Algoritmos

Created	@March 3, 2022 10:28 PM
Created by	
Tags	Introdução
Property	

Algoritmos - **Aula 1**

Algoritmos → operação → sequencia que leva ao cumprimento de uma tarefa - usado na solução de um problema (tem inicio e fim) → sequencia lógica que resolve o problema

Linguagem do computador → artificial, linguagem binária (0 desligado/1 ligado) → inflexíveis (precisa estar correto)

Fluxo base de um programa → Início e fim.

Sinalizações → Informações (comentários no código - comunicar outro programador)
→ Comandos

Fornecendo e Recebendo dados → Input (entrada) e Output (saída) → receber dados, processar e devolver dados

Variáveis → Caixa que recebe informações - precisa ter uma etiqueta (nome único) → guarda somente o último valor atribuído, o anterior é perdido para sempre

Tipos de dados → (tipos de dados (caixa - grande/média/pequena) na memória) → let nome : string = “José” ;

Concatenação → +

Interpolação → ${ }

Constantes → const (tem um valor que não pode ser alterado) - > constante em todos os momentos do código.

Enumeradores →

O **enum**
é um dos tipos do TypeScript que nos permite declarar um conjunto de valores/constantes pré-definidos

Os **enums**
são muito utilizados em cenários onde a mudança dos dados não é constante ou não mudam como: cadastro de sexo, dias da semana, cadastro de redes sociais … etc.

```
→ enum Prioridade {
                    Alta,
                    Normal,
                    Baixa
}
```

TYPESCRIPT - SEMPRE USAR O QUOKKA EXTENSÃO VS CODE (command+shift+p)

Aula 2

console.clear() → limpa o console

Operadores Aritméticos

Aula 5

Operadores Relacionais (>, < , ===...)

Operadores Lógicos (&&, ||, !)

Desvio Condicional (if, if..else, if.. else if..else)

Operador Ternário (? :) → condição **?** executa se a condição for verdadeira **:** executa se a condição for falsa)

Switch

Estudar Expressões Regulares e Orientação a Objetos

Aula 6

Estruturas de Repetições (while, do..while, for (;;))

Interrupção de fluxo (continue, break)

Estruturas aninhadas (repetição dentro de repetição)

Aula 08

Instanciar → criar um espacinho na memória

Funções: (método/ procedimento) →

estrada principal → pausa → outra estrada (subprograma) → encerra → volta para a estrada principal.

```
function nomeDaFunção(parametros: tipos) : tipo de retorno da função {
corpo da função
}
```

parâmetro → informação de entrada → levar a informação para outra estrada (outro código), (será utilizado lá).

Estudar recursividade e POO

Aula 10

Estruturas de dados estáticas:

- **Arrays** → matriz(duas dimensões)/ vetor (uma dimensão) → mesmo tipo

- `let arrayBidimensional : number[][] = [[9,5],[3,8]] → console.log(arrayBidimensional[0][1])`

- **Tuplas** → trabalham com tipos diferentes
  - `let umAluno: [sting, string, number] = [”Arivaldo”, “tuma2”, 10];`
- **Type** →

```
type Nome = string
type Nota = number
let alunos [ Nome , Nota ] = [Alice, 10]
```

- **Programação Orientada a Objetos** → Objeto é um modelo de padrão que criamos e definimos

```
type Aluno = {
nome: string
sobrenome: string
notafinal: number
}
let aluno : Aluno = {
nome: “Zé”
sobrenome: “Ferreira”
notafinal : 1.0
}
```

Acessar: referencia.qualinfo = mudar

```
aluno.nome = “José”
```