Aula Teórica 05

Material Didático

Conteúdos da Aula

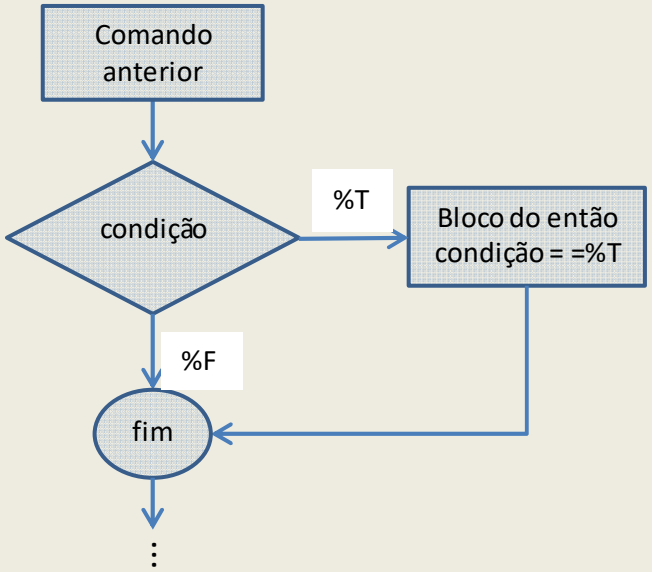
- **Comando de Desvio de Fluxo If e elseif**
- **Exercícios**

Comando de Desvio de Fluxo

O comando `if` é um comando de desvio do fluxo de execução.

Duas estruturas para o comando `if` são:

`if <condição> then <comandos> end`

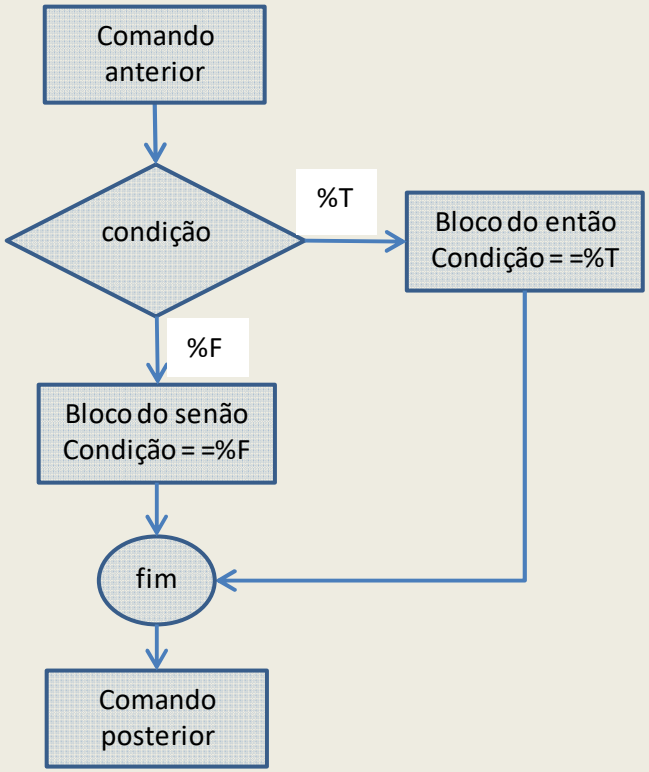


Se a condição for verdadeira, executa o Bloco do então e vai para o primeiro comando após o fim (end).

Se a condição for falsa, não faz nada e segue para o primeiro comando após o fim (end)

Comando de Desvio de Fluxo

```
if <condição> then <Bloco1> else <Bloco2> end
```



Se a condição for verdadeira então Executa o Bloco do então e vai para o comando posterior

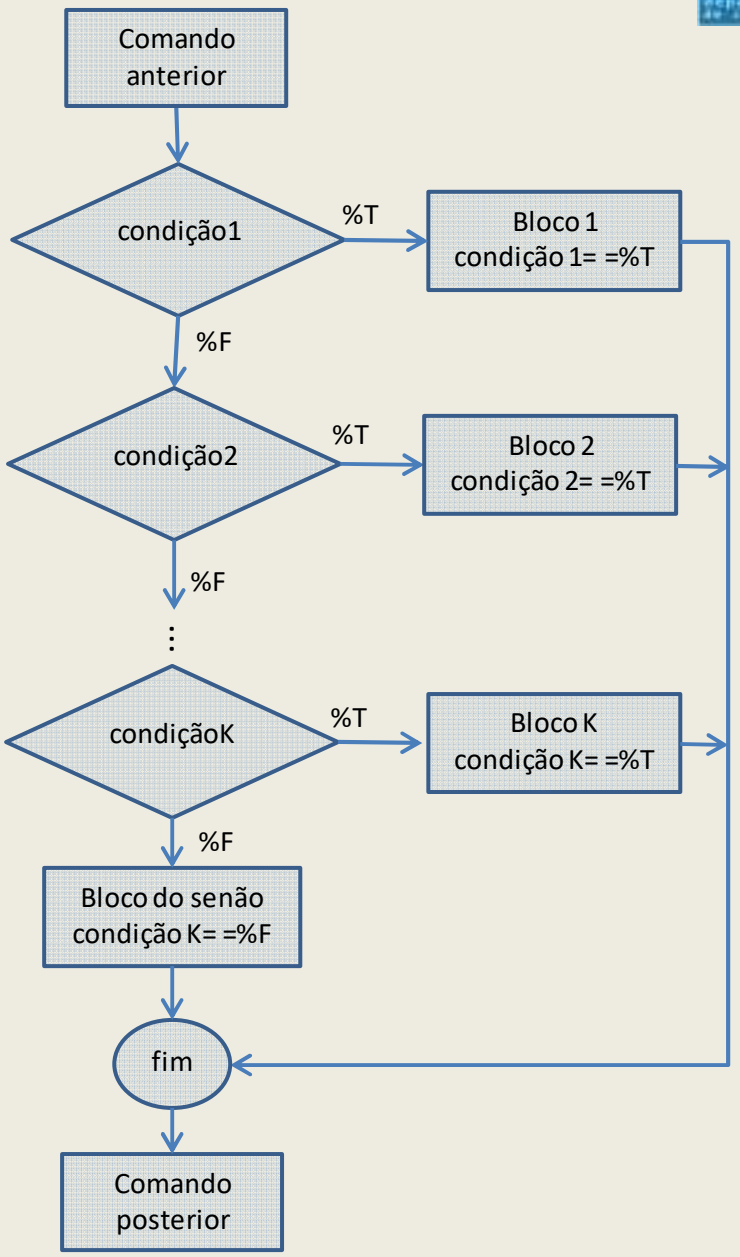
Caso contrário, se a condição for falsa, executa o Bloco do senão e vai para o comando posterior

