

- 7.1 Desenvolva um algoritmo que receba um valor inteiro positivo digitado pelo usuário, e apresente os números inteiros maiores que zero, enquanto o somatório deles for menor que o número digitado.

```

1  algoritmo "Lista 07_01 – Somatório Limitado"
2  var
3      limite, atual, soma: inteiro
4
5  inicio
6      // Passo 1: Receber o valor inteiro positivo do usuário
7      escreva("Digite um valor inteiro positivo: ")
8      leia(limite)
9
10     // Inicialização das variáveis de controle
11     atual <- 1
12     soma <- 0
13
14     escreval("Números inteiros cuja soma é menor que", limite, ":")
15
16     // Passo 2: Laço enquanto o somatório for menor que o número digitado
17     enquanto (soma + atual < limite) faca
18         escreval(atual) // Apresenta o número
19         soma <- soma + atual // Acumula no somatório
20         atual <- atual + 1 // Prepara o próximo número inteiro
21     fimenquanto
22
23     // Exibição opcional do resultado da soma final para conferência
24     escreval("-----")
25     escreval("Somatório final obtido: ", soma)
26 fimAlgoritmo

```

```

Digite um valor inteiro positivo:
6
Números inteiros cuja soma é menor que 6:
1
2
-----
Somatório final obtido: 3
>>> Fim da execução do programa !

```

7.2 Elabore um algoritmo que permita que um usuário insira, via teclado, um número inteiro, quantas vezes desejar, sendo que para cada número digitado o algoritmo deve calcular e apresentar o número digitado e o seu quadrado, caso esse número digitado seja ímpar; e o número digitado e seu cubo, caso o número digitado seja par. O algoritmo será encerrado quando o usuário digitar o valor 0, para tanto, essa informação deverá ser fornecida ao usuário, como parte integrante do algoritmo.

```
1  algoritmo "Lista 07_02 - Calcula Quadrados e Cubos"
2  var
3      num, resultado: inteiro
4  inicio
5      // Informar a condição de encerramento ao usuário
6      escreval("--- CALCULADOR DE QUADRADOS E CUBOS ---")
7      escreval("Instruções: Digite 0 a qualquer momento para encerrar.")
8      escreval("-----")
9
10     repita
11         escreva("Digite um número inteiro: ")
12         leia(num)
13         // Verifica se o número é diferente de zero para processar
14         se (num <> 0) entao
15             // Verifica se é par usando o operador % (resto da divisão por 2)
16             se (num % 2 = 0) entao
17                 resultado <- num * num * num // Calcula o Cubo
18                 escreval("O número ", num, " é PAR. Seu cubo é: ", resultado)
19             senao
20                 resultado <- num * num // Calcula o Quadrado
21                 escreval("O número ", num, " é ÍMPAR. Seu quadrado é: ", resultado)
22             fimse
23             escreval("-----")
24         fimse
25     ate (num = 0) // O laço encerra quando o usuário digita 0
26
27     escreval("Programa encerrado pelo usuário.")
28 fimalgoritmo
```

```
--- CALCULADOR DE QUADRADOS E CUBOS ---
Instruções: Digite 0 a qualquer momento para encerrar.
-----
Digite um número inteiro: 2
O número 2 é PAR. Seu cubo é: 8
-----
Digite um número inteiro: 3
O número 3 é ÍMPAR. Seu quadrado é: 9
-----
Digite um número inteiro: 0
Programa encerrado pelo usuário.

>>> Fim da execução do programa !
```

