PROGRAMAÇÃO BÁSICA

MODULO 00

A04 – Primeiro Programa

Comando ‘escreva()’

PROGRAMAÇÃO BÁSICA

MODULO 01

A01 – O que são variáveis?

São espaços reservados na memória para guardar valores, com nome, tipo e valor.

Regras:

-Começa com uma letra

-Só usar letras e números

-Nada de acentos

-Não pode conter espaços

-Não pode ter símbolos (só o underline)

-Não pode ser uma palavra reservada

A02 – Tipos primitivos

Os tipos primitivos podem ser divididos em 3 tipos

-Número (Inteiro e Real) Ex: 130 e 4.5

-Letras (Carácter e Cadeia) Ex: ‘A’ e “Olá”

-Respostas (Lógico) Ex: falso

A03 – Declarando variáveis

-Variável

-Identificador

-Tipos primitivos

-Valores

Tipo 🡪 Identificador = Valor

Ex: inteiro idade = 30

Real nota = 7.5

Cadeia nome = “Zé”

Logico casado = falso

Atribuição (=)

Nome 🡨 “Rogério”

Idade 🡨 30

Sexo 🡨 ‘M’

Aprovado 🡨 falso

A04 – Usando constantes

A uma forma de declarar uma variável que não varia, e uma boa prática é usar a constantes com letras maiúsculas.

Tem que inicializar a constante com um valor.

const 🡪 tipo 🡪 identificador = valor;

A05 – Entrada e saída de dados

Para entradas de dados usar o comando (leia)

Para saídas de dados usar o comando (escreva)

E para números randômicos (sorteia)

A06 – Como usar bibliotecas

Para usar alguns recursos tem que importar bibliotecas.

-Linguagem mais leve

-Economia de memória

-Execução mais otimizada

-Mais extensibilidade

A07 – Operações aritméticas

+ Adição

- Subtração

\* Multiplicação

/ Divisão

% Modulo (resto da divisão)

Ordem de precedência

↓ ()

↓ + -

↓ \* / %

↓ + -

A08 – Exercícios extras

A09 – Operadores de atribuição

PROGRAMAÇÃO BÁSICA

MODULO 02

A01 – Operadores relacionais

São feitos para comparação entre valores

== 🡪 Igual

!= 🡪 Diferente

> 🡪 Maior

< 🡪 Menor

>= 🡪 Maior ou Igual

<= 🡪 Menor ou Igual

Só podem ser usados uma vez dentro de uma expressão simples.

Toda operação vai ser um valor lógico (Verdadeiro ou Falso).

A02 – Operadores lógicos

E 🡪 Conjunção



OU 🡪 Disjunção



NÂO 🡪 Negação



Se tiver operadores relacionais e lógicos, o resultado sempre vai ser um valor lógico.

A ordem de precedência dos operadores lógicos.

“NÃO 🡪 E 🡪 OU”

Precedência global

-Parênteses

-Aritméticos

-Relacionais

-Lógicos

A03 – Condições simples

São operações básicas com operadores lógicos e relacionais

PROGRAMAÇÃO BÁSICA

MODULO 03

-ITERAÇÕES

A01 – REPETIÇÃO COM TESTE LOGICO NO INICIO

enquanto (tem prato) {

Lavarprato ()

}

A02 – REALIZANDO CONTAGENS

A03 – CONTINUAÇÃO

Lendo vários valores com laços

A04 – REALIZANDO CÁLCULOS EM LAÇOS

A05 – TESTE DE MESA NO PORTUGOL STUDIO

A06 – CONTANDO ELEMENTOS POR CATEGORIA

A07 – MAIOR E MENOR VALORES

A08 – REPETIÇÃO COM FLAG

A09 – REPETIÇÃO COM INTERRUPÇÃO

Fazendo um método para que a condição fique verdadeira e só pare com outra condição

A10 – REPETIÇÃO COM TESTE NO FINAL

faca {

Bloco

} enquanto(condição)

A11 – EXERCÍCIOS COM FAÇA ENQUANTO

A12 – REPETIÇÃO COM VARIÁVEL DE CONTROLE

para (c=1;c<=4;c++){

escreva (c)

}

Hexágono representação gráfica

A13 – EXERCICIOS COM ESTRUTURA PARA

A14 – LAÇOS ANINHADOS

Uma coisa dentro da outra.

A15 – VALIDAÇÃO DE DADOS COM LAÇO

Usando as formas de repetição para validar as entradas de dados.

faca {

escreva(“Digite Sexo: “)

leia(sx)

}enquanto(sx!=’M’ e sx!=’F’)

faca {

escreva(“Digite idade: “)

leia(id)

}enquanto (id <=0 ou id >= 130)

faca {

escreva(“Digite Sexo: “)

leia(sx)

}enquanto(nao(sexo==’M’ ou sexo == ‘F’))

faca {

escreva(“Digite idade: “)

leia(id)

}enquanto (nao(idade >= 0 ou idade <= 130))

Cadeia nome

enquanto (verdadeiro){

escreva(“Digite seu nome: “)

leia(nome)

se (não(txt.numero\_caracteres(nome) >= 3)) {

para

} senão {

escreva(“<<ERRO>> O nome deve ter pelo menos 3 letras\n”)

}

}

Inclua biblioteca Texto 🡪 txt

Inclua biblioteca Tipos 🡪 ti

Inteiro idade

Cadeia teclado

enquanto (verdadeiro) {

escreva(“Digite a idade: “)

leia(teclado)

se (ti.cadeia\_e\_inteiro(teclado,10) {

idade = ti.cadeia\_para\_inteiro(teclado,10)

se (idade >=0 e idade <=130){

pare

} senao {

escreva(“<<ERRO>> A idade deve estar entre 0 e 130 anos!\n”)

}

pare

} senao {

escreva(“<<ERRO>> A idade deve ser um numero\n”)

}

{