Capítulo 5 – Desenvolvimento de algoritmos

Introdução

Uma solução de problema que comporte o uso de um computador deve ser bem caracterizada. Existe um tipo diferente para cada uma das etapas de seu desenvolvimento completo.

Fase	Domínio	Requisitos
Conceitual / Modelo	Entidades abstratas Associações lógicas Valores abstratos	Habilidade em decompor o modelo em ações componentes
Algorítmica	Objetos (dados) Operações abstratas Valores construídos	Adequação à linguagem Compreensão da linguagem e métodos computacionais
Implementação / Programa	Estruturas de dados Operações primitivas Valores básicos	Precisão de mapeamento
Física / Processo	Armazenamento Operações reais Valores binários	Eficiência de código

Existem vários métodos que podem ser aplicados durante a fase de desenvolvimento de algoritmos. Muitos desses métodos são conhecidos como *métodos de análise estruturada*, ou seja, aqueles cujas metodologias permitem, a partir de uma definição formal de um problema, chegar a algoritmo pronto capaz de resolvê-lo.

O mais importante para o desenvolvimento é conhecer o problema e os elementos que compõem o seu universo de relações, onde, possivelmente, um caminho para a solução pode ser encontrado.

A maior dificuldade na etapa de elaboração da solução por meio de algoritmos é vencer a *distância conceitual* entre o quê deve ser feito (a ação) e como expressá-la (a descrição). Um algoritmo não se limita ao texto (aspecto estático), mas exprime ações (aspecto dinâmico), coordenadas por um *fluxo de controle*.

Ao desenvolver um algoritmo deve-se preocupar :

- com a ***estrutura de dados***, ou seja, a representação das entidades com as quais se irá trabalhar;
- com a ***estrutura lógica***, ou seja, a seqüência e necessidade dos processos que alterarão as entidades;
- com a ***decomposição lógica***, ou seja, a organização da estrutura lógica em *módulos funcionais* (descrições mais gerais);
- com a ***complexidade lógica*** dos módulos, ou seja, a descrição de cada processo por meio de ações mais simples, até que se consiga a sua expressão por meio da notação adotada.

Um algoritmo deve reunir as seguintes qualidades :

- ser claro, legível e confiável;
- ser auto-explicativo (bem documentado);
- permitir a sua verificação e modificação.

Para tentar atender estes requisitos sugere-se :

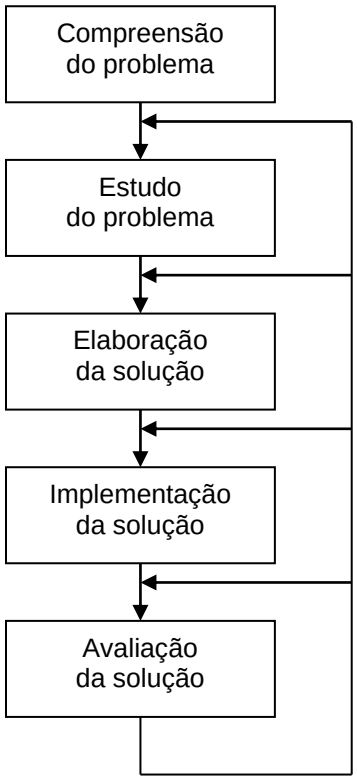
- evitar o crescimento da complexidade;
- colocação de comentários :
- para descrição da função do algoritmo;
- para mostrar como utilizá-lo;
- para explicar o significado e uso de variáveis;
- para descrever estruturas de dados;
- para especificar métodos e referências utilizadas;
- para indicar autor, data e identificação;
- utilização de espaços em branco e parênteses;
- colocação de um comando por linha;
- agrupamento de comandos em blocos.

Qualquer metodologia empregada deverá permitir flexibilidade bastante para que o desenvolvimento possa ser feito de modo a diminuir a complexidade e aumentar as facilidades para se atingir o texto final.

Apresentaremos a seguir, como exemplo, uma destas metodologias.

Desenvolvimento de soluções por algoritmos

- Etapas de desenvolvimento



- Compreensão do problema

  É preciso compreender, de forma bem abrangente, antes de buscar uma solução.

  Qual é a incógnita ?
  Quais são os dados ?
  Qual é a condição ?

  É possível satisfazer a condição ?
  A condição é suficiente para determinar a incógnita ?
  Ou é insuficiente ? Ou redundante ? Ou contraditória ?

  Traçar figuras, quando possível.
  Adotar uma notação adequada, se necessário.
  Separar as diversas partes da condição, se complexa.

- Estudo do problema

  É necessário encontrar a conexão entre os dados e o resultado.

  Já viu o problema antes ? Ou apresentado de forma diferente ?
  Conhece um problema correlato ? Ou que lhe poderia ser útil ?

  É possível que seja obrigado a considerar problemas auxiliares se não puder encontrar uma conexão imediata.

  Conhece outro problema que determine o mesmo resultado, ou semelhante ?

  Se existe tal problema já resolvido é possível usá-lo ?
  É possível utilizar o seu método ? Ou adaptá-lo ?
  Se for introduzido algum elemento auxiliar, pode-se usá-lo ?

  É possível reformular o problema ?

- Elaboração da solução

  É necessário expressar a solução de maneira clara e completa.

  Voltar às definições.

  É possível imaginar um problema correlato mais acessível ?
  É possível imaginar um problema mais genérico ?
  É possível imaginar um problema mais específico ?
  É possível resolver parte do problema ?

  Há algum problema análogo ?

  Todos os dados são necessários ?
  Toda a condição é necessária ?
  Todas as noções essenciais implicadas foram consideradas ?
  Se usar parte da condição pode-se determinar o resultado ?
  Se variar a incógnita, ou dados, ou todos eles, melhora a compreensão do problema ?
  É possível variar a condição ?
  É possível tirar mais alguma coisa de útil dos dados ?
  É possível imaginar outros dados úteis ?

- Implementação da solução

  É necessário executar a solução passo a passo.

  É possível verificar se o passo está correto ?
  É possível demonstrar que ele está correto ?

- Avaliação da solução

  É necessário examinar a solução obtida.

  É possível verificar o resultado ?
  É possível verificar o argumento ?
  É possível chegar ao resultado por um caminho diferente ?
  É possível utilizar o resultado, ou o método, em outro problema ?

Desenvolvimento de soluções por refinamentos sucessivos

A literatura concernente ao assunto sugere algumas formas para o desenvolvimento de soluções (ou programas) que tentam integrar estas questões ao processo. Dentre as mais genéricas pode-se citar o refinamento sucessivo por abordagem *top-down* ou por abordagem *bottom-up*.

O refinamento sucessivo por abordagem *top-down* parte de uma idéia bastante simples: a partir de uma formulação inicial do problema e de um esboço da estratégia de solução, expressam-se as ações necessárias para obtê-la. A forma de expressão pode ser feita através de diagramas ou de uma linguagem (ou notação) apropriada, a linguagem natural é a mais utilizada na maioria das vezes. Em seguida, cada uma das ações pode ser decomposta em outras ações menores, mais primitivas, que permitam uma descrição com um maior nível de especificidade e menor complexidade.

A abordagem *bottom-up*, por outro lado, parte da solução desejada, e vai agregando os passos necessários para se atingir as etapas imediatamente anteriores àquela que leva ao resultado desejado. Sucessivamente, cada passo é reavaliado e transformado em uma nova etapa. O objetivo é fazer com que todo o processo seja obtido de *trás-para-frente*, desde o resultado até se atingir a descrição de dados conhecida.

A primeira abordagem, a princípio, pode ser tomada como de mais fácil execução e a que leva a resultados práticos mais imediatos. A segunda pode ser considerada mais complexa, mais exigente da experiência do profissional. O ideal é aceitar uma certa flexibilidade, permitindo-se escolher uma composição de caminhos que melhor possa ser desenvolvida a cada momento e ir agregando partes novas àquelas já prontas, como se fosse o fechar de um quebra-cabeça. O importante é não se desviar da linha mestra traçada inicialmente, tentar aumentar o grau de conhecimento do problema e da solução em si, fazendo-a evoluir e diminuir a complexidade original.

Qualquer que seja a abordagem, ela deve também permitir uma boa distribuição da carga de trabalho quer dentro de um cronograma, quer por uma equipe de profissionais.

A maior dificuldade na etapa de elaboração da solução é vencer a *distância conceitual* entre o que deve ser feito (a ação) e como expressá-la (a descrição). Um algoritmo não se limita ao texto ou diagrama (aspecto estático), mas exprime ações (aspecto dinâmico), coordenadas por um *fluxo de controle*.

Ao desenvolver um algoritmo é preciso considerar:

- a *estrutura de dados*, ou seja, a representação das entidades com as quais se irá trabalhar;

- a *estrutura lógica*, ou seja, a seqüência e necessidade dos processos que alterarão as entidades;

- a *decomposição lógica*, ou seja, a organização da estrutura lógica em *módulos funcionais* (descrições mais gerais);

- a *complexidade lógica* dos módulos, ou seja, a descrição de cada processo por meio de ações mais simples, até que se consiga a sua expressão por meio da notação adotada.

Para tanto, sugere-se:

- evitar o crescimento da complexidade;
- escolher uma definição apropriada para os dados;
- levantar todas as condições essenciais para o seu uso;
- colocar comentários:
  - para descrever a função de cada bloco ou módulo;
  - para mostrar como utilizá-lo;
  - para explicar o significado e uso de variáveis;
  - para descrever estruturas de dados;
  - para especificar métodos e referências utilizadas;
  - para indicar autor, data e identificação;
- utilizar espaços em branco e parênteses;
- agrupar ações em blocos ou módulos
- destacar cada processo ou ação primitiva.

Qualquer metodologia escolhida deverá permitir um desenvolvimento contínuo que possa favorecer a diminuição da complexidade, e aumentar as facilidades para se obter uma descrição precisa e fiel da estratégia proposta.

Estruturas de dados

Dados são representações abstratas de informações que podem ser tratadas computacionalmente.

Chama-se *tipo* de um dado à classe de valores à qual pertença. Os tipos de dados podem ser:

- simples:
  quando representarem valores indivisíveis

  Exemplos:

  inteiro, real, lógico, caractere

- agrupado ou estruturado:
  quando representarem vários valores homogêneos ou heteregêneos

  Exemplos:

  arranjo - sequência de valores de mesmo tipo
  conjuntos - combinação de valores de mesmo tipo
  registro - combinação de valores não necessariamente do mesmo tipo

Quando um tipo representar um modelo, acompanhado do conjunto de operações definidas sobre ele, será chamado *tipo abstrato de dados* (TAD). O termo "*abstrato*" indica que o tipo não está diretamente associado a uma única implementação.

Exemplo:
Tipo : lista de inteiros

Operações : criar a lista vazia
inserir um item na lista
obter o primeiro da lista

Através de uma *estrutura de dados* conveniente esse modelo *abstrato* poderá ser implementado em uma linguagem de programação.

O estudo de estruturas de dados é importante porque possibilita representar um tipo abstrato de dados para um problema de forma a otimizar o espaço de armazenamento disponível e/ou o tempo de execução.

Exemplo:

Uma escola possui (*n)* alunos, e cada um deles está associado a um número de matrícula com (*k)* dígitos. Dado um número de matrícula é desejado saber o nome do aluno.

Modelo A:

Arranjo com 10^k posições contendo 256 caracteres em cada.
Operação básica: recuperar_nome (matrícula)
Considerações:

- espaço necessário: 256 x 10^k
- tempo de procura: 1 acesso

Modelo B:

Lista de pares do tipo [matrícula, nome]
Operação básica: recuperar_nome (matrícula)
Considerações:

- espaço necessário: 256 x *n*
- tempo de procura: *n*/2 acessos (média)

Qual seria o melhor modelo ?

Qual seria a melhor solução de compromisso ?

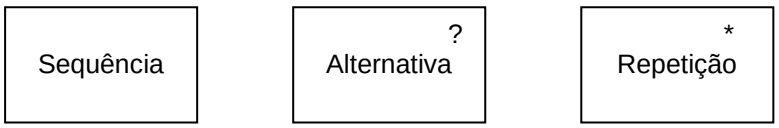
Qual seria a de implementação mais rápida ?

Qual seria a de implementação mais barata ?

Qual seria a de implementação mais segura ?

Essas são questões que precisarão ser avaliadas para ajudar na escolha entre opções.

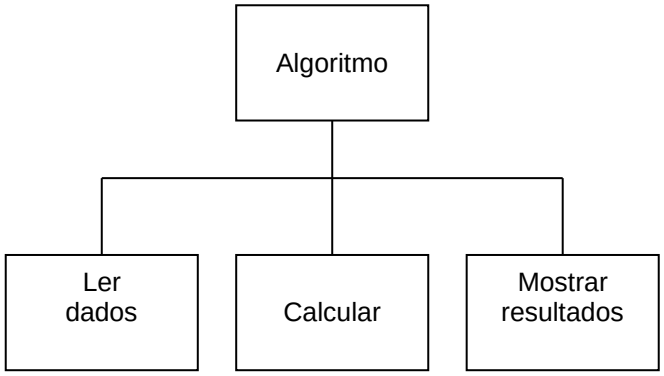
Desenvolvimento de algoritmos por diagramas básicos :



Regras de montagem :

- cada diagrama deve representar uma única ação fundamental;
- os diagramas podem se estruturar em níveis, executando-se uma ação por vez, em ordem, da esquerda para a direita;
- cada diagrama deve ser refinado até representar a ação fundamental por meio de ações primitivas.

Exemplo de montagem de um algoritmo típico :

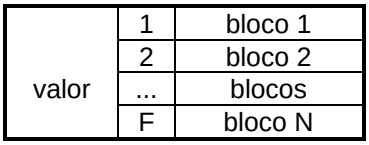


O diagrama acima deve ser entendido como a representação de um algoritmo que faz a leitura de dados, executa algum cálculo sobre eles e mostra os resultados.

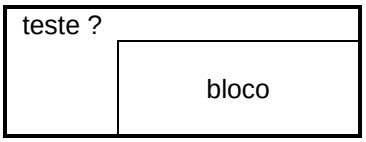
Para esboço de um algoritmo podem ser empregados diagramas semelhantes às estruturas de controle.

Estrutura	Diagrama
Seqüência simples	comando 1 ... comando N
Alternativa simples	teste ? | V | bloco
Alternativa dupla	teste ? | V | bloco 1 F | bloco 2

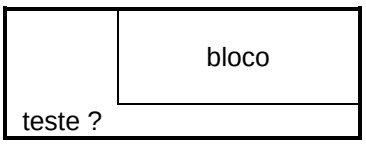
Alternativa múltipla



Repetição com teste no início



Repetição com teste no fim



Cada diagrama pode ser combinada com os demais formando blocos maiores, ou mais complexos, dependendo da necessidade do algoritmo.

Exemplos.

Exemplo 1.

Fazer um algoritmo para:
- ler os valores de dois resistores do teclado;
- calcular e mostrar o valor do resistor equivalente em série.

Análise de dados:

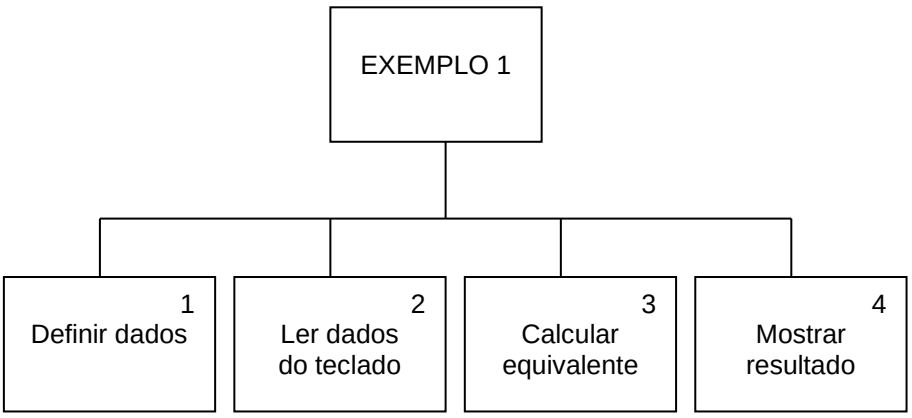
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R1	real		resistor 1
R2	real		resistor 2
R3	real		resistor equivalente

- Fórmulas que relacionam os dados :

R3 = R1 + R2

Diagrama funcional:



- Avaliação da solução :

- Para teste podem ser usados os seguintes valores:

Dados	Resultado
R1 = 10 [ohms] R2 = 5 [ohms]	R3 = 15 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 1	v.1
Ação	Bloco
! definir dados	1
! ler dados do teclado	2
! calcular equivalente em série	3
! mostrar resultado	4

Segunda versão, refinar o primeiro bloco.

Exemplo 1	v.2
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados do teclado	2
! calcular equivalente em série	3
! mostrar resultado	4

Terceira versão, refinar o segundo bloco.

Exemplo 1	v.3
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados do teclado tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor	2
! calcular equivalente em série	3
! mostrar resultado	4

Quarta versão, refinar o segundo bloco.

Exemplo 1	v.4
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados do teclado tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor	2
! calcular equivalente em série R3 ← R1 + R2;	3
! mostrar resultado	4

Quinta versão, refinar o quarto bloco.