7.3 Construa um algoritmo que calcule e apresente a tabuada de um número escolhido pelo usuário, sendo que o usuário deverá inserir, também, o limite superior da tabuada. O algoritmo não deverá permitir que o usuário insira um valor menor que dois, tanto para o número como para o limite, mostrando a mensagem de entrada inválida e solicitando uma nova entrada, até que o valor inserido seja válido.

```
Qual tabuada você quer ver? (Digite um valor >= 2): 5
Você quer a tabuada do 5 até qual número? (>= 2): 9
=====
RESULTADO: Tabuada do 5 até o 9
=====
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
=====
>>> Fim da execução do programa !
```

```
1  algoritmo "Lista 07_03 – Tabuada Dinâmica"
2  var
3      num, limite, i, resultado: inteiro
4  inicio
5      // Validação do número da tabuada
6      repita
7          escreva("Qual tabuada você quer ver? (Digite um valor >= 2): ")
8          leia(num)
9          se (num < 2) entao
10             escreval("Entrada inválida! O número deve ser maior ou igual a 2.")
11         fimse
12     ate num >= 2
13
14     // Validação do limite superior
15     repita
16         escreva("Você quer a tabuada do ", num, " até qual número? (>= 2): ")
17         leia(limite)
18         se (limite < 2) entao
19             escreval("Entrada inválida! O limite deve ser maior ou igual a 2.")
20         fimse
21     ate limite >= 2
22
23     // Cabeçalho do resultado
24     escreval("=====")
25     escreval("RESULTADO: Tabuada do ", num, " até o ", limite)
26     escreval("=====")
27
28     // Laço para calcular e mostrar a tabuada
29     para i de 1 ate limite faca
30         resultado <- num * i
31         escreval(num, " x ", i, " = ", resultado)
32     fimpara
33 fimalgoritmo
```

```

Qual tabuada você quer ver? (Digite um valor >= 2):
1
Entrada inválida! O número deve ser maior ou igual a 2.
Qual tabuada você quer ver? (Digite um valor >= 2): 4
Você quer a tabuada do 4 até qual número? (>= 2): 1
Entrada inválida! O limite deve ser maior ou igual a 2.
Você quer a tabuada do 4 até qual número? (>= 2): 6

=====
RESULTADO: Tabuada do 4 até o 6
=====
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24

>>> Fim da execução do programa !

```

7.4 Escreva um algoritmo que apresente todos os números naturais divisíveis por 5 e que sejam inferiores a um valor digitado pelo usuário.

```

1  Algoritmo "Lista 07_04 – Números Divisíveis por 5"
2  Var
3      limite, i : Inteiro
4  Inicio
5      // Entrada de dados
6      Escreva("Digite um valor limite: ")
7      Leia(limite)
8
9      Escreval("Números naturais divisíveis por 5 e menores que ", limite, ":")
10
11     // Laço de repetição
12     // Começamos em 0, pois 0 é um número natural divisível por qualquer inteiro positivo
13     Para i de 0 ate (limite - 1) faca           // limite - 1, valores inferiores ao nr digitado
14         Se (i % 5 = 0) entao
15             Escreva(i, " ")
16         FimSe
17     FimPara
18 FimAlgoritmo

```

```

Digite um valor limite:
50
Números naturais divisíveis por 5 e menores que 50:
0 5 10 15 20 25 30 35 40 45
>>> Fim da execução do programa !

```

7.5 Utilizando o laço “enquanto” em pseudocódigo, elabore um algoritmo que imprima todas as tabuadas do 1 ao 10. A apresentação da tabuada deverá está no formato, conforme mostrado abaixo:

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

```
1  Algoritmo "Lista 07_05 – Tabuada Aninhada"
2  Var
3      linha, coluna : Inteiro
4
5  Inicio
6      linha <- 1
7      // Laço externo: controla as linhas da tabela (1 a 10)
8      Enquanto (linha <= 10) faça
9          coluna <- 1
10
11         // Laço interno: controla as colunas e calcula os produtos
12         Enquanto (coluna <= 10) faça
13             // O comando 'Escreva' não pula linha, permitindo a impressão lado a lado
14             Escreva(linha * coluna:4)      // coluna:4 faz com que a coluna ocupe 4 posições
15             coluna <- coluna + 1
16         FimEnquanto
17
18         // Após imprimir todos os produtos de uma linha, pulamos para a próxima
19         Escreval("")
20         linha <- linha + 1
21     FimEnquanto
22
23 FimAlgoritmo
```

```
 1  2  3  4  5  6  7  8  9 10
 2  4  6  8 10 12 14 16 18 20
 3  6  9 12 15 18 21 24 27 30
 4  8 12 16 20 24 28 32 36 40
 5 10 15 20 25 30 35 40 45 50
 6 12 18 24 30 36 42 48 54 60
 7 14 21 28 35 42 49 56 63 70
 8 16 24 32 40 48 56 64 72 80
 9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100

>>> Fim da execução do programa !
```