Comando de Desvio de Fluxo

Podemos ter uma série de estruturas do tipo

```
if <condição> then
    Conjunto comandos 1
else // não tem condição
    Conjunto comandos 2
end
```

compondo o bloco de comandos, teremos o aninhamento de `if`'s, ou seja, um `if` dentro de outro `if`.



if□s aninhados

Quando existe este tipo de disposição do comando if, ocorre o que denominamos de if's aninhados

um end para cada if correspondente

```
if <condição_a> then
      <comando_1>;
      • • •
      <comando_n>;
else
      if <condição_b> then
            <comando_1>;
            • • •
            <comando_m>;
      else
            <comando_1>;
            • • •
            <comando_p>;
      end
end
```

Faça um programa que leia o valor de X maior ou igual a zero e determine em que faixa ele se encontra. Caso o valor seja menor do que zero dar uma mensagem de erro.

Faixa 1: $0 \leq X < 100$
Faixa 2: $100 \leq X < 500$
Faixa 3: $500 \leq X < 1.000$
Faixa 4: $1.000 \leq X < 5.000$
Faixa 5: $5.000 \leq X$

Execução 1:
Entre com o valor de X: 435
Resposta: X está na Faixa 2

Execução 2:
Entre com o valor de X: 2750
Resposta: X está na Faixa 4

Execução 3:
Entre com o valor de X: -20
Resposta: Erro! Valor inválido

```
Faixa 1: valores até 100
Faixa 2: maiores do que 100 e até 500
Faixa 3: maiores do que 500  e  até 1.000
Faixa 4: maiores do que 1.000 e até 5.000
Faixa 5: valores acima de 5.000
x = input("Digite o valor de x >= 0")
if x < 0 then
   printf("Erro! Valor inválido")
else // o valor em x é válido, >= zero. Encontrar a faixa
   if x <= 100   then
      printf("O valor está na Faixa 1")
   else // o valor em x > 100
      if x <= 500 then
            printf("O valor está na Faixa 2")
      else // o valor em x > 500
            if x <= 1000 then
                  printf("O valor está na Faixa 3")
            else // o valor em x > 1000
                  if x <= 5000 then
                        printf("O valor está na Faixa 4")
                  else // o valor em x > 5000
                        printf("O valor está na Faixa 5")
                  end
            end
      end
   end
end
```

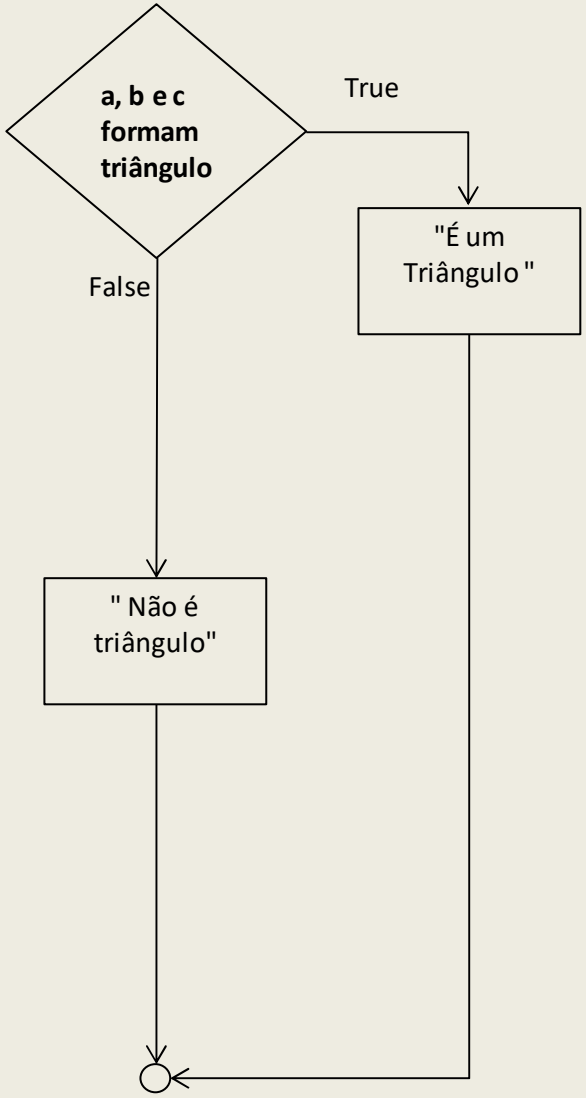
usando `elseif`

```
if <condição_a> then
      <comando_1>;
      • • •
      <comando_n>;
elseif <condição_b>
      <comando_1>;
      • • •
      <comando_m>;
else // opcional
      <comando_1>;
      • • •
      <comando_p>;
end
```