Exemplo 1	v.5
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
	1

! ler dados do teclado
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor
	2

! calcular equivalente em série R3 ← R1 + R2;	3
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	4

Programa em SCILAB:

```
// Exemplo 1
// Dados dois resistores, calcular o resistor equivalente em serie.
//
// 1. definir dados
  R1 = 0.0; // primeiro resistor
  R2 = 0.0; // segundo resistor
  R3 = 0.0; // resistor equivalente
// 2. ler dados do teclado
  clc;                          // limpar a area de trabalho
  R1 = input ( "\nR1 " ); // ler primeiro valor
  R2 = input ( "\nR2 " ); // ler segundo valor
// 3. calcular equivalente em serie
  R3 = R1 + R2;
// 4. mostrar resultado
  printf ( "\nR3=R1+R2= %f [ohms]", R3 );
// pausa para terminar
  printf ( "\nPressionar ENTER para terminar.\n" );
  halt;
// fim do programa
```

Programa em C:

```
// Exemplo 1
// Dados dois resistores, calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R1, // primeiro resistor
        R2, // segundo resistor
        R3; // resistor equivalente
// 2. ler dados do teclado
  printf ( "\nR1=" );
  scanf ( "%f", &R1; // ler primeiro valor
  printf ( "\nR2=" );
  scanf ( "%f", &R2; // ler segundo valor
// 3. calcular equivalente em serie
  R3 = R1 + R2;
// 4. mostrar resultado
  printf ( "\nR3=R1+R2=%f %s", R3, " [ohms]" );
// pausa para terminar
  printf ( "\nPressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C++:

```
      // Exemplo 1
      // Dados dois resistores, calcular o resistor equivalente em serie.
      //
      // bibliotecas necessarias
      #include <iostream>
      using namespace std;
      //
      int main (void)
      {
      // 1. definir dados
        double R1, // primeiro resistor
                   R2, // segundo resistor
                   R3; // resistor equivalente
      // 2. ler dados do teclado
        cout << "\nR1=";   cin >> R1; // ler primeiro valor
        cout << "\nR2=";   cin >> R2, // ler segundo valor
      // 3. calcular equivalente em serie
        R3 = R1 + R2;
      // 4. mostrar resultado
        cout << "\nR3=R1+R2=" << R3 << " [ohms]";
      // pausa para terminar
        cout << "\nPressionar ENTER para terminar.";
        cin.get( );
        return EXIT_SUCCESS;
      } // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 1
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_1
{
  public static void Main ( )
  {
  // 1. definir dados
    double R1, // primeiro resistor
               R2, // segundo resistor
               R3; // resistor equivalente
  // 2. ler dados do teclado
    Console.Write ( "\nR1=" );
    R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
    Console.Write ( "\nR2=" );
    R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
  // 3. calcular equivalente em serie
    R3 = R1 + R2;
  // 4. mostrar resultado
    Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
  // pausa para terminar
    Console.Write ( "\nPressionar ENTER para terminar." );
    Console.ReadLine ( );
  } // end Main ( )

} // fim Exemplo_1 class
```

Programa em Java:

```
/**
  * Exemplo 1
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_1
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                          // primeiro resistor
             R2,                          // segundo resistor
             R3;                          // resistor equivalente
   // 2. ler dados do teclado
      System.out.print ( "\nR1 = " );     // ler primeiro valor
      R1 = Integer.parseInt ( System.console( ).readLine( ) );
      System.out.print ( "\nR2 = " );     // ler segundo valor
      R2 = Integer.parseInt ( System.console( ).readLine( ) );
   // 3. calcular equivalente em serie
      R3 = R1 + R2;
   // 4. mostrar resultado
      System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_1 class
```

Programa em Python:

```
# Exemplo 1
# Dados dois resistores, calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
# 2. ler dados do teclado
R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
R2 = float ( input ( "\nR2 = " ) ); # ler segundo valor
# 3. calcular equivalente em serie
R3 = R1 + R2;
# 4. mostrar resultado
print ( "\nR3=R1+R2= ", R3, " [ohms]" );
# pausa para terminar
print ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler o valor de um raio de círculo
   - calcular e mostrar o volume do cilindro de altura igual ao diâmetro.

2. Fazer um algoritmo para :
   - ler três valores reais (lados de um triângulo);
   - calcular e mostrar cada lado e o ângulo oposto a ele.

3. Repetir o exercício anterior para calcular e mostrar :
   - o perímetro e
   - a área do triângulo.

4. Fazer um algoritmo para :
   - calcular e mostrar a força elétrica entre duas cargas;
   - ler o valor das cargas (em Coulombs)
   - ler o raio (em metros)
   - supor :

$k = 9 \times 10^9$ e $F = k \cdot \frac{Q_1 \cdot Q_2}{R^2}$

5. Refazer o exercício anterior para um valor de raio lido em centímetros.

Exemplo 2.

Fazer um algoritmo para:
- ler os valores de dois resistores do teclado;
- calcular e mostrar o valor de resistor equivalente em série, se os dados forem válidos.

Análise de dados:

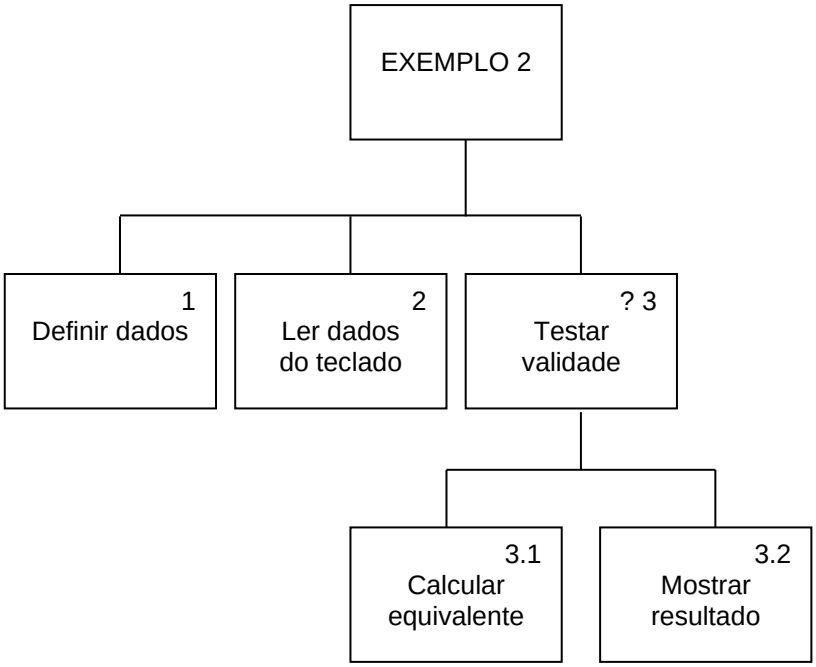
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R1	real		resistor 1 > 0 (válido)
R2	real		resistor 2 > 0 (válido)
R3	real		resistor equivalente

- Fórmulas que relacionam os dados :

R3 = R1 + R2

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
R1 = 10 [ohms] R2 = 5 [ohms]	R3 = 15 [ohms]
R1 = 0 [ohms] R2 = 5 [ohms]	(sem resultado)
R1 = 10 [ohms] R2 = 0 [ohms]	(sem resultado)
R1 = 0 [ohms] R2 = 0 [ohms]	(sem resultado)

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 2	v.1
Ação	Bloco
! definir dados	1
! ler dados do teclado	2
! testar validade dos dados	3
! calcular equivalente em série	3.1
! mostrar resultado	3.2

Segunda versão, refinar o primeiro bloco.

Exemplo 2	v.2
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados do teclado	2
! testar validade dos dados	3
! calcular equivalente em série	3.1
! mostrar resultado	3.2

Terceira versão, refinar o segundo bloco.

Exemplo 2			v.3
Ação			Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
			1

! ler dados do teclado
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor
			2

! testar validade dos dados			3
R1>0 & R2>0?	V	! calcular equivalente em série R3 ← R1 + R2;	3.1
		! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	3.2

Quarta versão, refinando novamente o terceiro bloco.

Exemplo 2	v.4
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
	1

! ler dados do teclado
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor
	2

! testar validade dos dados
  se (R1>0 & R2 > 0)
  ! calcular equivalente em série
  R3 ← R1 + R2;
  ! mostrar resultado
  tela ← ("\nR3=R1+R2=", R3, " [ohms]");
  fim se ! fim se dados válidos
	3
	
	3.1
	
	3.2

Programa em SCILAB:

```
// Exemplo 2
// Dados dois resistores, calcular o resistor equivalente em serie.
//
// 1. definir dados
   R1 = 0.0; // primeiro resistor
   R2 = 0.0; // segundo resistor
   R3 = 0.0; // resistor equivalente
// 2. ler dados do teclado
   clc;                          // limpar area de comandos
   R1 = input ( "\nR1 " ); // ler primeiro valor
   R2 = input ( "\nR2 " ); // ler segundo valor
// 3. testar a validade dos dados
   if ( R1 > 0 & R2 > 0 )
   // 3.1. calcular equivalente em serie
      R3 = R1 + R2;
   // 3.2. mostrar resultado
      printf ( "\nR3=R1+R2= %f [ohms]", R3 );
   end // se dados validos
// pausa para terminar
   printf ( "\nPressionar ENTER para terminar.\n" );
   halt;
// fim do programa
```

Programa em C:

```
// Exemplo 1
// Dados dois resistores, calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
   float R1, // primeiro resistor
         R2, // segundo resistor
         R3; // resistor equivalente
// 2. ler dados do teclado
   printf ( "\nR1=" );
   scanf ( "%f", &R1; // ler primeiro valor
   printf ( "\nR2=" );
   scanf ( "%f", &R2; // ler segundo valor
// 3. testar a validade dos dados
   if ( R1>0 && R2 > 0 )            // se dados validos
   {  // 3.1. calcular equivalente em serie
      R3 = R1 + R2;
      // 3.2. mostrar resultado
      printf ( "\nR3=R1+R2=%f %s", R3, " [ohms]" );
   } // fim se dados validos
// pausa para terminar
   printf ( "\nPressionar ENTER para terminar." );
   getchar ( );
   return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 2
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R1, // primeiro resistor
          R2, // segundo resistor
          R3; // resistor equivalente
// 2. ler dados do teclado
   cout << "\nR1=";   cin >> R1; // ler primeiro valor
   cout << "\nR2=";   cin >> R2, // ler segundo valor
// 3. testar a validade dos dados
   if ( R1>0 && R2 > 0 )          // se dados validos
   {  // 3.1. calcular equivalente em serie
      R3 = R1 + R2;
      // 3.2. mostrar resultado
      cout << "\nR3=R1+R2=" << R3 << " [ohms]";
   } // fim se dados validos
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

```
/*
 * Exemplo 2
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_2
{
   public static void Main ( )
   {
   // 1. definir dados
      double R1, // primeiro resistor
                R2, // segundo  resistor
                R3; // resistor equivalente
   // 2. ler dados do teclado
      Console.Write ( "\nR1=" );
      R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
      Console.Write ( "\nR2=" );
      R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
   // 3. testar a validade dos dados
      if ( R1 > 0 && R2 > 0 )          // se dados validos
      { // 3.1. calcular equivalente em serie
          R3 = R1 + R2;
        // 3.2. mostrar resultado
          Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
      } // fim se dados validos
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_2 class
```

Programa em Java:

```
/**
  * Exemplo 2
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_2
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                              // primeiro resistor
             R2,                              // segundo  resistor
             R3;                              // resistor equivalente
   // 2. ler dados do teclado
      System.out.print ( "\nR1 = " );          // ler primeiro valor
      R1 = Integer.parseInt ( System.console( ).readLine( ) );
      System.out.print ( "\nR2 = " );          // ler segundo valor
      R2 = Integer.parseInt ( System.console( ).readLine( ) );
   // 3. testar a validade dos dados
      if ( R1 > 0 && R2 > 0 )          // se dados validos
      { // 3.1. calcular equivalente em serie
          R3 = R1 + R2;
        // 3.2. mostrar resultado
          System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );
      } // fim se dados validos
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_2 class
```

Programa em Python:

```
# Exemplo 2
# Dados dois resistores, calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
# 2. ler dados do teclado
R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
R2 = float ( input ( "\nR2 = " ) ); # ler segundo valor
# 3. testar a validade dos dados
if ( R1 > 0.0 and R2 > 0.0 ):
   # 3.1. calcular equivalente em serie
      R3 = R1 + R2;
   # 3.2. mostrar resultado
      print ( "\nR3=R1+R2= ", R3, " [ohms]" );
# se dados validos
# pausa para terminar
```

```
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um valor de um raio de círculo válido (maior que zero) e
   - calcular e mostrar o volume do cilindro de altura igual ao diâmetro do círculo.

2. Fazer um algoritmo para :
   - ler três valores reais (lados de um triângulo), todos maiores que zero, e
   - calcular e mostrar cada lado e o ângulo oposto a ele.

3. Fazer um algoritmo para :
   - ler um valor válido da diagonal de um retângulo e,
   - sabendo que um dos lados é a metade do outro,
   - calcular e mostrar o tamanho de cada lado e a área do retângulo.

4. Fazer um algoritmo para :
   - ler um valor válido de um ângulo em graus,
   - convertê-lo para radianos, e
   - calcular e mostrar a área do setor circular de raio unitário.

5. Fazer um algoritmo para :
   - ler o valor das cargas (em Coulombs),
   - ler um valor válido para o raio (em metros),
   - calcular e mostrar a força elétrica entre duas cargas;
   - supor :

$k = 9 \times 10^9$ e $F = k \cdot \frac{Q_1 \cdot Q_2}{R^2}$

Exemplo 3.

Fazer um algoritmo para:
- ler os valores de dois resistores do teclado;
- calcular e mostrar o valor de resistor equivalente em série, se os dados forem válidos;
- caso não sejam fornecidos dados válidos, indicar ocorrência de erro.

Análise de dados:

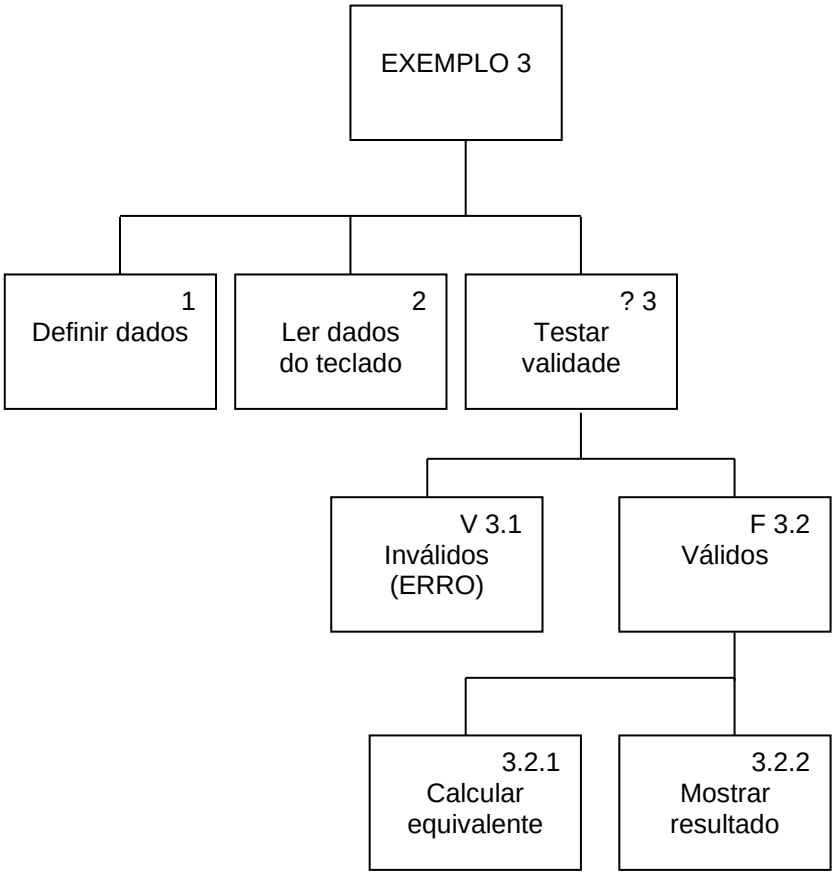
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R1	real		resistor 1 > 0 (válido)
R2	real		resistor 2 > 0 (válido)
R3	real		resistor equivalente

- Fórmulas que relacionam os dados :

R3 = R1 + R2

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
R1 = 10 [ohms] R2 = 5 [ohms]	R3 = 15 [ohms]
R1 = 0 [ohms] R2 = 5 [ohms]	(sem resultado)
R1 = 10 [ohms] R2 = 0 [ohms]	(sem resultado)
R1 = 0 [ohms] R2 = 0 [ohms]	(sem resultado)

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 3	v.1
Ação	Bloco
! definir dados	1
! ler dados do teclado	2
! testar validade dos dados	3
! calcular equivalente em série	3.1
! mostrar resultado	3.2

Segunda versão, refinar o primeiro bloco.

Exemplo 3	v.2
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados do teclado	2
! testar validade dos dados	3
! inválidos, indicar erro	3.1
! válidos	3.2
! calcular equivalente em série	3.2.1
! mostrar resultado	3.2.2

Terceira versão, refinar o segundo bloco.

Exemplo 3	v.3
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
	1

! ler dados do teclado
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor
	2

! testar validade dos dados	3
! inválidos, indicar erro	3.1
! válidos	3.2
! calcular equivalente em série	3.2.1
! mostrar resultado	3.2.2

Quarta versão, refinar o terceiro bloco.

Exemplo 3			v.4
Ação			Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
			1

! ler dados do teclado
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor
			2

! testar validade dos dados			3

R1≤0
  |
R2≤0?
	V	! inválidos
		  tela ← “\nERRO: Dados inválidos”;
			3.1

	F	! calcular equivalente em série R3 ← R1 + R2;	3.2.1
		! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	3.2.2

Quinta versão, refinando novamente o terceiro bloco.

Exemplo 3	v.5
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
	1

! ler dados do teclado
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor
	2

! testar validade dos dados
  se (R1 ≤ 0 | R2 ≤ 0)
  ! inválidos
  tela ← “\nERRO: Dados inválidos”;
  senão ! válidos
  ! calcular equivalente em série
  R3 ← R1 + R2;
  ! mostrar resultado
  tela ← ("\nR3=R1+R2=", R3, " [ohms]");
  fim se ! dados válidos
	3

Programa em SCILAB:

```
// Exemplo 3
// Dados dois resistores, calcular o resistor equivalente em serie.
//
// 1. definir dados
   R1 = 0.0; // primeiro resistor
   R2 = 0.0; // segundo resistor
   R3 = 0.0; // resistor equivalente
// 2. ler dados do teclado
   clc;                        // limpar a area de trabalho
   R1 = input ( “\nR1 “ ); // ler primeiro valor
   R2 = input ( “\nR2 “ ); // ler segundo valor
// 3. testar a validade dos dados
   if ( R1 <= 0 | R2 <= 0 )
   // 3.1. invalidos
      printf ( “\nERRO: Dados invalidos” );
   else   // validos
   // 3.2.1. calcular equivalente em serie
      R3 = R1 + R2;
   // 3.2.2. mostrar resultado
      printf ( “\nR3=R1+R2= %f [ohms]”, R3 );
   end // se dados validos
// pausa para terminar
  printf ( “\nPressionar ENTER para terminar.\n” );
  halt;
// fim do programa
```

Programa em C:

```
// Exemplo 3
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
   float R1, // primeiro resistor
         R2, // segundo resistor
         R3; // resistor equivalente
// 2. ler dados do teclado
   printf ( "\nR1=" );
   scanf ( "%f", &R1; // ler primeiro valor
   printf ( "\nR2=" );
   scanf ( "%f", &R2; // ler segundo valor

// 3. testar a validade dos dados
   if ( R1<=0 || R2<= 0 )
   {
     // 3.1. invalidos
        printf ( “\nERRO: Dados invalidos” );
   }
   else   // validos
   {
     // 3.2.1. calcular equivalente em serie
        R3 = R1 + R2;
     // 3.2.2. mostrar resultado
        printf ( "\nR3=R1+R2=%f %s", R3, " [ohms]" );
   } // fim se dados validos
// pausa para terminar
   cout << "\nPressionar ENTER para terminar.";
   getchar ( );
   return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 3
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R1, // primeiro resistor
          R2, // segundo resistor
          R3; // resistor equivalente
// 2. ler dados do teclado
   cout << "\nR1=";   cin >> R1; // ler primeiro valor
   cout << "\nR2=";   cin >> R2, // ler segundo valor
// 3. testar a validade dos dados
   if ( R1<=0 || R2<= 0 )
   {
     // 3.1. invalidos
      cout << “\nERRO: Dados invalidos”;
   }
   else   // validos
   {
     // 3.2.1. calcular equivalente em serie
      R3 = R1 + R2;
     // 3.2.2. mostrar resultado
      cout << "\nR3=R1+R2=" << R3 << " [ohms]";
   } // fim se dados validos
// pausa para terminar
   cout << "\nPressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

```
/*
 * Exemplo 3
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_3
{
   public static void Main ( )
   {
   // 1. definir dados
      double R1, // primeiro resistor
                R2, // segundo  resistor
                R3; // resistor equivalente
   // 2. ler dados do teclado
      Console.Write ( "\nR1=" );
      R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
      Console.Write ( "\nR2=" );
      R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
   // 3. testar a validade dos dados
      if ( R1<=0 || R2<= 0 )
      {
        // 3.1. invalidos
          Console.WriteLine ( “\nERRO: Dados invalidos” );
      }
      else   // validos
      {
        // 3.2.1. calcular equivalente em serie
          R3 = R1 + R2;
       // 3.2.2. mostrar resultado
         Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
      } // fim se dados validos
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_3 class
```

Programa em Java:

```
/**
  * Exemplo 3
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_3
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                          // primeiro resistor
             R2,                          // segundo  resistor
             R3;                          // resistor equivalente
   // 2. ler dados do teclado
      System.out.print ( "\nR1 = " );     // ler primeiro valor
      R1 = Integer.parseInt ( System.console( ).readLine( ) );
      System.out.print ( "\nR2 = " );     // ler segundo valor
      R2 = Integer.parseInt ( System.console( ).readLine( ) );
   // 3. testar a validade dos dados
      if ( R1 <= 0 || R2 <= 0 )           // se dados validos
      { // 3.1. invalidos
          System.out.println ( "\nERRO: Dados invalidos" );
      }
      else
      { // 3.2.1. calcular equivalente em serie
          R3 = R1 + R2;
        // 3.2. mostrar resultado
          System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );
      } // fim se dados validos
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_3 class
```

Programa em Python:

```
# Exemplo 3
# Dados dois resistores, calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
# 2. ler dados do teclado
R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
R2 = float ( input ( "\nR2 = " ) ); # ler segundo valor
# 3. testar a validade dos dados
if ( R1 <= 0.0 or R2 <= 0.0 ):
  # 3.1. invalidos
    print ( "\nERRO: Dados invalidos" );
else:   # validos
  # 3.2.1. calcular equivalente em serie
    R3 = R1 + R2;
  # 3.2.2. mostrar resultado
    print ( "\nR3=R1+R2= ", R3, " [ohms]" );
# se dados validos
# pausa para terminar
print  ( "\nPressionar ENTER tecla para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um valor de um raio de círculo válido (maior que zero) e
   - calcular e mostrar o volume do cilindro de altura igual ao diâmetro do círculo;
   - se o valor for inválido, informar o erro.

