Estruturas condicionais

Estruturas Condicionais

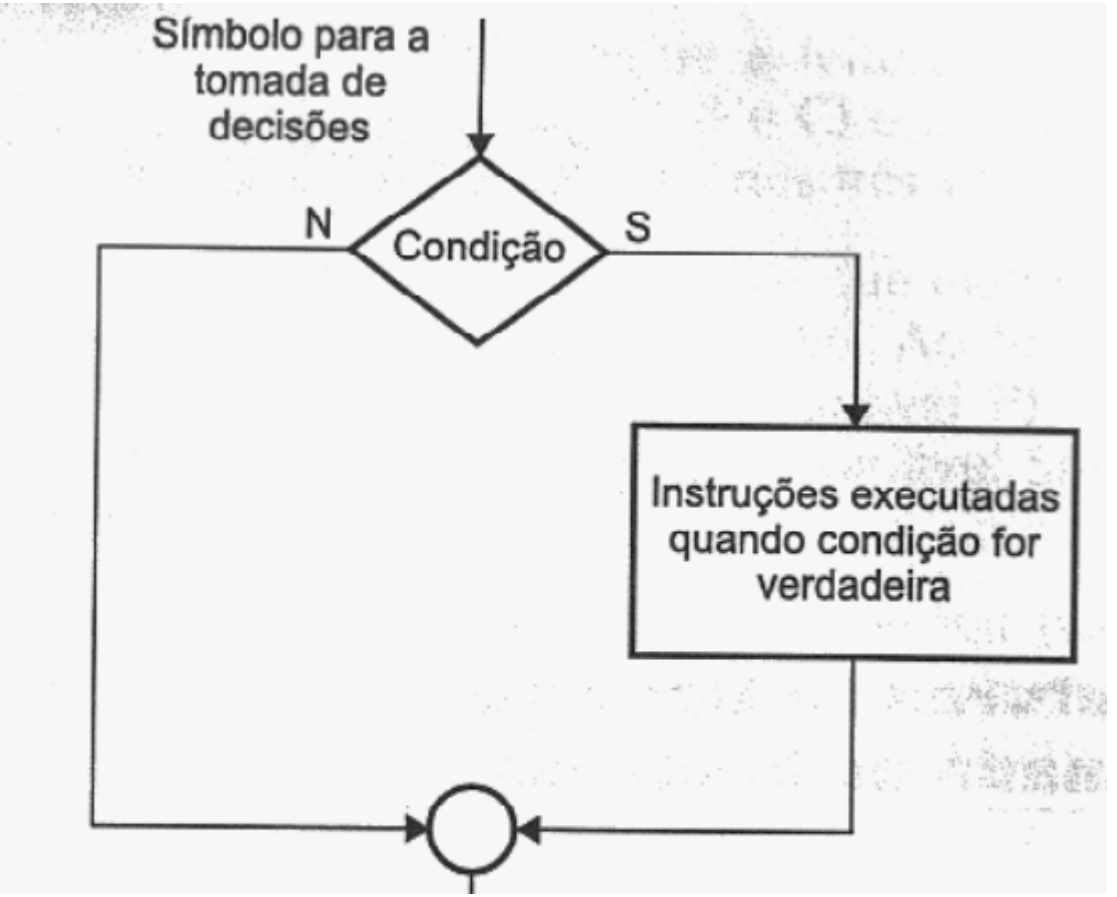
- Muitas vezes é necessário direcionar o fluxo de um programa;
- Isso, junto com variáveis, é o motivo pelo qual temos programas extremamente flexíveis;
- Maioria dos problemas lida com a necessidade dessas estruturas.

Desvio condicionais simples

- A estrutura mais simples é com a instrução se... então... fim_se;
- Ela possibilita a avaliação de uma condição antes de continuar o código;
- Também possibilita desvio de partes do código.