7.6 Escreva um algoritmo que receba a idade e o estado civil de vários militares e, ao final, imprima a quantidade de militares casados, solteiros, separados e viúvos. O algoritmo finaliza quando for informado o valor zero para idade.

```
1  algoritmo "Lista 07_06 – Estatística de Estado Civil e Militar"
2  var
3      idade, solteiros, casados, separados, viuvos: inteiro
4      estadoCivil: caractere
5  inicio
6      // Inicialização dos contadores
7      solteiros <- 0
8      casados <- 0
9      separados <- 0
10     viuvos <- 0
11
12     escreva("Informe a idade do militar (ou 0 para encerrar): ")
13     leia(idade)
14
15     enquanto (idade <> 0) faça
16         escreval("Informe o estado civil:")
17         escreval("S - Solteiro | C - Casado | P - Separado | V - Viúvo")
18         escreva("Opção: ")
19         leia(estadoCivil)
20
21         // Seleção para incrementar o contador específico
22         escolha estadoCivil
23             caso "S"
24                 solteiros <- solteiros + 1
25             caso "C"
26                 casados <- casados + 1
27             caso "P"
28                 separados <- separados + 1
29             caso "V"
30                 viuvos <- viuvos + 1
31             outrocaso
32                 escreval(">> Opção inválida! Este registro não será contabilizado.")
33         fimescolha
34
35         escreval("-----")
36         escreva("Informe a idade do próximo militar (ou 0 para encerrar): ")
37         leia(idade)
38     fimenquanto
39
40     // Exibição dos resultados finais
41     escreval("")
42     escreval("=====")
43     escreval("    RELATÓRIO FINAL DE ESTADO CIVIL")
44     escreval("=====")
45     escreval("Militares Solteiros: ", solteiros)
46     escreval("Militares Casados: ", casados)
47     escreval("Militares Separados: ", separados)
```

```

48 escreval("Militares Viúvos: ", viuvos)
49 escreval("=====")
50 fimalgoritmo

```

```

Informe a idade do militar (ou 0 para encerrar): 33
Informe o estado civil:
S - Solteiro | C - Casado | P - Separado | V - Viúvo
Opção: S
-----
Informe a idade do próximo militar (ou 0 para encerrar): 43
Informe o estado civil:
S - Solteiro | C - Casado | P - Separado | V - Viúvo
Opção: R
>> Opção inválida! Este registro não será contabilizado.
-----
Informe a idade do próximo militar (ou 0 para encerrar): 55
Informe o estado civil:
S - Solteiro | C - Casado | P - Separado | V - Viúvo
Opção: S
-----
Informe a idade do próximo militar (ou 0 para encerrar): 0

=====
RELATÓRIO FINAL DE ESTADO CIVIL
=====
Militares Solteiros: 2
Militares Casados: 0
Militares Separados: 0
Militares Viúvos: 0
=====

>>> Fim da execução do programa !

```

7.7 Construa um algoritmo que leia números inteiros até que seja inserido um número negativo. Ao final, o algoritmo deverá informar a média dos números digitados, o maior e o menor valor inserido.

```

1 algoritmo "Lista 07_07 – Estatísticas de Números"
2 var
3   num, soma, cont, maior, menor: inteiro
4   media: real
5 inicio
6   soma <- 0
7   cont <- 0
8
9   escreva("Digite um número inteiro (negativo para sair): ")
10  leia(num)
11
12  // Inicializamos o maior e menor com o primeiro número lido
13  se (num >= 0) entao
14    maior <- num
15    menor <- num
16  fimse
17
18  enquanto (num >= 0) faca
19    soma <- soma + num
20    cont <- cont + 1
21  // Verificação de maior e menor

```

```

22     se (num > maior) entao
23         maior <- num
24     fimse
25     se (num < menor) entao
26         menor <- num
27     fimse
28     escreva("Digite o próximo número (negativo para sair): ")
29     leia(num)
30     fimenquanto
31
32     // Verificação para evitar divisão por zero
33     se (cont > 0) entao
34         media <- soma / cont
35         escreval("-----")
36         escreval("Média dos números: ", media)
37         escreval("Maior valor inserido: ", maior)
38         escreval("Menor valor inserido: ", menor)
39         escreval("-----")
40     senao
41         escreval("Nenhum número positivo foi inserido.")
42     fimse
43
44     escreva(">>> Fim da execução.")
45     fimalgoritmo

```

```

Digite um número inteiro (negativo para sair): 10
Digite o próximo número (negativo para sair): 8
Digite o próximo número (negativo para sair): 6
Digite o próximo número (negativo para sair): 7
Digite o próximo número (negativo para sair): -1
-----
Média dos números:  7.75
Maior valor inserido:  10
Menor valor inserido:  6
-----
>>> Fim da execução.
>>> Fim da execução do programa !

