



**Universidade Federal de Ouro Preto - UFOP**  
**Departamento de Computação - DECOM**  
**Comissão para Coordenação das Atividades Pedagógicas**  
**da Disciplina BCC701 – CAP-BCC701**  
**[www.decom.ufop.br/bcc701](http://www.decom.ufop.br/bcc701)**



# **Aula Teórica 03**

## **Comandos Condicionais (Decisão)**

**Semana 03**

**Material Didático Proposto**

# Conteúdos da Aula

- ▶ **Programação Estruturada**
- ▶ **Comandos Condicionais (Decisão)**
- ▶ **Operadores Relacionais**

## »» Programação Estruturada

# Programação Estruturada

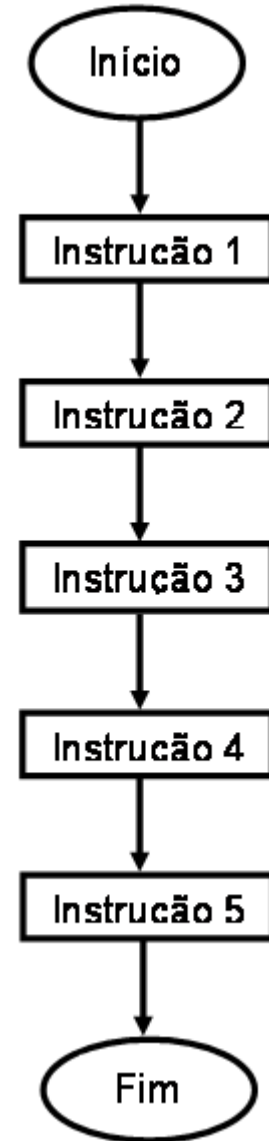
## Conceito:

**Programação estruturada é uma forma de Programação de computadores que preconiza que todos os programas possíveis podem ser reduzidos a apenas três estruturas:**

- **sequência**
- **decisão**
- **iteração**

## Sequência

**Até a última aula, os programas constituíram-se por uma sequência de instruções, ou comandos, executados sequencialmente, conforme o fluxograma ao lado.**