2. Fazer um algoritmo para :
   - ler três valores reais (lados de um triângulo), todos maiores que zero, e
   - calcular e mostrar cada lado e o ângulo oposto a ele;
   - se o valor for inválido, informar o erro.

3. Fazer um algoritmo para :
   - ler um valor válido da diagonal de um retângulo e,
   - sabendo que um dos lados é a metade do outro,
   - calcular e mostrar o tamanho de cada lado e a área do retângulo;
   - se o valor for inválido, usar o valor absoluto da diagonal.

4. Fazer um algoritmo para :
   - ler um valor válido de um ângulo em graus,
   - convertê-lo para radianos, e
   - calcular e mostrar a área do setor circular de raio unitário;
   - se o valor for negativo, converter para o primeiro quadrante.

5. Fazer um algoritmo para :
   - ler o valor das cargas (em Coulombs),
   - ler um valor válido para o raio (em metros),
   - calcular e mostrar a força elétrica entre duas cargas;
   - se o valor do raio for negativo, usar o valor absoluto,
   - se o valor do raio for nulo, informar o erro;
   - supor :

$k = 9 \times 10^9$ e $F = k \cdot \frac{Q_1 \cdot Q_2}{R^2}$

Exemplo 4.

Fazer um algoritmo para:
- ler os valores de dois resistores do teclado e garantir que sejam válidos;
- calcular e mostrar o valor de resistor equivalente em série.

Análise de dados:

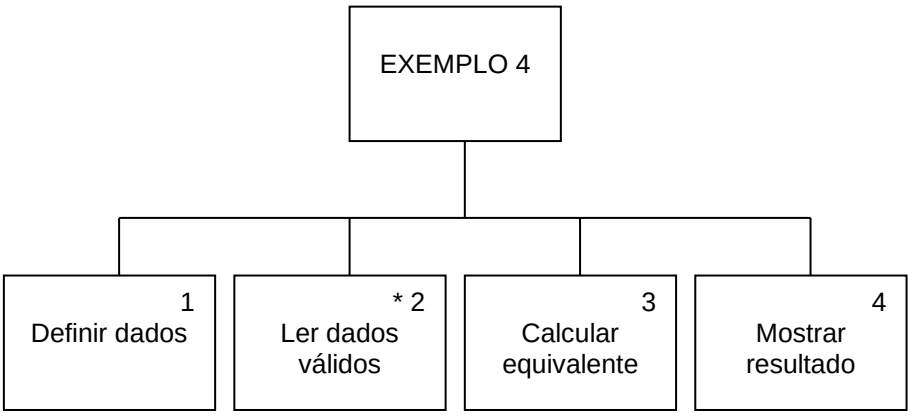
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R1	real		resistor 1 > 0 (válido)
R2	real		resistor 2 > 0 (válido)
R3	real		resistor equivalente

- Fórmulas que relacionam os dados :

R3 = R1 + R2

Diagrama funcional:



- Avaliação da solução :

- Para teste podem ser usados os seguintes valores:

Dados	Resultado
R1 = 10 [ohms] R2 = 5 [ohms]	R3 = 15 [ohms]
R1 = 0 [ohms] R2 = 5 [ohms]	(sem resultado)
R1 = 10 [ohms] R2 = 0 [ohms]	(sem resultado)
R1 = 0 [ohms] R2 = 0 [ohms]	(sem resultado)

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 4	v.1
Ação	Bloco
! definir dados	1
! ler dados válidos do teclado	2
! calcular equivalente em série	3
! mostrar resultado	4

Segunda versão, refinar o primeiro bloco.

Exemplo 4	v.2
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados do teclado	2
! calcular equivalente em série	3
! mostrar resultado	4

Terceira versão, refinar o segundo bloco.

Exemplo 4	v.3
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados válidos do teclado	2
tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor R1 ≤ 0 ?	2.1
tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor R2 ≤ 0 ?	2.2
! calcular equivalente em série	3
! mostrar resultado	4

Quarta versão, refinar o terceiro bloco.

Exemplo 4	v.4
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados válidos do teclado	2
tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor R1 ≤ 0 ?	2.1
tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor R2 ≤ 0 ?	2.2
! calcular equivalente em série R3 ← R1 + R2;	3
! mostrar resultado	4

Quinta versão, refinar o quarto bloco.

Exemplo 4	v.5
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados válidos do teclado	2
tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor R1 ≤ 0 ?	2.1
tela ← "\nR2 = "; R2 ← teclado, ! ler segundo valor R2 ≤ 0?	2.2
! calcular equivalente em série R3 ← R1 + R2;	3
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	4

Sexta versão, refinar novamente o segundo bloco.

Exemplo 4	v.6
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! ler dados válidos do teclado	2
repetir até (R1>0) tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor fim repetir ! enquanto (R1 ≤ 0)	2.1
repetir até (R2>0) tela ← "\nR2 = "; R2 ← teclado; ! ler primeiro valor fim repetir ! enquanto (R2 ≤ 0)	2.2
! calcular equivalente em série R3 ← R1 + R2;	3
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	4

Programa em SCILAB:

```
// Exemplo 4
// Dados dois resistores, calcular o resistor equivalente em serie.
//
// 1. definir dados
   R1 = 0.0; // primeiro resistor
   R2 = 0.0; // segundo resistor
   R3 = 0.0; // resistor equivalente
// 2. ler dados do teclado
// 2.1. ler primeiro valor
   clc;                          // limpar a area de trabalho
   R1 = input ( “\nR1 “ )   ; // ler primeiro valor
   while ( R1 <= 0 )
      R1 = input ( “\nR1 “ ); // ler primeiro valor
   end // ( R1 <= 0 )
// 2.2. ler segundo valor
   R2 = input ( “\nR2 “ );    // ler segundo valor
   while ( R2 <= 0 )
      R2 = input ( “\nR2 “ ); // ler segundo valor
   end // ( R2 <= 0 )
// 3. calcular equivalente em serie
   R3 = R1 + R2;
// 4. mostrar resultado
   printf ( “\nR3=R1+R2= %f [ohms]”, R3 );
// pausa para terminar
   printf ( “\nPressionar ENTER para terminar.\n” );
   halt;
// fim do programa
```

Programa em C:

```
// Exemplo 4
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
   float R1, // primeiro resistor
         R2, // segundo resistor
         R3; // resistor equivalente
// 2. ler dados do teclado
// 2.1. ler primeiro valor
   do
   {
      printf ( "\nR1=" );
      scanf ( "%f", &R1; // ler primeiro valor
   }
   while ( R1<=0 );
// 2.2. ler segundo valor
   do
   {
      printf ( "\nR2=" );
      scanf ( "%f", &R2; // ler segundo valor
   }
   while ( R2<=0 );
// 3. calcular equivalente em serie
   R3 = R1 + R2;
// 4. mostrar resultado
   printf ( "\nR3=R1+R2= %f %s", R3, " [ohms]" );
// pausa para terminar
   printf ( "Pressionar ENTER para terminar." );
   getchar ( );
   return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 4
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R1, // primeiro resistor
          R2, // segundo resistor
          R3; // resistor equivalente
// 2. ler dados do teclado
// 2.1. ler primeiro valor
   do
   {
    cout << "\nR1=";   cin >> R1; // ler primeiro valor
   }
   while ( R1<=0 );
// 2.2. ler segundo valor
   do
   {
    cout << "\nR2=";   cin >> R2, // ler segundo valor
   }
   while ( R2<=0 );
// 3. calcular equivalente em serie
   R3 = R1 + R2;
// 4. mostrar resultado
   cout << "\nR3=R1+R2=" << R3 << " [ohms]";
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 4
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_4
{
   public static void Main ( )
   {
   // 1. definir dados
      double R1, // primeiro  resistor
             R2, // segundo resistor
             R3; // resistor equivalente
   // 2. ler dados do teclado
   // 2.1. ler primeiro valor
      do
      {
         Console.Write ( "\nR1=" );
         R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
      }
      while ( R1 <= 0 );
   // 2.2. ler segundo valor
      do
      {
         Console.Write ( "\nR2=" );
         R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
      }
      while ( R2 <= 0 );
   // 3. calcular equivalente em serie
      R3 = R1 + R2;
   // 4. mostrar resultado
      Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_4 class
```

Programa em Java:

```
/**
  * Exemplo 4
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_4
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                              // primeiro  resistor
             R2,                              // segundo resistor
             R3;                              // resistor equivalente
   // 2. ler dados do teclado
   // 2.1. ler primeiro valor
      do
      {
         System.out.print ( "\nR1 = " );         // ler primeiro valor
         R1 = Double.parseDouble ( System.console( ).readLine( ) );
      }
      while ( R1 <= 0 );
   // 2.2. ler segundo valor
      do
      {
         System.out.print ( "\nR2 = " );        // ler segundo   valor
         R2 = Double.parseDouble ( System.console( ).readLine( ) );
      }
      while ( R2 <= 0 );
   // 3. calcular equivalente em serie
      R3 = R1 + R2;
   // 4. mostrar resultado
      System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_4 class
```

Programa em Python:

```
# Exemplo 4
# Dados dois resistores, calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
# 2. ler dados do teclado
# 2.1. ler primeiro valor
R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
while ( R1 <= 0 ):
   R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
# ( R1 <= 0 )
# 2.2. ler segundo valor
R2 = float ( input ( "\nR2 = " ) ); # ler segundo  valor
while ( R2 <= 0 ):
   R2 = float ( input ( "\nR2 = " ) ); # ler segundo  valor
# ( R2 <= 0 )
# 3. calcular equivalente em serie
R3 = R1 + R2;
# 4. mostrar resultado
print  ( "\nR3=R1+R2= ", R3, " [ohms]" );
# pausa para terminar
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um valor de um raio de círculo, garantido que seja válido (maior que zero) e
   - calcular e mostrar o volume do cilindro de altura igual ao diâmetro do círculo.

2. Fazer um algoritmo para :
   - ler três valores reais (lados de um triângulo), todos maiores que zero,
   - verificar se formam mesmo um triângulo (todo lado deve ser menor que a soma dos outros),
   - calcular e mostrar cada lado e o ângulo oposto a ele.

3. Fazer um algoritmo para :
   - ler um valor de diagonal de um retângulo, garantindo que esteja no intervalo [1,100] e
   - sabendo que um dos lados é a metade do outro,
   - calcular e mostrar o tamanho de cada lado e a área do retângulo.

4. Fazer um algoritmo para :
   - ler um valor válido de um ângulo em graus, e se não for,
   - convertê-lo para o equivalente em radianos no primeiro quadrante, e
   - calcular e mostrar a área do setor circular de raio unitário.

5. Fazer um algoritmo para :
   - ler o valor das cargas (em Coulombs),
   - ler um valor válido (maior que zero) para o raio (em metros),
   - calcular e mostrar a força elétrica entre duas cargas;
   - supor :

$k = 9 \times 10^9$ e $F = k \cdot \frac{Q_1 \cdot Q_2}{R^2}$

Exemplo 5.

Fazer um algoritmo para:
- repetir as ações abaixo 5 vezes:
  - ler os valores de dois resistores do teclado e
  - garantir que sejam válidos;
  - calcular e mostrar o valor de resistor equivalente em série.

Análise de dados:

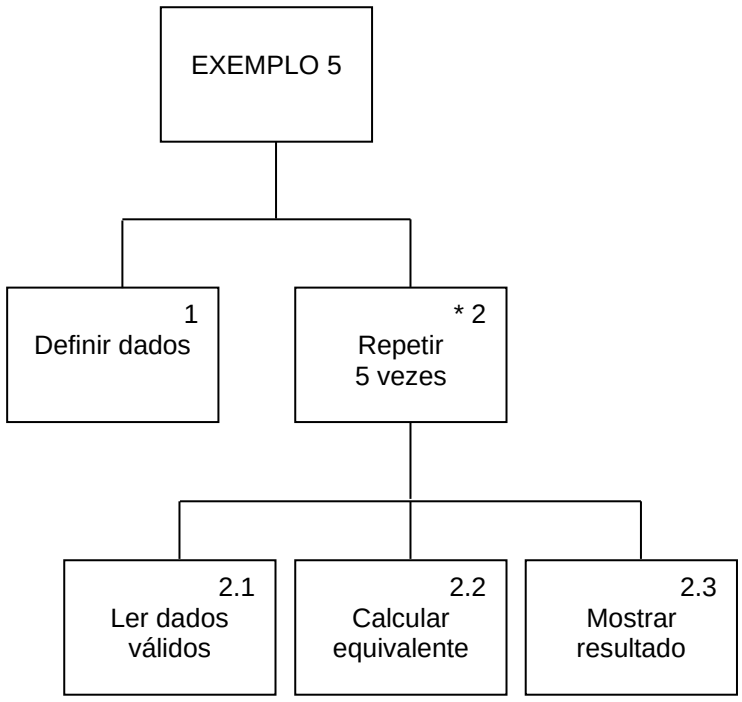
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R1	real		resistor 1 > 0 (válido)
R2	real		resistor 2 > 0 (válido)
R3	real		resistor equivalente

- Fórmulas que relacionam os dados :

R3 = R1 + R2

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
R1 = 10 [ohms] R2 = 5 [ohms]	R3 = 15 [ohms]
R1 = 10 [ohms] R2 = 2 [ohms]	R3 = 12 [ohms]
R1 = 10 [ohms] R2 = 1 [ohms]	R3 = 11 [ohms]
R1 = 5 [ohms] R2 = 2 [ohms]	R3 = 7 [ohms]
R1 = 2 [ohms] R2 = 1 [ohms]	R3 = 3 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 5	v.1
Ação	Bloco
! definir dados	1
! repetir 5 vezes	2
! ler dados válidos do teclado	2.1
! calcular equivalente em série	2.2
! mostrar resultado	2.3

Segunda versão, refinar o primeiro bloco.

Exemplo 5	v.2
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! repetir 5 vezes	2
! ler dados válidos do teclado	2.1
! calcular equivalente em série	2.2
! mostrar resultado	2.3

Terceira versão, refinar o segundo bloco.

Exemplo 5	v.3
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
  inteiro X; ! contador do número de vezes
	1

! repetir 5 vezes	2
X = 1:5:1 ! (de 1 até 5 de 1 em 1)	
! ler dados válidos do teclado	2.1
! calcular equivalente em série	2.2
! mostrar resultado	2.3

Quarta versão, refinar o segundo bloco.

Exemplo 5	v.4
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
  inteiro X; ! contador do número de vezes
	1

! repetir 5 vezes	2
X = 1:5:1 ! (de 1 até 5 de 1 em 1)	
! ler dados válidos do teclado	2.1
tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor	
R1 ≤ 0 ?	
tela ← "\nR2 = "; R2 ← teclado; ! ler segundo valor	
R2 ≤ 0 ?	
! calcular equivalente em série R3 ← R1 + R2;	2.2
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	2.3

Quinta versão, refinar novamente o segundo bloco.

Exemplo 5	v.5
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
  inteiro X; ! contador do número de vezes
	1

! repetir 5 vezes	
X = 1:5:1 ! (de 1 até 5 de 1 em 1)	
! ler dados válidos do teclado repetir até (R1 > 0) tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor fim repetir ! enquanto (R1 ≤ 0) repetir até (R2 > 0) tela ← "\nR2 = "; R2 ← teclado; ! ler primeiro valor fim repetir ! enquanto (R2 ≤ 0)	2.1
! calcular equivalente em série R3 ← R1 + R2;	2.2
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	2.3

Sexta versão, refinar novamente o segundo bloco.

Exemplo 5	v.5
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
  inteiro X; ! contador do número de vezes
	1

! repetir 5 vezes	
repetir para (X = 1:5:1) ! (de 1 até 5 de 1 em 1) ! ler dados válidos do teclado repetir até (R1 > 0) tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor fim repetir ! enquanto (R1 ≤ 0) repetir até (R2 > 0) tela ← "\nR2 = "; R2 ← teclado; ! ler primeiro valor fim repetir ! enquanto (R2 ≤ 0) ! calcular equivalente em série R3 ← R1 + R2; ! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]"); fim repetir ! para X = 1:5:1	2.1 2.2 2.3

Programa em SCILAB:

```
// Exemplo 5a
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// 1. definir dados
   R1 = 0.0; // primeiro resistor
   R2 = 0.0; // segundo resistor
   R3 = 0.0; // resistor equivalente
   X  = 0;   // contador do numero de vezes
// 2. repetir 5 vezes (primeira forma)
   clc;          // limpar a area de trabalho
   for X = 1:1:5  // repetir 5 vezes
   // 2.1.1 ler primeiro valor
      R1 = input ( "\nR1 " );    // ler primeiro valor
      while ( R1 <= 0 )
         R1 = input ( "\nR1 " ); // ler primeiro valor
      end // ( R1 <= 0 )
   // 2.2. ler segundo valor
      R2 = input ( "\nR2 " );    // ler segundo valor
      while ( R2 <= 0 )
         R2 = input ( "\nR2 " ); // ler segundo valor
      end // ( R2 <= 0 )
   // 3. calcular equivalente em serie
      R3 = R1 + R2;
   // 4. mostrar resultado
      printf ( "\nR3=R1+R2= %f [ohms]", R3 );
   end // repetir para X = 1:5:1
// pausa para terminar
   printf ( "\nPressionar ENTER para terminar.\n" );
   halt;
// fim do programa
```

Programa em SCILAB:

