualizadas, e as *const*, não podem ser atualizadas nem retribuídas, além disso elas precisam já ser inicializadas no momento que está sendo declarada. Isso ocorre porque *var*  sofrem *hoisting* de escopo.

Obs: lembrando também que variáveis não podem ser declaradas começando com números, não podem ter espaço entre uma palavra e outra ou usar palavras reservadas à linguagem do JavaScript.

Tipos de dados

JavaScript é uma linguagem dinamicamente tipada. O tipo de dado pode ser alterado conforme a necessidade do algoritmo.

***String:*** texto

***Number:*** números inteiros e decimais

***Null:*** objeto vazio

***Undefined****:* valor que não foi definido

***Boolean:*** tipo lógico (por exemplo: true or false)

***Objeto****:* mapeamento entre chaves e valores, (valores devem ser declarados em chaves) pode declarar um conjunto de dados

***Array:*** conjunto de informações que podem ser perfeitos para representação de lista e de conjuntos

***Function:*** um objeto que tem a capacidade adicional de ser executado

O operador *typeof* pode ser utilizado para identificar o tipo de um dado.

Operadores

**Operadores aritméticos**

Adição: +

Subtração: –

Multiplicação: \*

Divisão: /

Módulo ou resto de uma divisão: %

Incremento: ++

Decremento: --

**Operadores de atribuição**

=: atribuiu um valor

//serve pra nada//

**Operador de comparação**

Comparar valores: ==

Comparar valores e tipo: ===

Diferente de: !=

Menor: < ou <=

Maior: > ou >=

**Operadores lógicos**

e: &&

ou: ||

negação: !

Condicionais

Para fazer condicionais, utilizamos *if*, *else* e *switch*.

Estruturas de repetição

*for*, *while* e *do while*

*for* é muito utilizado quando se precisa aliterar *arrays*, quando se usa matrizes e conjuntos. *while* é uma estrutura bem parecida com *for*, mas faz a verificação primeiro para depois executar o bloco de código. *do while* primeiro faz a primeira rodada de bloco de código e depois vai verificar se aquela condição continua verdadeira para continuar executando o looping.

Tudo que foi ensinado nessas aulas está em uma fork do gituhub que a professora disponibilizou.

**TypeScript**

O TypeScript é um superset do JavaScript. Ele extrapola a linguagem do JavaScript, adicionando tipagem estática, interfaces, enums, decorators, generics.

Os browsers não leem TypeScript. Portanto, os programadores escrevem códigos em TypeScript, e esse código é transpilado para JavaScript.

Tipos básicos

*boolean, number, string, array*, *tuple, enum, any, void, null* e *undefined, object, never.*

Para se referir ao tipo, o declare com letra minúscula, como escrito acima. Se declarar com letra maiúscula, vai declarar interface. *Array* só pode ser usada como interface, o tipo dentro do conjunto precisa ser declarado. *Tuple* é uma variação de *Array*.

**Union Types**

Quando um parâmetro ou uma variável pode ter mais de um tipo. *number | string* = *number* ou *string.*

**Type Elias**

Ao invés de listar os vários tipos possíveis de um parâmetro ou variável, você pode criar o seu próprio type.

**Type Assertion**

Redefine o tipo de uma determinada varável.

Interfaces

Dá pra fazer a mesma função que o Type Elias, só que contempla mais coisas.

Classes

Também existe no JavaScript, mas tem algumas propriedades que não tem nele. *public* permite que uma variável seja modificável, enquanto que *private* impede que uma variável seja alterada fora da classe.

Herança

Permite que uma classe herde as características de outra. Igual ao JavaScript.

**React**

**O que é?**

- Lib para a criação de interfaces;

- Utilizada para a construção de SPA (Single Page Applications);

- Podemos chamar de framework, devido a seu ecossistema:

- React JS

- React Native

- Redux

- Webpack

- Etc...

-Tudo fica dentro do JavaScript (elementos visuais, lógica e estilo).

Vantagens

- Organização do código:

- Dividir nosso app/código em blocos específicos;

- O funcionamento de um componente não interfere em outro;

- Se um componente é removido, o resto continua funcionando

- Divisão de responsabilidades:

- Back-end: Regras de negócio

- Front-end: Interface

- Programação declarativa:

- Dizemos ao código o que queremos, ele se encarrega do resto.