Se Então

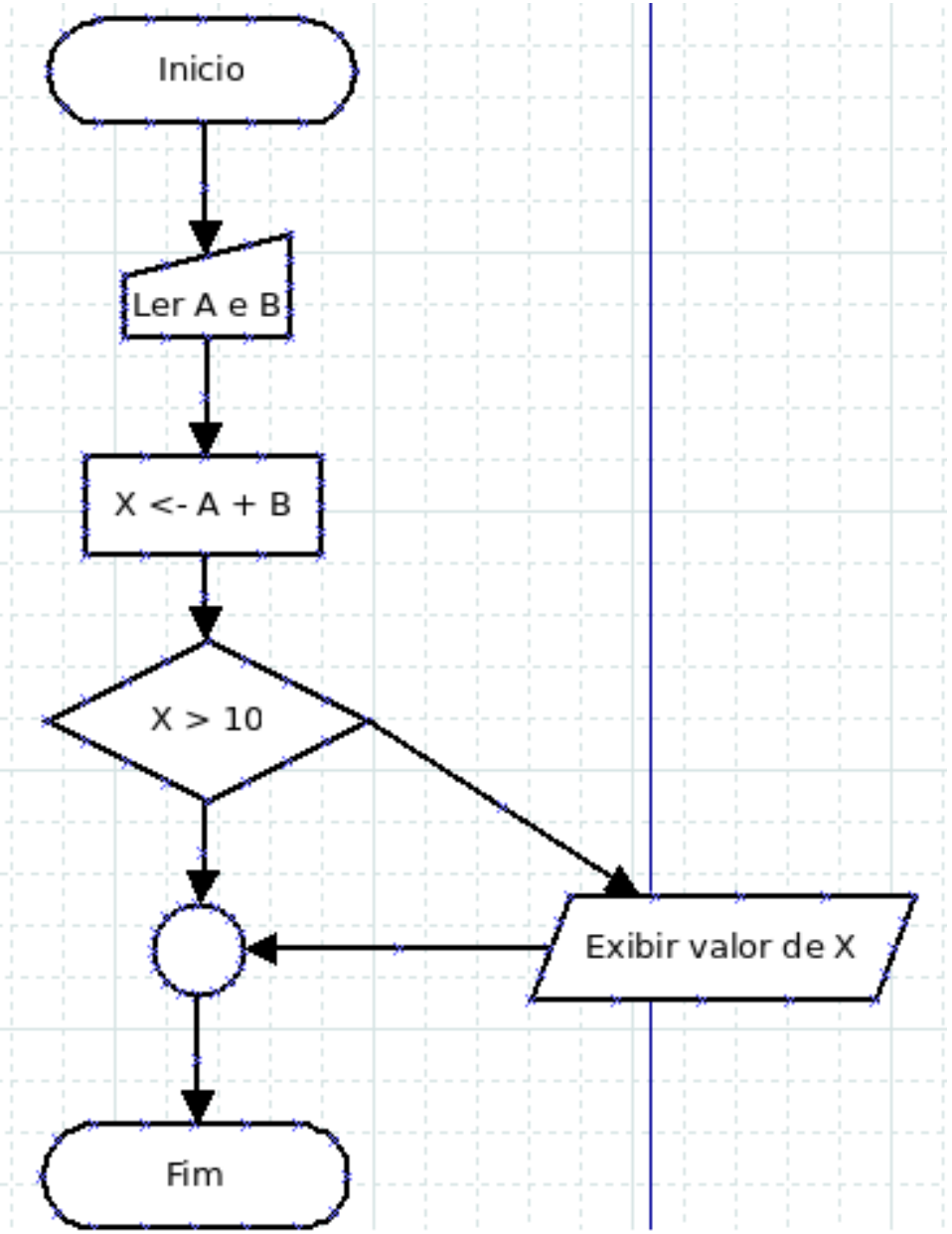
- se (<condição>) então
  - Instruções executadas quando condição for verdadeira
- fim_se
- No diagrama de blocos:



Exemplo

- Ler dois valores (variáveis A e B);
- Efetuar a soma dos dois valores;
- Apresentar o valor da soma, caso ela seja maior que 10.

