**Aula 01 – Exercícios Propostos**

Nome do Aluno: Fábio Lucas Romeiro de Castro Data: 10/04/2018

RA do Aluno: 1460281813011

**Capítulo 4 – Exercício 1** – Crie um algoritmo que leia um vetor de 30 números inteiros e gere um segundo vetor cujas posições pares são o dobro do vetor original e as impares o triplo.

Algoritmo valor\_dobrado\_triplicado

tipo vinteiros = vetor [1..30] de inteiros

var vinteiros: lista, listaMultiplicada

inicio

para i<-1 até 30 passo 1 faça  
 escreva "Digite um numero inteiro:"  
 leia lista[i]

se (i mod(2) == 0) então  
 listaMultiplicada[i] <- lista[i] \* 2   
 senão  
 listaMultiplicada[i] <- lista[i] \* 3  
 fim se  
 fim para

escreva lista  
 escreva listaMultiplicada

fim

**Capítulo 4 – Exercício 2** – Desenvolva um algoritmo que permita a leitura de um vetor de 20 números inteiros e gere um segundo vetor com os mesmos dados, só que de maneiras invertida, ou seja, o primeiro elemento ficará na ultima posição, o segundo na penúltima posição, e assim por diante.

Algoritmo inversao\_de\_vetores

tipo vinteiros = vetor [1..30] de inteiros

var   
 vinteiros: lista, listaInvertida  
 i, invertido :inteiro

inicio

inteiro = 30

para i<-1 até 30 passo 1 faça  
 escreva "Digite um numero inteiro:"  
 leia lista[i]

listaInvertida[invertido] <- lista[i]  
 invertido <- invertido – 1  
 fim para

escreva lista  
 escreva listaInvertida

fim

**Capítulo 4 – Exercício 3** – Elabore um algoritmo que leia 50 números inteiros e obtenha qual o tamanho da maior sequencia consecutiva de números em ordem crescente.

Algoritmo sequencia\_consecutiva

tipo vinteiros = vetor [1..50] de inteiros

var   
 vinteiros: lista, sequencia  
 i, contadorSequencia:inteiro

inicio

contadorSequencia <- 1

para i<- 1 até 50 passo 1 faça  
 escreva "Digite um número inteiro"  
 leia lista[i]

se(i <> 1) então

se (lista[i]<lista[i-1]) então  
 sequencia[contadorSequencia] <- lista[i]  
 contadorSequencia <- contadorSequencia + 1  
 fim se  
 fim se  
 fim para

escreva sequencia

fim

**Capítulo 4 – Exercício 4** – Elabore um algoritmo que leia uma serie de 50 notas, e calcule quantas são 10% acima da média e quantas são 10% abaixo.

Algoritmo notas\_acima\_abaixo

tipo vreais = vetor [1..50] de reais

var   
 vreais: notas  
 soma, media, dezPorc, dezPorcAcima, dezPorcAbaixo :Reais  
 acima, abaixo :inteiro

inicio

soma<-0  
 acima<-0  
 abaixo<-0

para i<-1 até 50 passo 1 faça  
 escreva "Digite uma nota: "  
 leia notas[i]

soma <- soma + notas[i]  
 fim para

media<- soma/50  
 dezPorc<-(10\*media)/100  
 dezPorcAcima<- media + dezPorc  
 dezPorcAbaixo<- media - dezPorc

para i<-1 até 50 faça

se(notas[i] == dezPorcAcima) então  
 acima <- acima + 1  
 fim se

se(notas[i] == dezPorcAbaixo) então  
 abaixo <- abaixo + 1  
 fim se  
 fim para

escreva "Quantidade de notas 10% acima da média: ", acima  
 escreva "Quantidade de notas 10% abaixo da média: ", abaixo

fim

**Capítulo 4 – Exercício 5** – Faça um algoritmo que leia o nome, o custo e o preço de 50 produtos. Ao final devera relacionar os produtos que:

1. Tem lucro menor que 10%;
2. Tem lucro entre 10% e 30%;
3. Tem lucro maior que 30%.

Algoritmo lucratividade

tipo vcaracter = vetor [1..50] de caracter

var

vcaracter: resultado  
 custo, preco, lucro :Reais

inicio

para i<-1 até 50 passo 1 faça  
 escreva "Digite seu nome: "  
 leia nome

escreva "Digite o custo: "  
 leia custo

escreva "Digite o preço: "  
 leia preco

lucro <- ((preco - custo) \* 100)/preco

se(lucro < 10) então  
 resultado[i] <- nome, ", tem lucro menor do que 10% - Lucro = R$",   
 (preco - custo)  
 fim se

se(lucro >= 10 e lucro < 30) então  
 resultado[i] <- nome, ", tem lucro maior do que 10% e menor do que 30%   
 - Lucro = R$", (preco - custo)  
 fim se

se(lucro >= 30) então  
 resultado[i] <- nome, ", tem lucro maior do que do que 30% - Lucro = R$",  
 (preco - custo)  
 fim se

escreva resultado[i]  
 fim para

fim

**Capítulo 4 – Exercício 6** – Construa um algoritmo que permita informar dados para 2 vetores inteiros de 20 posições, e apresente a intersecção dos vetores. Lembrando que intersecção são os elementos repetidos em ambos os vetores, mas sem repetição (Cada número pode aparecer uma única vez no resultado):