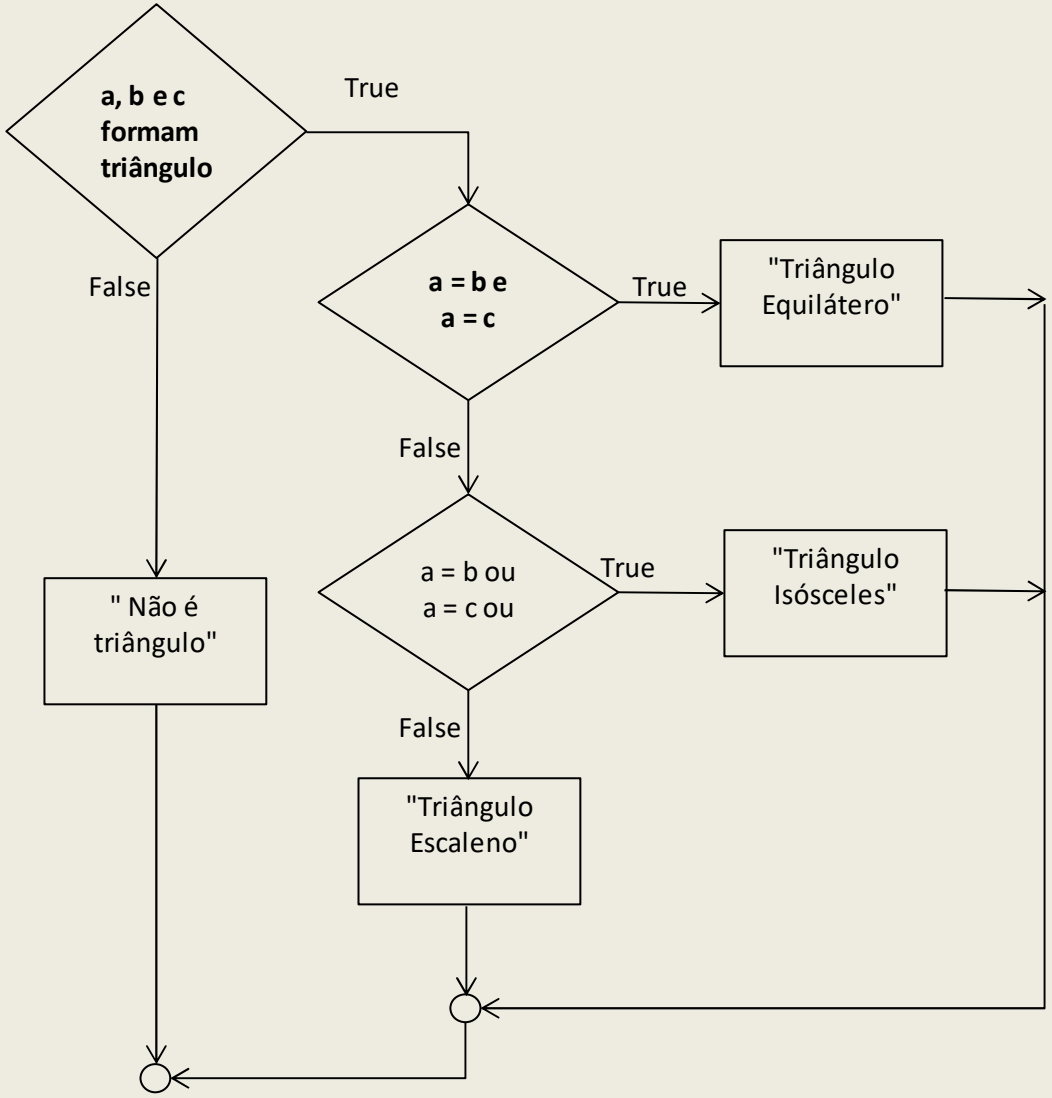
independente da quantidade de testes teremos somente um `end`

```
Faixa 1: valores até 100
Faixa 2: maiores do que 100 e até 500
Faixa 3: maiores do que 500  e  até 1.000
Faixa 4: maiores do que 1.000 e até 5.000
Faixa 5: valores acima de 5.000
x = input("Digite o valor de x >= 0")
if x < 0 then
   printf("Erro! Valor inválido")
else // o valor em x é válido, >= zero. Encontrar a faixa
   if x <= 100   then
      printf("O valor está na Faixa 1")
   else // o valor em x > 100
      if x <= 500 then
            printf("O valor está na Faixa 2")
      else // o valor em x > 500
            if x <= 1000 then
                  printf("O valor está na Faixa 3")
            else // o valor em x > 1000
                  if x <= 5000 then
                        printf("O valor está na Faixa 4")
                  else // o valor em x > 5000
                        printf("O valor está na Faixa 5")
                  end
            end
      end
   end
end
```

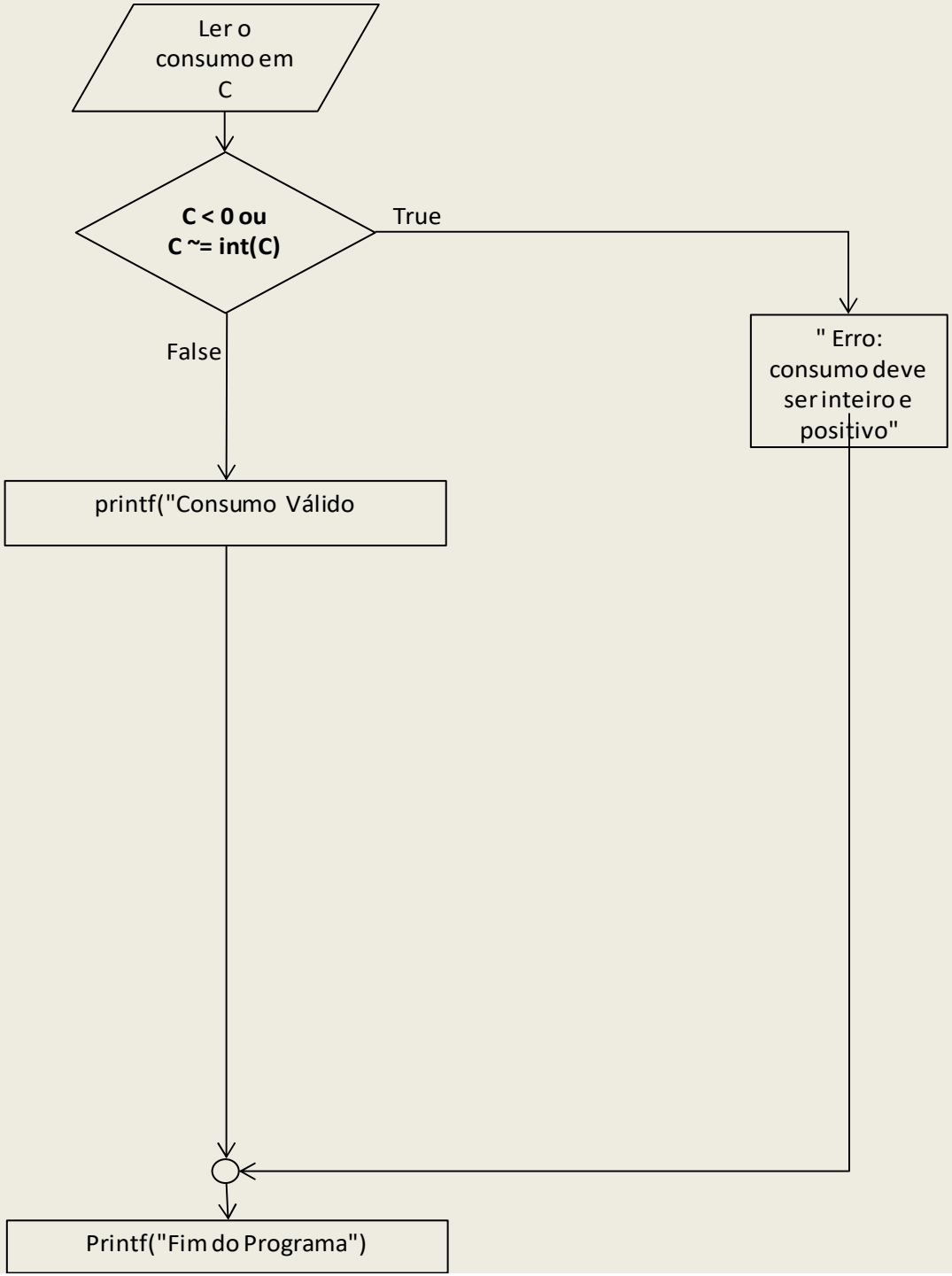
```
Faixa 1: valores até 100
Faixa 2: maiores do que 100 e até 500
Faixa 3: maiores do que 500  e  até 1.000
Faixa 4: maiores do que 1.000 e até 5.000
Faixa 5: valores acima de 5.000
if x < 0 then
   printf("Erro! Valor inválido")
else // o valor em x é válido, >= zero. Encontrar a faixa

  if x <= 100   then
       printf("O valor está na Faixa 1")
  elseif x <= 500 then
       printf("O valor está na Faixa 2")
  elseif x <= 1000 then
       printf("O valor está na Faixa 3")
  elseif x <= 5000 then
       printf("O valor está na Faixa 4")
  else
       printf("O valor está na Faixa 5")
  end

end
```

