



Programador de Sistemas

Lógica de Programação

Estrutura de Repetição – 3ª parte

Prof. Mauricio Wieler Orellana

mauricioow@gmail.com

4. Crie um algoritmo que leia vários números, calcule e mostre:

- A soma dos números digitados
- A quantidade de números digitados
- A média dos números digitados
- O maior número digitado
- O menor número digitado
- A média dos números pares, dentre os números digitados.
- A porcentagem dos números ímpares entre todos os números digitados.

algoritmo "Vários Números"

var

num, soma, qtde, maior, menor, contapar, contaimpar, somapar: inteiro

media, mediapar, porcimpar: real

inicio

soma <- 0

qtde <- 0

contapar <- 0

somapar <- 0

escreval("Relatório de Vários Números")

escreval("Digite um número inteiro positivo: ")

leia(num)

maior <- num

menor <- num

enquanto (num > 0) faça

 se (num%2 = 0) então

 contapar <- contapar + 1

 somapar <- somapar + num

 senão

 contaimpar <- contaimpar + 1

fimse

qtde <- qtde + 1 //Contador

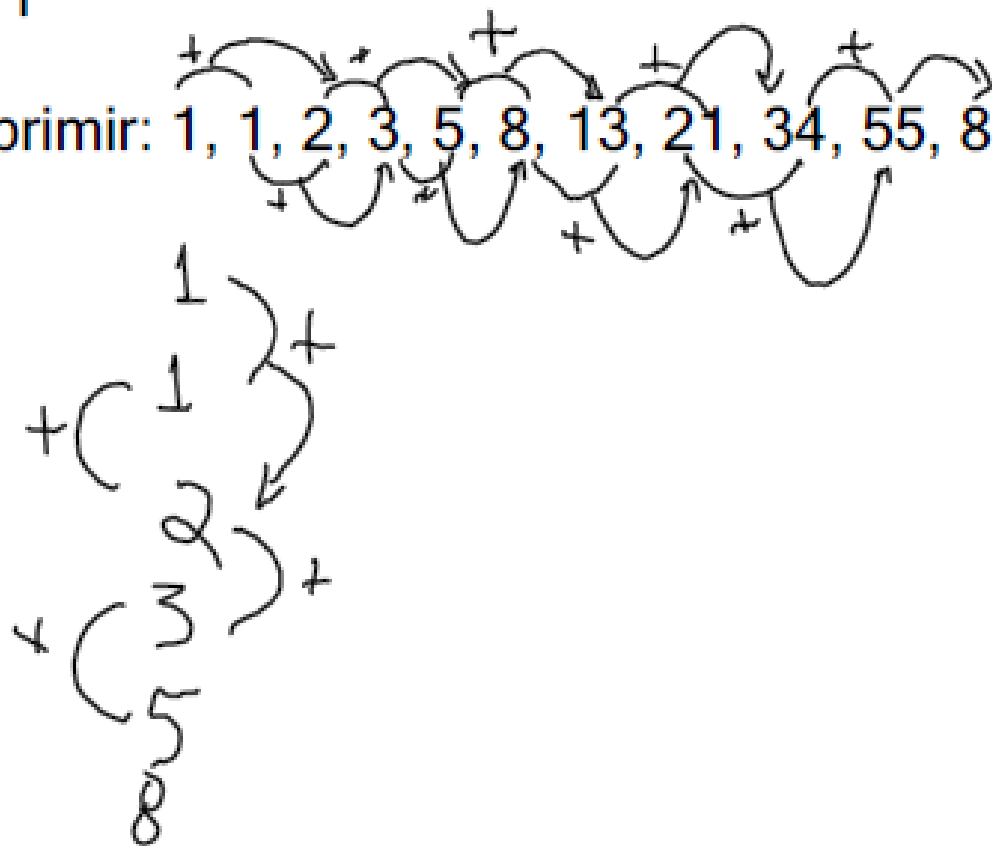
soma <- soma + num //Acumulador



8. Crie um algoritmo que leia um número n e imprima os n primeiros termos da Série de Fibonacci:

Ex: $n = 11$

Deve imprimir: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89



Senac

```
algoritmo "Série de Fibonacci"
var
    i, n, ant, atual, prox: inteiro
inicio
    ant <- 0
    atual <- 1
    escreval("SÉRIE DE FIBONACCI")
    escreval("_____")
    escreval("Digite o número de termos da série: ")
    leia(n)
    escreval("Série de Fibonacci com ",n," termos:")
    para i de 1 ate n faca
        escreval(atual)
        prox <- ant + atual
        ant <- atual
        atual <- prox
    fimpara
fimalgoritmo
```