```

7.8 Elabore um algoritmo que solicite ao usuário um número inteiro positivo (maior que 0). O programa deve descobrir se esse número é um Quadrado Perfeito, ou seja, se existe um número inteiro N tal que $N \times N$ seja igual ao número digitado.

Observação: a) utilize o comando interrompa para que o laço de repetição não verifique as multiplicações além do valor que extrapole o número informado; b) não use a potenciação fracionária (ex: $n ** 0.5$) para achar a raiz, já que o objetivo é treinar laço.

```
1  algoritmo "Lista 07_08 – Verifica Quadrado Perfeito"
2  var
3    num, n, produto: inteiro
4    eQuadrado: logico
5  inicio
6    eQuadrado <- falso
7    // Entrada de dados com validação simples
8    repita
9      escreva("Digite um número inteiro positivo (maior que 0): ")
10     leia(num)
11     se (num <= 0) entao
12       escreval("Valor inválido! Por favor, insira um número > 0.")
13     fimse
14   ate num > 0
15
16   // Laço para testar os possíveis números N. Começamos de 1 e vamos incrementando
17   para n de 1 ate num faca
18     produto <- n * n
19
20     // Se encontramos o N tal que N * N = num
21     se (produto = num) entao
22       eQuadrado <- verdadeiro
23       interrompa // Interrompe conforme solicitado na observação (a)
24     fimse
25
26     // Se o produto já passou do número, não faz sentido continuar testando
27     se (produto > num) entao
28       interrompa // Otimização para não verificar além do necessário
29     fimse
30   fimpara
31
32   // Saída de resultados
33   se (eQuadrado = verdadeiro) entao
34     escreval("O número ", num, " É um QUADRADO PERFEITO!")
35     escreval("Pois ", n, " x ", n, " = ", num)
36   senao
37     escreval("O número ", num, " NÃO é um quadrado perfeito.")
38   fimse
39 fimalgoritmo
```

```
Digite um número inteiro positivo (maior que 0): 40000
O número 40000 É um QUADRADO PERFEITO!
Pois 200 x 200 = 40000

>>> Fim da execução do programa !
```

7.9 Desenvolva um algoritmo que lê um número inteiro positivo do teclado e informa se ele é par ou ímpar. Caso o número lido seja não positivo (menor do que 1), é apresentada uma mensagem de erro e solicitado o número novamente até que ele seja válido. Ao final da execução, o programa pergunta ao usuário se ele deseja executar o programa novamente. O usuário poderá escolher entre parar ou continuar. Neste último caso, é solicitado novamente a entrada e executa até o usuário optar por parar.

```
1  Algoritmo "Lista 07_09 – Verifica vários números se par ou ímpar"
2  Var
3      num : Inteiro
4      opcao : Caractere
5  Inicio
6      Repita
7          // Passo 1: Validação da entrada (Garante que seja positivo)
8          Escreva("Digite um número inteiro positivo: ")
9          Leia(num)
10
11         Enquanto (num < 1) faça
12             Escreval("ERRO: O número deve ser maior que zero!")
13             Escreva("Digite novamente um número inteiro positivo: ")
14             Leia(num)
15         FimEnquanto
16
17         // Passo 2: Verificação de Paridade
18         Se (num % 2 = 0) entao
19             Escreval("O número ", num, " é PAR.")
20         Senao
21             Escreval("O número ", num, " é ÍMPAR.")
22         FimSe
23
24         // Passo 3: Controle de continuidade
25         Escreval("")
26         Escreva("Deseja executar novamente? (S/N): ")
27         Leia(opcao)
28         Escreval("")
29         Ate (opcao = "N") ou (opcao = "n")
30
31         Escreval("Programa encerrado.")
32     FimAlgoritmo
```

```
Digite um número inteiro positivo: 10
O número 10 é PAR.

Deseja executar novamente? (S/N): S

Digite um número inteiro positivo: 5
O número 5 é ÍMPAR.

Deseja executar novamente? (S/N): S

Digite um número inteiro positivo: 0
ERRO: O número deve ser maior que zero!
Digite novamente um número inteiro positivo: 3
O número 3 é ÍMPAR.

Deseja executar novamente? (S/N): N

Programa encerrado.

>>> Fim da execução do programa !
```