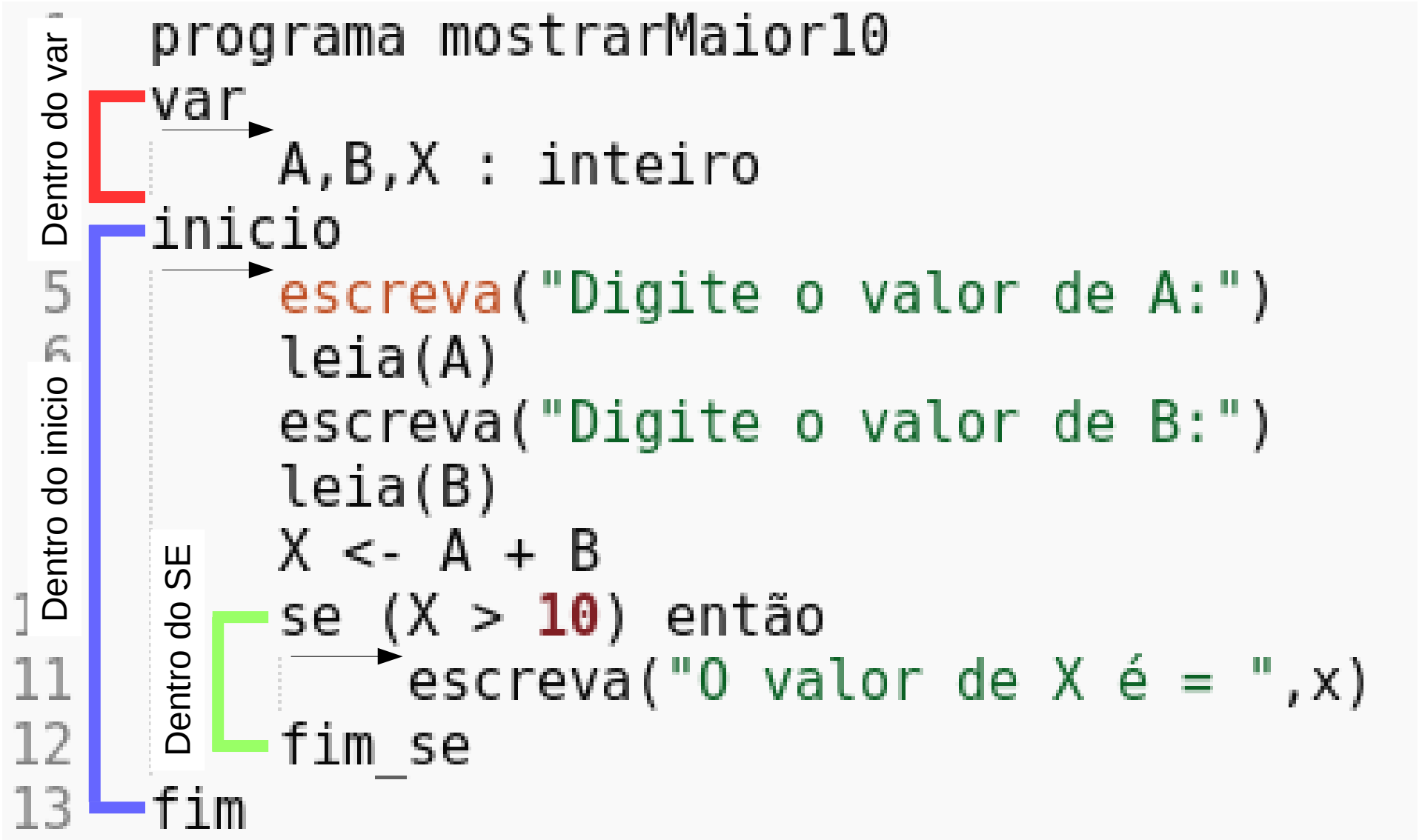
Diagrama de blocos



Algoritmo

```
1  programa mostrarMaior10
2  var
3      A,B,X : inteiro
4  inicio
5      escreva("Digite o valor de A:")
6      leia(A)
7      escreva("Digite o valor de B:")
8      leia(B)
9      X <- A + B
10     se (X > 10) então
11         escreva("O valor de X é = ",x)
12     fim_se
13 fim
```

Algoritmo


Dentro do var
Dentro do inicio
Dentro do SE
programa mostrarMaior10
var
A,B,X : inteiro
inicio
escreva("Digite o valor de A:")
leia(A)
escreva("Digite o valor de B:")
leia(B)
X <- A + B
se (X > 10) então
escreva("O valor de X é = ",x)
fim_se
fim