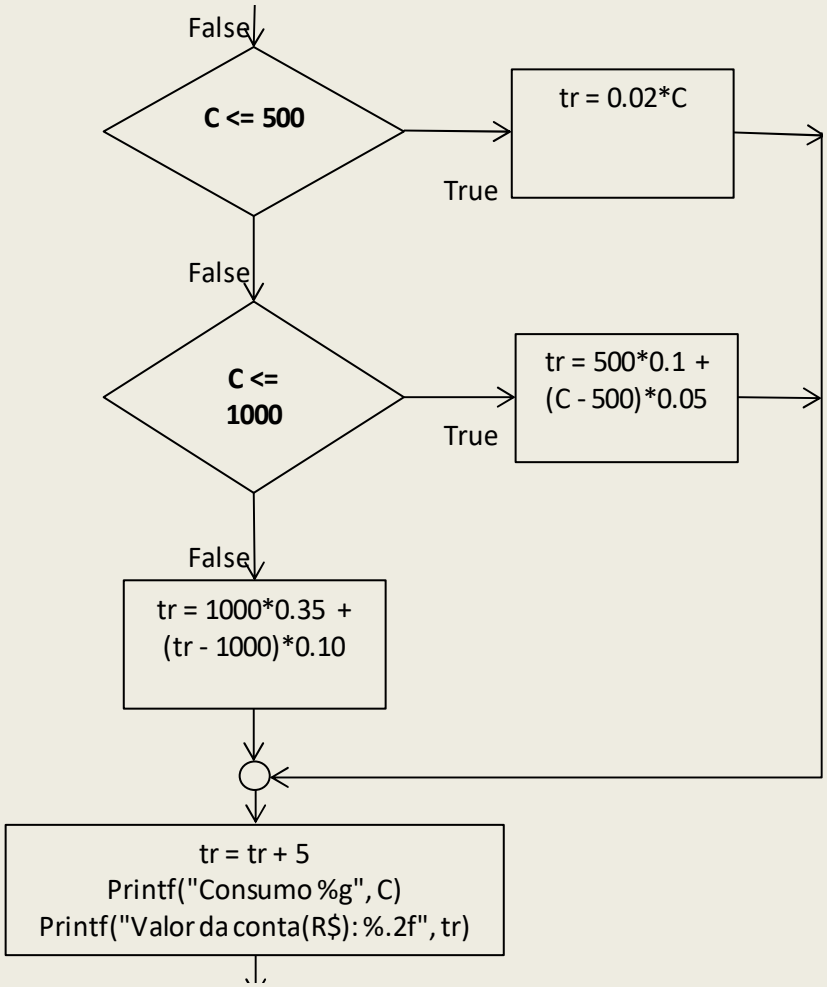


if (a, b, c formam um triângulo) then			
	Printf("É um triângulo")		
else			
	Printf("Não é triângulo")		
end			