## Decisão (comandos condicionais)

**A segunda estrutura é utilizada quando é necessário realizar um desvio de fluxo, realizado com base em uma decisão**

**Se** "condição" for verdadeiro **Então**

**Faça a tarefa A;**

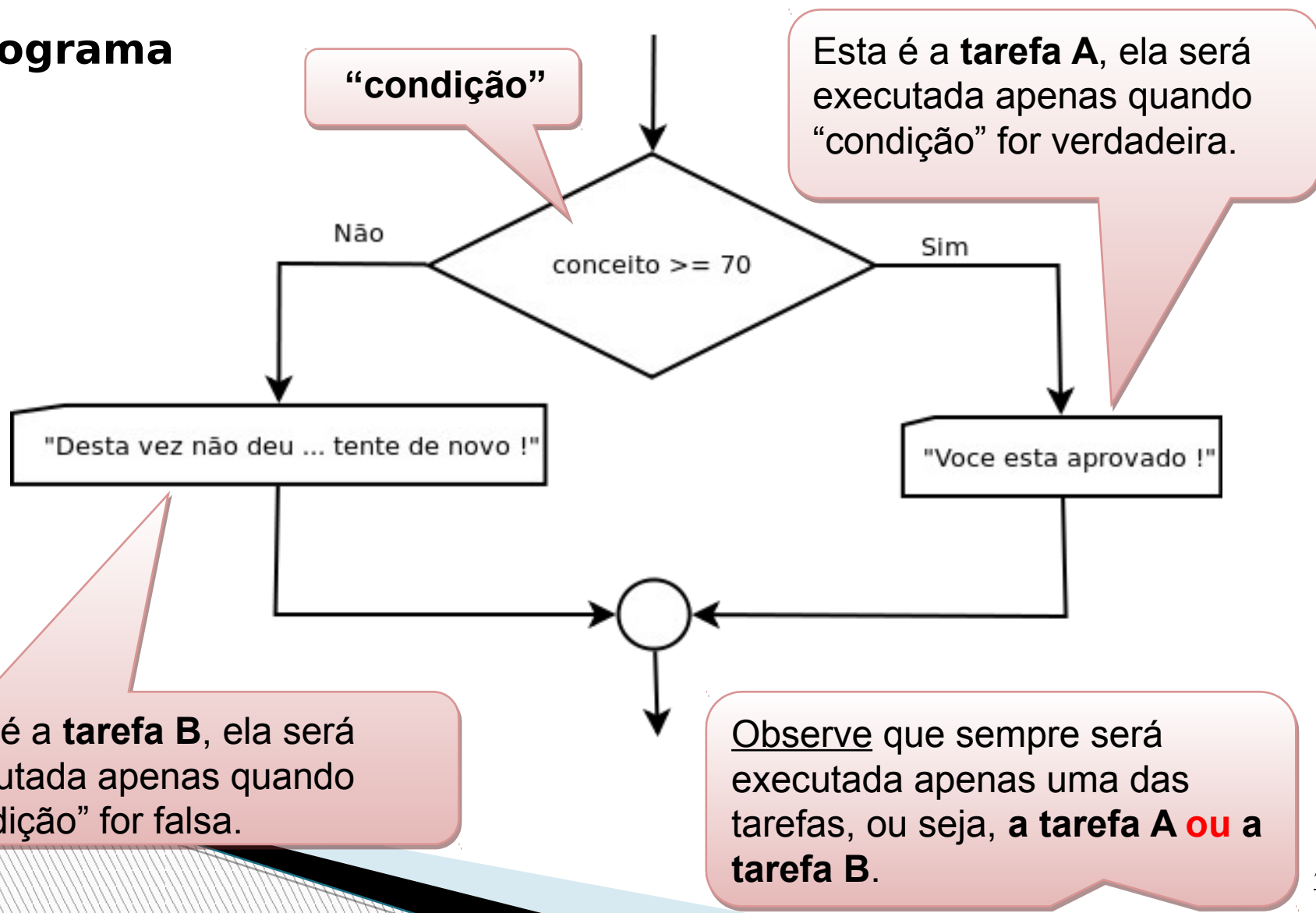
**Senão** // caso falso

**Faça a tarefa B;**

**O desvio de fluxo é caracterizado pela "escolha" (condição) entre executar a *tarefa A* ou executar a *tarefa B***

## Decisão (comandos condicionais)

### Fluxograma



## Iteração

**Repete um conjunto de instruções,  
comandos, um certo número de vezes ou  
conforme uma condição.**

**Será estudado no 2º módulo.**



# **Comandos Condicionais**

# Equações de Segundo Grau:

- ▶ Equação

$$ax^2 + bx + c = 0$$

- ▶ Raízes (reais se  $\Delta > 0$ )

$$r_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$r_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

$$\Delta = b^2 - 4ac$$

- ▶ Calcular as raízes para  $a = 534.2765$ ,  $b = 9987.3431$  e  $c = 225.7690$

# Equação de 2º grau

## Como obter a solução?

- ▶ Ler os valores dos coeficientes  $a$ ,  $b$  e  $c$
- ▶ Calcular o valor de delta
- ▶ Obter as raízes da equação
- ▶ Imprimir as raízes da equação

# Equações de Segundo Grau: Lendo os valores dos coeficientes

```
// Cálculo das raízes de equação de 2o grau  
disp("Raízes de equação de 2o grau")
```

```
// Entrada de dados
```

```
a = input("Digite o valor de a:")
```

```
b = input("Digite o valor de b:")
```

```
c = input("Digite o valor de c:")
```

Diálogo com  
o usuário

# Equações de Segundo Grau: Calculando e imprimindo as raízes

```
// Resolvendo a equação
delta = b^2 - 4*a*c
r1 = (-b + sqrt(delta)) / (2*a)
r2 = (-b - sqrt(delta)) / (2*a)

// Imprimindo resultados
printf("Raiz 1 = %g", r1)
printf("Raiz 2 = %g", r2)
```

# Execução do programa

Raízes de equação de 2o grau

Digite o valor de a: 534.2765

Digite o valor de b: 9987.3431

Digite o valor de c: 225.7690

Raiz 1 = -0.0226329

Raiz 2 = -18.6706

Os valores  
digitados pelo  
usuário estão em  
vermelho

# Teste a Solução

- ▶ Quando propomos uma solução para um problema temos que pensar em testes que verifiquem a correção do que fazemos
- ▶ Nesse caso, o teste é simples: se  $r$  é um valor calculado para uma raiz, o valor da expressão  $a*r^2 + b*r + c$  deve ser zero

# Teste

```
-->a*r1^2 + b*r1 + c
```

```
ans =
```

```
1.017D-11
```

```
-->a*r2^2 + b*r2 + c
```

```
ans =
```

```
2.888D-11
```

Notação científica:

**$2.888 \times 10^{-11}$**

(muito próximo a zero)



# Erros Comuns

- ▶ Escrever `delta = b^2 - 4ac`, omitindo os operadores de multiplicação
  - Um erro de sintaxe, que é apontado pelo Scilab
- ▶ Escrever `r1 = (-b+sqrt(delta))/2*a`, o que na verdade calcula

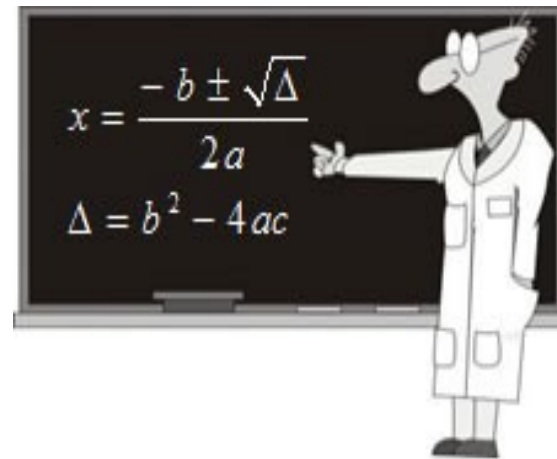
$$r_1 = \left( \frac{-b + \sqrt{\Delta}}{2} \right) a$$

- Um erro de semântica, que só pode ser descoberto por meio de testes, que o programador deve fazer

# Equações de Segundo Grau:

- ▶ Equação
- ▶ Bhaskara ( $\Delta > 0$ ,  $a \neq 0$ )