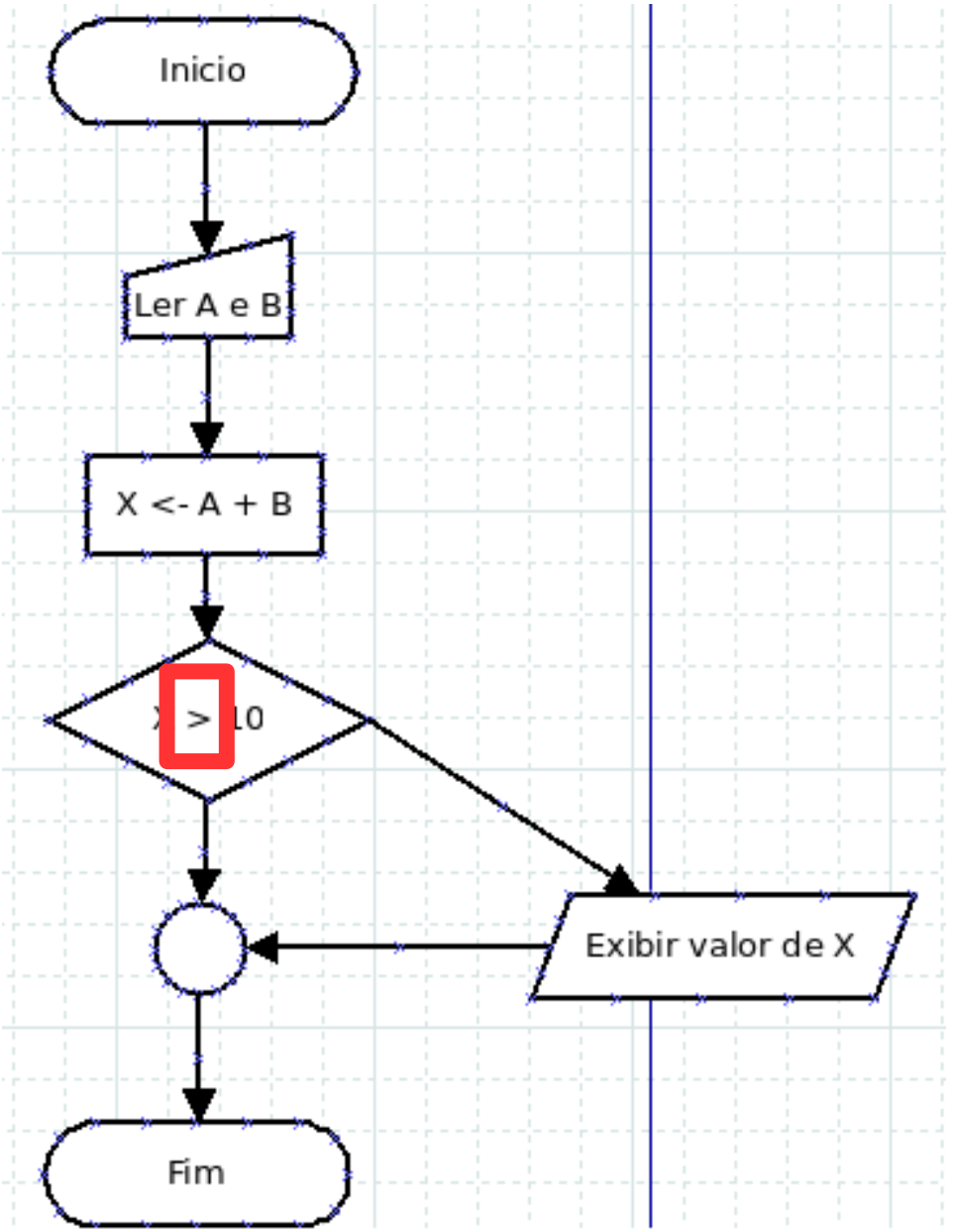
Operadores Relacionais

- A condição em um SE espera que seja feita uma relação entre dois valores;
- Existem uma grande variedade desses operadores;
- Utilizaremos eles para efetuar a comparação entre diferentes itens.

Operadores Relacionais

Símbolo	Significado
=	Igual a
<>	Diferente de
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a

Diagrama de blocos

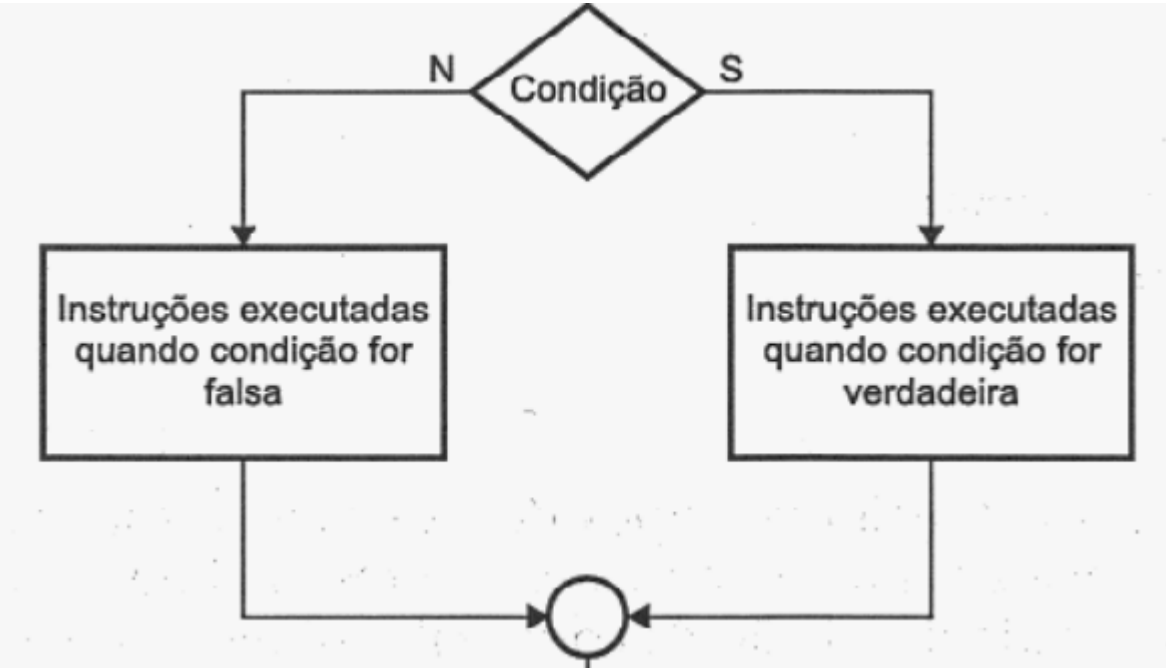


Desvio condicional composto

- É possível utilizar uma instrução para ser executada caso a condição seja falsa;
- É a clausula senão.