TESTE DE MESA

n	i	ant	atual	prox
6	1	0	1	$0 + 1 = 1$
	2	1	1	$1 + 1 = 2$
	3	1	2	$1 + 2 = 3$
	4	2	3	$2 + 3 = 5$
	5	3	5	$3 + 5 = 8$
	6	5	8	

TELA: 1

1

2

3

5

8

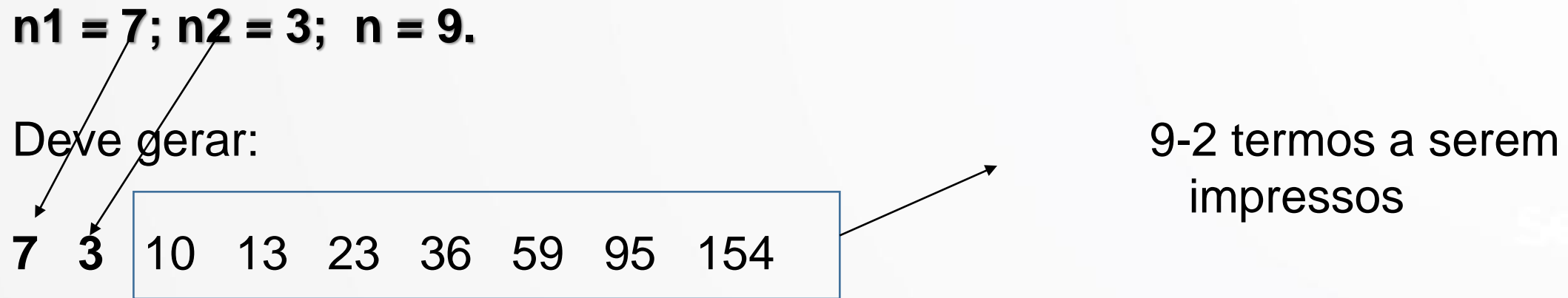
Crie um algoritmo que leia dois números inteiros $n1$ e $n2$ e um número inteiro n (número de termos da Série) e gere os n primeiros termos da Série de Ricci:

Exemplo:

$n1 = 7$; $n2 = 3$; $n = 9$.

Deve gerar:

7 **3** 10 13 23 36 59 95 154



9-2 termos a serem
impressos

algoritmo "Série de Ricci"

var

i, n, n1, n2, prox: inteiro

inicio

escreval("SÉRIE DE RICCI")

escreval("_____")

escreval

escreval("Digite o 1o numero: ")

leia(n1)

escreval("Digite o 2o numero: ")

leia(n2)

escreval("Digite o número de termos da série: ")

leia(n)

escreval("Série de Ricci com ",n," termos:")

escreval(n1)

escreval(n2)

para i de 1 ate n-2 faca

prox <- n1 + n2

n1 <- n2

n2 <- prox

escreval(prox)

fimpara

fimalgoritmo

Crie um algoritmo que leia vários números inteiros positivos e imprima o produto dos números ímpares digitados e a soma dos números pares digitados.

```
algoritmo "Vários números - Pares e ímpares"
var
    num, i, prod, soma: inteiro
inicio
    prod <- 1
    soma <- 0
    escreval("_____Operações com vários números_____")
    escreval("_____")
    escreval
    escreval("Digite um número inteiro positivo ou 0 para terminar ")
    leia(num)
    enquanto (num > 0) faca
        se (num%2 = 0) entao // Se num é par, soma
            soma <- soma + num
        senao
            prod <- prod * num // Se num é ímpar, multiplica
        fimse
        escreval("Digite um número inteiro positivo ou 0 para terminar")
        leia(num)
    fimenquanto
    escreval
    escreval("Soma dos números pares: ", soma)
    escreval("Produto dos números ímpares: ", prod)
fimalgoritmo
```

Crie um algoritmo que leia a idade e o sexo de 6 pessoas e mostre:

- A média das idades
- A porcentagem de maiores de idade
- A porcentagem de menores de idade
- A porcentagem de mulheres, maiores de idade.