```
// Exemplo 5b
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// 1. definir dados
   R1 = 0.0; // primeiro resistor
   R2 = 0.0; // segundo resistor
   R3 = 0.0; // resistor equivalente
   X   = 0;    // contador do numero de vezes
// 2. repetir 5 vezes (segunda forma)
   clc;          // limpar a area de trabalho
   X = 1; // valor inicial
   while ( X <= 5 )
   // 2.1.1 ler primeiro valor
      R1 = input ( “\nR1 “ );   // ler primeiro valor
      while ( R1 <= 0 )
        R1 = input ( “\nR1 “ ); // ler primeiro valor
      end // ( R1 <= 0 )
   // 2.2. ler segundo valor
      R2 = input ( “\nR2 “ );   // ler segundo valor
      while ( R2 <= 0 )
        R2 = input ( “\nR2 “ ); // ler segundo valor
      end // ( R2 <= 0 )
   // 3. calcular equivalente em serie
      R3 = R1 + R2;
   // 4. mostrar resultado
      printf ( “\nR3=R1+R2= %f [ohms]”, R3 );
      X = X + 1; // proximo valor
   end // repetir para X = 1:5:1
// pausa para terminar
   printf ( “\nPressionar ENTER para terminar.\n” );
   halt;
// fim do programa
```

Programa em C:

```
// Exemplo 5a
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R1, // primeiro resistor
        R2, // segundo resistor
        R3; // resistor equivalente
  int    X;   // contador do numero de vezes
// 2. repetir 5 vezes (primeira forma)
  for ( X = 1; X<=5; X = X+1 )
  {
   // 2.1.1 ler primeiro valor
   do
   {
     printf ( "\nR1=" );
     scanf ( "%f", &R1 ); // ler primeiro valor
   }
   while ( R1 <= 0 );
   // 2.1.2. ler segundo valor
   do
   {
     printf ( "\nR2=" );
     scanf ( "%f", &R2 ); // ler primeiro valor
   }
   while ( R2 <= 0 );
   // 2.2. calcular equivalente em serie
   R3 = R1 + R2;
   // 2.3. mostrar resultado
   printf ( "\nR3=R1+R2=%f %s", R3, " [ohms]" );
  } // fim repetir para X = 1:5:1
// pausa para terminar
  printf ( "Pressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C:

```
// Exemplo 5b
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R1, // primeiro resistor
        R2, // segundo resistor
        R3; // resistor equivalente
  int   X;  // contador do numero de vezes
// 2. repetir 5 vezes (segunda forma)
  X = 1;    // valor inicial
  while ( X <= 5 )
  {
   // 2.1.1 ler primeiro valor
   do
   {
     printf ( "\nR1=" );
     scanf ( "%f", &R1 ); // ler primeiro valor
   }
   while ( R1 <= 0 );
   // 2.1.2. ler segundo valor
   do
   {
     printf ( "\nR2=" );
     scanf ( "%f", &R2 ); // ler primeiro valor
   }
   while ( R2 <= 0 );
   // 2.2. calcular equivalente em serie
   R3 = R1 + R2;
   // 2.3. mostrar resultado
   printf ( "\nR3=R1+R2=%f %s", R3, " [ohms]" );
   X = X + 1;  // próximo valor
  } // fim repetir para X = 1:5:1
// pausa para terminar
  printf ( "Pressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 5a
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R1, // primeiro resistor
          R2, // segundo resistor
          R3; // resistor equivalente
   int    X;   // contador do numero de vezes
// 2. repetir 5 vezes (primeira forma)
   for ( X = 1; X<=5; X = X+1 )
   {
    // 2.1.1 ler primeiro valor
    do
    {
     cout << "\nR1=";
     cin   >> R1; // ler primeiro valor
    }
    while ( R1 <= 0 );
    // 2.1.2. ler segundo valor
    do
    {
     cout << "\nR2=";
     cin   >> R2, // ler segundo valor
    }
    while ( R2 <= 0 );
    // 2.2. calcular equivalente em serie
    R3 = R1 + R2;
    // 2.3. mostrar resultado
    cout << "\nR3=R1+R2=" << R3 << " [ohms]";
   } // fim repetir para X = 1:5:1
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C++:

```
// Exemplo 5b
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R1, // primeiro resistor
          R2, // segundo resistor
          R3; // resistor equivalente
   int    X;   // contador do numero de vezes
// 2. repetir 5 vezes (segunda forma)
   X = 1;     // valor inicial
   while ( X <= 5 )
   {
     // 2.1.1 ler primeiro valor
     do
     {
       cout << "\nR1=";
       cin >> R1; // ler primeiro valor
     }
     while ( R1 <= 0 );
     // 2.1.2. ler segundo valor
     do
     {
       cout << "\nR2=";
       cin >> R2, // ler segundo valor
     }
     while ( R2 <= 0 );
     // 2.2. calcular equivalente em serie
     R3 = R1 + R2;
     // 2.3. mostrar resultado
     cout << "\nR3=R1+R2=" << R3 << " [ohms]";
     X = X + 1;   // próximo valor
   } // fim repetir para X = 1:5:1
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 5a
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_5a
{
   public static void Main ( )
   {
   // 1. definir dados
      double R1, // primeiro resistor
                 R2, // segundo  resistor
                 R3; // resistor equivalente
   // 2. ler dados do teclado
   // 1. definir dados
      double R1, // primeiro resistor
                 R2, // segundo  resistor
                 R3; // resistor equivalente
      int        X;   // contador do numero de vezes
   // 2. repetir 5 vezes (primeira forma)
      for ( X = 1; X <= 5; X = X+1 )
      {
      // 2.1.1. ler primeiro valor
         do
         {
            Console.Write ( "\nR1=" );
            R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
         }
         while ( R1 <= 0 );
      // 2.1.2. ler segundo valor
         do
         {
            Console.Write ( "\nR2=" );
            R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
         }
         while ( R2 <= 0 );
      // 2.2. calcular equivalente em serie
         R3 = R1 + R2;
      // 2.3. mostrar resultado
         Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
      } // fim repetir para X = 1:5:1
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_5a class
```

Outra versão do programa em C#:

```
/*
 * Exemplo 5b
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_5b
{
   public static void Main ( )
   {
   // 1. definir dados
      double R1,                                  // primeiro resistor
             R2,                                  // segundo  resistor
             R3;                                 // resistor equivalente
      int    X;                                   // contador do numero de vezes
   // 2. repetir 5 vezes (primeira forma)
      X = 1;
      while ( X <= 5 )
      {
      // 2.1.1. ler primeiro valor
         do
         {
            Console.Write ( "\nR1=" );
            R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
         }
         while ( R1 <= 0 );
      // 2.1.2. ler segundo valor
         do
         {
            Console.Write ( "\nR2=" );
            R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
         }
         while ( R2 <= 0 );
      // 2.2. calcular equivalente em serie
         R3 = R1 + R2;
      // 2.3. mostrar resultado
         Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
         X = X + 1;                  // proximo valor
      } // fim repetir para X = 1:5:1
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_5b class
```

Programa em Java:

```
/**
  * Exemplo 5a
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_5a
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                          // primeiro resistor
             R2,                          // segundo  resistor
             R3;                          // resistor equivalente
      int    X;                           // contador do numero de vezes
   // 2. repetir 5 vezes (primeira forma)
      for ( X = 1; X <= 5; X = X+1 )
      {
       // 2.1.1. ler primeiro valor
         do
         {
           System.out.print ( "\nR1 = " ); // ler primeiro valor
           R1 = Double.parseDouble ( System.console( ).readLine( ) );
         }
         while ( R1 <= 0 );
       // 2.1.2. ler segundo valor
         do
         {
           System.out.print ( "\nR2 = " ); // ler segundo   valor
           R2 = Double.parseDouble ( System.console( ).readLine( ) );
         }
         while ( R2 <= 0 );
       // 2.2. calcular equivalente em serie
         R3 = R1 + R2;
       // 2.3. mostrar resultado
         System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );
      } // fim repetir para X = 1:5:1
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_5a class
```

Outra versão do programa em Java:

```
/**
  * Exemplo 5b
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_5a
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                    // primeiro resistor
             R2,                    // segundo  resistor
             R3;                    // resistor equivalente
      int    X;                     // contador do numero de vezes
    // 2. repetir 5 vezes (segunda forma)
      X = 1;
      while ( X <= 5 )
      {
       // 2.1.1. ler primeiro valor
          do
          {
            System.out.print ( "\nR1 = " ); // ler primeiro valor
            R1 = Double.parseDouble ( System.console( ).readLine( ) );
          }
          while ( R1 <= 0 );
       // 2.1.2. ler segundo valor
          do
          {
            System.out.print ( "\nR2 = " ); // ler segundo   valor
            R2 = Double.parseDouble ( System.console( ).readLine( ) );
          }
          while ( R2 <= 0 );
       // 2.2. calcular equivalente em serie
          R3 = R1 + R2;
       // 2.3. mostrar resultado
          System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );

          X = X + 1;                        // proximo valor

      } // fim repetir para X = 1:5:1
    // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_5b class
```

Programa em Python:

```
# Exemplo 5a
# Dados dois resistores,
# calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
X   = 0;    # contador do numero de vezes
# 2. repetir 5 vezes (primeira forma)
for X in range ( 1, 5+1, 1 ):              # repetir 5 vezes
  # 2.1.1 ler primeiro valor
    R1 = float ( input ( "\nR1 = " ) );    # ler primeiro valor
    while ( R1 <= 0 ):
       R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
    # ( R1 <= 0 )
  # 2.2. ler segundo valor
    R2 = float ( input ( "\nR2 = " ) );    # ler segundo valor
    while ( R2 <= 0 ):
       R2 = float ( input ( "\nR2 = " ) ); # ler segundo valor
  # ( R2 <= 0 )
  # 2.3. calcular equivalente em serie
    R3 = R1 + R2;
  # 2.4. mostrar resultado
    print ( "\nR3=R1+R2= ", R3, " [ohms]" );
# repetir para X = 1:5:1
# pausa para terminar
print ( "\nPressionar <ENTER> para terminar.\n" );
input  ( );
# fim do programa
```

Outra versão do programa em Python:

```
# Exemplo 5b
# Dados dois resistores,
# calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
X   = 0;    # contador do numero de vezes
# 2. repetir 5 vezes (segunda forma)
X = 1;      # valor inicial
while ( X <= 5 ):                          # repetir 5 vezes
  # 2.1.1 ler primeiro valor
    R1 = float ( input ( "\nR1 = " ) );    # ler primeiro valor
    while ( R1 <= 0 ):
       R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
    # ( R1 <= 0 )
  # 2.2. ler segundo valor
    R2 = float ( input ( "\nR2 = " ) );    # ler segundo valor
    while ( R2 <= 0 ):
       R2 = float ( input ( "\nR2 = " ) ); # ler segundo valor
  # ( R2 <= 0 )
  # 2.3. calcular equivalente em serie
    R3 = R1 + R2;
  # 2.4. mostrar resultado
    print ( "\nR3=R1+R2= ", R3, " [ohms]" );
    X = X + 1; # proximo valor
# repetir para X = 1:5:1
# pausa para terminar
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um número inteiro (N) do teclado;
   - calcular e mostrar a soma dos (N) primeiros números naturais.

2. Fazer um algoritmo para :
   - calcular e mostrar a soma dos pares entre100 e 500.

3. Fazer um algoritmo para :
   - ler dois números inteiros (M e N, M < N) do teclado;
   - calcular e mostrar a soma dos números entre (M) e (N).

4. Fazer um algoritmo para :
   - ler dois números inteiros (M e N, M < N) do teclado;
   - calcular e mostrar a soma dos quadrados dos números entre eles.

5. Fazer um algoritmo para :
   - ler um número inteiro (N) do teclado;
   - ler N outros valores reais (P) do teclado, um por vez;
   - calcular e mostrar o produto destes valores.

Exemplo 6.

Fazer um algoritmo para:
- repetir para um número indeterminado de vezes:
  - ler os valores de dois resistores do teclado e garantir que sejam válidos;
  - calcular e mostrar o valor de resistor equivalente em série.

Análise de dados:

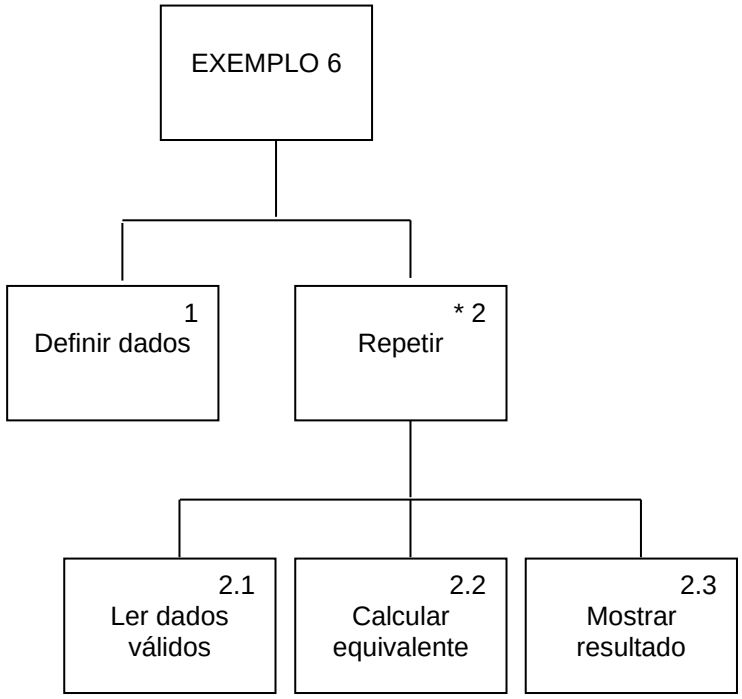
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R1	real		resistor 1 > 0 (válido)
R2	real		resistor 2 > 0 (válido)
R3	real		resistor equivalente

- Fórmulas que relacionam os dados :

R3 = R1 + R2

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
R1 = 10 [ohms] R2 = 5 [ohms]	R3 = 15 [ohms]
R1 = 10 [ohms] R2 = 2 [ohms]	R3 = 12 [ohms]
R1 = 10 [ohms] R2 = 1 [ohms]	R3 = 11 [ohms]
R1 = 5 [ohms] R2 = 2 [ohms]	R3 = 7 [ohms]
R1 = 2 [ohms] R2 = 1 [ohms]	R3 = 3 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 6	v.1
Ação	Bloco
! definir dados	1
! repetir até parar	2
! ler dados válidos do teclado	2.1
! calcular equivalente em série	2.2
! mostrar resultado	2.3

Segunda versão, refinar o primeiro bloco.

Exemplo 6	v.2
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! repetir até parar	2
! ler dados válidos do teclado	2.1
! calcular equivalente em série	2.2
! mostrar resultado	2.3

Terceira versão, refinar o segundo bloco.

Exemplo 6	v.3
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! repetir até parar	2
! ler dados válidos do teclado	2.1
! calcular equivalente em série	2.2
! mostrar resultado	2.3
enquanto houver dados	

Quarta versão, refinar o segundo bloco.

Exemplo 6	v.4
Ação	Bloco
! definir dados real R1, ! primeiro resistor R2, ! segundo resistor R3; ! resistor equivalente	1
! repetir até parar	2
! ler dados válidos do teclado	2.1
tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor	
R1 ≤ 0 ?	
tela ← "\nR2 = "; R2 ← teclado; ! ler segundo valor	
R2 ≤ 0 ?	
! calcular equivalente em série R3 ← R1 + R2;	2.2
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	2.3
enquanto houver dados	

Quinta versão, refinar novamente o segundo bloco.

Exemplo 6	v.5
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
  inteiro Resposta; ! controle da repetição
	1

! repetir até parar	

! ler dados válidos do teclado
  repetir até (R1 > 0)
  tela ← "\nR1 = "; R1 ← teclado; ! ler primeiro valor
  fim repetir ! enquanto (R1 ≤ 0)
  repetir até (R2 > 0)
  tela ← "\nR2 = "; R2 ← teclado; ! ler segundo valor
  fim repetir ! enquanto (R2 ≤ 0)
	2.1

! calcular equivalente em série R3 ← R1 + R2;	2.2
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	2.3

! verificar se há mais dados
  tela ← “\nMais dados (Sim=1,Não=0) ? ";
  Resposta ← teclado;
	2.4

Resposta = 1 ?	

Sexta versão, refinar novamente o segundo bloco.