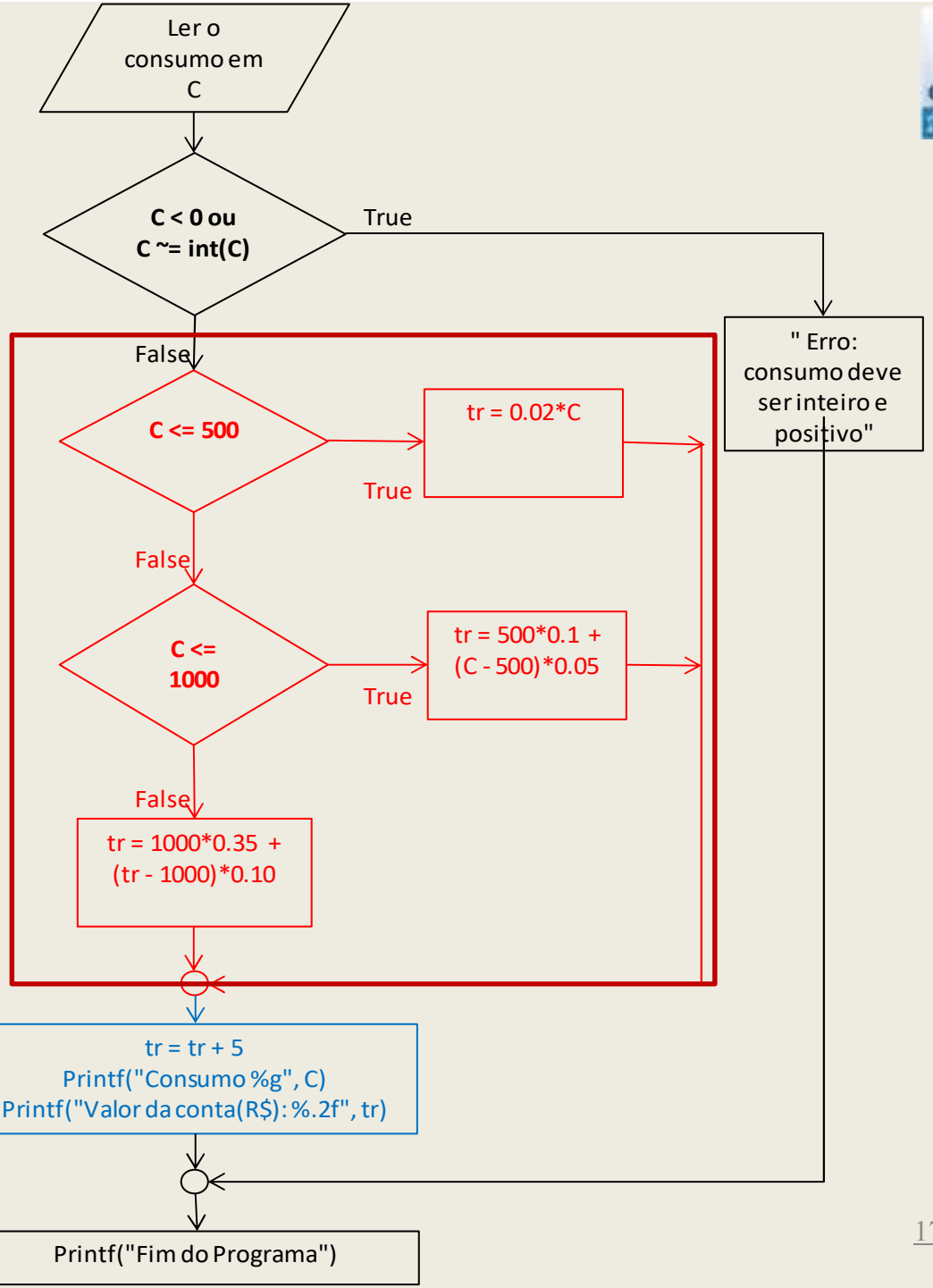


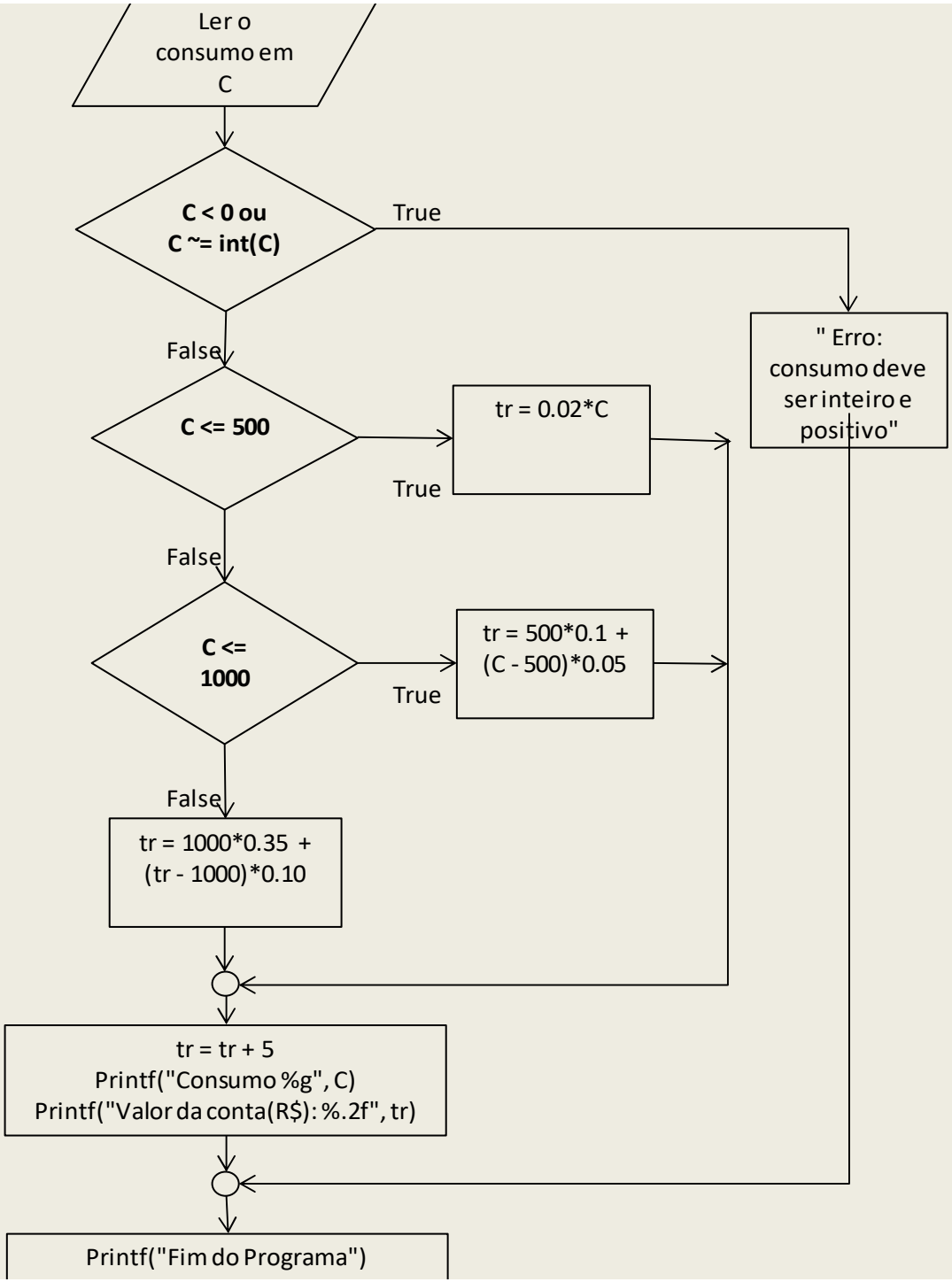

```
if (a, b, c formam um triângulo) then
        if (a = b e b = c) then
                Prinf("Triângulo Equilátero")
        elseif( a = b ou a = c ou b = c ) then
                Prinf("Triângulo Isósceles")
        else
                Prinf("Escaleno")
        end
else
        Printf("Não é triângulo")
end
```