SUGESTÃO:

Variável sexo pode ser inteira. Ex: Sexo [0 – M | 1 – F]

```

algoritmo "Idade e Sexo de 6 pessoas"
var
    i, idade, soma, maior, menor, mulh_maior, sexo, cont: inteiro
    media, perc_maior, perc_menor, percmm: real
inicio
    soma <- 0
    maior <- 0
    menor <- 0
    mulh_maior <- 0
    cont <- 0
    escreval("Idade e sexo de várias pessoas")
    escreval("_____")
    escreval("Digite o sexo da pessoa : [0]Masc | [1]Fem.")
    leia(sexo)
    escreval("Digite a idade da pessoa :")
    leia(idade)
    enquanto (idade > 0) faca
        cont <- cont + 1
        soma <- soma + idade
        se (idade >= 18) entao
            maior <- maior + 1
            se (sexo = 1) entao //feminino
                mulh_maior <- mulh_maior + 1
            fimse
        senao
            menor <- menor + 1
        fimse
        escreval("Digite o sexo da pessoa : [0]Masc | [1]Fem.")
        leia(sexo)
        escreval("Digite a idade da pessoa:")
        leia(idade)
    fimenquanto
    media <- soma / cont
    perc_maior <- (maior / cont) * 100
    perc_menor <- (menor / cont) * 100
    percmm <- (mulh_maior / cont) * 100
    escreval("A média das idades vale: ", media:2:2)
    escreval("Maiores de idade: ", perc_maior:2:2, "%")
    escreval("Menores de idade: ", perc_menor:2:2, "%")
    escreval("Mulheres, maiores de idade: ", percmm:2:2, "%")
finalgoritmo

```

Crie um algoritmo que leia dois números inteiros positivos e determine o máximo divisor comum entre eles usando o algoritmo de Euclides:

$$a = 15$$

$$b = 24$$

	1	1	1	2
24	15	9	6	3
9	6	3	0	

$$\text{MDC} = 3$$

$$\begin{array}{r} 24 \overline{) 15} \\ 9 \end{array} \quad \begin{array}{r} 15 \overline{) 9} \\ 6 \end{array}$$

Senac


```
algoritmo "M D C"
var
    a,b,aux,resto: inteiro
inicio
    escreval("M D C  entre dois números")
    escreval("Digite o 1º número:")
    leia(a)
    escreval("Digite o 2º número:")
    leia(b)
    se (a < b) entao
        aux <- a
        a <- b
        b <- aux
    fimse
    resto <- a%b
    enquanto (resto <> 0) faca
        a <- b
        b <- resto
        resto <- a%b
    fimenquanto
    escreval("MDC = ",b)
fimalgoritmo
```

TESTE DE MESA

a	b	aux	resto
15	24	15	$24 \% 15 = 9$
24	15		$15 \% 9 = 6$
15	9		$9 \% 6 = 3$
9	6		$6 \% 3 = 0$
6	3		

TELA DO PROGRAMA:

Máximo Divisor Comum = 3

Crie um algoritmo que leia vários números inteiros e apresente o fatorial de cada número.

O algoritmo se encerra quando se digita um número menor que 1.

Ex: $6 \Rightarrow 6! = 720 = 1 * 2 * 3 * 4 * 5 * 6$

$3 \Rightarrow 3! = 6 = 1 * 2 * 3$

$7 \Rightarrow 7! = 5040$

$8 \Rightarrow 8! = 40320$

$4 \Rightarrow 4! = 24$

- 1 (PAROU!)

fat ← 1
para i de 1 até num faça
 fat ← fat * i
fimpara

```
algoritmo "Fatorial de Vários Números"
var
    num, i: inteiro
    fat: real
inicio
    escreval("Fatorial de vários números inteiros")
    escreval("_____")
    escreval("Digite um número inteiro positivo ou 0 para terminar")
    leia(num)
    enquanto (num > 0) faca
        fat <- 1
        para i de 1 ate num faca //Cálculo do fatorial de cad anum
            fat <- fat * i
        fimpara
        escreval(num, "! = ", fat)
        escreval("Digite um número inteiro positivo ou 0 para
terminar")
        leia(num)
    fimenquanto
fimalgoritmo
```

Crie um algoritmo que leia um número inteiro positivo, determine a sua decomposição em fatores primos calculando também a multiplicidade de cada fator.