Exemplo 6	v.6
Ação	Bloco

! definir dados
  real R1, ! primeiro resistor
  R2, ! segundo resistor
  R3; ! resistor equivalente
  inteiro Resposta; ! controle da repetição
	1

! repetir até parar	

repetir até Resposta ≠ 1
  ! ler dados válidos do teclado
  repetir até (R1 > 0)
  tela ← "\nR1="; R1 ← teclado; ! ler primeiro valor
  fim repetir ! enquanto (R1 ≤ 0)
  repetir até (R2 > 0)
  tela ← "\nR2="; R2 ← teclado; ! ler primeiro valor
  fim repetir ! enquanto (R2 ≤ 0)
	2.1

! calcular equivalente em série R3 ← R1 + R2;	2.2
! mostrar resultado tela ← ("\nR3=R1+R2=", R3, " [ohms]");	2.3

! verificar se há mais dados
  tela ← “\nMais dados (Sim=1,Não=0) ? ";
  Resposta ← teclado;
	2.4

fim repetir ! enquanto Resposta = 1	

Programa em SCILAB:

```
// Exemplo 6
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// 1. definir dados
   R1 = 0.0;       // primeiro resistor
   R2 = 0.0;       // segundo resistor
   R3 = 0.0;       // resistor equivalente
   Resposta = 0; // contador do numero de vezes
// 2. repetir até parar
   clc;            // limpar a area de trabalho
   Resposta = input ( “\nMais dados (Sim=1,Nao=0) ? “ );
   while ( Resposta == 1 )
   // 2.1.1 ler primeiro valor
      R1 = input ( “\nR1 “ );   // ler primeiro valor
      while ( R1 <= 0 )
         R1 = input ( “\nR1 “ ); // ler primeiro valor
      end // ( R1 <= 0 )
   // 2.1.2. ler segundo valor
      R2 = input ( “\nR2 “ );   // ler segundo valor
      while ( R2 <= 0 )
        R2 = input ( “\nR2 “ ); // ler segundo valor
      end // ( R2 <= 0 )
   // 2.2. calcular equivalente em serie
      R3 = R1 + R2;
   // 2.3. mostrar resultado
      printf ( “\nR3=R1+R2= %f [ohms]”, R3 );
   // 2.4. verificar se ha' mais dados
      Resposta = input ( “\nMais dados (Sim=1,Nao=0) ? “ );
   end // enquanto houver dados
// pausa para terminar
   printf ( “\nPressionar ENTER para terminar.\n” );
   halt;
// fim do programa
```

Programa em C:

```
// Exemplo 6
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R1, // primeiro resistor
        R2, // segundo resistor
        R3; // resistor equivalente
  int   Resposta; // controle da repeticao
// 2. repetir ate' parar
  do
  {
   // 2.1.1 ler primeiro valor
   do
   {
     printf ( "\nR1=" );
     scanf ( "%f", &R1 ); // ler primeiro valor
   }
   while ( R1 <= 0 );
   // 2.1.2. ler segundo valor
   do
   {
     printf ( "\nR2=" );
     scanf ( "%f", &R2 ); // ler primeiro valor
   }
   while ( R2 <= 0 );
   // 2.2. calcular equivalente em serie
   R3 = R1 + R2;
   // 2.3. mostrar resultado
   printf ( "\nR3=R1+R2=%f %s", R3, " [ohms]" );
   // 2.4. verificar se ha' mais dados
   cout << "\nMais dados (Sim=1,Nao=0) ? ";
   cin  >> Resposta;
  }
  while ( Resposta == 1 ); // enquanto houver dados
// pausa para terminar
  printf ( "\nPressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 6
// Dados dois resistores,
// calcular o resistor equivalente em serie.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R1, // primeiro resistor
          R2, // segundo resistor
          R3; // resistor equivalente
   int    Resposta; // controle da repeticao
// 2. repetir ate' parar
   do
   {
    // 2.1.1 ler primeiro valor
    do
    {
     cout << "\nR1=";
     cin  >> R1; // ler primeiro valor
    }
    while (R1 <= 0);
    // 2.1.2. ler segundo valor
    do
    {
     cout << "\nR2=";
     cin  >> R2, // ler segundo valor
    }
    while (R2 <= 0);
    // 2.2. calcular equivalente em serie
    R3 = R1 + R2;
    // 2.3. mostrar resultado
    cout << "\nR3=R1+R2=" << R3 << " [ohms]";
    // 2.4. verificar se ha' mais dados
    cout << "\nMais dados (Sim=1,Nao=0) ? ";
    cin  >> Resposta;
   }
   while ( Resposta == 1 ); // enquanto houver dados
// pausa para terminar
   cout << "\nPressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 6
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_6
{
  public static void Main ( )
  {

  // 1. definir dados
    double R1, // primeiro resistor
           R2, // segundo  resistor
           R3; // resistor equivalente
    int    Resposta;  // controle da repeticao
  // 2. repetir ate' parar
    do
    {
    // 2.1.1. ler primeiro valor
      do
      {
        Console.Write ( "\nR1=" );
        R1 = int.Parse ( Console.ReadLine ( ) ); // ler primeiro valor
      }
      while ( R1 <= 0 );
    // 2.1.2. ler segundo valor
      do
      {
        Console.Write ( "\nR2=" );
        R2 = int.Parse ( Console.ReadLine ( ) ); // ler segundo valor
      }
      while ( R2 <= 0 );
    // 2.2. calcular equivalente em serie
      R3 = R1 + R2;
    // 2.3. mostrar resultado
      Console.WriteLine ( "\nR3=R1+R2=" + R3 + " [ohms]" );
    // 2.4. verificar se ha' mais dados
      Console.Write ( "\nMais dados (Sim=1,Nao=0) ? " );
      Resposta = int.Parse ( Console.ReadLine ( ) );
    }
    while ( Resposta == 1 ); // enquanto houver dados
  // pausa para terminar
    Console.Write ( "\nPressionar ENTER para terminar." );
    Console.ReadLine ( );
  } // end Main ( )

} // fim Exemplo_6 class
```

Programa em Java:

```
/**
  * Exemplo 6
  * Dados dois resistores, calcular o resistor equivalente em serie.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_6
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R1,                          // primeiro resistor
             R2,                          // segundo  resistor
             R3;                          // resistor equivalente
      int    Resposta;                    // controle da repeticao
    // 2. repetir ate" parar
      do
      {
       // 2.1.1. ler primeiro valor
         do
         {
           System.out.print ( "\nR1 = " ); // ler primeiro valor
           R1 = Double.parseDouble ( System.console( ).readLine( ) );
         }
         while ( R1 <= 0 );
       // 2.1.2. ler segundo valor
         do
         {
           System.out.print ( "\nR2 = " ); // ler segundo   valor
           R2 = Double.parseDouble ( System.console( ).readLine( ) );
         }
         while ( R2 <= 0 );
       // 2.2. calcular equivalente em serie
         R3 = R1 + R2;
       // 2.3. mostrar resultado
         System.out.println ( "\nR3 = R1+R2 = " + R3 + " [ohms]" );
       // 2.4. verificar se ha' mais dados
         System.out.print ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = Integer.parseInt ( System.console( ).readLine( ) );
      }
      while ( Resposta == 1 ); // enquanto houver dados
    // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
    } // end main ( )

} // fim Exemplo_6 class
```

Programa em Python:

```
# Exemplo 6
# Dados dois resistores,
# calcular o resistor equivalente em serie.
#
# 1. definir dados
R1 = 0.0; # primeiro resistor
R2 = 0.0; # segundo resistor
R3 = 0.0; # resistor equivalente
Resposta  = 0;       # controle da repeticao
# 2. repetir ate' parar
Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
while ( Resposta == 1 ):
  # 2.1.1 ler primeiro valor
    R1 = float ( input ( "\nR1 = " ) );    # ler primeiro valor
    while ( R1 <= 0 ):
       R1 = float ( input ( "\nR1 = " ) ); # ler primeiro valor
    # ( R1 <= 0 )
  # 2.2. ler segundo valor
    R2 = float ( input ( "\nR2 = " ) );    # ler segundo valor
    while ( R2 <= 0 ):
       R2 = float ( input ( "\nR2 = " ) ); # ler segundo valor
  # ( R2 <= 0 )
  # 2.3. calcular equivalente em serie
    R3 = R1 + R2;
  # 2.4. mostrar resultado
    print ( "\nR3=R1+R2= ", R3, " [ohms]" );
  # 2.5. verificar se ha' mais dados
    Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
# enquanto houver dados
# pausa para terminar
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um número indeterminado de dados, contendo cada um, a idade de um indivíduo;
   - calcular e mostrar o número de dados lidos e quantos valores são maiores que 18 anos.

2. Fazer um algoritmo para :
   - ler um número indeterminado de dados, contendo cada um, a idade de um indivíduo;
   - sabendo-se que o último dado conterá o valor zero e não entrará nos cálculos,
   - calcular e mostrar o número de dados lidos e quantos valores são maiores que 18 anos.

3. Fazer um algoritmo para :
   - ler um conjunto de dados contendo, cada um, uma nota;
   - determinar e mostrar quantas notas estão acima de 60 pontos e quantas estão abaixo;
   - o último dado, e que não será processado, conterá a nota = 999.

4. Fazer um algoritmo para :
   - ler um número indeterminado de valores inteiros positivos,
   - o último dado, que não será processado, conterá o valor 9999;
   - calcular e mostrar a porcentagem de valores pares e ímpares.

5. Fazer um algoritmo para :
   - ler um número indeterminado de valores inteiros,
   - o último dado, que não será processado, conterá o valor 9999;
   - calcular e mostrar a porcentagem de valores negativos, nulos e positivos.

Exemplo 7.

Fazer um algoritmo para:
- ler 10 valores de resistores testados em laboratório;
- calcular e mostrar o valor médio desta amostra.

Análise de dados:

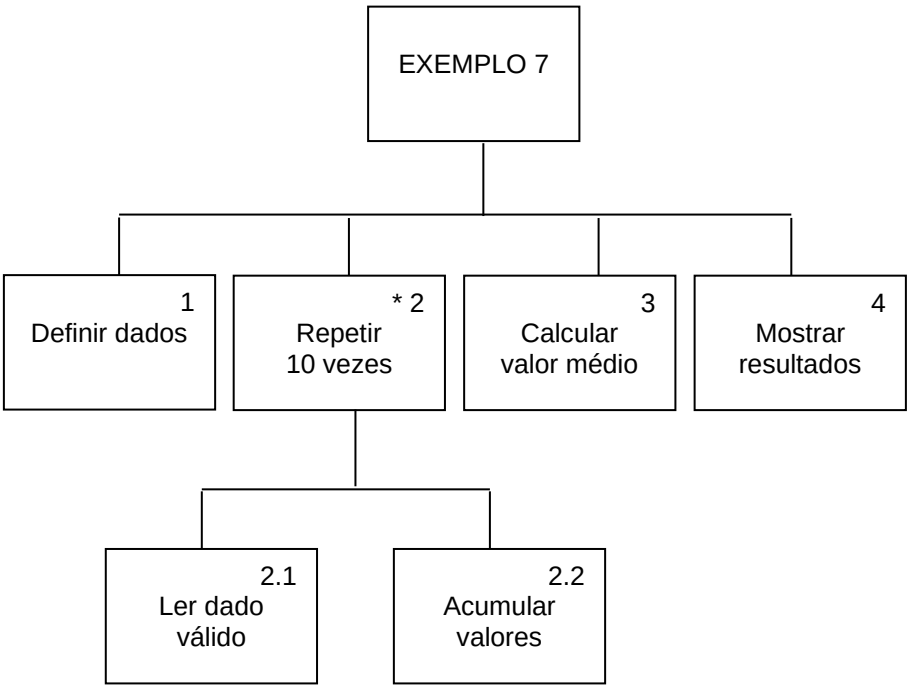
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R	real		resistor > 0 (válido)
SOMA	real	0.0	somatório de valores
MÉDIA	real		valor médio

- Fórmulas que relacionam os dados :

MÉDIA = SOMA / 10;

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
10.00 [ohms]	
10.04 [ohms]	
10.01 [ohms]	
10.05 [ohms]	
10.00 [ohms]	
09.96 [ohms]	
10.00 [ohms]	
09.95 [ohms]	
09.99 [ohms]	
10.00 [ohms]	Valor médio = 10.00 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 7	v.1
Ação	Bloco
! definir dados	1
! repetir 10 vezes	2
! ler dado válido do teclado	2.1
! acumular valores	2.2
! calcular o valor médio	3
! mostrar resultado	4

Segunda versão, refinar o primeiro bloco.

Exemplo 7	v.2
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA; ! valor médio	1
! repetir 10 vezes	2
! ler dado válido do teclado	2.1
! acumular valores	2.2
! calcular o valor médio	3
! mostrar resultado	4

Terceira versão, refinar o segundo bloco.

Exemplo 7	v.3
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA; ! valor médio inteiro X; ! contador do numero de vezes	1
! repetir 10 vezes	2
X ← 1:10:1	
! ler dados válidos do teclado	2.1
! acumular valores	2.2
! calcular valor médio	3
! mostrar resultado	4

Quarta versão, refinar o segundo bloco.

Exemplo 7	v.4
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA; ! valor médio inteiro X; ! contador do numero de vezes	1
! repetir 10 vezes	2
X ← 1:10:1	
! ler dados válidos do teclado	2.1
tela ← "\nR = "; R ← teclado; ! ler valor	
R ≤ 0 ?	
! acumular valores SOMA ← SOMA + R;	2.2
! calcular valor médio	3
! mostrar resultado	4

Quinta versão, refinar o terceiro e quarto blocos.

Exemplo 7	v.5
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA; ! valor médio inteiro X; ! contador do numero de vezes	1
! repetir 10 vezes	2
X ← 1:10:1	
! ler dados válidos do teclado	2.1
tela ← "\nR = "; R ← teclado; ! ler valor	
R ≤ 0 ?	
! acumular valores SOMA ← SOMA + R;	2.2
! calcular valor médio MEDIA ← SOMA / 10;	3
! mostrar resultado tela ← (“\nValor médio =”, MEDIA, “ [ohms]”);	4

Sexta versão, refinar novamente o segundo bloco.

Exemplo 7	v.5
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA; ! valor médio inteiro X; ! contador do numero de vezes	1
! repetir 10 vezes	2
repetir para (X ← 1:10:1) ! ler dados válidos do teclado repetir até (R > 0) tela ← "\nR="; R ← teclado; ! ler valor fim repetir ! enquanto (R<=0) ! acumular valores SOMA ← SOMA + R; fim repetir ! para (X ← 1:10:1)	
! calcular valor médio MEDIA ← SOMA / 10;	3
! mostrar resultado tela ← (“\nValor médio =”, MEDIA, “ [ohms]”);	4

Programa em SCILAB:

```
// Exemplo 7
// Dados valores de resistores, calcular o valor medio.
//
// 1. definir dados
   R       = 0.0;  // resistor
   SOMA = 0.0;  // somatorio de valores
   MEDIA = 0.0; // valor medio
   X       = 0;    // contador do numero de vezes
// 2. repetir 10 vezes
   clc;               // limpar a area de trabalho
   for  X = 1 : 1 : 10
   // 2.1. ler um valor
      R = input ( “\nR “ );   // ler primeiro valor
      while ( R <= 0 )
        R = input ( “\nR “ ); // ler outro valor
      end // ( R <= 0 )
   // 2.2. acumular valores
      SOMA = SOMA + R;
   end // repetir para ( X = 1 : 10 : 1 )
// 3. calcular valor medio
   MEDIA = SOMA / 10;
// 4. mostrar resultado
   printf ( “\nValor medio = %f [ohms]”, MEDIA );
// pausa para terminar
   printf ( “\nPressionar ENTER para terminar.\n” );
   halt;
// fim do programa
```

Programa em C:

```
// Exemplo 7
// Dados valores de resistores, calcular o valor medio.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
   float R,                // resistor
         SOMA = 0.0,  // somatorio de valores
          MEDIA;        // valor medio
   int    X;              // contador do numero de vezes
// 2. repetir 10 vezes
   for ( X=1; X<=10; X=X+1 )
   {
     // 2.1. ler um valor
     do
     {
       printf ( "\nR=" );
       scanf ( "%f", &R ); // ler valor
     }
     while ( R <= 0 );
     // 2.2. acumular valores
     SOMA = SOMA + R;
   } // fim repetir
// 3. calcular valor medio
   MEDIA = SOMA / 10;
// 4. mostrar resultado
   printf ( “\nValor medio = %f, %s”, MEDIA, “ [ohms]” );
// pausa para terminar
   printf ( "Pressionar ENTER para terminar." );
   getchar ( );
   return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 7
// Dados valores de resistores, calcular o valor medio.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R,                // resistor
          SOMA = 0.0,  // somatorio de valores
          MEDIA;          // valor medio
   int    X;                // contador do numero de vezes
// 2. repetir 10 vezes
   for ( X=1; X<=10; X=X+1 )
   {
    // 2.1. ler um valor
    do
    {
      cout << "\nR=";
      cin   >> R; // ler valor
    }
    while ( R <= 0 );
    // 2.2. acumular valores
    SOMA = SOMA + R;
   } // fim repetir
// 3. calcular valor medio
   MEDIA = SOMA / 10;
// 4. mostrar resultado
   cout << "\nValor medio =" << MEDIA << " [ohms]";
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 7
 * Dados dois resistores, calcular o resistor equivalente em serie.
 */

using System;

class Exemplo_7
{
   public static void Main ( )
   {
   // 1. definir dados
      double R,                 // resistor
                SOMA = 0.0, // segundo  resistor
                MEDIA= 0.0; // resistor equivalente
      int       X;              // contador do numero de vezes
   // 2. repetir 10 vezes
      for ( X = 1; X <= 10; X = X+1 )
      {
      // 2.1. ler dado valido do teclado
         do
         {
            Console.Write ( "\nR=" );
            R = int.Parse ( Console.ReadLine ( ) ); // ler valor
         }
         while ( R <= 0 );
      // 2.2. acumular valores
         SOMA = SOMA + R;
      } // fim repetir
   // 3. calcular o valor medio
      MEDIA = SOMA / 10;
   // 4. mostrar resultado
      Console.WriteLine ( "\nValor medio=" + MEDIA + " [ohms]"  );
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_7 class
```

Programa em Java:

```
/**
  * Exemplo 7
  * Dados valores de resistores, calcular o valor medio.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_7
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R,                        // resistor
             SOMA = 0.0,               // segundo  resistor
             MEDIA= 0.0;               // resistor equivalente
      int    X;                        // contador do numero de vezes
   // 2. repetir 10 vezes
      for ( X = 1; X <= 10; X = X+1 )
      {
      // 2.1. ler dado valido do teclado
         do
         {
           System.out.print ( "\nR = " );// ler valor
           R = Integer.parseInt ( System.console( ).readLine( ) );
         }
         while ( R <= 0 );
      // 2.2. acumular valores
         SOMA = SOMA + R;
      } // fim repetir
   // 3. calcular o valor medio
      MEDIA = SOMA / 10;
   // 4. mostrar resultado
      System.out.println ( "\nValor medio = " + MEDIA + " [ohms]" );
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_7 class
```

Programa em Python:

```
# Exemplo 7
# Dados valores de resistores, calcular o valor medio.
#
# 1. definir dados
R      = 0.0;  # resistor
SOMA   = 0.0;  # somatorio de valores
MEDIA = 0.0;  # valor medio
X      = 0;    # contador do numero de vezes
# 2. repetir 10 vezes
for  X in range ( 1, 10+1, 1 ):
  # 2.1. ler um valor
    R = float ( input ( "\nR = " ) );     # ler primeiro valor
    while ( R <= 0 ):
        R = float ( input ( "\nR = " ) ); # ler outro valor
    # enquanto ( R <= 0 )
  # 2.2. acumular valores
    SOMA = SOMA + R;
# repetir para ( X = 1 : 10 : 1 )
# 3. calcular valor medio
MEDIA = SOMA / 10.0;
# 4. mostrar resultado
print ( "\nValor medio = ", MEDIA, " [ohms]" );
# pausa para terminar
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler valores de idade de 10 indivíduos;
   - calcular e mostrar a idade média deste grupo de indivíduos.

2. Fazer um algoritmo para :
   - ler valores de idade de 10 indivíduo;
   - calcular e mostrar a idade média dos maiores que 18 anos.

3. Fazer um algoritmo para :
   - ler o número de valores em um conjunto de dados (N) contendo, cada um, uma nota;
   - ler o valor de cada nota;
   - determinar e mostrar a média dos valores maiores que 60 pontos.

4. Fazer um algoritmo para :
   - ler o número de valores em um conjunto de dados (N);
   - ler (N) valores inteiros positivos,
   - calcular e mostrar a soma dos valores pares e a soma dos valores ímpares.

5. Fazer um algoritmo para :
   - ler o número de valores em um conjunto de dados (N);
   - ler (N) valores inteiros positivos,
   - calcular e mostrar a diferença entre o valor médio negativo e o valor médio positivo.

Exemplo 8.

Fazer um algoritmo para:
- ler um número indeterminado de valores de resistores testados em laboratório;
- calcular e mostrar o valor médio desta amostra.