Algoritmo interseccao

tipo vinteiros = vetor [1..20] de inteiros

var   
 vinteiros: conjuntoA, conjuntoB, intersecção  
 contador :Inteiro  
 existente :Logico

inicio

para i<-1 até 20 passo 1 faça  
 escreva "Digite um elemento do conjunto A"  
 leia conjuntoA[i]  
 fim para

para i<-1 até 20 passo 1 faça  
 escreva "Digite um elemento do conjunto B"  
 leia conjuntoB[i]  
 fim para

contador<-0

para i<-1 até 20 passo 1 faça

para x<-1 até 20 passo 1 faça  
 se (conjuntoA[i] == conjuntoB[x]) então  
 existente<- falso

para y<-1 até 20 passo 1 faça

se(conjuntoA[i] == interseccao[y]) então  
 existente <- verdadeiro  
 fim se  
 fim para

se(existente == falso) então  
 contador <- contador + 1  
 interseccao[i] <- conjuntoA[i]  
 fim se  
 fim se  
 fim para  
 fim para

para y<-1 até contador passo 1 faça  
 escreva interseccao[y]  
 fim para   
fim

**Capítulo 4 – Exercício 7** – Construa um algoritmo que permita informar dados para 2 vetores inteiros de 20 posições e apresente o conjunto união dos vetores. Lembrando que conjunto união são todos os elementos que existem em ambos os vetores, mas sem repetição (Cada número pode aparecer uma única vez no resultado).

Algoritmo uniao

tipo vinteiros = vetor [1..20] de inteiros

var   
 vinteiros: conjuntoA, conjuntoB, união  
 contador :Inteiro  
 acabado :Logico

inicio

para i<-1 até 20 faça  
 escreva "Digite um elemento do conjunto A"  
 leia conjuntoA[i]

escreva "Digite um elemento do conjunto B"  
 leia conjuntoB[i]  
 fim para

contador<-1  
 contador2<-1  
 contador3<-1

enquanto (acabado == falso) faça

se conjuntoA[i] == conjuntoB[i] então  
 uniao[contador2] <-conjuntoA[contador]  
 contador2 <- contador2 + 1  
 fim se

contador <- contador + 1

se(contador > 20) então  
 contador <- 1  
 contador3<- contador3 + 1  
 fim se

se(contador3 > 20) então  
 acabado <- verdadeiro  
 fim se  
 fim enquanto

contador2 <- 1  
 contador3 <- 20  
 acabado <- falso

enquanto (terminado == falso) faça

para i<- 1 até contador3 passo 1 faça

se (uniao[contador2+i] == uniao[contador2]) então  
 uniao[contador2+i] <- 0  
 fim se

se(i == contador3) então  
 contador2 <- contador2 + 1  
 contador3 <- contador3 – 1  
 fim se

se(contador2 > 20) então  
 acabado <- verdadeiro  
 fim se  
 fim para  
 fim enquanto

escreva uniao

fim

**Capítulo 4 – Exercício 8** – Crie um algoritmo que leia a pontuação final de 200 provas de um concurso e os nomes dos respectivos participantes, e apresente um ranking dos colocados que obtiveram mais de 70 pontos.

Algoritmo ranking

tipo vreais = vetor [1..200] de reais

tipo vcaracter = vetor [1..200] de caracter

var   
 vreais: pontuacao  
 vcaracter :participante, ranking

inicio

para i<-1 até 200 passo 1 faça  
 escreva "Digite seu nome: "  
 leia participante[i]

escreva "Digite sua pontuacao: "  
 leia pontuacao[i]

se (pontuacao[i] > 70) então  
 ranking[i]<- "Nome: ", participante[i], " | Pontuacao: ", pontuacao[i], " pontos"   
 escreve ranking[i]  
 fim se  
 fim para

fim

**Capítulo 4 – Exercício 9** – Dado um vetor com dados de 50 alturas, elabore um algoritmo que permita calcular:

1. A média das alturas;
2. O desvio padrão das alturas. Lembrando que desvio padrão é dado por (∑(alturas²)/número de alturas) – media²;
3. A moda das alturas. Lembrando que moda é o valor que tem maior incidência de repetições;
4. A mediana das alturas. Lembrando que a mediana é o elemento central de uma lista ordenada;

Algoritmo alturas

tipo   
 vreais = vetor [1..50] de reais  
 vinteiros = vetor [1..50] de inteiros

var   
 vreais: alturas  
 vinteiro: contador  
 soma, media, desvioPadrao :Reais  
 maisRepitido :Inteiro

inicio

soma<-0  
 maisRepitido<-0

para i<-1 até 50 passo 1 faça  
 contador[i] <- 0  
 fim para

para i<-1 até 50 passo 1 faça  
 escreva "Digite a altura: "  
 leia alturas[i]

para x<-1 até 50 passo 1 faça

se (alturas[i] == alturas[x]) então  
 contador[i] <- contador[i] + 1

se(contador[i]>maisRepitido) então  
 maisRepitido<- alturas[i]  
 fim se  
 fim se  
 fim para

soma<- soma + alturas[i]  
 fim para

media<- soma/50  
 desvioPadrao <-((soma \* soma)/50) - (media \* media)  
 mediana <- (alturas[25] + alturas[26])/2

escreva "Media: ", media, " | Desvio padrão: ", desvioPadrao, " | Moda: ",   
 maisRepitido, " | Mediana: ", mediana

fim

**Observações**:

* Finalizada a resolução de um exercício, efetue uma quebra de folha. Deve constar no documento um enunciado e resolução de exercício por página. Caso o exercício ocupe mais de uma página, efetue a quebra no final da resolução do exercício;
* Fonte: Times New Roman – Tamanho: 12 (sem variar tamanho de fonte);
* Em negrito deve constar somente “Capítulo 1 – Exercício 1”;

*“Nossas atitudes escrevem nosso destino. Nós somos responsáveis pela vida que temos. Culpar os outros pelo que nos acontece é cultivar a ilusão. A aprendizagem é nossa e ninguém poderá fazê-la por nós, assim como nós não poderemos fazer pelos outros. Quanto mais depressa aprendermos isso, menos sofreremos”*

Zibia Gasparetto