Desvio condicional composto

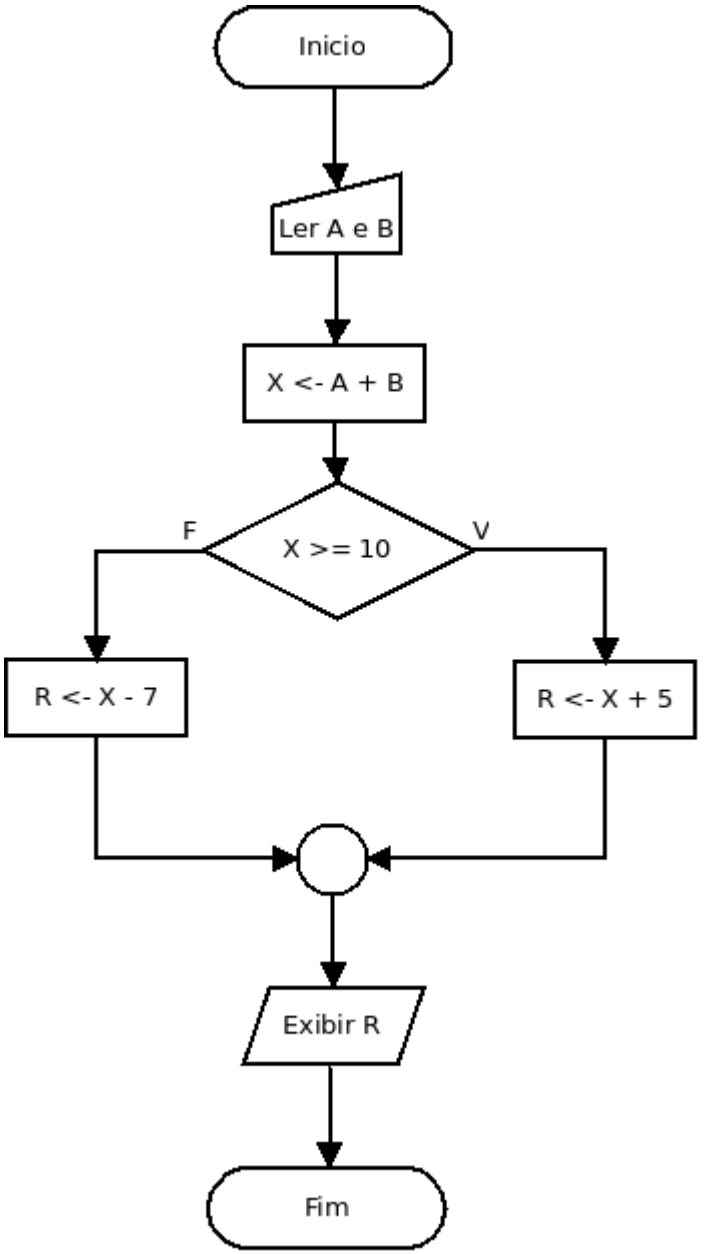
- se (<condição>) então
  - Instruções executadas quando condição for verdadeira
- senão
  - Instruções executadas para condição falsa
- fim_se



Exemplo

- Ler dois valores (A e B);
- Somar ambos;
- Se a soma for maior ou igual à 10 calcule valor da soma + 5, caso contrário calcule valor da soma – 7.