Análise de dados:

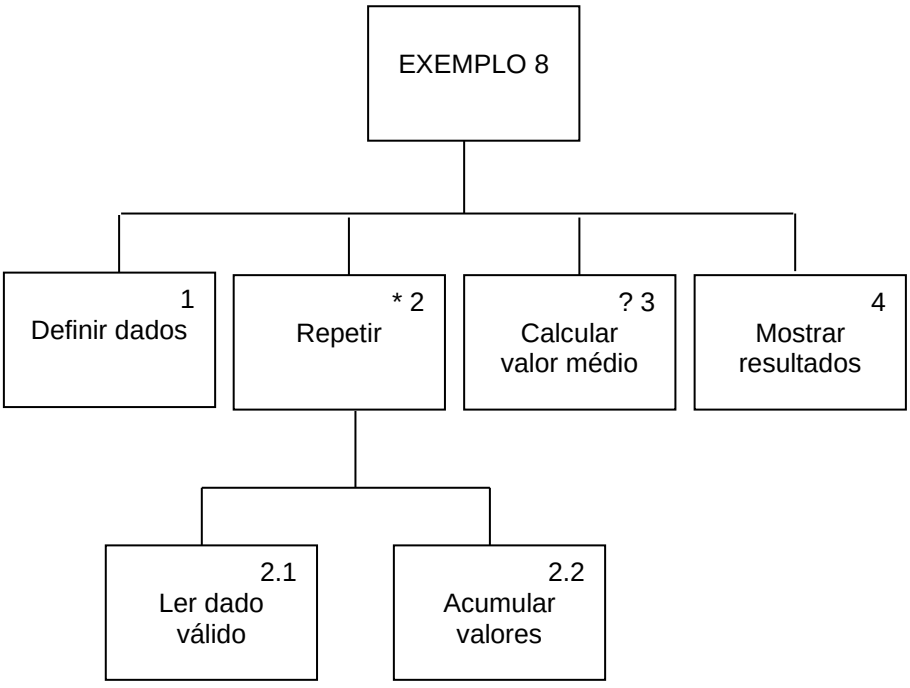
- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R	real		resistor > 0 (válido)
SOMA	real	0.0	somatório de valores
N	inteiro	0	número de elementos > 0
MÉDIA	real	0.0	valor médio

- Fórmulas que relacionam os dados :

MÉDIA = SOMA / N; ! se N diferente de zero

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
10.00 [ohms]	
10.04 [ohms]	
10.01 [ohms]	
10.05 [ohms]	
10.00 [ohms]	
09.96 [ohms]	
10.00 [ohms]	
09.95 [ohms]	
09.99 [ohms]	
10.00 [ohms]	Valor médio = 10.00 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 8	v.1
Ação	Bloco
! definir dados	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! acumular valores	2.2
! calcular o valor médio, se houver dados	3
! mostrar resultado	4

Segunda versão, refinar o primeiro bloco.

Exemplo 8	v.2
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA ← 0.0; ! valor médio inteiro N=0; ! numero de elementos	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! acumular valores	2.2
! calcular o valor médio, se houver dados	3
! mostrar resultado	4

Terceira versão, refinar o segundo bloco.

Exemplo 8	v.4
Ação	Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA ← 0.0; ! valor médio inteiro N=0; ! numero de elementos	1
! repetir enquanto houver dados	2
! ler dado válido do teclado tela ← "\nR="; R ← teclado; ! ler valor R ≤ 0 ?	2.1
! acumular valores SOMA ← SOMA + R; N ← N + 1; ! mais um dado valido	2.2
até parar	
! calcular valor médio	3
! mostrar resultado	4

Quarta versão, refinar o terceiro e quarto blocos.

Exemplo 8			v.4
Ação			Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA ← 0.0; ! valor médio inteiro N=0; ! numero de elementos			1
! repetir enquanto houver dados			2
! ler dado válido do teclado tela ← "\nR="; R ← teclado; ! ler valor R ≤ 0 ?			2.1
! acumular valores SOMA ← SOMA + R; N ← N + 1; ! mais um dado valido			2.2
até parar			
! calcular valor médio			3
N=0?	V	! não houve dados tela ← “\nNão houve dados”;	3.1
	F	! houve dados MEDIA ← SOMA / N;	3.2
! mostrar resultado tela ← (“\nValor médio = “, MEDIA, “ [ohms]”);			4

Quinta versão, refinar novamente o segundo bloco.

Exemplo 8			v.4
Ação			Bloco
! definir dados real R, ! resistor SOMA ← 0.0, ! somatório de valores MEDIA ← 0.0; ! valor médio inteiro N ← 0, ! numero de elementos Resposta; ! controle da repetição			1
! repetir enquanto houver dados			2
! ler dado válido do teclado			2.1
tela ← "\nR="; R ← teclado; ! ler valor			
R ≤ 0 ?			
! acumular valores SOMA ← SOMA + R; N ← N + 1; ! mais um dado valido			2.2
! verificar se há mais dados tela ← "\nMais dados (Sim=1,Não=0) ?"; Resposta ← teclado;			2.3
Resposta = 1			
! calcular valor médio			3
N=0?	V	! não houve dados tela ← "\nNão houve dados";	
	F	! houve dados MEDIA ← SOMA / N;	
! mostrar resultado tela ← ("\nValor médio = ", MEDIA, " [ohms]");			4

Sexta versão, refinar novamente o segundo e terceiro blocos.

Exemplo 8	v.4
Ação	Bloco

! definir dados
  real R, ! resistor
  SOMA ← 0.0, ! somatório de valores
  MEDIA=0.0; ! valor médio
  inteiro N ← 0, ! numero de elementos
  Resposta; ! controle da repetição
	1

! repetir enquanto houver dados	2
repetir até (Resposta ≠ 1) ! 2.1 ler dado válido do teclado repetir até (R > 0) tela ← "\nR="; R ← teclado; ! ler valor fim repetir ! enquanto (R ≤ 0) ! 2.2 acumular valores SOMA ← SOMA + R; N ← N + 1; ! 2.3 verificar se há mais dados tela ← “\nMais dados (Sim=1,Não=0) ?”; Resposta ← teclado; fim repetir ! enquanto (Resposta = 1);	
! calcular valor médio	3
se (N = 0) ! não houve dados tela ← “\nNão houve dados”; senão ! houve dados, calcular a média MEDIA ← SOMA / N; fim se ! houve dados	
! mostrar resultado tela ← (“\nValor médio = “, MEDIA, “ [ohms]”);	4

Programa em SCILAB:

```
// Exemplo 8
// Dados valores de resistores, calcular o valor medio.
//
// 1. definir dados
  R        = 0.0;  // resistor
  SOMA     = 0.0;  // somatorio de valores
  MEDIA    = 0.0;  // valor medio
  N        = 0;    // numero de elementos
  Resposta = 0;    // controle da repeticao
// 2. repetir ate' parar
  clc;                  // limpar a area de trabalho
  Resposta = input ( "\nMais dados (Sim=1,Nao=0) ? " );
  while ( Resposta == 1 )
  // 2.1.1 ler dado valido do teclado
    R = input ( "\nR " );   // ler valor
    while ( R <= 0 )
      R = input ( "\nR " ); // ler valor
    end // ( R<=0 )
  // 2.2. acumular valores
    SOMA = SOMA + R;
    N = N + 1; // mais um dado valido
  // 2.3. verificar se ha' mais dados
    Resposta = input ( "\nMais dados (Sim=1,Nao=0) ? " );
  end // enquanto houver dados
// 3. calcular o valor medio
  if ( N == 0 )
  // nao houve dados
    printf ( "\nNao houve dados" );
  else
  // houve dado, calcular a media
    MEDIA = SOMA / N;
  end // fim se houve dados
// 4. mostrar resultado
  printf ( "\nValor medio= %f [ohms]", MEDIA );
// pausa para terminar
  printf ( "\nPressionar ENTER para terminar.\n" );
  halt;
// fim do programa
```

Programa em C:

```
// Exemplo 8
// Dados valores de resistores, calcular o valor medio.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R,              // resistor
        SOMA = 0.0, // somatorio de valores
        MEDIA= 0.0; // valor medio
  int   N = 0;          // numero de elementos
        Resposta;     // controle da repeticao
// 2. repetir ate' parar
  do
  {
  // 2.1 ler dado valido do teclado
    do
    {
      printf ( "\nR=" );
      scanf ( "%f", &R ); // ler valor
    }
    while (R <= 0);
  // 2.2 acumular valores
    SOMA = SOMA + R;
    N = N + 1;
  // 2.3 verificar se ha' mais dados
    printf ( “\nMais dados (Sim=1,Não=0) ? ” );
    scanf ( "%d", &Resposta );
  }
  while ( Resposta == 1 );
// 3. calcular o valor medio
  if ( N == 0 )
  {
  // nao houve dados
    printf ( “\nNao houve dados” );
  }
  else
  {
  // houve dados, calcular a media
    MEDIA = SOMA / N;
  } // fim se houve dados
// 4. mostrar resultado
  printf ( “\nValor medio = %f %s”, MEDIA, “ [ohms]” );
// pausa para terminar
  printf ( "\nPressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 8
// Dados valores de resistores, calcular o valor medio.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R,              // resistor
          SOMA = 0.0, // somatorio de valores
          MEDIA= 0.0; // valor medio
   int    N = 0;          // numero de elementos
          Resposta;    // controle da repeticao
// 2. repetir ate' parar
   do
   {
   // 2.1 ler dado valido do teclado
      do
      {
        cout << "\nR=";
        cin   >> R; // ler valor
      }
      while (R <= 0);
   // 2.2 acumular valores
      SOMA = SOMA + R;
      N = N + 1;
   // 2.3 verificar se ha' mais dados
      cout << "\nMais dados (Sim=1,Não=0) ?";
      cin   >> Resposta;
   }
   while ( Resposta == 1 );
// 3. calcular o valor medio
   if ( N == 0 )
   {
   // nao houve dados
      cout << "\nNao houve dados";
   }
   else
   {
   // houve dados, calcular a media
      MEDIA = SOMA / N;
   } // fim se houve dados
// 4. mostrar resultado
   cout << "\nValor medio=" << MEDIA << " [ohms]";
// pausa para terminar
   cout << "\nPressionar ENTER para terminar.";
   cing.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 8
 * Dados valores de resistores, calcular o valor medio.
 */
using System;

class Exemplo_8
{
   public static void Main ( )
   {
   // 1. definir dados
      double R,               // resistor
             SOMA = 0.0, // segundo  resistor
             MEDIA= 0.0; // resistor equivalente
      int    N = 0,           // numero de elementos
             Resposta;    // controle da repeticao
   // 2. repetir ate' parar
      do
      {
      // 2.1. ler dado valido do teclado
         do
         {
            Console.Write ( "\nR=" );
            R = int.Parse ( Console.ReadLine ( ) ); // ler valor
         }
         while ( R <= 0 );
      // 2.2. acumular valores
         SOMA = SOMA + R;
         N = N + 1;
      // 2.3. verificar se ha' mais dados
         Console.WriteLine ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = int.Parse ( Console.ReadLine ( ) );
      }
      while ( Resposta == 1 );
   // 3. calcular o valor medio
      if ( N == 0 )
      {
      // nao houve dados
         Console.WriteLine ( "\nNao houve dados" );
      }
      else
      {
      // houve dados, calcular a media
         MEDIA = SOMA / N;
      } // fim se houve dados
   // 4. mostrar resultado
      Console.WriteLine ( "\nValor medio=" + MEDIA + " [ohms]"  );
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )

} // fim Exemplo_8 class
```

Programa em Java:

```
/**
  * Exemplo 8
  * Dados valores de resistores, calcular o valor medio.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_8
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R,                    // resistor
             SOMA = 0.0,           // segundo  resistor
             MEDIA= 0.0;           // resistor equivalente
      int    N = 0,                // numero de elementos
             Resposta;             // controle da repeticao
   // 2. repetir ate' parar
      do
      {
      // 2.1. ler dado valido do teclado
         do
         {
           System.out.print ( "\nR = " );  // ler valor
           R = Integer.parseInt ( System.console( ).readLine( ) );
         }
         while ( R <= 0 );
      // 2.2. acumular valores
         SOMA = SOMA + R;
         N = N + 1;
      // 2.3. verificar se ha' mais dados
         System.out.print ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = Integer.parseInt ( System.console( ).readLine( ) );
      }
      while ( Resposta == 1 );
   // 3. calcular o valor medio
      if ( N == 0 )
      {
      // nao houve dados
         System.out.println ( "\nNao houve dados" );
      }
      else
      {
      // houve dados, calcular a media
         MEDIA = SOMA / N;
      } // fim se houve dados
   // 4. mostrar resultado
      System.out.println ( "\nValor medio = " + MEDIA + " [ohms]" );
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_8 class
```

Programa em Python:

```
# Exemplo 8
# Dados valores de resistores, calcular o valor medio.
#
# 1. definir dados
R        = 0.0; # resistor
SOMA     = 0.0; # somatorio de valores
MEDIA    = 0.0; # valor medio
N        = 0;   # numero de elementos
Resposta = 0;   # controle da repeticao
# 2. repetir ate' parar
Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
while ( Resposta == 1 ):
  # 2.1.1 ler dado valido do teclado
    R = float ( input ( "\nR = " ) );   # ler valor
    while ( R <= 0 ):
      R = float ( input ( "\nR = " ) ); # ler valor
    # ( R<=0 )
  # 2.2. acumular valores
    SOMA = SOMA + R;
    N = N + 1; # mais um dado valido
  # 2.3. verificar se ha' mais dados
    Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
# enquanto houver dados
# 3. calcular o valor medio
if ( N == 0 ):
  # nao houve dados
     print ( "\nNao houve dados" );
else:
  # houve dado, calcular a media
     MEDIA = SOMA / N;
# fim se houve dados
# 4. mostrar resultado
print  ( "\nValor medio= ", MEDIA, " [ohms]" );
# pausa para terminar
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um número indeterminado de dados, contendo cada um, a idade de um indivíduo;
   - o último dado, não entrará nos cálculos, e conterá o valor da idade igual a zero;
   - calcular e mostrar a idade média deste grupo de indivíduos.

2. Fazer um algoritmo para :
   - ler um número indeterminado de dados, contendo cada um, a idade de um indivíduo;
   - sabendo-se que o último dado conterá o valor zero e não entrará nos cálculos,
   - calcular e mostrar a idade média dos maiores que 18 anos.

3. Fazer um algoritmo para :
   - ler um conjunto de dados contendo, cada um, uma nota;
   - determinar e mostrar a média dos valores maiores que 60 pontos;
   - o último dado, e que não será processado, conterá a nota = 999.

4. Fazer um algoritmo para :
   - ler um número indeterminado de valores inteiros positivos,
   - o último dado, que não será processado, conterá o valor 9999;
   - calcular e mostrar a soma dos valores pares e a soma dos valores ímpares.

5. Fazer um algoritmo para :
   - ler um número indeterminado de valores inteiros,
   - o último dado, que não será processado, conterá o valor 9999;
   - calcular e mostrar a diferença entre o valor médio negativo e o valor médio positivo.

Exemplo 9.

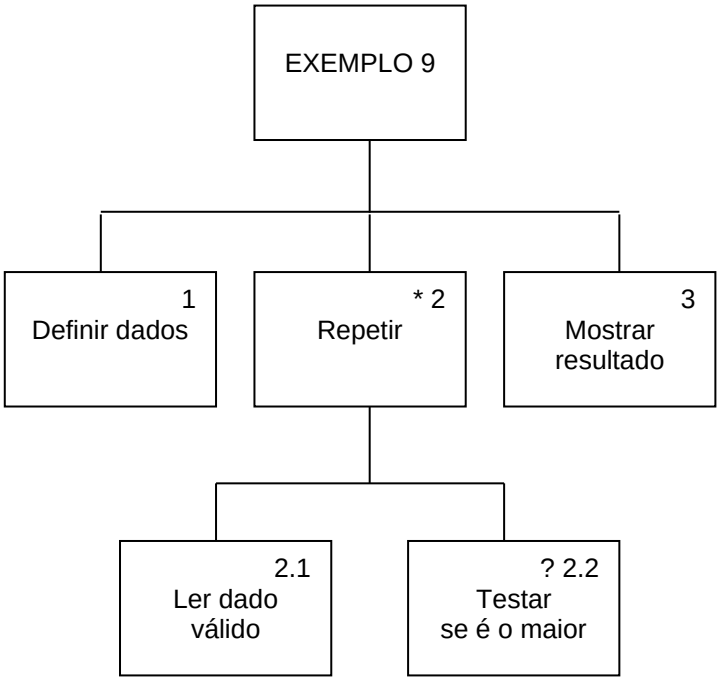
Fazer um algoritmo para:
- ler um número indeterminado de valores de resistores testados em laboratório;
- calcular o maior valor desta amostra.

Análise de dados:

- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R	real		resistor > 0 (válido)
MAIOR	real	0.0	maior valor

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
10.00 [ohms]	
10.04 [ohms]	
10.01 [ohms]	
10.05 [ohms]	
10.00 [ohms]	
09.96 [ohms]	
10.00 [ohms]	
09.95 [ohms]	
09.99 [ohms]	
10.00 [ohms]	Maior valor = 10.05 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 9	v.1
Ação	Bloco
! definir dados	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! testar se é o maior	2.2
! mostrar resultado	3

Segunda versão, refinar o primeiro bloco.

Exemplo 9	v.2
Ação	Bloco
! definir dados real R, ! resistor MAIOR ← 0.0; ! maior valor	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! testar se é o maior	2.2
! mostrar resultado	3

Terceira versão, refinar o segundo bloco.

Exemplo 9	v.3
Ação	Bloco
! definir dados real R, ! resistor MAIOR ← 0.0; ! maior valor	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
tela ← "\nR="; R ← teclado; ! ler valor	
R ≤ 0 ?	
! testar se é o maior	2.2
R>MAIOR ? | V | MAIOR ← R; ! guardar o novo	
enquanto houver dados	
! mostrar resultado	3

Quarta versão, refinar novamente o segundo bloco.

Exemplo 9	v.4
Ação	Bloco
! definir dados real R, ! resistor MAIOR ← 0.0; ! maior valor inteiro Resposta; ! controle da repetição	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
tela ← "\nR="; R ← teclado; ! ler valor	
R ≤ 0 ?	
! testar se é o maior	2.2
R>MAIOR ? | V | MAIOR ← R; ! guardar o novo	
! verificar se há mais dados tela ← “\nMais dados (Sim=1,Não=0) ?”; Resposta ← teclado;	2.3
Resposta = 1 ?	
! mostrar resultado	3

Quinta versão, refinar o terceiro bloco.

Exemplo 9	v.5
Ação	Bloco
! definir dados real R, ! resistor MAIOR ← 0.0; ! maior valor inteiro Resposta; ! controle da repetição	1
! repetir enquanto houver dados	2
! ler dado válido do teclado tela ← "\nR="; R ← teclado; ! ler valor R ≤ 0 ?	2.1
! testar se é o maior	2.2
R>MAIOR ? | V | MAIOR ← R; ! guardar o novo	
! verificar se há mais dados tela ← "\nMais dados (Sim=1,Não=0) ?"; Resposta ← teclado;	2.3
Resposta = 1 ?	
! mostrar resultado tela ← ("\nMaior valor = ", Maior, " [ohms]");	3

Sexta versão, refinar novamente o segundo bloco.

Exemplo 9	v.6
Ação	Bloco
! definir dados real R, ! resistor MAIOR ← 0.0; ! maior valor inteiro Resposta; ! controle da repetição	1
! repetir enquanto houver dados	2
repetir até (Resposta ≠ 1) ! 2.1 ler dado válido do teclado repetir até (R > 0) tela ← "\nR="; R ← teclado; ! ler valor fim repetir ! enquanto (R ≤ 0) ! 2.2 testar se é o maior se (R>MAIOR) MAIOR = R; ! guardar o novo fim se ! maior ! 2.3 verificar se há mais dados tela ← "\nMais dados (Sim=1,Não=0) ?"; Resposta ← teclado; fim repetir ! enquanto (Resposta = 1)	
! mostrar resultado tela ← ("\nMaior valor = ", Maior, " [ohms]");	3

Programa em SCILAB:

```
// Exemplo 9
// Dados valores de resistores, calcular o maior valor.
//
// 1. definir dados
   R           = 0.0; // resistor
   MAIOR    = 0.0, // maior valor
   Resposta = 0;    // controle da repeticao
// 2. repetir enquanto houver dados
  clc;                  // limpar a area de trabalho
  Resposta = input ( “\nMais dados (Sim=1,Nao=0) ? “ );
  while ( Resposta == 1 )
  // 2.1 ler dado valido do teclado
     R = input ( “\nR “ );   // ler valor
     while ( R <= 0 )
        R = input ( “\nR “ ); // ler valor
     end // ( R <= 0 )
  // 2.2. testar se e’ o maior
     if ( R > MAIOR )
       MAIOR = R;
     end // fim do teste se maior
  // 2.3. verificar se ha’ mais dados
     Resposta = input ( “\nMais dados (Sim=1,Nao=0) ? “ );
  end // enquanto houver dados
// 3. mostrar resultado
  printf ( “\nMaior valor = %f [ohms]”, MAIOR );
// pausa para terminar
  printf ( “\nPressionar ENTER para terminar.\n” );
  halt;
// fim do programa
```

Programa em C:

```
// Exemplo 9
// Dados valores de resistores, calcular o maior valor.
//
// bibliotecas necessárias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R,                  // resistor
        MAIOR = 0.0, // maior valor
  int   Resposta;       // controle da repeticao
// 2. repetir enquanto houver dados
  do
  {
  // 2.1. ler dado valido do teclado
    do
    {
    // 2.1 ler um valor do teclado
      printf ( "\nR=" );
      scanf ( "%f", &R ); // ler valor
    }
    while ( R <= 0 );
  // 2.2. testar se é o maior
    if ( R > MAIOR )
    {
      MAIOR = R; // guardar o novo
    } // fim do teste do maior
  // 2.3. verificar se ha' mais dados
    printf ( "\nMais dados (Sim=1,Não=0) ? " ) ;
    scanf ( "%d", &Resposta );
  }
  while ( Resposta == 1 );
// 3. mostrar resultado
  printf ( "\nMaior valor = %f %s", MAIOR, " [ohms]";
// pausa para terminar
  printf ( "Pressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 9
// Dados valores de resistores, calcular o maior valor.
//
// bibliotecas necessárias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R,                // resistor
          MAIOR = 0.0, // maior valor
   int    Resposta;     // controle da repeticao
// 2. repetir enquanto houver dados
   do
   {
   // 2.1. ler dado valido do teclado
      do
      {
      // 2.1 ler um valor do teclado
         cout << "\nR=";
         cin >> R; // ler valor
      }
      while ( R <= 0 );
   // 2.2. testar se é o maior
      if ( R > MAIOR )
      {
         MAIOR = R; // guardar o novo
      } // fim do teste do maior
   // 2.3. verificar se ha' mais dados
      cout << "\nMais dados (Sim=1,Não=0) ?";
      cin >> Resposta;
   }
   while ( Resposta == 1 );
// 3. mostrar resultado
   cout << "\nMaior valor = " << MAIOR << " [ohms]";
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 9
 * Dados valores de resistores, calcular o valor medio.
 */
using System;

class Exemplo_9
{
  public static void Main ( )
  {
  // 1. definir dados
    double R,                  // resistor
           MAIOR = 0.0, // maior valor
           MEDIA = 0.0; // resistor equivalente
    int    Resposta;       // controle da repeticao
  // 2. repetir enquanto houver dados
    do
    {
    // 2.1. ler dado valido do teclado
      do
      {
        Console.Write ( "\nR=" );
        R = int.Parse ( Console.ReadLine ( ) ); // ler valor
      }
      while ( R <= 0 );
    // 2.2. testar se e' o maior
      if ( R > MAIOR )
      {
       MAIOR = R; // guardar o novo maior
      } // fim do teste do maior
    // 2.3. verificar se ha' mais dados
      Console.WriteLine ( "\nMais dados (Sim=1,Nao=0) ? " );
      Resposta = int.Parse ( Console.ReadLine ( ) );
    }
    while ( Resposta == 1 );
  // 3. mostrar resultado
    Console.WriteLine ( "\nMaior valor = " + MAIOR + " [ohms]" );
  // pausa para terminar
    Console.Write ( "\nPressionar ENTER para terminar." );
    Console.ReadLine ( );
  } // end Main ( )

} // fim Exemplo_9 class
```

Programa em Java:

```
/**
  * Exemplo 9
  * Dados valores de resistores, calcular o maior valor.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_9
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double  R,                    // resistor
              MAIOR = 0.0,          // maior valor
              MEDIA = 0.0;          // resistor equivalente
      int     Resposta;             // controle da repeticao
    // 2. repetir enquanto houver dados
      do
      {
      // 2.1. ler dado valido do teclado
         do
         {
           System.out.print ( "\nR = " );  // ler valor
           R = Integer.parseInt ( System.console( ).readLine( ) );
         }
         while ( R <= 0 );
      // 2.2. testar se e' o maior
         if ( R > MAIOR )
         {
           MAIOR = R; // guardar o novo maior
         } // fim do teste do maior
      // 2.3. verificar se ha' mais dados
         System.out.print ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = Integer.parseInt ( System.console( ).readLine( ) );
      }
      while ( Resposta == 1 );
    // 3. mostrar resultado
      System.out.println ( "\nMaior valor = " + MAIOR + " [ohms]" );
    // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )

} // fim Exemplo_9 class
```

Programa em Python:

```
# Exemplo 9
# Dados valores de resistores, calcular o maior valor.
#
# 1. definir dados
R           = 0.0; # resistor
MAIOR    = 0.0; # maior valor
Resposta = 0;    # controle da repeticao
# 2. repetir enquanto houver dados
Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
while ( Resposta == 1 ):
  # 2.1 ler dado valido do teclado
    R = float ( input ( "\nR = " ) );     # ler valor
    while ( R <= 0 ):
        R = float ( input ( "\nR = " ) ); # ler valor
    # ( R <= 0 )
  # 2.2. testar se e' o maior
    if ( R > MAIOR ):
       MAIOR = R;
    # fim do teste se maior
  # 2.3. verificar se ha' mais dados
    Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
# enquanto houver dados
# 3. mostrar resultado
print ( "\nMaior valor = ", MAIOR, " [ohms]" );
# pausa para terminar
print ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um número indeterminado de dados, contendo cada um, a idade de um indivíduo;
   - o último dado, não entrará nos cálculos, e conterá o valor da idade igual a zero;
   - calcular e mostrar a maior idade neste grupo de indivíduos.

2. Fazer um algoritmo para :
   - ler um número indeterminado de dados, contendo cada um, a idade de um indivíduo;
   - o último dado, não entrará nos cálculos, e conterá o valor da idade igual a zero;
   - calcular e mostrar a menor idade neste grupo de indivíduos.

3. Fazer um algoritmo para :
   - ler um conjunto de dados contendo, cada um, uma nota;
   - determinar e mostrar quantas notas são iguais a 60 pontos;
   - o último dado, e que não será processado, contém nota = 999.

4. Fazer um algoritmo para :
   - ler um número indeterminado de dados;
   - cada dado possui um valor, o último dado, e que não será processado, contém o valor 9999;
   - calcular e mostrar os dois maiores valores lidos.

5. Modificar o algoritmo anterior de forma a :
   - ler um número indeterminado de dados;
   - cada dado possui um valor, mas só serão válidos os valores maiores que zero,
   - o último dado, e que não será processado, contém o valor 9999;
   - calcular e mostrar os dois maiores valores lidos.

Exemplo 10.

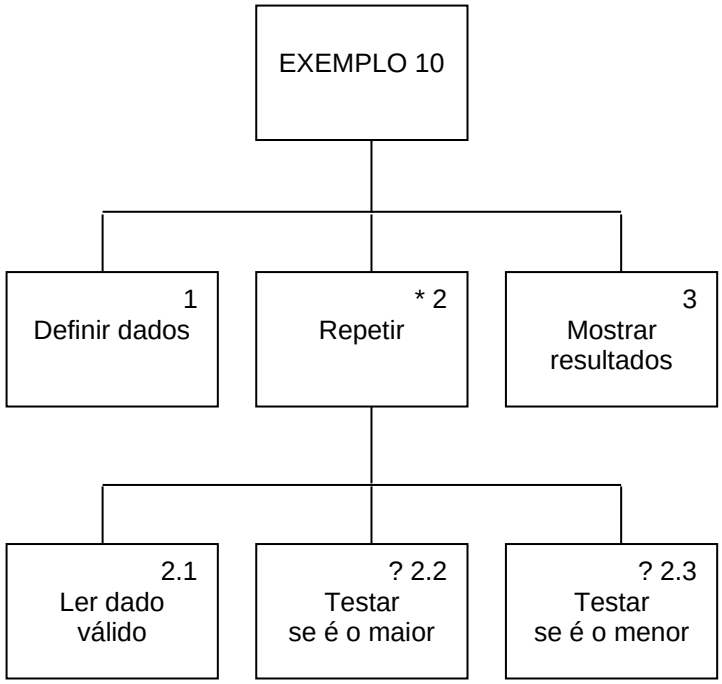
Fazer um algoritmo para:
- ler um número indeterminado de valores de resistores testados em laboratório;
- calcular o maior e o menor valor desta amostra.

Análise de dados:

- Dados do problema :

Dado	Tipo	Valor Inicial	Obs.
R	real		resistor > 0 (válido)
MAIOR	real	primeiro lido	maior valor
MENOR	real	primeiro lido	menor valor

Diagrama funcional:



- Avaliação da solução :

  - Para teste podem ser usados os seguintes valores:

Dados	Resultado
10.00 [ohms]	
10.04 [ohms]	
10.01 [ohms]	
10.05 [ohms]	
10.00 [ohms]	
09.96 [ohms]	
10.00 [ohms]	
09.95 [ohms]	
09.99 [ohms]	
10.00 [ohms]	Maior valor = 10.05 [ohms]
	Menor valor= 09.95 [ohms]

Algoritmo:

Esboço:

Primeira versão, só comentários.

Exemplo 10	v.1
Ação	Bloco
! definir dados	1
! ler primeiro valor	1.1
! usar o dado lido como valor inicial	1.2
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! testar se é o maior	2.2
! testar se é o menor	2.3
! mostrar resultado	3

Segunda versão, refinar o primeiro bloco.

Exemplo 10	v.2
Ação	Bloco
! definir dados real R, ! resistor MAIOR, ! maior valor MENOR; ! menor valor	1
! ler o primeiro valor tela ← "\nQual o primeiro valor ?"; R ← teclado; ! supor válido	1.1
! usar dado lido como valor inicial MAIOR ← R; MENOR ← R;	1.2
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! testar se é o maior	2.2
! testar se é o menor	2.3
! mostrar resultado	3

Terceira versão, refinar o segundo bloco.

Exemplo 10	v.3
Ação	Bloco
! definir dados real R, ! resistor MAIOR, ! maior valor MENOR; ! menor valor ! 1.1 ler o primeiro valor tela ← "\nQual o primeiro valor ?"; R ← teclado; ! supor válido ! 1.2 usar dado lido como valor inicial MAIOR ← R; MENOR ← R;	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
! testar se é o maior	2.2
! testar se é o menor	2.3
até parar	
! mostrar resultado	3

Quarta versão, refinar novamente o segundo bloco.

Exemplo 10			v.4
Ação			Bloco
! definir dados real R, ! resistor MAIOR, ! maior valor MENOR; ! menor valor ! 1.1 ler o primeiro valor tela ← "\nQual o primeiro valor ?"; R ← teclado; ! supor válido ! 1.2 usar dado lido como valor inicial MAIOR ← R; MENOR ← R;			1
! repetir enquanto houver dados			2
! ler dado válido do teclado			2.1
tela ← "\nR="; R ← teclado; ! ler valor			
R ≤ 0 ?			
! testar se é o maior			2.2
R>MAIOR?	V	MAIOR ← R;	
	F	! testar se é o menor	2.3
		R < MENOR ? V MENOR←R;	
! verificar se há mais dados tela ← "\nMais dados (Sim=1,Não=0) ?"; Resposta ← teclado;			2.4
Resposta = 1 ?			
! mostrar resultado			3

Quinta versão, refinar o terceiro bloco.

Exemplo 10	v.5
Ação	Bloco
! definir dados real R, ! resistor MAIOR, ! maior valor MENOR; ! menor valor ! 1.1 ler o primeiro valor tela ← "\nQual o primeiro valor ?"; R ← teclado; ! supor válido ! 1.2 usar dado lido como valor inicial MAIOR ← R; MENOR ← R;	1
! repetir enquanto houver dados	2
! ler dado válido do teclado	2.1
tela ← "\nR="; R ← teclado; ! ler valor	
R ≤ 0 ?	
! testar se é o maior	2.2
R>MAIOR? V: MAIOR ← R;	
R>MAIOR? F: ! testar se é o menor	2.3
R < MENOR ? V: MENOR←R;	
! verificar se há mais dados tela ← "\nMais dados (Sim=1,Não=0) ?"; Resposta ← teclado;	2.4
Resposta = 1 ?	
! mostrar resultado tela ← ("\nMaior valor = ", MAIOR , " [ohms]"); tela ← ("\nMenor valor = ", MENOR, " [ohms]");	3

Sexta versão, refinar novamente o segundo bloco.

Exemplo 10	v.6
Ação	Bloco

! definir dados
  real R, ! resistor
  MAIOR, ! maior valor
  MENOR; ! menor valor
! 1.1 ler o primeiro valor
  tela ← “\nQual o primeiro valor ?”;
  R ← teclado; ! supor válido
! 1.2 usar dado lido como valor inicial
  MAIOR ← R;
  MENOR ← R;
	1

! repetir enquanto houver dados	2

repetir até (Resposta != 1)
  ! 2.1 ler dado válido do teclado
  repetir até (R > 0)
  tela ← "\nR=";
  R ← teclado; ! ler valor
  fim repetir ! enquanto (R ≤ 0)
  ! 2.2 testar se é o maior
  se (R>MAIOR)
  MAIOR ← R; ! guardar o novo maior
  senão
  ! 2.3 testar se é o menor
  se (R < MENOR)
  MENOR ← R; ! guardar o novo menor
  fim se ! menor
  fim se ! maior
  ! 2.4 verificar se há mais dados
  tela ← “\nMais dados (Sim=1,Não=0) ?”;
  Resposta ← teclado;
  fim repetir ! enquanto (Resposta = 1)	

! mostrar resultado
  tela ← (“\nMaior valor = “, MAIOR , “ [ohms]”);
  tela ← (“\nMenor valor = “, MENOR, “ [ohms]”);
	3

Programa em SCILAB:

```
// Exemplo 10
// Dados valores de resistores, calcular o maior valor.
//
// 1. definir dados
   R          = 0.0; // resistor
   MAIOR    = 0.0; // maior valor
   MENOR   = 0.0 // menor valor
   Resposta = 0;   // controle da repeticao
//
// 1.1 ler primeiro valor valido
  clc;                  // limpar a area de trabalho
  R = input ( “\nQual o primeiro valor ? “ ); // ler valor
  while ( R <= 0 )
    R = input ( “\nR “ ); // ler valor
  end // ( R <= 0 )
// 1.2 usar dado lido como valor inicial
  MAIOR  = R;
  MENOR = R;
// 2. repetir enquanto houver dados
  Resposta = input ( “\nMais dados (Sim=1,Nao=0) ? “ );
  while ( Resposta == 1)
  // 2.1 ler dado valido do teclado
    R = input ( “\nR “ );    // ler valor
    while ( R<=0 )
      R = input ( “\nR “ ); // ler valor
    end // ( R <= 0 )
  // 2.2 testar se e' o maior
    if ( R > MAIOR )
      MAIOR = R;           // guardar o novo maior
    else
    // 2.3 testar se e’ o menor
      if ( R < MENOR )
        MENOR=R;       // guardar o novo menor
       end // fim do teste do menor
    end // fim do teste se maior
  // 2.4 verificar se ha' mais dados
    Resposta = input ( “\nMais dados (Sim=1,Nao=0) ? “ );
  end // enquanto houver dados
// 3. mostrar resultado
  printf ( “\nMaior  valor = %f [ohms]”, MAIOR  );
  printf ( “\nMenor valor = % [ohms]”, MENOR );
// pausa para terminar
  printf ( “\nPressionar ENTER para terminar.\n” );
  halt;
// fim do programa
```

Programa em C:

```
// Exemplo 10a
// Dados valores de resistores, calcular o maior e o menor valor.
//
// bibliotecas necessarias
#include <stdio.h>
#include <stdlib.h>
//
int main (void)
{
// 1. definir dados
  float R,             // resistor
        MAIOR,     // maior valor
        MENOR;   // menor valor
  int   Resposta;  // controle da repeticao
// 1.1 ler o primeiro valor
  printf ( “\nQual o primeiro valor ?” );
  scanf ( "%f", &R );       // supor válido
// 1.2 usar dado lido como valor inicial
  MAIOR  = R;
  MENOR = R;
// 2. repetir
  Resposta = 1;
  while ( Resposta == 1 )
  {
  // 2.1. ler dado valido do teclado
    do
    {
      printf ( "\nR=" );
      scanf ( "%f", &R ); // ler valor
    }
    while ( R <= 0 );
  // 2.2. testar se é o maior
    if ( R>MAIOR )
    {
     MAIOR  = R;      // guardar o novo maior
    }
    else
    {
    // 2.3. testar se e’ o menor
      if ( R < MENOR )
      {
        MENOR = R;   // guardar o novo menor
      } // fim do teste do menor
    } // fim do teste do maior
  // 2.4. verificar se ha’ mais dados
    printf ( “\nMais dados (Sim=1,Não=0) ? ” );
    scanf ( "%f", &Resposta );
  }
// 3. mostrar resultados
  printf ( “\nMaior  valor = %f %s“, MAIOR, “ [ohms]” );
  printf ( “\nMenor valor = %f %s“, MENOR, “ [ohms]” );
// pausa para terminar
  printf ( "Pressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C:

```
// Exemplo 10b
// Dados valores de resistores, calcular o maior e o menor valor.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
  float R,            // resistor
        MAIOR,    // maior valor
        MENOR;  // menor valor
  int    Resposta; // controle da repeticao

