Aula 1 - Comando básicos

**início** {

//Criando váriaveis(unidade básica de armazenamento).

**caractere**: nome;

**inteiro**: idade;

**real**: altura;

**caractere**: info;

//colocando valor na variável (Usa-se um valor de atribuição "=").

// nome = "Pedro!";

idade = 30;

altura = 1.85;

info = nome + ", " + idade + ", " + altura;

//comandos do algoritimo

**leia**("Digite o nome", nome);

**escreva**(nome);

**escreva**(idade);

**escreva**(altura);

**escreva**(nome + ", " + idade + ", " + altura);

}

Aula 2 - Calculando média das notas

**início** {

**real**: nota1;

**real**: nota2;

**real**: nota3;

**leia**("Nota 1", nota1);

**leia**("Nota 2", nota2);

**leia**("Nota 3", nota3);

**real**: soma;

soma = nota1 + nota2 + nota3;

**real**: media;

media = soma / 3;

**escreva**("O valor da Média é: ", media);

}

Aula 3 - Calculando Juros Compostos

**início** {

//valor futuro (Juros)

**real**: fv;

//valor atual (capital)

**real**: pv;

// taxa de juros;

**real**: i;

// tempo

**real**: n;

**leia**("Valor atual: ", pv);

**leia**("Taxa de juros: ", i);

**leia**("tempo: ", n);

//calculando fórmula do juros composto

//para calcular potenciação utilizar comando "pot"

fv = pv \* pot(1 + i/100, n);

**escreva**("O Juros é de: " + fv);

}

Aula 4 - Convertendo temperatura F° para C°

**início** {

//Convertendo temperatura fahrenheit para Celsius

**real**: tc;

**real**: tf;

**leia**("Qual é a temperatura em fahrenheit?", tf);

tc = (tf - 32)/9 \* 5;

**escreva** ("A temperatura é: ", tc, "C°.");

}

Aula 5 - Calculando Equação do 2° Grau

//Criando calulador de equação de 2° grau.

// ex. de equação --> 2(x2) + 4x - 6 = 0 --> a=2, b=4, c=-6

// --> 3(x2) - 27 = 0 --> a=3, b=0, c=-27

**início** {

**real**: a;

**real**: b;

**real**: c;

**real**: delta;

**real**: x1;

**real**: x2;

**leia**("A", a);

**leia**("B", b);

**leia**("C", c);

delta = b \* b - 4 \* a \* c;

x1 = (-b + rad(delta)) / (2 \* a);

x2 = (-b - rad(delta)) / (2 \* a);

**escreva**("x1 = " + x1);

**escreva**("x2 = " + x2);

}

Aula 6 - Estrutura de Condição

//Cadastro de um hotel onde divide a categoria de idade entre:

// Criança, adolescente e Adulto

**início** {

**inteiro**: idade;

**leia**("Idade: ", idade);

**caractere**: categoria;

**se** (idade < 0) {

categoria = "idade inválida";

} **senão se** (idade <13) {

categoria = "Criança";

} **senão se** (idade <18) {

categoria = "Adolescente";

} **senão** {

categoria = "Adulto";

}

**escreva**(categoria);

}

**Aula** 7 - Encontrando números pares de um intervalo

//Mostre os números pares ou impares em um intervalo de MIN e MAX

// (n % 2 == 0) representa uma formula para encontrar os números pares

// -> "%" dessa formula representa a divisão "/" porém....

// -> pórem não queremos o VALOR da divisão, e sim o RESTO da divisão

// (n % 2 <> 0) subistitua por essa fórmula caso queira encontrar os números pares

**início** {

**inteiro**: min;

**inteiro**: max;

// "n" representa o valor atual, o "n" pecorre cada número entre o Min e o Max

**inteiro**: n;

**leia** ("Valor Mínimo: ", min);

**leia** ("Valor Máximo: ", max);

n = min;

// "enquanto" trabalha com uma estrutura de repetição, até a condição se torna verdadeira

**enquanto**(n <= max){

**se** (n % 2 == 0){

**escreva**(n);

}

n = n + 1;

