**Raoni Martins Raposo. 🡪** Aluno do Fluency sem Limites.

**Exercícios Unidade 3.**

**Variáveis e Conceitos Básicos.**

**Obs: Em alguns exercícios abaixo você está livre para escolher realizar com JavaScript ou Python.**

**Descreva o que chamou sua atenção na sintaxe da linguagem Python?**

A sintaxe da linguagem Python é notável por várias razões:

Legibilidade: Python é frequentemente elogiado por sua sintaxe clara e legível, que é quase como o inglês. Isso torna o código Python fácil de ler e entender.

Indentação: Ao contrário de muitas outras linguagens de programação, Python usa indentação para delimitar blocos de código. Isso força o código a ser bem estruturado e melhora a legibilidade.

Sintaxe concisa: Python permite expressar conceitos complexos em menos linhas de código do que muitas outras linguagens de programação.

Tipagem dinâmica: Python é uma linguagem de tipagem dinâmica, o que significa que o tipo de uma variável é determinado em tempo de execução. Isso oferece uma grande flexibilidade.

Suporte a múltiplos paradigmas de programação: Python suporta programação orientada a objetos, programação imperativa e funcional.

Ausência de caracteres de terminação de linha: Python não usa ponto e vírgula (;) ou outros caracteres para indicar o final de uma linha de código. O final de uma linha é o final de uma instrução, tornando o código mais limpo.

Sendo assim, essas características fazem do Python uma linguagem de programação muito atraente para iniciantes e profissionais experientes, porém, sou mais Java e Javascript.

**O que é controle de fluxo em linguagem de programação?**

OControle de fluxo em linguagem de programação refere-se ao conjunto de instruções que gerenciam a ordem e a condição sob as quais diferentes partes de um código são executadas. Em outras palavras, é como o programa decide o que fazer e quando fazer, dependendo das condições que são atendidas ou não. Essas estruturas permitem que você tome decisões com base em condições específicas: Decisão, Repetição e Seleção.

**Como o conceito anterior é aplicado sobre a linguagem JavaScript?**

No JavaScript, o controle de fluxo é aplicado através de estruturas de decisão, repetição e seleção, assim como em outras linguagens de programação o controle é executado linha por linha.

**Declare 3 tipos diferentes de variáveis em Javascript.**

var nome: “Raposo”;

let idade: 44;

const pi = 3.14;

**Declare 3 tipos diferentes de variáveis em Python.**

nome: “Raposo”;

lista\_de\_numeros: [ 1, 2, 3, 4, 5]

informações\_do\_usuario = {“nome”: Raoni, “idade”: 44, “email”: “raoni.raposo@gmail.com”}

**Explicar o conceito de escopo de variáveis com suas palavras. Mostre exemplos para ilustrar os diferentes conceitos existentes.**

O escopo de uma variável refere-se à parte do programa onde a variável é acessível e pode ser modificada. O escopo determina a visibilidade da variável no código. Existem dois tipos principais de escopo em muitas linguagens de programação: escopo global e escopo local.

Escopo Global: Uma variável declarada fora de qualquer função tem escopo global, o que significa que ela pode ser acessada e modificada em qualquer lugar do seu código.

var variavelGlobal = "Eu sou global";

function exibirMensagem() {

console.log(variavelGlobal); // Acessível aqui

}

console.log(variavelGlobal); // Acessível aqui também

Escopo Local: Uma variável declarada dentro de uma função tem escopo local para essa função, o que significa que ela só pode ser acessada e modificada dentro dessa função.

function minhaFuncao() {

var variavelLocal = "Eu sou local";

console.log(variavelLocal); // Acessível apenas dentro desta função

}

minhaFuncao();

// console.log(variavelLocal); // Erro! variavelLocal não é acessível aqui

**Faça um algoritmo para definir e imprimir um número inteiro.**

\* Em Javascript

// Definindo um número inteiro

let numeroInteiro = 7;

// Imprimindo o número inteiro no console

console.log(numeroInteiro);

**Faça um algoritmo que realize a conversação de um número inteiro para string.**

* Em Javascript

// Definindo um número inteiro

let numeroInteiro = 123;

// Convertendo o número inteiro para string

let numeroString = numeroInteiro.toString();

// Imprimindo a string no console

console.log(numeroString);

**Faça um algoritmo para somar dois números inteiros (a + b).**

* Em Javascript

// Definindo dois números inteiros

let a = 5;

let b = 3;

// Somando os números

let soma = a + b;

// Imprimindo o resultado da soma no console

console.log('A soma de a + b é:', soma);

**Faça um algoritmo para multiplicar dois números inteiros (a \* b).**

* Em Javascript

// Definindo dois números inteiros

let a = 4;

let b = 7;

// Multiplicando os números

let multiplicacao = a \* b;

// Imprimindo o resultado da multiplicação no console

console.log('O resultado de a \* b é:', multiplicacao);

**Faça um algoritmo para verificar se um número inteiro é primo.**

* Em Javascript

function ehPrimo(numero) {

// Verificar se o número é menor que 2

if (numero < 2) return false;

// Verificar se o número é 2

if (numero === 2) return true;

// Verificar se o número é divisível por qualquer número entre 2 e a raiz quadrada do número

let raiz = Math.sqrt(numero);

for (let i = 2; i <= raiz; i++) {

if (numero % i === 0) return false;

}

// Se não encontrou nenhum divisor, então o número é primo

return true;

}

// Testando a função

let numeroTeste = 29;

console.log(`O número ${numeroTeste} é primo?`, ehPrimo(numeroTeste));

**Faça um algoritmo para calcular a área de um retângulo com base e altura conhecidas (fórmula: área = base \* altura).**

* Em Javascript

// Definindo a base e a altura do retângulo

let base = 10; // substitua pelo valor da base

let altura = 5; // substitua pelo valor da altura

// Calculando a área do retângulo

let area = base \* altura;

// Imprimindo a área do retângulo no console

console.log('A área do retângulo é:', area);

**Faça um algoritmo para inserir um elemento em uma lista.**

* Em Javascript

// Definindo uma lista

let lista = [1, 2, 3];

// Elemento a ser inserido

let novoElemento = 0;

// Inserindo o elemento no início da lista

lista.unshift(novoElemento);

// Imprimindo a lista atualizada no console

console.log(lista);

**Faça um algoritmo para remover um elemento de uma lista.**

* Em Javascript

// Definindo uma lista

let lista = [1, 2, 3, 4];

// Índice do elemento a ser removido (por exemplo, índice 2 que corresponde ao número 3)

let indice = 2;

// Removendo o elemento pelo índice

let elementosRemovidos = lista.splice(indice, 1);

// Imprimindo a lista atualizada e os elementos removidos

console.log(lista); // [1, 2, 4]

console.log('Elementos removidos:', elementosRemovidos); // [3]