Exemplo



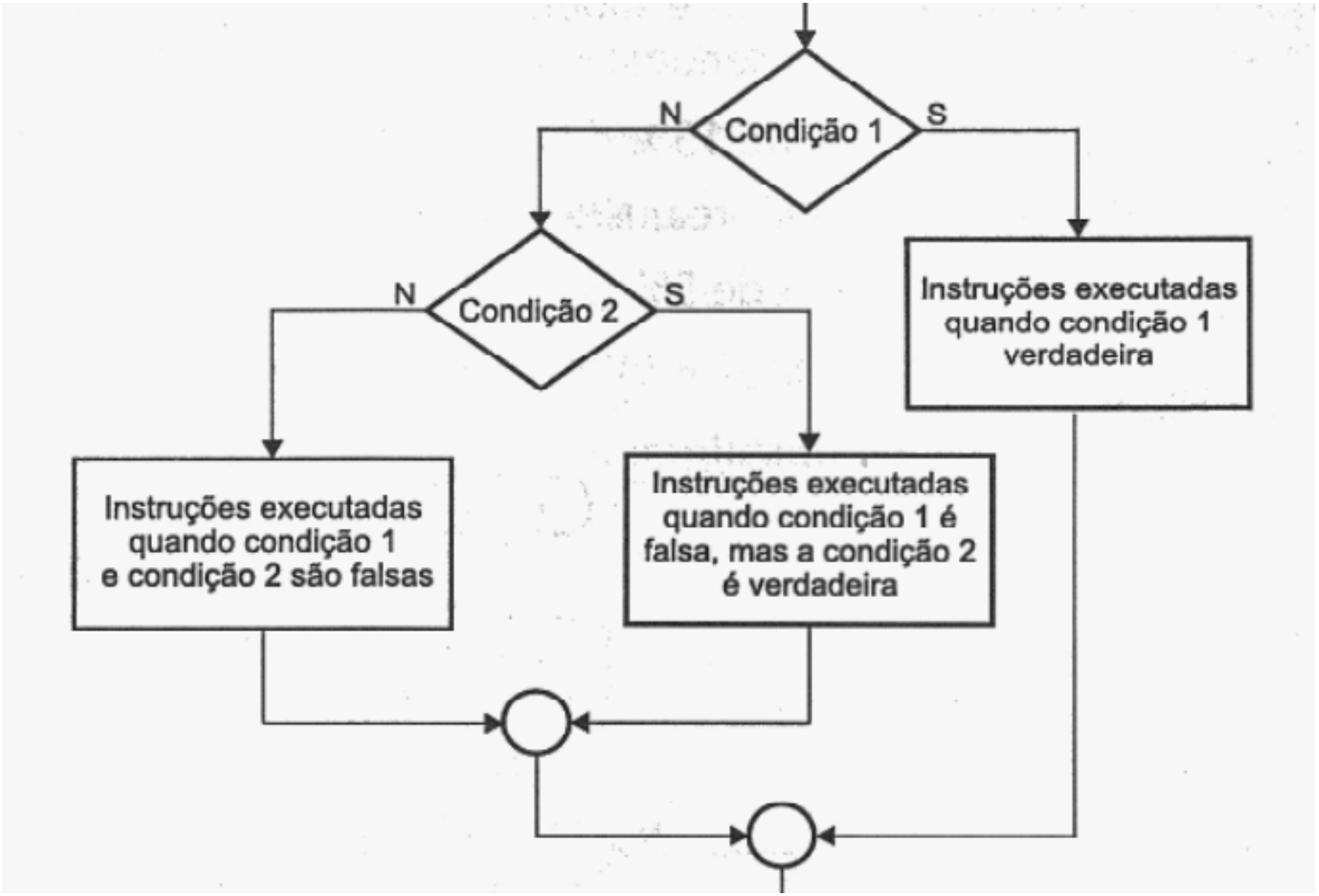
Exemplo

```
1  programa mostrarMaior10
2  var
3      A,B,X,R : inteiro
4  inicio
5      escreva("Digite o valor de A:")
6      leia(A)
7      escreva("Digite o valor de B:")
8      leia(B)
9      X <- A + B
10     se (X >= 10) então
11         R <- X + 5
12     senão
13         R <- X - 7
14     fim_se
15     escreva("Valor de R = ",R)
16 fim
```

Operadores encadeados

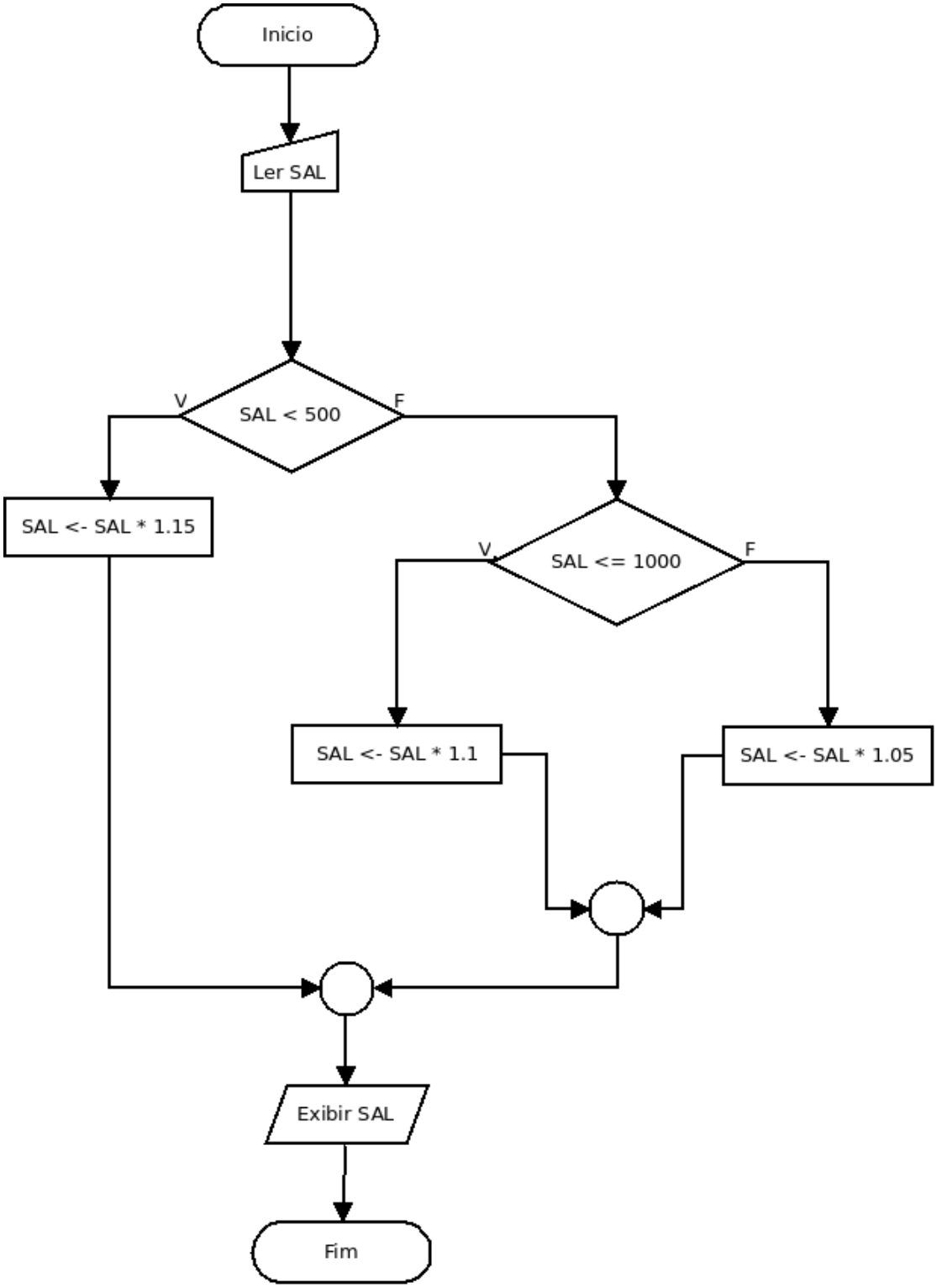
- É possível colocar “um se dentro de outro”;
- se (condicao_1) então
  - Instruções para condicao_1 verdadeira
- senão
  - se (condicao_2) então
    - Instruções para condicao_2 verdadeira
  - senão
    - Instruções para condicao_1 e condicao_2 falsa
  - fim_se
- fim_se