}

}

Aula 8 - Gerador de Tabuada

// Criando Tabuada

**início** {

**inteiro**: t;

**inteiro**: n;

**leia**("Tabuada do .... ", t);

**para** n **de** 1 **até** 10 **passo** 1 {

**escreva**(t + " x " + n + " = " + (t \* n));

}

// Utilizando o Comando “**enquanto**”

n = 1;

**enquanto**(n <= 10) {

**escreva**(t + " x " + n + " = " + (t \* n));

n = n + 1;

}

}

Aula 9 - Outra forma de calcular a média das notas

**início** {

**real**: nota;

**real**: media;

**real**: soma;

**inteiro**: n;

n = 1;

soma = 0;

**enquanto**(nota >= 0){

**leia**("Nota " + n, nota);

**se** (nota >=0) {

soma = soma + nota;

n = n + 1;

}

}

media = soma / (n-1);

**escreva**("A média é " + media);

}

Aula 10 - Gerando a sequência de Fibonacci

**início** {

// sequência de Fibonacci: 0 1 1 2 3 5 8 13 21.....

**inteiro**: n;

**inteiro**: i;

**inteiro**: j;

**inteiro**: p;

**inteiro**: t;

i = 1;

j = 0;

**escreva** (j);

**escreva** (i);

**para** n **de** 3 **até** 10 **passo** 1 {

p = i + j;

**escreva**(p);

t = i;

i = p;

j = t;

}

}

Aula 10.1 - Sequência de Fibonacci de forma mais simples

**início** {

// sequência de Fibonacci: 0 1 1 2 3 5 8 13 21.....

**inteiro**: n;

**inteiro**: i;

**inteiro**: j;

i = 0;

j = 1;

**para** n **de** 1 **até** 10 **passo** 1 {

**escreva**(i);

i = i + j;

j = i - j;

}

}

Aula 11 - Estudando tabela da verdade

**início** {

// Criando sistema de passaportes e vistos

**lógico:** tem\_passaporte;

**lógico:** tem\_residencia\_fixa;

**lógico:** tem\_antecedentes;

**lógico:** tem\_oferta\_trabalho;

tem\_passaporte = **verdadeiro**;

tem\_residencia\_fixa = **verdadeiro**;

tem\_antecedentes = **falso**;

tem\_oferta\_trabalho = **verdadeiro**;

**se**(tem\_passaporte **e** tem\_residencia\_fixa **e** **não** tem\_antecedentes

**ou** tem\_passaporte **e** tem\_oferta\_trabalho){

**escreva**("visto aprovado");

}

**senão**{

**escreva**("visto negado");

}

}

Aula 12 - Ordenando elementos de um vetor

**início** {

**inteiro**: valores[5];

**inteiro**: x;

valores[0] = 4;

valores[1] = 6;

valores[2] = 1;

valores[3] = 2;

valores[4] = 0;

// usando o comando

**para** x **de** 0 **até** 4 **passo** 1 {

**escreva**(valores[x]);

}

//Também pode usar o comando ENQUANTO

x = 0;

**enquanto**(x < 5){

**escreva**(valores[x]);

x = x + 1;

}

}

Aula 12.1 - Ordenando elementos de um vetor (bubble sort)

**início** {

**inteiro**: valores[5];

valores[0] = 4;

valores[1] = 6;

valores[2] = 1;

valores[3] = 2;

valores[4] = 0;

**inteiro**: i;

**inteiro**: j;

**inteiro**: temp;

i = 0;

**enquanto**(i < 4){

j = i + 1;

**enquanto**(j < 5){

**se** (valores[i] > valores[j]){

temp = valores[j];

valores[j] = valores[i];

valores[i] = temp;

}

j = j + 1;

}

i = i + 1;

}

**escreva**(valores);

}

Aula 13 - Somando elementos em uma matriz de duas dimensões

**início** {

**inteiro**: matriz1[3][4];

//

**inteiro**: matriz2[3][4];

matriz1[0] [0] = 3;

matriz1[0] [1] = 1;

matriz1[0] [2] = 4;

matriz1[0] [3] = 6;

matriz1[1] [0] = 0;

matriz1[1] [1] = 9;

matriz1[1] [2] = 8;

matriz1[1] [3] = 7;

matriz1[2] [0] = 7;

matriz1[2] [1] = 6;

matriz1[2] [2] = 1;

matriz1[2] [3] = 2;