$$ax^2 + bx + c = 0$$

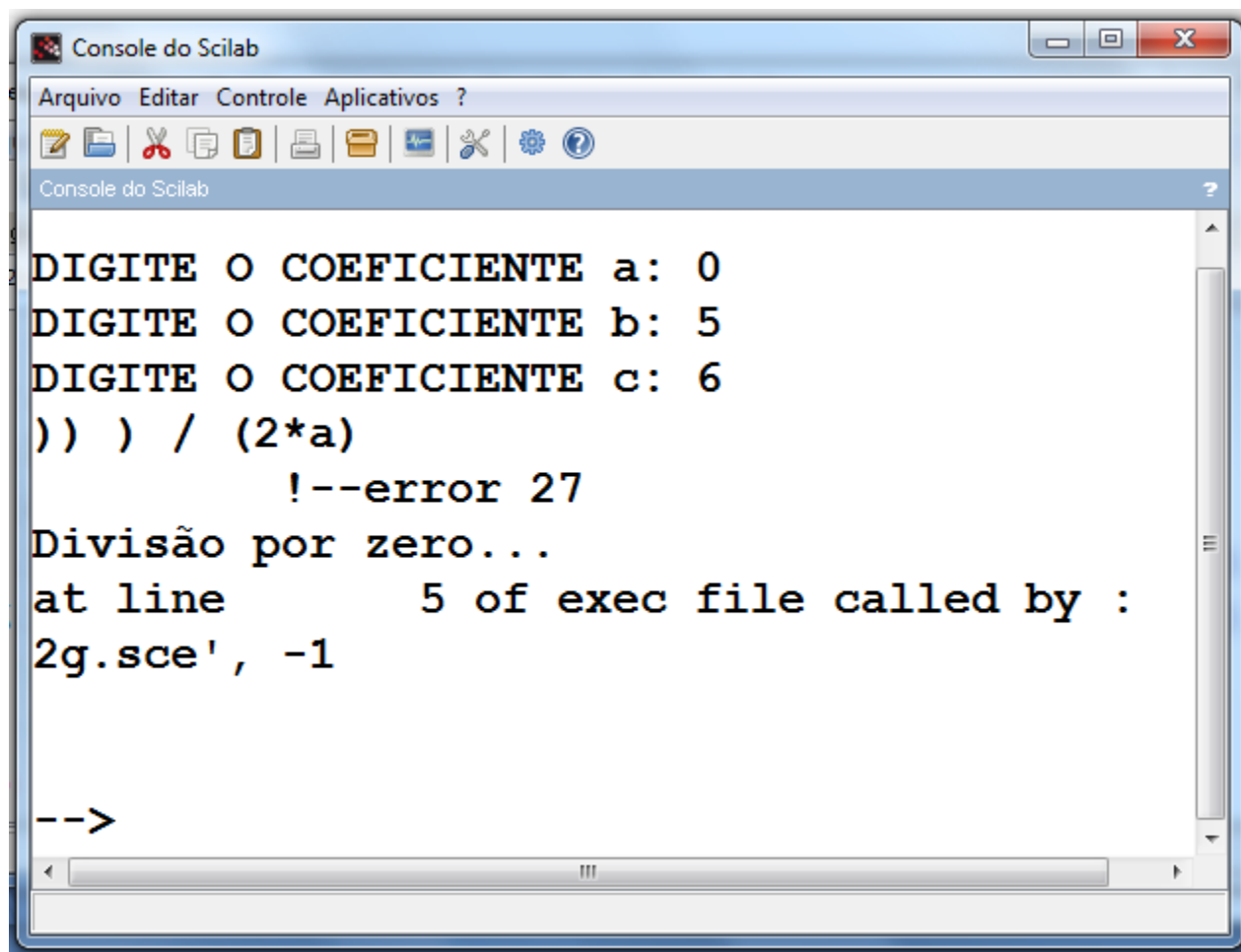


- ▶ **Quaisquer** valores de coeficiente!

# Falta analisar outras situações

- ▶  $a = 0$ 
  - Equação de primeiro grau
  - Divisão por zero no nosso programa!
- ▶  $\Delta < 0$ 
  - Raízes complexas
  - Só imprime a parte real no nosso programa!

# Executando com $a = 0$



```
Console do Scilab
Arquivo  Editar  Controle  Aplicativos  ?
[Icons]
Console do Scilab
DIGITE O COEFICIENTE a: 0
DIGITE O COEFICIENTE b: 5
DIGITE O COEFICIENTE c: 6
)) ) / (2*a)
      |--error 27
Divisão por zero...
at line      5 of exec file called by :
2g.sce', -1

-->
```

**OBS.: verificando o valor do coeficiente  $a$ , pode-se evitar este erro !**

# Analizando o valor de a

**Condição** (expressão  
relacional)

**se** a igual a 0 **então**

mensagem de erro;

resolver equação de primeiro grau;

**senão** // **caso contrário**

apresentamos as raízes reais;

# Operadores Relacionais

**<condição>** é uma expressão relacional :

**<expr 1> <operador Relacional> <expr 2>**

Onde:

<expr n> é uma expressão, que pode ser um valor numérico, ou uma expressão matemática que resulta em um valor numérico.

A avaliação de uma expressão relacional pode resultar em:

# Operadores Relacionais - Scilab

Operador	Descrição
>	Maior que.
>=	Maior ou igual a.
<	Menor que.
<=	Menor ou igual a.
==	Igual a.
<> ou ~=	Diferente de.

# Expressões com Tipo Lógico

```

-->p = %t
p =
T
-->q = 5+3 < 2
q =
F
-->a = 0
a =
0
-->a == 0
ans =
T
-->a <> 0
ans =
F

```

Literal True

Expressão lógica, usando o operador relacional <

igual

diferente

Note que operadores aritméticos têm precedência sobre operadores relacionais



# Operadores Relacionais - Exemplos

## Prioridade de Execução

- Quando temos uma combinação entre expressões matemáticas e expressões lógicas, primeiramente o Scilab calcula as expressões matemáticas; a seguir, o Scilab calcula as expressões lógicas.
- os operadores matemáticos tem maior prioridade de execução, com relação aos operadores relacionais.

# O comando **if** (versão simples)

A condição deve ser  
uma expressão lógica

```
if <condição> then  
    <bloco do então>  
end
```

## **OBSERVAÇÕES:**

um bloco é um conjunto de quaisquer comandos Scilab sintaticamente corretos (inclusive outro `if`).