Operadores encadeados



Exemplo

- Calcular o reajuste do salário de um funcionário;
  - 15% caso seu salário seja menor que 500;
  - 10% se o seu salário esteja entre 500 e 1000;
  - 5% caso seja maior que 1000.


Inicio
Ler SAL
SAL < 500
V
F
SAL <- SAL * 1.15
SAL <= 1000
V
F
SAL <- SAL * 1.1
SAL <- SAL * 1.05
Exibir SAL
Fim

Exemplo

```
1  programa calcularAumento
2  var
3      SAL : real
4  inicio
5      escreva("Digite seu salário: ")
6      leia(SAL)
7      se (SAL < 500) então
8          SAL <- SAL * 1.15
9      senão
10         se (SAL <= 1000) então
11             SAL <- SAL * 1.1
12         senão
13             SAL <- SAL * 1.05
14         fim_se
15     fim_se
16     escreva("Seu novo salário é de: ",SAL)
17 fim
```

Operadores lógicos

- É possível combinar duas ou mais condições com operadores lógicos;
  - Operador e
  - Operador ou
- Também é possível inverter o valor de um operador;
  - Operador não

Operadores lógicos

- se (condicao_1) e (condicao_2) então
  - Executar instruções
- fim_se
- se (condicao_1) ou (condicao_2) então
  - Executar instruções
- fim_se
- se não (condicao_1) então
  - Executar instruções
- fim_se

Operadores Lógicos

```
programa TESTA_LÓGICA_E
var
   NÚMERO : inteiro
início
   leia NÚMERO
   se (NÚMERO >= 20) .e. (NÚMERO <= 90) então
        escreva "O número está na faixa de 20 a 90"
   senão
        escreva "O número está fora da faixa de 20 a 90"
   fim_se
fim
```

Operadores Lógicos

```
programa TESTA_LÓGICA_OU
var
   SEXO : caractere
início
   leia SEXO
   se (SEXO = "masculino") .ou. (SEXO = "feminino") então
        escreva "O seu sexo é válido"
   senão
        escreva "O seu sexo é inválido"
   fim_se
fim
```

Operadores Lógicos

```
programa TESTA_LÓGICA_NÃO
var
   A, B, C, X : inteiro
início
   leia A, B, X
   se .não. (X > 5) então
        C ← (A + B) * X
   senão
        C ← (A - B) * X
   fim_se
   escreva C
fim
```