Ex: $n=72$

$$\begin{array}{r|l} 72 & \\ 36 & \\ 18 & \\ 9 & \\ 3 & \\ 1 & \end{array} \begin{array}{l} 2 \\ 2 \\ 2 \\ 3 \\ 3 \end{array} \rightarrow 2, \text{ multiplicidade } 3$$

$$\begin{array}{r|l} 9 & \\ 3 & \\ 1 & \end{array} \begin{array}{l} 3 \\ 3 \end{array} \rightarrow 3, \text{ multiplicidade } 2$$

```
algoritmo "Decomposição em fatores primos"
var
    num, fator, mult: inteiro
inicio
    escreval("Decomposição em Fatores Primos")
    escreval("Digite um número inteiro positivo (maior que 1): ")
    leia(num)
    escreval("Decomposição em fatores primos: ")
    fator <- 2
    enquanto (num > 1) faca
        mult <- 0
        enquanto (num % fator = 0) faca
            mult <- mult + 1
            num <- num \ fator
        fimenquanto
        se (mult <> 0) entao
            escreval("Fator ",fator," multiplicidade ",mult)
        fimse
        fator <- fator + 1
    fimenquanto
fim algoritmo
```


Crie um programa que leia 10 elementos inteiros e mostre a quantidade de números primos dentre os números que foram digitados:

Ex:

4

7

12

5 => Existem 3 números primos

14

36

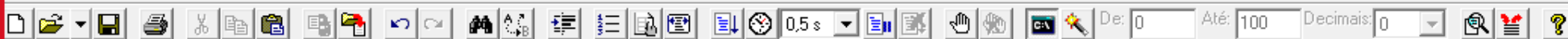
16

180

666

13

```
para divisor de 1 ate num faça  
  cont ← 0  
  se (num % divisor = 0) então  
    cont ← cont + 1  
fimse  
se cont = 2 então  
  escreva (num, "é primo!")
```



```
algoritmo "Números Primos - Coleção de 10 números digitados"

var
    i,num, divisor, conta_divisor, conta_primo: inteiro
inicio
    escreval("Números Primos - Coleção de 10 números digitados")
    conta_primo <- 0 //Contador de números primos, começa com ZERO
    para i de 1 ate 10 faca
        escreval("Digite o ",i,"o. número positivo: ")
        leia(num)
        conta_divisor <- 0 //CONTADOR DE DIVISORES começa com zero
        para divisor de 1 ate num faca //Cálculo de divisores
            se (num % divisor = 0) entao //num é divisível pelo divisor
                conta_divisor <- conta_divisor + 1
            fimse
        fimpara
        se (conta_divisor = 2) entao // o número lido é primo
            conta_primo <- conta_primo + 1 //contador de primos acionado
        fimse
    fimpara
    escreval("Existem ",conta_primo," números primos entre os 10 digitados...")
finalgoritmo
```

Escopo	Nome	Tipo	Valor
GLOBAL	I	I	11

Crie um programa que leia vários elementos inteiros positivos e mostre os divisores de cada um dos números que foram digitados:

{O algoritmo termina quando for digitado 0}

```
algoritmo "Divisores de vários números"
var
    num, divisor: inteiro
inicio
    escreval("Digite um número inteiro ou 0 para acabar: ")
    leia(num)
    escreval("Divisores de ", num, ":")
enquanto (num > 0) faça
    para divisor de 1 ate num faça
        se (num%divisor = 0) então
            escreval(divisor)
        fimse
    fimpara
    escreval("Digite um número inteiro ou 0 para acabar: ")
    leia(num)
    escreval("Divisores de ", num, ":")
fimenquanto
finalgoritmo
```