Ler o consumo em C
C < 0 ou C ~= int(C)
True
False
" Erro: consumo deve ser inteiro e positivo"
printf("Consumo Válido
Printf("Fim do Programa")


False
C <= 500
True
tr = 0.02*C
False
C <= 1000
True
tr = 500*0.1 + (C - 500)*0.05
False
tr = 1000*0.35 + (tr - 1000)*0.10
tr = tr + 5
Printf("Consumo %g", C)
Printf("Valor da conta(R$): %.2f", tr)

```
C = input("Consumo")
if (C < 0 ou C ~= int(C) then
        Printf("Erro: consumo deve ser ...")
else
        if (C <= 500) then
                tr = 0.02*C
        elseif(C <= 1000) then
                tr = 500*0.1 +
                (C - 500)*0.05
        else
                tr = 1000*0.35 +
                (C - 1000)*0.10
        end
        tr = tr + 5
        Printf("Consumo %g", C)
        Printf("Valor da conta(R$): %.2f", tr)
end
printf("Fim do programa")
```




Ler o consumo em C
C < 0 ou C ~= int(C)
True
" Erro: consumo deve ser inteiro e positivo"
False
C <= 500
tr = 0.02*C
True
False
C <= 1000
tr = 500*0.1 + (C - 500)*0.05
True
False
tr = 1000*0.35 + (tr - 1000)*0.10
tr = tr + 5
Printf("Consumo %g", C)
Printf("Valor da conta(R$): %.2f", tr)
Printf("Fim do Programa")

Comando de Desvio de Fluxo

Comparação

```
if ... then ... else ... end

if <condição1> then
   <bloco "então1">
else
   if <condição2> then
      <bloco "então2">
   else
       if <condição3> then
           <bloco "então3">
       else
          if <condiçãoN> then
             <bloco "entãoN">
          else
             <bloco "senão">
          end
       end
   end
end
```

if ... then ... elseif ... end

```
if <condição1> then
  <bloco "então1">
elseif <condição2>
  <bloco "então2">
elseif <condição3>
  <bloco "então3">
elseif <condiçãoN>
  <bloco "entãoN">
else
  <bloco "senão">
end
```

Exemplo

Considere um aplicativo que apresenta uma série de opções, como por exemplo um menu eletrônico de uma pizzaria com tele-entrega

```
Pizzaria BCC vc que escolhe:
  escolha a sua opção:
1 - A moda
2 - Atum
3 - Brócolis com bacon
4 - Calabresa
.
.
.
15 - Vegetariana
Entre com a escolhida:
```

```
...
opcao = input("Entre com a
   escolha: ")
      if opcao == 1 then
        <bloco "então1">
      elseif opcao == 2
        <bloco "então2">
      elseif opcao == 3
        <bloco "então3">
      ...
      elseif opcao == 14
        <bloco "entãoN">
      else
        <bloco "senão">
      end
```

Operadores Lógicos

Operadores lógicos definidos no Scilab:

Operador	Descrição
&	E (conjunção).
|	OU (disjunção não exclusiva).
~	Não (negação).

- **E (Conjunção):**
  - **Resulta verdadeiro se as duas expressões forem verdadeiras;**
- **OU (disjunção não exclusiva):**
  - **Resulta verdadeiro se pelo menos uma das duas expressões forem verdadeiras;**
- **Não (negação):**
  - **Inverte o valor lógico da expressão, se ela for verdadeira o resultado é falso, se ela for falsa o resultado é verdadeiro.**

Operadores lógicos podem ser definidos por Tabelas Verdade

A	B	A & B	A | B	~A
%t	%t	%t	%t	%f
%t	%f	%f	%t	%f
%f	%t	%f	%t	%t
%f	%f	%f	%f	%t

O consumo de energia c é válido se: **c >= 0 e c é inteiro**,
if (c >= 0 & c == int(c)) then printf("\n Consumo válido") end

O consumo de energia c é inválido se: **c < 0 ou c não é inteiro**,
if (c < 0 | c ~= int(c)) then printf("\n Consumo Inválido") end

Exercícios

Exercício 1

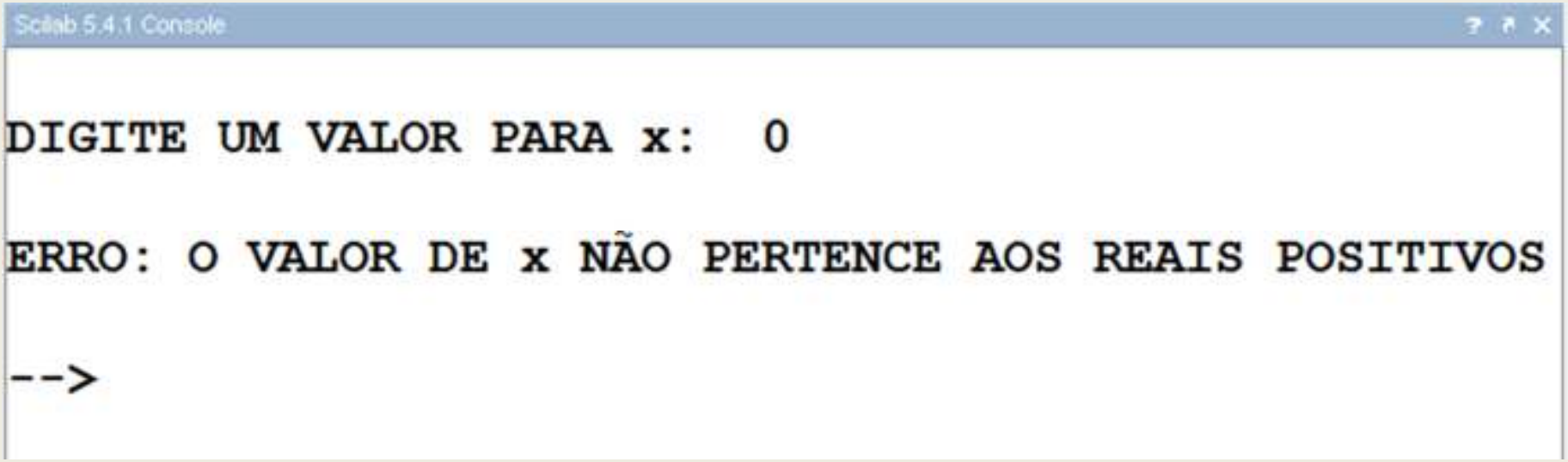
A função $f(x)$, no domínio $\Re^+$ (reias não negativos), é definida por:

$$f(x) = \begin{cases} (x^2 + 0.5)^3 & , \text{se } 0 < x \leq 2 \\ 1 / (x^2 - 4) & , \text{se } 2 < x \leq 10 \\ 2\,\text{sen}(x) + \cos(4x) & , \text{se } 10 < x \leq 20 \\ 23.8 & , \text{se } x > 20 \end{cases}$$

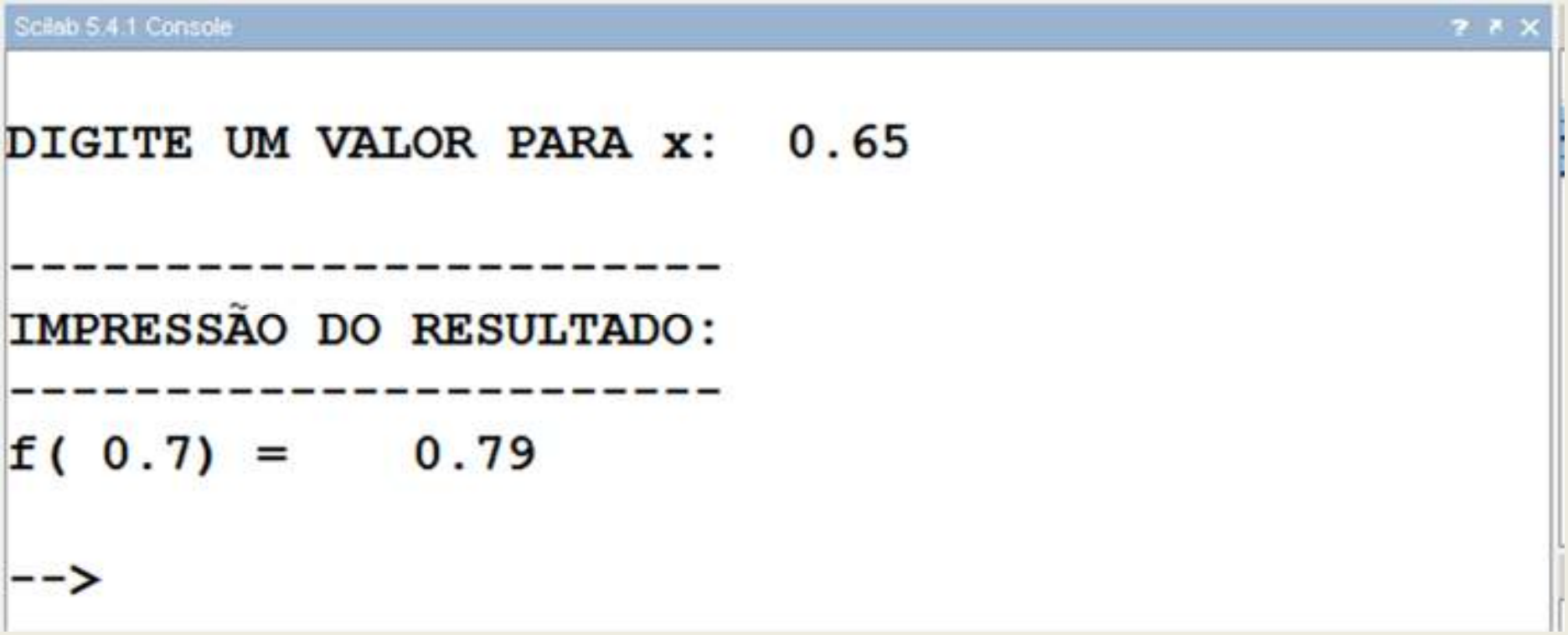
Codifique um programa que calcule o valor da função para um argumento de entrada pertencente ao domínio da função (x > 0). Caso o valor não pertença ao domínio emitir mensagem de erro.

A seguir a ilustração de execução do programa.

Exercício 1 – Execução A



Exercício 1 – Execução B



Exercício 1 - Código

```
1  clc; clear;
2  x = input("DIGITE UM VALOR PARA x:  ");
3  if x <= 0 then
4      printf("\nERRO: O VALOR DE x ");
5      printf("NÃO PERTENCE AOS REAIS POSITIVOS\n");
6  else
7      if x <= 2 then
8          fx = (x^2 + 0.5)^3;
9      elseif x <= 10      // x já é maior que 2
10         fx = 1 / (x^2 - 4);
11     elseif x <= 20      // x já é maior que 10
12         fx = 2 * sin(x) + cos(4*x);
13     else      //      x é maior que 20
14         fx = 23.8;
15     end
16     printf("\n--------------------");
17     printf("\nIMPRESSÃO DO RESULTADO:")
18     printf("\n--------------------");
19     printf("\nf(%4.1f) = %7.2f\n", x, fx);
20 end
```

Exercício 1 – Código Alternativo

O código alternativo exibido a seguir não representa uma boa

prática de programação, pois:

- não se beneficia dos valores em ordem crescente, naturalmente especificados no intervalo;
- a cada `elseif`, devemos testar o subintervalo usando o operador `and`, sem reaproveitar o teste do subintervalo anterior;

Exercício 1 – Código Alternativo

```
1  clc; clear;
2  x = input("DIGITE UM VALOR PARA x: ");
3  if x <= 0 then
4      printf("\nERRO: O VALOR DE x ");
5      printf("NÃO PERTENCE AOS REAIS POSITIVOS\n");
6  else
7      if x <= 2 then
8          fx = (x^2 + 0.5)^3;
9      elseif x > 10 & x <= 20
10         fx = 2 * sin(x) + cos(4*x);
11     elseif x > 2 & x <= 10
12         fx = 1 / (x^2 - 4);
13     elseif x > 20
14         fx = 23.8;
15     end
16     printf("\n----------");
17     printf("\nIMPRESSÃO DO RESULTADO:")
18     printf("\n----------");
19     printf("\nf(%4.1f) = %7.2f\n", x, fx);
20 end
```

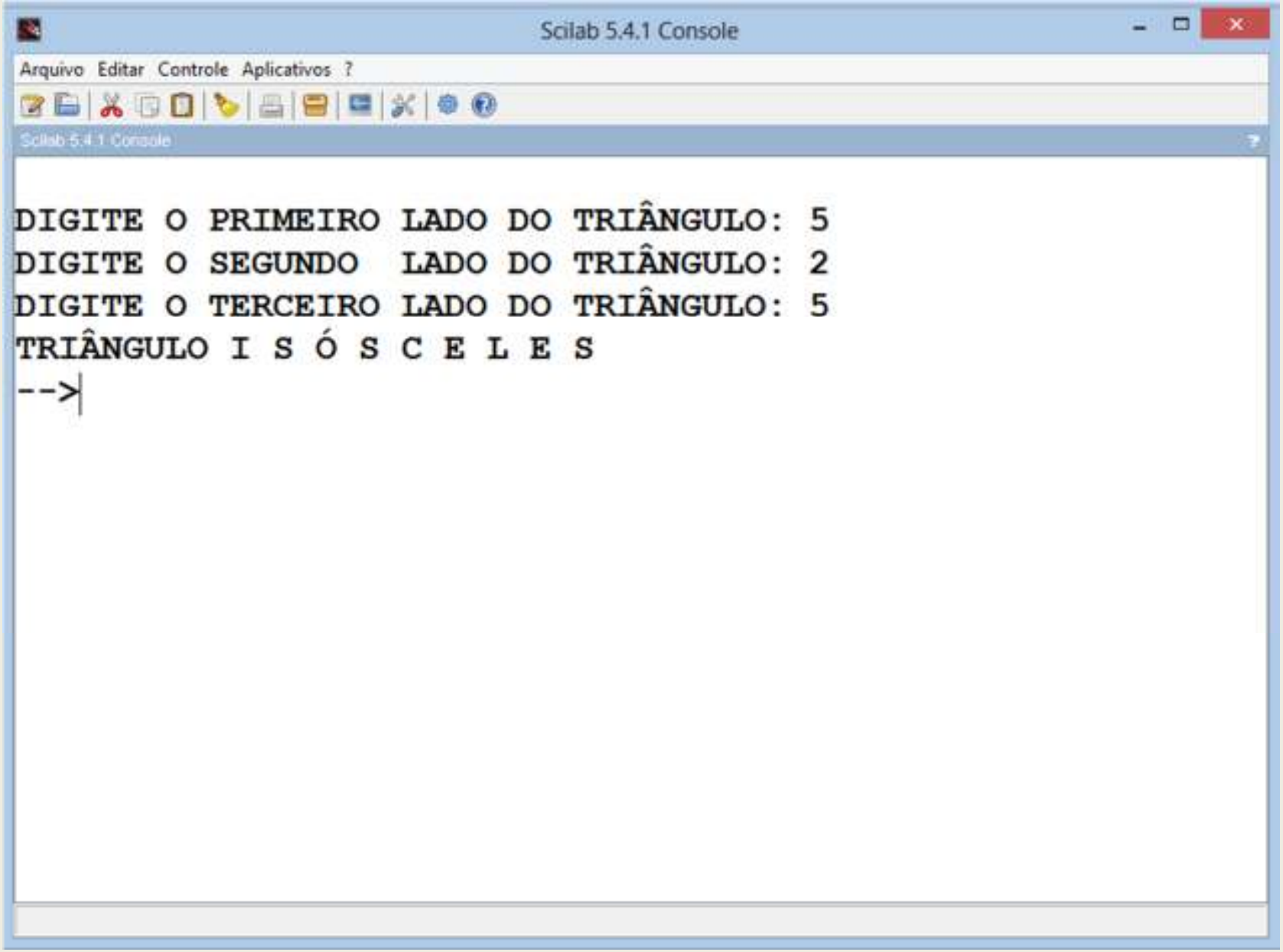
Exercício 2

Codifique um programa em Scilab que leia 3 valores inteiros a partir do teclado.

Estes valores inteiros representam os lados de um triângulo. O Programa deve verificar a condição de existência do triângulo. Em caso afirmativo, o programa classifica o triângulo de acordo com seus lados: equilátero, ou isósceles, ou escaleno.

A seguir uma ilustração da execução do programa.

Exercício 2 - Execução



Exercício 2 - Código

A04Ex2.sce

```
1  clc; clear;
2  // Entrada dos lados do triângulo
3  a = input("DIGITE O PRIMEIRO LADO DO TRIÂNGULO: ")
4  b = input("DIGITE O SEGUNDO  LADO DO TRIÂNGULO: ")
5  c = input("DIGITE O TERCEIRO LADO DO TRIÂNGULO: ")
6  // Início dos cálculos
7  if (a < b + c) & (b < a + c) & (c < a + b) then
8      if (a == b) & (b == c) then
9          printf("TRIÂNGULO E Q U I L Á T E R O")
10     elseif (a == b) | (a == c) | (b == c)
11         printf("TRIÂNGULO I S Ó S C E L E S")
12     else
13       printf("TRIÂNGULO E S C A L E N O")
14     end
15 else
16     printf("TRIÂNGULO INEXISTENTE")
17 end
```