Operadores lógicos

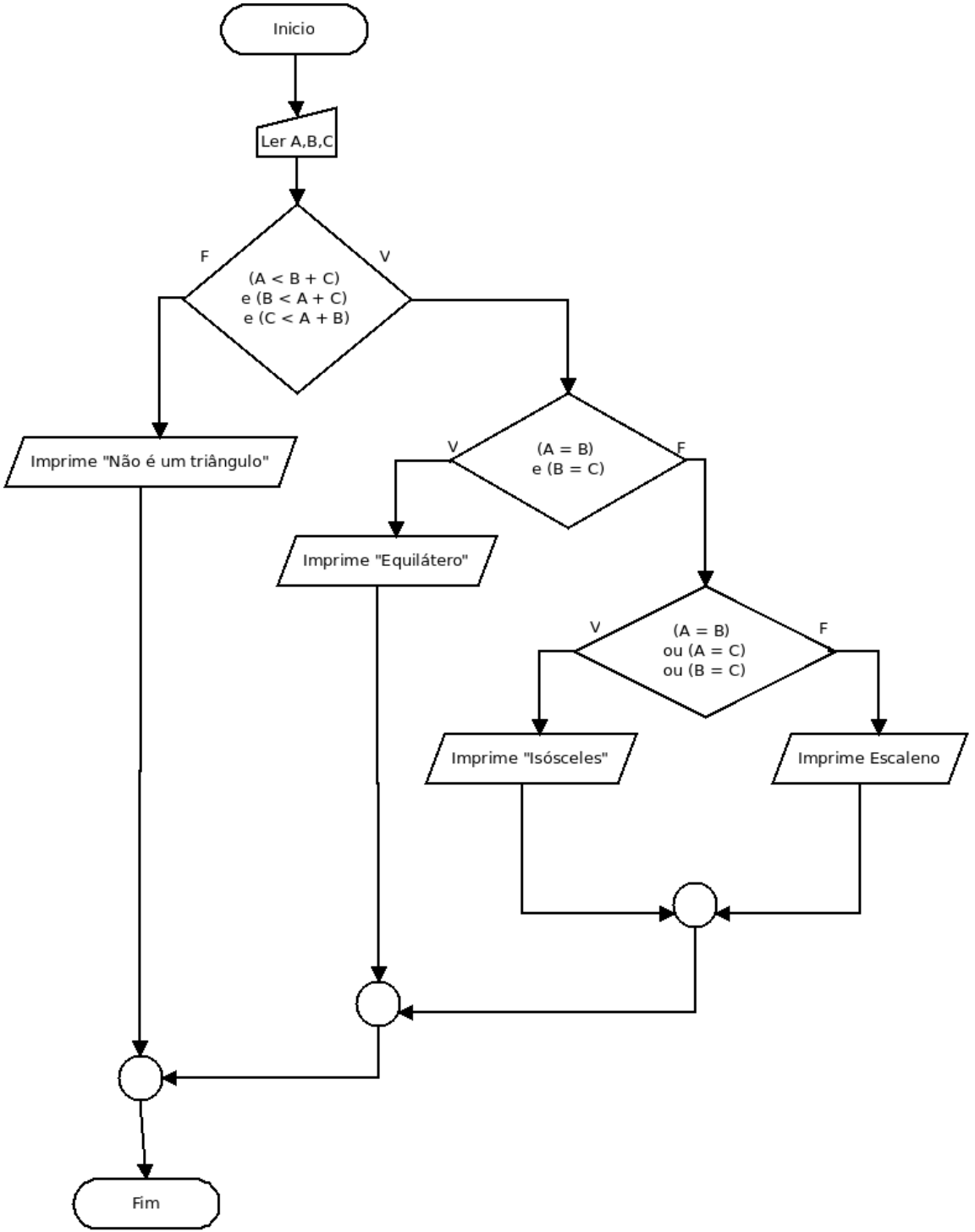
Operador Lógico e		
C1	C2	C1 e C2
V	V	V
F	V	F
V	F	F
F	F	F

Operador Lógico ou		
C1	C2	C1 ou C2
V	V	V
F	V	V
V	F	V
F	F	F

Operador Lógico não	
C1	não C1
V	F
F	V

Exemplo

- Dadas as medidas de um triângulo mostre se ele é isósceles, escaleno ou equilátero;


Inicio
Ler A,B,C
F
V
(A < B + C)
e (B < A + C)
e (C < A + B)
Imprime "Não é um triângulo"
V
F
(A = B)
e (B = C)
Imprime "Equilátero"
V
F
(A = B)
ou (A = C)
ou (B = C)
Imprime "Isósceles"
Imprime Escaleno
Fim

Exemplo

```
1  programa verificarTriangulo
2  var
3      a,b,c : real
4  inicio
5      escreva("Digite os três lados de um triângulo: ")
6      leia(a,b,c)
7      se (a < (b + c)) e (b < (a + c)) e (c < (b + a)) então
8          se (a = b) e (b = c) então
9              escreva("Triângulo Equilátero")
10         senão
11             se (a = b) ou (a = c) ou (b = c) então
12                 escreva("Triângulo Isósceles")
13             senão
14                 escreva("Triângulo Escaleno")
15             fim_se
16         fim_se
17     senão
18         escreva("As medidas não são de um triângulo")
19     fim_se
20 fim
```