// 1.1 ler primeiro valor valido
  do
  {
    printf ( “\nQual o primeiro valor ?” );
    scanf ( "%f", &R );       // ler apenas valor válido
  }
  while ( R <= 0 );
// 1.2 usar dado lido como valor inicial
  MAIOR  = R;      MENOR = R;
// 2. repetir
  do
  {
  // 2.1. ler dado valido do teclado
    do
    {
      printf ( "\nR=" );
      scanf ( "%f", &R ); // ler valor
    }
    while ( R <= 0 );
  // 2.2. testar se e’ o maior
    if ( R>MAIOR )
    {   MAIOR  = R;   }       // guardar o novo maior
    else
    {
    // 2.3. testar se e’ o menor
      if ( R < MENOR )
      {   MENOR = R;   }   // guardar o novo menor
    } // fim do teste do maior
  // 2.4. verificar se ha’ mais dados
    printf ( “\nMais dados (Sim=1,Não=0) ? ” );
    scanf ( "%f", &Resposta );
  }
  while ( Resposta == 1 );
// 3. mostrar resultado
  printf ( “\nMaior  valor = %f %s“, MAIOR, “ [ohms]” );
  printf ( “\nMenor valor = %f %s“, MENOR, “ [ohms]” );
// pausa para terminar
  printf ( "Pressionar ENTER para terminar." );
  getchar ( );
  return ( 0 );
} // fim do programa
```

Programa em C++:

```
// Exemplo 10a
// Dados valores de resistores, calcular o maior e o menor valor.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
   double R,             // resistor
          MAIOR,     // maior valor
          MENOR;   // menor valor
   int       Resposta;  // controle da repeticao
// 1.1 ler o primeiro valor
   cout << “\nQual o primeiro valor ?”;
   cin   >> R;       // supor válido
// 1.2 usar dado lido como valor inicial
   MAIOR  = R;
   MENOR = R;
// 2. repetir
   Resposta = 1;
   while ( Resposta == 1 )
   {
   // 2.1. ler dado valido do teclado
      do
      {
         cout << "\nR=";
         cin   >> R; // ler valor
      }
      while ( R <= 0 );
   // 2.2. testar se é o maior
      if ( R>MAIOR )
      {
        MAIOR  = R;      // guardar o novo maior
      }
      else
      {
      // 2.3. testar se e’ o menor
         if ( R < MENOR )
         {
           MENOR = R;   // guardar o novo menor
         } // fim do teste do menor
      } // fim do teste do maior
   // 2.4. verificar se ha’ mais dados
      cout << “\nMais dados (Sim=1,Não=0) ? ”;
      cin   >> Resposta;
   }
// 3. mostrar resultados
   cout << “\nMaior valor = “  << MAIOR   << “ [ohms]”;
   cout << “\nMenor valor = “ << MENOR << “ [ohms]”;
// pausa para terminar
   cout << "Pressionar ENTER para terminar.";
   cin.get ( );
   return EXIT_SUCCESS;
} // fim do programa
```

Programa em C++:

```
// Exemplo 10b
// Dados valores de resistores, calcular o maior e o menor valor.
//
// bibliotecas necessarias
#include <iostream>
using namespace std;
//
int main (void)
{
// 1. definir dados
  float R,            // resistor
        MAIOR,    // maior valor
        MENOR;  // menor valor
  int    Resposta; // controle da repeticao

// 1.1 ler primeiro valor valido
  do
  {
   cout << "\nQual o primeiro valor ?";
   cin   >> R; // ler apenas valor válido
  }
  while ( R <= 0 );
// 1.2 usar dado lido como valor inicial
  MAIOR  = R;      MENOR = R;
// 2. repetir
  do
  {
  // 2.1. ler dado valido do teclado
    do
    {
      cout << "\nR=";
      cin   >> R; // ler valor
    }
    while ( R <= 0 );
  // 2.2. testar se e' o maior
    if ( R>MAIOR )
    {   MAIOR  = R;   }       // guardar o novo maior
    else
    {
    // 2.3. testar se e' o menor
      if ( R < MENOR )
      {   MENOR = R;   }   // guardar o novo menor
    } // fim do teste do maior
  // 2.4. verificar se ha' mais dados
    cout << "\nMais dados (Sim=1,Não=0) ? ";
    cin   >> Resposta;
  }
  while ( Resposta == 1 );
// 3. mostrar resultado
  cout << "\nMaior valor = "  << MAIOR   << " [ohms]";
  cout << "\nMenor valor = " << MENOR << " [ohms]";
// pausa para terminar
  cout << "Pressionar ENTER para terminar.";
  cin.get ( );
  return EXIT_SUCCESS;
} // fim do programa
```

Programa em C#:

```
/*
 * Exemplo 10a
 * Dados valores de resistores, calcular o valor medio.
 */
using System;

class Exemplo_10a
{
  public static void Main ( )
  {
  // 1. definir dados
    double R,              // resistor
           MAIOR,          // maior valor
           MENOR;          // resistor equivalente
    int    Resposta;       // controle da repeticao
  // 1.1. ler o primeiro valor
    Console.Write ( "\n o primeiro valor ? " );
    R = int.Parse ( Console.ReadLine ( ); // ler primeiro valor
  // 1.2. usar dado lido como valor inicial
    MAIOR = R;   MENOR = R;
  // 2. repetir
    Resposta = 1;
    while ( Resposta == 1 )
    {
    // 2.1. ler dado valido do teclado
      do
      {
        Console.Write ( "\nR=" );
        R = int.Parse  ( Console.ReadLine ( ) ); // ler valor
      }
      while ( R <= 0 );
    // 2.2. testar se e' o maior
      if ( R > MAIOR )
      {   MAIOR = R;   }   // guardar o novo maior
      }
      else
      {
      // 2.3. testar se e' o menor
        if ( R < MENOR )
        {   MENOR = R;   }   // guardar o novo menor
      }// fim do teste do maior
    // 2.3. verificar se ha' mais dados
      Console.WriteLine ( "\nMais dados (Sim=1,Nao=0) ? " );
      Resposta = int.Parse ( Console.ReadLine ( ) );
    }
  // 3. mostrar resultado
    Console.WriteLine ( "\nMaior  valor = " + MAIOR  + " [ohms]" );
    Console.WriteLine ( "\nMenor valor = " + MENOR + " [ohms]" );
  // pausa para terminar
    Console.Write ( "\nPressionar ENTER para terminar." );
    Console.ReadLine ( );
  } // end Main ( )

} // fim Exemplo_10a class
```

Outra versão do programa em C#:

```
/*
 * Exemplo 10b
 * Dados valores de resistores, calcular o valor medio.
 */
using System;

class Exemplo_10b
{
   public static void Main ( )
   {
   // 1. definir dados
      double R,              // resistor
             MAIOR,          // maior valor
             MENOR;          // resistor equivalente
      int    Resposta;       // controle da repeticao
   // 1.1. ler o primeiro valor
      Console.Write ( "\n o primeiro valor ? " );
      R = int.Parse  ( Console.ReadLine ( ) ); // ler primeiro valor
   // 1.2. usar dado lido como valor inicial
      MAIOR = R;   MENOR = R;
   // 2. repetir
      do
      {
      // 2.1. ler dado valido do teclado
         do
         {
            Console.Write ( "\nR=" );
            R = int.Parse  ( Console.ReadLine ( ) ); // ler valor
         }
         while ( R <= 0 );
      // 2.2. testar se e' o maior
         if ( R > MAIOR )
         {
           MAIOR = R; // guardar o novo maior
         }
         else
         {
         // 2.3. testar se e' o menor
            if ( R < MENOR )
            {
              MENOR = R; // guardar o novo menor
            } // fim do teste do menor
         }// fim do teste do maior
      // 2.3. verificar se ha' mais dados
         Console.WriteLine ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = int.Parse ( Console.ReadLine ( ) );
      }
      while ( Resposta == 1 );
   // 3. mostrar resultado
      Console.WriteLine ( "\nMaior  valor = " + MAIOR  + " [ohms]" );
      Console.WriteLine ( "\nMenor valor = " + MENOR + " [ohms]" );
   // pausa para terminar
      Console.Write ( "\nPressionar ENTER para terminar." );
      Console.ReadLine ( );
   } // end Main ( )
} // fim Exemplo_10b class
```

Programa em Java:

```
/**
  * Exemplo 10a
  * Dados valores de resistores, calcular o maior e o menor valor.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_10a
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R,                        // resistor
             MAIOR,                    // maior  valor
             MENOR;                    // menor valor
      int    Resposta;                 // controle da repeticao
   // 1.1. ler o primeiro valor
      System.out.print ( "\nR = " );   // ler valor
      R = Integer.parseInt ( System.console( ).readLine( ) );
   // 1.2. usar dado lido como valor inicial
      MAIOR  = R;
      MENOR = R;
   // 2. repetir
      Resposta = 1;
      while ( Resposta == 1 )
      {
      // 2.1. ler dado valido do teclado
         do
         {
           System.out.print ( "\nR = " ); // ler valor
           R = Integer.parseInt ( System.console( ).readLine( ) );
         }
         while ( R <= 0 );
      // 2.2. testar se e' o maior
         if ( R > MAIOR )
         {    MAIOR = R;     }           // guardar o novo maior
         else
         {
          // 2.3. testar se e' o menor
            if ( R < MENOR )
            {    MENOR = R;     }       // guardar o novo menor
         }// fim do teste do maior
      // 2.3. verificar se ha' mais dados
         System.out.print ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = Integer.parseInt ( System.console( ).readLine( ) );
      }
   // 3. mostrar resultado
      System.out.println ( "\nMaior v alor = " + MAIOR   + " [ohms]" );
      System.out.println ( "\nMenor valor = " + MENOR + " [ohms]" );
   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )
} // fim Exemplo_10a class
```

Outra versão do programa em Java:

```
/**
  * Exemplo 10b
  * Dados valores de resistores, calcular o maior e o menor valor.
  */

// ---------------------------------------------- classes necessarias

// ---------------------------------------------- definicao de classe

class Exemplo_10b
{
   public static void main ( String [ ] args )
   {
   // 1. definir dados
      double R,                        // resistor
             MAIOR,                    // maior valor
             MENOR;                    // resistor equivalente
      int    Resposta;                 // controle da repeticao

   // 1.1. ler primeiro valor valido
      do
      {
        System.out.print ( "\nR = " );    // ler valor
        R = Integer.parseInt ( System.console( ).readLine( ) );
      }
      while ( R <= 0 );
   // 1.2. usar dado lido como valor inicial
      MAIOR  = R;
      MENOR = R;
   // 2. repetir
      do
      {
      // 2.1. ler dado valido do teclado
         do
         {
           System.out.print ( "\nR = " ); // ler valor
           R = Integer.parseInt ( System.console( ).readLine( ) );
         }
         while ( R <= 0 );
      // 2.2. testar se e' o maior
         if ( R > MAIOR )
         {
           MAIOR = R;                    // guardar o novo maior
         }
         else
         {
         // 2.3. testar se e' o menor
            if ( R < MENOR )
            {
              MENOR = R; // guardar o novo menor
            } // fim do teste do menor
         }// fim do teste do maior
      // 2.3. verificar se ha' mais dados
         System.out.print ( "\nMais dados (Sim=1,Nao=0) ? " );
         Resposta = Integer.parseInt ( System.console( ).readLine( ) );
      }
      while ( Resposta == 1 );
```

```
   // 3. mostrar resultado
      System.out.println ( "\nMaior  valor = " + MAIOR    + " [ohms]" );
      System.out.println ( "\nMenor valor = " + MENOR + " [ohms]" );

   // pausa para terminar
      System.out.print ( "\nPressionar ENTER para terminar." );
      System.console( ).readLine( );
   } // end main ( )
} // fim Exemplo_10b class
```

Programa em Python:

```
# Exemplo 10
# Dados valores de resistores, calcular o maior valor.
#
# 1. definir dados
R           = 0.0;          # resistor
MAIOR    = 0.0;          # maior valor
MENOR  = 0.0            # menor valor
Resposta =  0;          # controle da repeticao
#
# 1.1 ler primeiro valor valido
R = float ( input ( "\nQual o primeiro valor ? " ) ); # ler valor
while ( R <= 0 ):
    R = float ( input ( "\nR = " ) ); # ler valor
# ( R <= 0 )
# 1.2 usar dado lido como valor inicial
MAIOR  = R;
MENOR = R;
# 2. repetir enquanto houver dados
Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
while ( Resposta == 1):
  # 2.1 ler dado valido do teclado
    R = float ( input ( "\nR = " ) );    # ler valor
    while ( R<=0 ):
       R = float ( input ( "\nR = " ) ); # ler valor
    # ( R <= 0 )
  # 2.2 testar se e' o maior
    if ( R > MAIOR ):
       MAIOR = R;          # guardar o novo maior
    else:
  # 2.3 testar se e' o menor
       if ( R < MENOR ):
          MENOR=R;        # guardar o novo menor
       # fim do teste do menor
    # fim do teste se maior
  # 2.4 verificar se ha' mais dados
    Resposta = int ( input ( "\nMais dados (Sim=1,Nao=0) ? " ) );
# enquanto houver dados
# 3. mostrar resultado
print  ( "\nMaior valor  = ", MAIOR  , " [ohms]" );
print  ( "\nMenor valor = ", MENOR, " [ohms]" );
# pausa para terminar
print  ( "\nPressionar ENTER para terminar.\n" );
input ( );
# fim do programa
```

Exercícios

1. Fazer um algoritmo para :
   - ler um conjunto de dados contendo, cada um, uma nota;
   - determinar e mostrar a maior e a menor nota da turma;
   - o último dado, e que não será processado, contém nota = 999.

2. Fazer um algoritmo para :
   - ler um conjunto de dados contendo, cada um, uma nota;
   - determinar e mostrar as duas maiores e as duas menores notas da turma;
   - o último dado, e que não será processado, contém nota = 999.

3. Fazer um algoritmo para :
   - ler um número indeterminado de dados;
   - cada dado possui um valor, o último dado, e que não será processado, contém o valor 9999;
   - calcular e mostrar os dois maiores valores lidos, e que sejam diferentes.

4. Modificar o algoritmo anterior de forma a :
   - ler um valor (N) do teclado;
   - calcular e mostrar os dois maiores e o menor valor entre (N) outros valores lidos do teclado.

5. Fazer um algoritmo para :
   - ler um conjunto de 50 dados contendo, cada um, a altura e um código para masculino (1), e outro para feminino (2);
   - calcular e mostrar :
     - a maior e a menor altura da turma;
     - a média de altura das mulheres;
     - a média de altura da turma.

Exercícios propostos

1. Há três candidatos a uma vaga no senado. Feita a eleição a contagem de votos deverá ser feita através do computador. Fazer um algoritmo para :
   - ler um conjunto de dados contendo, cada um, o voto de um eleitor. O último dado deve conter um valor negativo. Os dados estão organizados segundo o seguinte critério :

     1, 2, 3 - número dos três candidatos, respectivamente;
     0 - voto em branco;
     4 - voto nulo;

   - calcular e mostrar :
     - o número do candidato vencedor e o quantos votos obteve;
     - o número de votos em branco e o número de votos nulos;
     - o número de eleitores que compareceram às urnas.

2. Pode-se calcular a raiz quadrada de um número positivo através do método de aproximação sucessivas de Newton, descrito a seguir :

   - seja "a" o número do qual deseja-se obter a raiz quadrada;
   - a primeira aproximação para a raiz quadrada será dada por :

$$x_1 = a / 2$$

   - a próxima ou sucessiva aproximação é dada por :

$$x_{n+1} = \frac{(x_n^2 + a)}{2x_n}$$

   - fazer um algoritmo para :
   - ler o valor de "a" do teclado;
   - calcular e mostrar a 25ª. aproximação.

3. A conversão de graus Farenheit para Centígrados é obtida por :

$$C = 5 (F - 32) / 9$$

   - fazer um algoritmo para calcular e mostrar uma tabela de graus Centígrados em função de graus Farenheit, que variem de 50 a 150 de 1 em 1.

4. Fazer um algoritmo para gerar e mostrar a seguinte seqüência :

289 - 256 - 225 - 196 -...- 9 - 4 - 1

5. Fazer um algoritmo para calcular e mostrar o valor "s" :

$$s = 1 + 3/2 + 5/3 + 7/4 +...+ 99/50$$

6. Fazer um algoritmo para calcular e mostrar o enésimo termo da série abaixo, onde o valor de (n) é lido do teclado.

5, 6, 11, 12, 17, 18, 23, 24, ...

7. Sendo $s = 1^2 + 2^2 + 3^2 +...+ n^2$, e "k", um número inteiro maior que 1, fazer um algoritmo para calcular e mostrar o maior valor de "n" que torne a relação s < k verdadeira. O valor de "k" será lido do teclado.

8. O valor aproximado de π (PI) pode ser calculado usando a série :

$$s = 1 - 1/3^3 + 1/5^3 - 1/7^3 + 1/9^3 \ldots \qquad \text{e} \qquad \pi = \sqrt[3]{s \cdot 32}$$

Fazer um algoritmo para calcular e mostrar o valor de π usando os 51 primeiros termos da série.

9. Supondo que a população de um país "a" seja de 90.000.000 de habitantes, com uma taxa anual de crescimento de 3//; e que a população de um país "b" seja, aproximadamente, de 200.000.000 de habitantes, com uma taxa anual de crescimento de 1,5//.
Fazer um algoritmo para calcular e mostrar o número de anos necessários para que a população do país "a" ultrapasse ou se igual a população do país "b", mantidas essas taxas de crescimento.

10. O número 3025 possui a seguinte característica :

$$30 + 25 = 55$$
$$55^2 = 3025$$

- fazer um algoritmo para calcular e mostrar todos os números de 4 algarismos que apresentam esta propriedade.

11. O número 1221 possui a propriedade de que lido de "*trás-para-frente*" é igual lido de "*frente-para-trás*". Estes números são chamados de "palíndromos". Calcular e mostrar todos os números palíndromos de 5 algarismos.

12. Calcular e mostrar todos os números palíndromos menores que 30.000 e que sejam quadrados perfeitos.

13. Fazer um algoritmo para :
- ler 1000 dados contendo, cada, o valor de uma nota fiscal;
- calcular e mostrar :
- o número de notas fiscais, cujo valor é menor ou igual a R$1000,00;
- o número de notas fiscais, cujo valor é maior que R$1000,00 e menor ou igual R$2000,00;
- o número das notas fiscais, cujo valor é maior que R$2000,00;
- o total arrecadado durante o mês.

14. Fazer um algoritmo para :
- ler um conjunto de dados contendo, cada um, uma quantidade expressa em milímetros.
- o último dado, que não entrará nos cálculos, conterá essa quantidade igual a zero;
- calcular e mostrar, para cada dado lido, a quantidade correspondente expressa em metros, decímetros, centímetros e milímetros.

Exemplo: 82453 milímetros
82 metros, 4 decímetros, 5 centímetros e 3 milímetros

15. Fazer um algoritmo para calcular e mostrar os 100 primeiros termos da série de Fibonacci, esta série é gerada da seguinte forma: o primeiro e segundos termos valem 1 e os seguintes são calculados somando-se os dois termos anteriores a ele.

f = 1, 1, 2, 3, 5, 8, ...

16. Fazer um algoritmo para calcular e mostrar os números primos compreendidos entre 500 e 600.