7.10 Desenvolva um algoritmo que realiza o somatório da sequência: 1, 5, 9, 13, 17, 21, ... ,53. Progressão Aritmética (PA), cujo primeiro termo é 1, último termo é 53 e razão 4.

```
1  Algoritmo "Lista 07_10 – Somatório de Sequência de PA"
2  Var
3      termo, soma : Inteiro
4  Inicio
5      // Inicialização das variáveis
6      soma <- 0
7      termo <- 1
8
9      Escreval("Sequência: ")
10
11     // Laço de repetição: vai do primeiro (1) ao último termo (53)
12     Enquanto (termo <= 53) faça
13         Escreva(termo, " ") // Exibe o termo atual para conferência
14         // Acumula o valor do termo atual na variável soma
15         soma <- soma + termo
16         // Incrementa o termo com a razão 4
17         termo <- termo + 4
18     FimEnquanto
19
20     Escreval("")
21     Escreval("O resultado do somatório é: ", soma)
22
23 FimAlgoritmo
```

```
Sequência:
1  5  9  13  17  21  25  29  33  37  41  45  49  53

O resultado do somatório é:  378

>>> Fim da execução do programa !
```

7.11 Desenvolva um algoritmo que receba como entrada, via teclado, um número inteiro não negativo (inclui o zero), digitado pelo usuário, e calcula e apresenta o fatorial desse número. Caso o usuário digite um valor negativo, o algoritmo deverá apresentar uma mensagem de entrada inválida, solicitando que o usuário digite um valor válido (maior ou igual zero). Dado: o fatorial de um número é obtido pela multiplicação desse número por todos os seus antecessores até o 1 (um). Ex: $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$. $0! = 1! = 1$

```
1  Algoritmo "Lista 07_11 – Cálculo de Fatorial"
2  Var
3    num, i, fatorial : Inteiro
4  Inicio
5    // Passo 1: Validação da entrada
6    Escreva("Digite um número inteiro não negativo: ")
7    Leia(num)
8
9    Enquanto (num < 0) faça
10     Escreva("ENTRADA INVÁLIDA! O número deve ser maior ou igual a zero.")
11     Escreva("Digite novamente: ")
12     Leia(num)
13   FimEnquanto
14
15   // Passo 2: Cálculo do Fatorial
16   // Inicializamos com 1, pois é o elemento neutro da multiplicação
17   fatorial <- 1
18
19   // Se o usuário digitar 0 ou 1, o laço não será executado e o resultado será 1
20   Para i de num ate 1 passo -1 faça
21     fatorial <- fatorial * i
22   FimPara
23
24   // Passo 3: Exibição do resultado
25   Escreva("O fatorial de ", num, " (", num, "!)" é:", fatorial)
26   FimAlgoritmo
```

```
Digite um número inteiro não negativo: -1
ENTRADA INVÁLIDA! O número deve ser maior ou igual a zero.
Digite novamente: 5
O fatorial de 5 ( 5!) é: 120

>>> Fim da execução do programa !
```

7.12 Elabore um algoritmo que calcule e imprima o valor de S na série abaixo. O valor de N será digitado pelo usuário. O algoritmo deverá ficar, repetidamente, solicitando que o usuário digite o valor de n, até que o usuário digite um valor inteiro positivo.

$$S = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{N}$$

[A série "S", ao lado, acabou omitida na apostila]

```
1  Algoritmo "Lista 07_12 – Soma de Série Harmônica"
2  Var
3      N, i : Inteiro
4      S : Real
5  Inicio
6      // Passo 1: Validação da entrada (Garante que N seja um inteiro positivo)
7      Escreva("Digite um valor inteiro positivo para N: ")
8      Leia(N)
9
10     Enquanto (N <= 0) faça
11         Escreval("ENTRADA INVÁLIDA! O valor de N deve ser maior que zero.")
12         Escreva("Digite novamente: ")
13         Leia(N)
14     FimEnquanto
15
16     // Passo 2: Cálculo da série
17     S <- 0
18     // O laço percorre de 1 até N, somando os inversos
19     Para i de 1 ate N faça
20         S <- S + (1 / i)
21     FimPara
22
23     // Passo 3: Exibição do resultado
24     Escreval("O valor de S para N =", N, " é:", S)
25
26 FimAlgoritmo
```

```
Digite um valor inteiro positivo para N: 0
ENTRADA INVÁLIDA! O valor de N deve ser maior que zero.
Digite novamente: 20
O valor de S para N = 20 é: 3.59773965714368

>>> Fim da execução do programa !
```