**if**, **then**, **else** e **end**: são palavras reservadas do Scilab e não podem ser usadas para nomear variáveis.

# O comando **if** (completo)

```
if <condição> then  
    <bloco do então>  
else  
    <bloco do senão>  
end
```

<condição> é uma expressão relacional.  
Assim , resulta

em um valor verdadeiro (%t) ou  
falso (%f) .

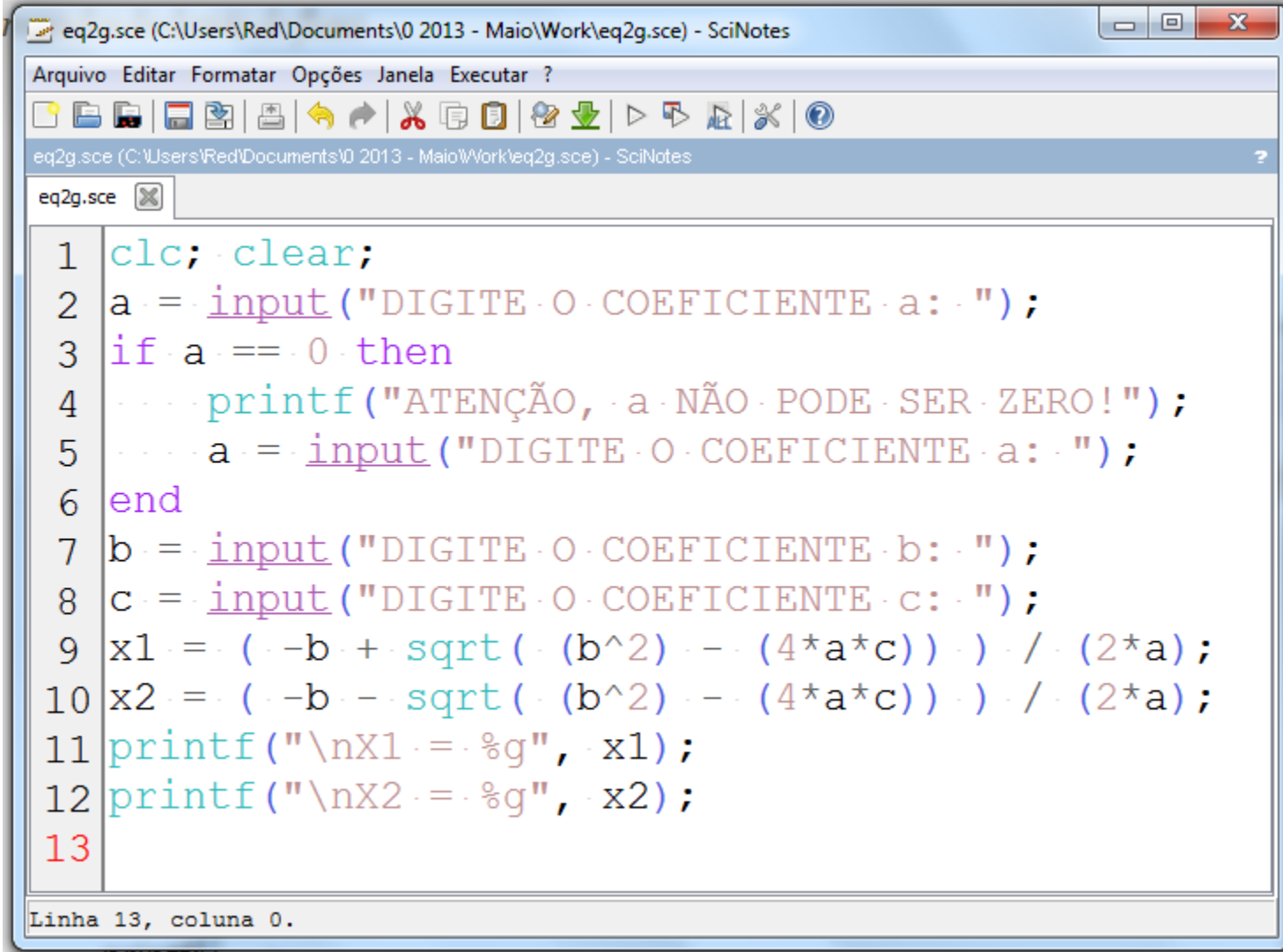
<bloco do então> será executado somente  
quando a

condição resultar em  
verdadeiro (%t) .

# Utilizando o **if** (versão simples)

```
if a == 0 then  
    printf("Coeficiente a não pode ser  
    0");  
end  
  
// programa como antes
```