**inteiro**: i;

**inteiro**: j;

//

**inteiro**: multiplicador;

multiplicador = 2;

//varredura

**para** i **de** 0 **até** 2 **passo** 1{

**para** j **de** 0 **até** 3 **passo** 1{

**escreva**(matriz1[i][j]);

matriz2[i][j] = matriz1[i][j] \* multiplicador;

}

}

**para** i **de** 0 **até** 2 **passo** 1{

**para** j **de** 0 **até** 3 **passo** 1{

**escreva**(matriz2[i][j]);

}

}

}

Aula 14 - Criando um novo tipo de Dados (ContaBancaria)

**tipo** conta{

**inteiro**: numero;

**caractere**: titular;

**real**: saldo;

}

**início** {

conta: c;

c.numero = 123;

c.titular = "Pedro Alves";

c.saldo = 1000;

// fazendo um deposito de 200,00

c.saldo = c.saldo + 200;

**escreva**(c);

}

Aula 14.1 - Criando um novo tipo de Dados (ContaBancaria)

**tipo** conta{

**inteiro**: numero;

**caractere**: titular;

**real**: saldo;

}

//UTILIZANDO Contas com ARREY/VETOR

**início** {

conta: contas[2];

contas[0].numero = 123;

contas[0].titular = "Pedro Alves";

contas[0].saldo = 1000;

contas[1].numero = 689;

contas[1].titular = "João Vitor";

contas[1].saldo = 2500;

**escreva**(contas);

**escreva**(contas[0].titular + ": " + contas[0].saldo);

}

Aula 15 - Criando função para encontrar o maior número de um vetor

// As variavieis dentro das funções existem somente dentro das funções.

**início** {

**inteiro**: s[3];

s[0] = 4;

s[1] = 9;

s[2] = 2;

**inteiro**: m;

m = encontrar\_maior(s);

**escreva**("O maior valor é " + m);

}

**função** encontrar\_maior(**inteiro**: v[3]) -> **inteiro**{

inteiro: maior;

//

**inteiro**: n;

maior = v[0];

**para** n **de** 1 **até** 2 **passo** 1{

**se**(v[n] > maior){

maior = v[n];

}

}

**retorne** maior;

}

Aula 16 - Criando Função para calcular taxa de crescimento

**início** {

**real**: v1;

**real**: v2;

v1 = 7;

v2 = 200;

**real**: taxa;

taxa = crescimento(v1, v2);

**escreva**("A taxa é de " + taxa + "%");

}

**função** crescimento(**real**: inicio, **real**: fim) -> **real** {

**real**: taxa;

taxa = (fim - inicio) / inicio;

**retorne** taxa \* 100;

}

Aula 17 - Calculando o fatorial de um número usando uma função recursiva.

// Exercício de chamada Recursiva (Função que chama ela mesma)

// Fatorial: n! n\*(n-1)! Obs: 0! = 1

// Toda função recursiva deve ter um critério de parada.

**início** {

**escreva**(fat(4));

}

**função** fat(**inteiro**:n) -> **inteiro** {

**se** (n==0){

**retorne** 1;

}

**retorne** n \* fat(n-1);

}

Aula 18 - Criando Jogo da Velha

[\\Criando](file:///\\Criando) novos tipos de Dados

**tipo** tabuleiro {

**caractere**: m[3][3];

}

**tipo** jogador {

**caractere**: nome;

**caractere**: simbolo;

}

**tipo** jogada{

**inteiro**:i;

**inteiro**:j;

}

**início** {

tabuleiro: t;

jogador: jogadores[2];

**inteiro**: njog;

njog = 0;

zerar(t);

criar\_jogadores(jogadores);

imprimir(t);

**enquanto**(não tabuleiro\_cheio(t) **e não** jogo\_acabou(t)){

jogada: jogada;

jogada = aguardar\_jogada(jogadores[njog]);

**se** (realizar\_jogada(t, jogada,jogadores[njog])){

imprimir(t);

njog = (njog + 1) % 2;