7.13 Elabore um algoritmo que permita que o usuário insira, via teclado, os limites (inferior e superior) de uma faixa de valores e imprima os números primos contidos nesta faixa. Caso o usuário digite, como limite inferior da faixa, um número menor ou igual a 1, o algoritmo deverá imprimir a mensagem de entrada inválida e solicitar que o usuário realize uma nova entrada, até que o valor digitado seja válido. O mesmo deverá ocorrer para o caso de o usuário digitar um limite superior menor do que o digitado como limite inferior da faixa.

```
1  Algoritmo "Lista 07_13 – Primos em uma faixa de valores inteiros"
2  Var
3      limiteInf, limiteSup, i, j, divisores : Inteiro
4  Inicio
5      // 1. Validação do Limite Inferior
6      Escreva("Digite o limite inferior (maior que 1): ")
7      Leia(limiteInf)
8      Enquanto (limiteInf <= 1) faça
9          Escreval(" ERRO: O limite inferior deve ser maior que 1!")
10         Escreva("Digite novamente: ")
11         Leia(limiteInf)
12     FimEnquanto
13
14     // 2. Validação do Limite Superior
15     Escreva("Digite o limite superior (maior ou igual a ", limiteInf, "): ")
16     Leia(limiteSup)
17     Enquanto (limiteSup < limiteInf) faça
18         Escreval(" ERRO: O limite superior não pode ser menor que o inferior!")
19         Escreva("Digite novamente: ")
20         Leia(limiteSup)
21     FimEnquanto
22
23     Escreval("")
24     Escreval("Números primos entre ", limiteInf, " e ", limiteSup, ":")
25
26     // 3. Laço principal: percorre a faixa
27     Para i de limiteInf ate limiteSup faça
28         divisores <- 0
29
30         // 4. Laço interno: testa se o número 'i' é primo
31         Para j de 1 ate i faça
32             Se (i % j = 0) entao
33                 divisores <- divisores + 1
34             FimSe
35         FimPara
36
37         // Se encontrou exatamente 2 divisores, o número é primo
38         Se (divisores = 2) entao
39             Escreva(i, " ")
40         FimSe
41     FimPara
42 FimAlgoritmo
```

```

Digite o limite inferior (maior que 1): 0
  ERRO: O limite inferior deve ser maior que 1!
Digite novamente: 5
Digite o limite superior (maior ou igual a 5): 3
  ERRO: O limite superior não pode ser menor que o inferior!
Digite novamente: 30

Números primos entre 5 e 30:
 5 7 11 13 17 19 23 29
>>> Fim da execução do programa !

```

7.14 Implemente um algoritmo que simula um cardápio e permite realizar pedidos dos produtos. Somente no início, o algoritmo mostra os produtos do cardápio na tela, contendo o código, o nome do item e o seu valor.

O usuário poderá fazer pedidos pelo código do item até que seja inserido o valor 0 (zero). Caso o usuário digite um código inexistente no cardápio (com exceção do valor 0 (zero) para sair), o programa deve ficar solicitando que o usuário digite o código de acordo com o cardápio, ou zero para sair. Para cada produto, a quantidade também deve ser informada.

No final, o algoritmo deverá apresentar na tela a quantidade, o nome e o valor de cada item que o usuário escolheu e o valor total a pagar.

Dados:

Código	Produto	Valor Unitário
1	Refrigerante	R\$ 5,00
2	Cerveja	R\$ 9,00
3	Suco	R\$ 7,00
4	Água	R\$ 4,00

Exemplo de execução do algoritmo:

```

----- CARDÁPIO -----
1 - Refrigerante (R$ 5,00)
2 - Cerveja      (R$ 9,00)
3 - Suco         (R$ 7,00)
4 - Água         (R$ 4,00)
0 - SAIR / FINALIZAR
-----
Digite o código do item: 1
Qual a quantidade desejada? 3
-> Adicionado: Refrigerante
---
Digite o código do item: 8
>> Código inválido! Digite entre 1 e 4 (ou 0 para sair).
Digite o código do item: 3
Qual a quantidade desejada? 2
-> Adicionado: Suco
---
Digite o código do item: 0
=====
RESUMO DO PEDIDO
=====
Refrigerante: 3 un x R$ 5,00 = R$ 15.00
Suco:         2 un x R$ 7,00 = R$ 14.00
-----
TOTAL A PAGAR:   R$ 29.00
=====