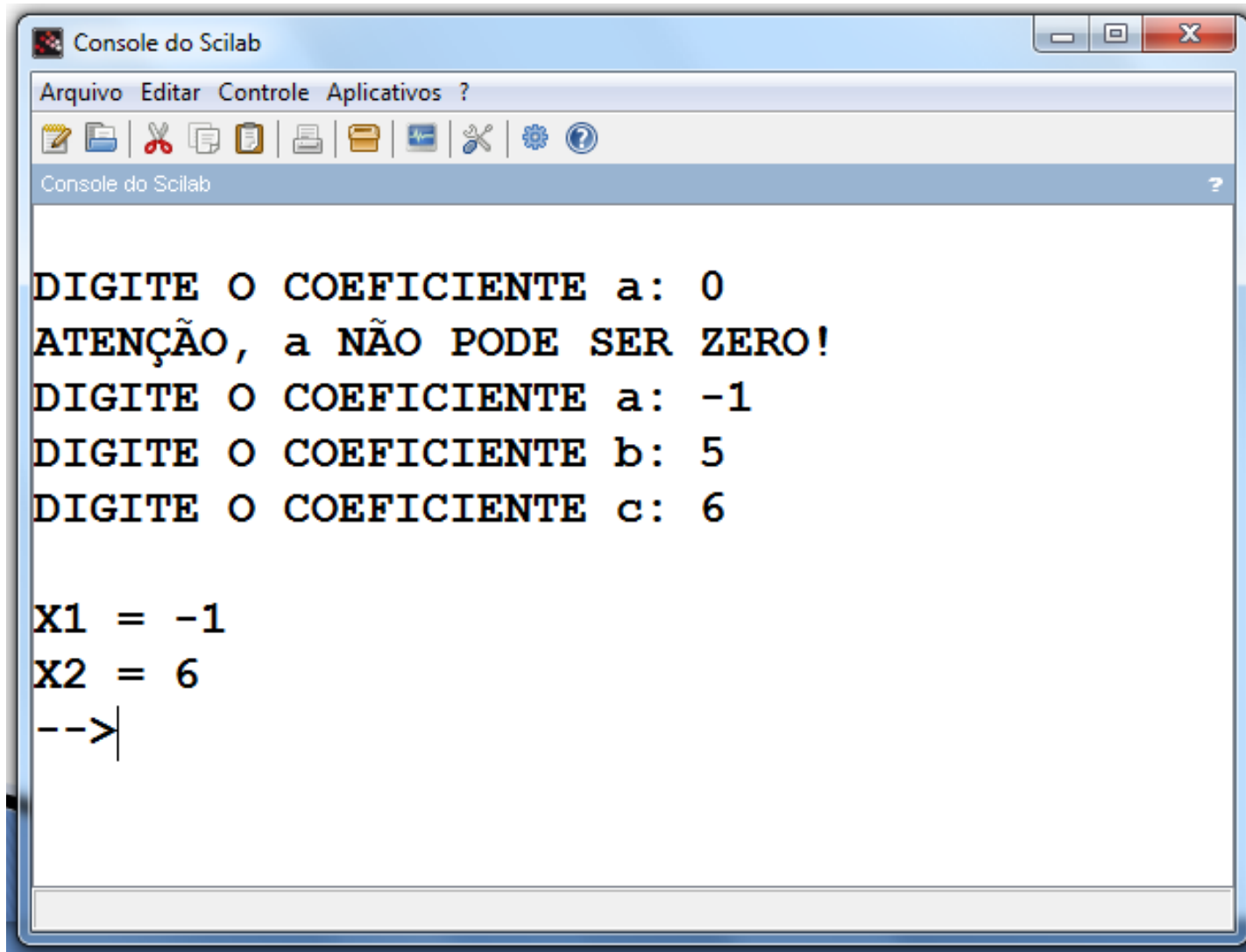
# Implementação com **if** (versão simples)



```
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
Arquivo  Editar  Formatar  Opções  Janela  Executar  ?
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
eq2g.sce
1  clc; clear;
2  a = input("DIGITE O COEFICIENTE a: ");
3  if a == 0 then
4      printf("ATENÇÃO, a NÃO PODE SER ZERO!");
5      a = input("DIGITE O COEFICIENTE a: ");
6  end
7  b = input("DIGITE O COEFICIENTE b: ");
8  c = input("DIGITE O COEFICIENTE c: ");
9  x1 = (-b + sqrt(b^2 - (4*a*c))) / (2*a);
10 x2 = (-b - sqrt(b^2 - (4*a*c))) / (2*a);
11 printf("\nX1 = %g", x1);
12 printf("\nX2 = %g", x2);
13
Linha 13, coluna 0.
```