} **senão** {

**escreva**("Jogada inválida. Tente novamente.");

}

}

**se** (jogo\_acabou(t)){

jogador: ganhador;

ganhador = jogadores[(njog + 1) % 2];

**escreva**("o jogador " + ganhador.nome + " Ganhou!");

} **senão** {

**escreva**("Niguém Venceu!");

}

} //Colchetes fim do “Inicio”

//Próximas linhas são Funções

**função** tabuleiro\_cheio(tabuleiro: t) -> **lógico**{

**inteiro**: i;

**inteiro**: j;

**para** i **de** 0 **até** 2 **passo** 1 {

**para** j **de** 0 **até** 2 **passo** 1 {

**se** (t.m[i][j] == " "){

**retorne** **falso**;

}

}

}

**retorne** **verdadeiro**;

}

**função** jogo\_acabou(tabuleiro: t) -> **lógico**{

// Diagonal 1

**se** (t.m[0][0] == t.m[1][1] **e** t.m[0][0] == t.m[2][2] **e** t.m[0][0] <> " ") {

**retorne** **verdadeiro**;

}

// Diagonal 2

**se** (t.m[0][2] == t.m[1][1] **e** t.m[0][2] == t.m[2][0] **e** t.m[0][2] <> " ") {

**retorne** **verdadeiro**;

}

// Coluna 1

**se** (t.m[0][0] == t.m[1][0] **e** t.m[0][0] == t.m[2][0] **e** t.m[0][0] <> " ") {

**retorne** **verdadeiro**;

}

// Coluna 2

**se** (t.m[0][1] == t.m[1][1] **e** t.m[0][1] == t.m[2][1] **e** t.m[0][1] <> " ") {

**retorne** **verdadeiro**;

}

// Coluna 3

**se** (t.m[0][2] == t.m[1][2] **e** t.m[0][2] == t.m[2][2] **e** t.m[0][2] <> " ") {

**retorne** **verdadeiro**;

}

// Linha 1

**se** (t.m[0][0] == t.m[0][1] **e** t.m[0][0] == t.m[0][2] **e** t.m[0][0] <> " ") {

**retorne** **verdadeiro**;

}

// Linha 2

**se** (t.m[1][0] == t.m[1][1] **e** t.m[1][0] == t.m[1][2] **e** t.m[1][0] <> " ") {

**retorne** **verdadeiro**;

}

// Linha 2

**se** (t.m[2][0] == t.m[2][1] **e** t.m[2][0] == t.m[2][2] **e** t.m[2][0] <> " ") {

**retorne** **verdadeiro**;

}

**retorne** **falso**;

} //Fim da Função

**função** zerar(tabuleiro: t){

**inteiro**: i;

**inteiro**:j;

**para** i **de** 0 **até** 2 **passo** 1{

**para** j **de** 0 **até** 2 **passo** 1{

t.m[i][j] = " ";

}

}

}

**função** imprimir(tabuleiro: t){

**inteiro**: i;

**para** i **de** 0 **até** 2 **passo** 1{

**escreval**(t.m[i][0]);

**escreval**(" | ");

**escreval**(t.m[i][1]);

**escreval**(" | ");

**escreva**(t.m[i][2]);

// Recusando a linha no ultimo parágrafo

**se** (i < 2){

**escreva**("----------");

}

}

**escreva**();

}

**função** criar\_jogadores(jogador: jogadores[2]){

jogadores[0].nome = "A";

jogadores[0].simbolo = "x";

jogadores[1].nome = "B";

jogadores[1].simbolo = "O";

}

**função** aguardar\_jogada(jogador: jogador) -> jogada {

jogada: jogada;

**leia**("jogador " + jogador.nome + " => Linha", jogada.i);

**leia**("jogador " + jogador.nome + " => Coluna", jogada.j);

**retorne** **jogada**;

}

**função** realizar\_jogada(tabuleiro: t, jogada: jogada, jogador: jogador) -> **lógico** {

**se** (t.m[jogada.i][jogada.j] <> " ") {

**retorne** **falso**;

}

t.m[jogada.i][jogada.j] = jogador.simbolo;

**retorne** **verdadeiro**;

}