```

```

1 algoritmo "Lista 07_14 - Controle de Pedidos"
2 var
3     codigo, qtd: inteiro
4     totalGeral, subRefri, subCerveja, subSuco, subAgua: real
5     qtdRefri, qtdCerveja, qtdSuco, qtdAgua: inteiro
6
7 inicio
8     // Inicialização de acumuladores
9     totalGeral <- 0
10    qtdRefri <- 0
11    qtdCerveja <- 0
12    qtdSuco <- 0
13    qtdAgua <- 0
14
15    // Exibição do Cardápio (Somente no início)
16    escreval("----- CARDÁPIO -----")
17    escreval("1 - Refrigerante (R$ 5,00)")
18    escreval("2 - Cerveja      (R$ 9,00)")
19    escreval("3 - Suco        (R$ 7,00)")
20    escreval("4 - Água         (R$ 4,00)")
21    escreval("0 - SAIR / FINALIZAR")
22    escreval("-----")
23
24    repita
25        escreva("Digite o código do item: ")
26        leia(codigo)
27
28        se (codigo <> 0) entao
29            // Validação do código
30            enquanto (codigo < 0) ou (codigo > 4) faca
31                escreval(">> Código inválido! Digite entre 1 e 4 (ou 0 para sair).")
32                escreva("Digite o código do item: ")
33                leia(codigo)
34                se (codigo = 0) entao
35                    interrompa
36                fimse
37            fimenquanto
38
39            // Se o código for válido e não for zero, pede a quantidade
40            se (codigo > 0) entao
41                escreva("Qual a quantidade desejada? ")
42                leia(qtd)
43
44            escolha codigo
45                caso 1
46                    qtdRefri <- qtdRefri + qtd
47                    escreval("-> Adicionado: Refrigerante")
48                caso 2
49                    qtdCerveja <- qtdCerveja + qtd
50                    escreval("-> Adicionado: Cerveja")
51                caso 3
52                    qtdSuco <- qtdSuco + qtd
53                    escreval("-> Adicionado: Suco")
54                caso 4
55                    qtdAgua <- qtdAgua + qtd
56                    escreval("-> Adicionado: Água")
57            fimescolha

```

```

58     escreval("---")
59     fimse
60     fimse
61     ate codigo = 0
62
63     // Cálculos finais
64     subRefri <- qtdRefri * 5.0
65     subCerveja <- qtdCerveja * 9.0
66     subSuco <- qtdSuco * 7.0
67     subAgua <- qtdAgua * 4.0
68     totalGeral <- subRefri + subCerveja + subSuco + subAgua
69
70     // Resumo do Pedido
71     escreval("=====")
72     escreval("RESUMO DO PEDIDO")
73     escreval("=====")
74
75     se (qtdRefri > 0) entao
76         escreval("Refrigerante: ", qtdRefri, " un x R$ 5,00 = R$ ", subRefri:5:2)
77     fimse
78     se (qtdCerveja > 0) entao
79         escreval("Cerveja:   ", qtdCerveja, " un x R$ 9,00 = R$ ", subCerveja:5:2)
80     fimse
81     se (qtdSuco > 0) entao
82         escreval("Suco:      ", qtdSuco, " un x R$ 7,00 = R$ ", subSuco:5:2)
83     fimse
84     se (qtdAgua > 0) entao
85         escreval("Água:     ", qtdAgua, " un x R$ 4,00 = R$ ", subAgua:5:2)
86     fimse
87
88     escreval("-----")
89     escreval("TOTAL A PAGAR: R$ ", totalGeral:5:2)
90     escreval("=====")
91     fimalgoritmo

```

```

----- CARDÁPIO -----
1 - Refrigerante (R$ 5,00)
2 - Cerveja      (R$ 9,00)
3 - Suco         (R$ 7,00)
4 - Água        (R$ 4,00)
0 - SAIR / FINALIZAR
-----
Digite o código do item: 1
Qual a quantidade desejada? 3
-> Adicionado: Refrigerante
---
Digite o código do item: 2
Qual a quantidade desejada? 5
-> Adicionado: Cerveja
---
Digite o código do item: 0
=====
RESUMO DO PEDIDO
=====
Refrigerante:  3 un x R$ 5,00 = R$ 15.00
Cerveja:       5 un x R$ 9,00 = R$ 45.00
=====
TOTAL A PAGAR:  R$ 60.00
=====
>>> Fim da execução do programa !

```