**Agora, verifica-se o valor de a,  
antes de continuar com a  
execução.**

# Testando com **if** (versão simples)



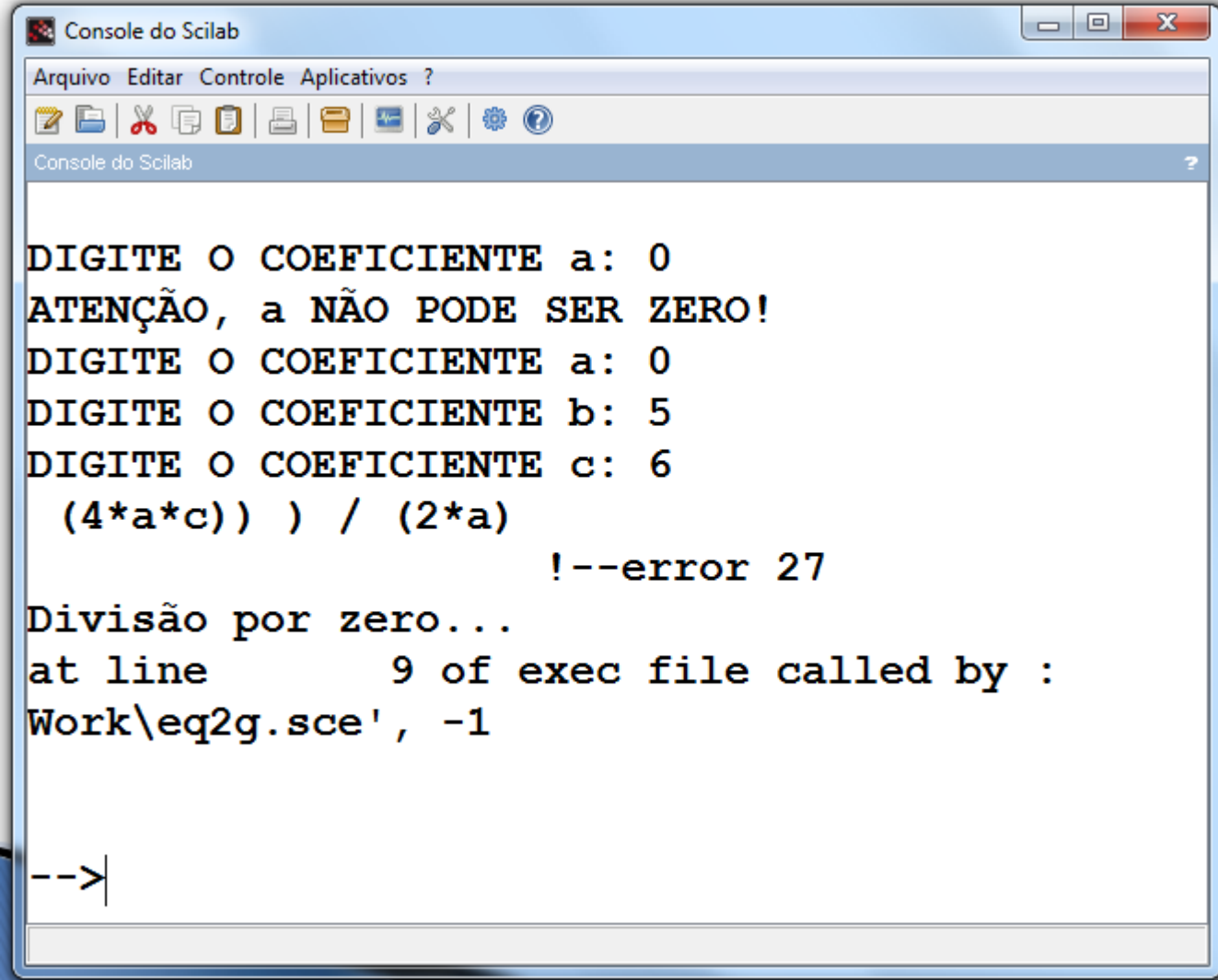
The screenshot shows the Scilab Console window with a menu bar (Arquivo, Editar, Controle, Aplicativos, ?) and a toolbar with icons for file operations and settings. The console text is as follows:

```
DIGITE O COEFICIENTE a: 0
ATENÇÃO, a NÃO PODE SER ZERO!
DIGITE O COEFICIENTE a: -1
DIGITE O COEFICIENTE b: 5
DIGITE O COEFICIENTE c: 6

X1 = -1
X2 = 6
-->
```

# Testando com **if** (versão simples)

## Nova execução, com entrada de valores dife



```
Console do Scilab
Arquivo Editar Controle Aplicativos ?
[Icons]
Console do Scilab ?

DIGITE O COEFICIENTE a: 0
ATENÇÃO, a NÃO PODE SER ZERO!
DIGITE O COEFICIENTE a: 0
DIGITE O COEFICIENTE b: 5
DIGITE O COEFICIENTE c: 6
(4*a*c)) ) / (2*a)
                                !--error 27
Divisão por zero...
at line          9 of exec file called by :
Work\eq2g.sce', -1

-->
```

**O erro ocorreu novamente, pois:**

- **O usuário teve somente duas oportunidades de inserir um valor não nulo para o coeficiente  $a$ ;**
- **Quando aprendermos a 3ª estrutura de programação estruturada, repetição, permitiremos ao usuário inserir o valor de  $a$ , quantas vezes forem necessárias, até que um valor não nulo seja fornecido.**

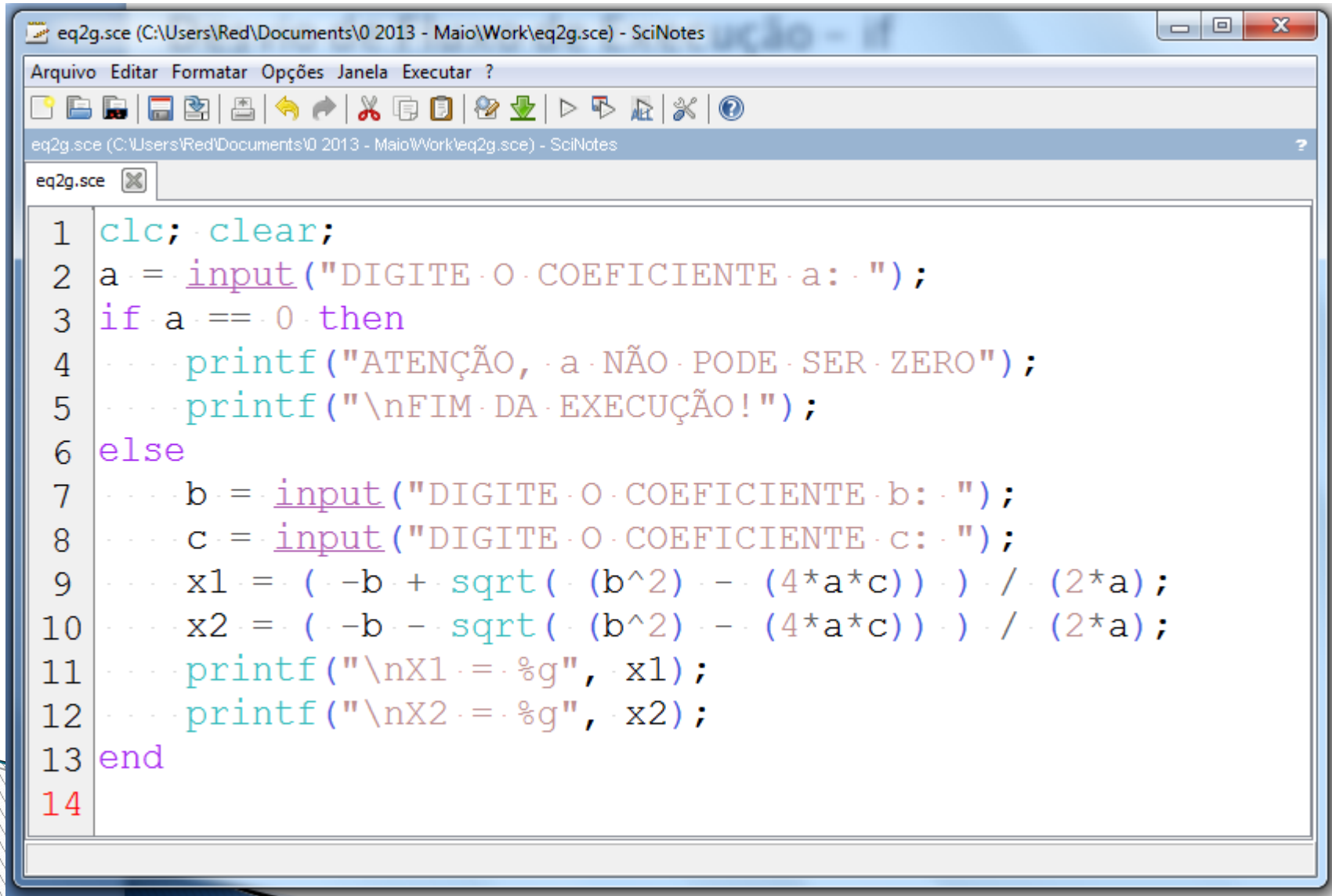


# Utilizando o **if** (completo)

```
if a == 0 then
    printf("Coeficiente a não pode ser
0");
else
    // programa como antes
end
```

# Implementação com **if** (completo)

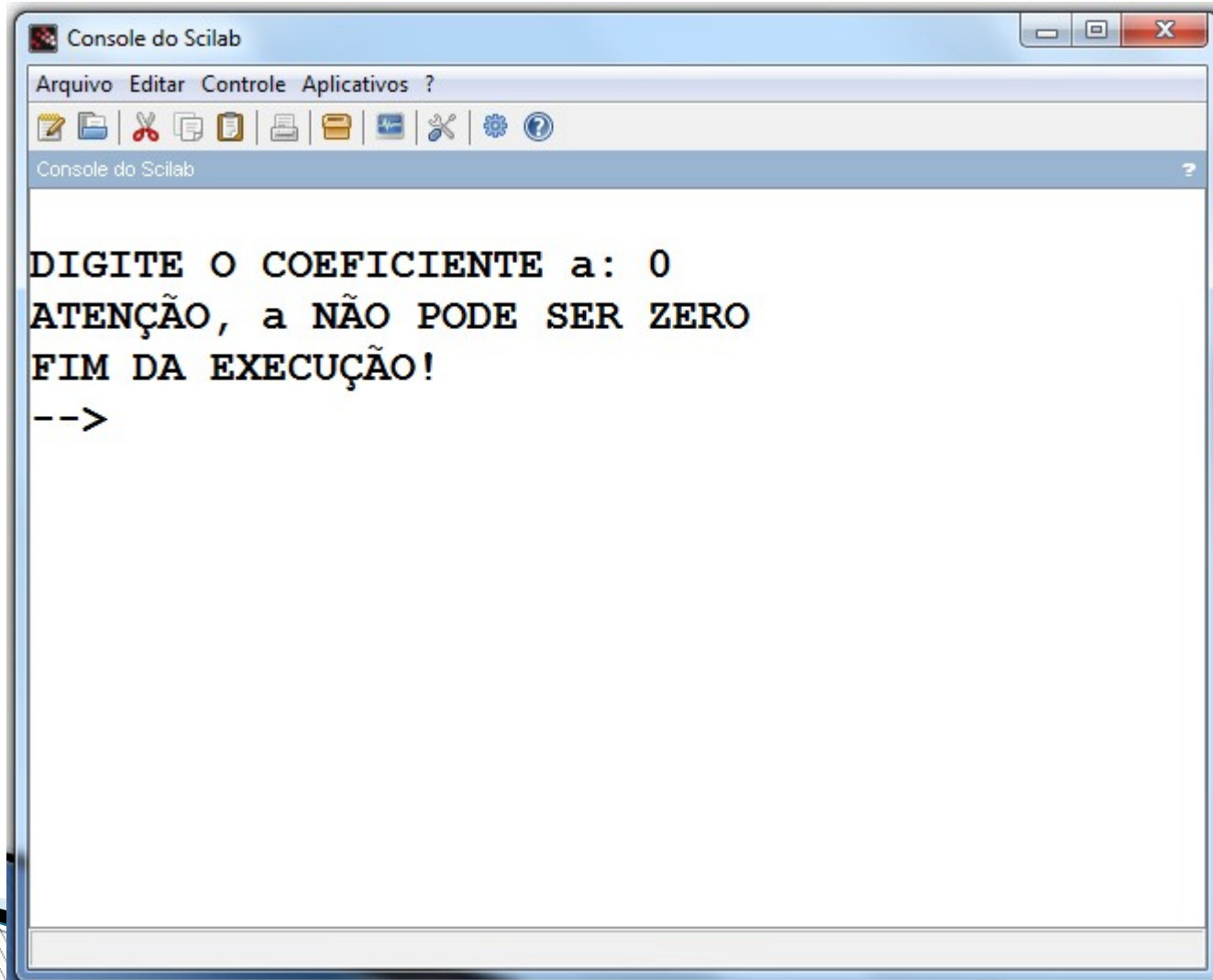
## Aprimorando a lógica da programação:



```
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
Arquivo Editar Formatar Opções Janela Executar ?
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
eq2g.sce
1 clc; clear;
2 a = input("DIGITE O COEFICIENTE a: ");
3 if a == 0 then
4     printf("ATENÇÃO, a NÃO PODE SER ZERO");
5     printf("\nFIM DA EXECUÇÃO!");
6 else
7     b = input("DIGITE O COEFICIENTE b: ");
8     c = input("DIGITE O COEFICIENTE c: ");
9     x1 = (-b + sqrt(b^2 - (4*a*c))) / (2*a);
10    x2 = (-b - sqrt(b^2 - (4*a*c))) / (2*a);
11    printf("\nX1 = %g", x1);
12    printf("\nX2 = %g", x2);
13 end
14
```

# Testando com **if** (completo)

## Exemplo 1 de execução do programa:

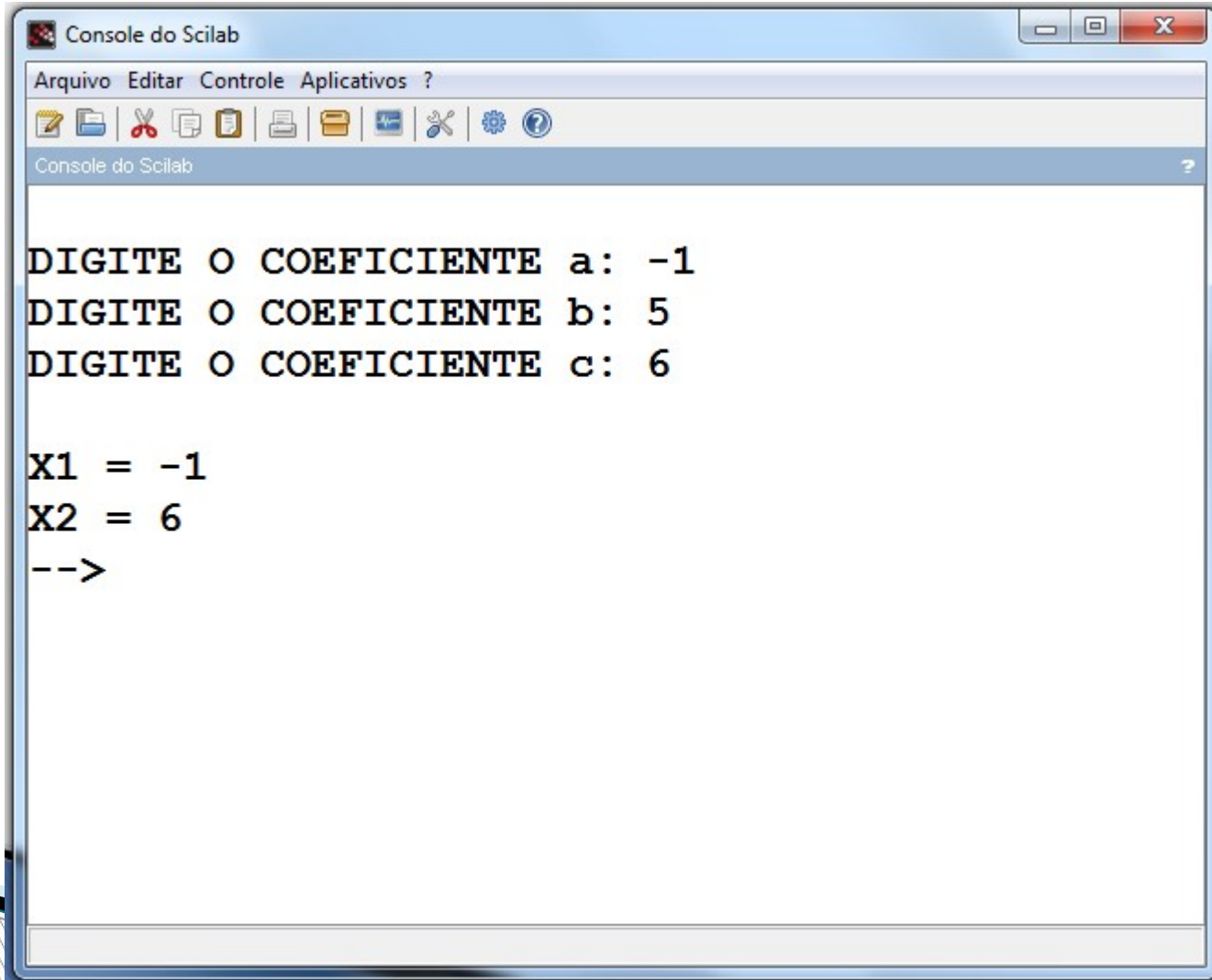


The screenshot shows a window titled "Console do Scilab" with a menu bar (Arquivo, Editar, Controle, Aplicativos, ?) and a toolbar. The console area contains the following text:

```
DIGITE O COEFICIENTE a: 0  
ATENÇÃO, a NÃO PODE SER ZERO  
FIM DA EXECUÇÃO!  
-->
```

# Testando com **if** (completo)

## Exemplo 2 de execução do programa:



The screenshot shows the 'Console do Scilab' window. The title bar includes standard window controls (minimize, maximize, close). The menu bar contains 'Arquivo', 'Editar', 'Controle', and 'Aplicativos ?'. The toolbar has icons for file operations (new, open, save, print, copy, paste, delete), editing (undo, redo), and help. The main text area displays the following text:

```
DIGITE O COEFICIENTE a: -1
DIGITE O COEFICIENTE b: 5
DIGITE O COEFICIENTE c: 6

X1 = -1
X2 = 6
-->
```

# Analizando o valor do cálculo de delta : $\Delta < 0$

**Se** delta for positivo **Então**  
apresentamos as raízes reais;  
**Senão** // caso contrário  
apresentamos as raízes complexas;

# Números Complexos

**O Scilab possibilita manipular números complexos de forma simples.**

**`real(x1)` → retorna a parte real.**

**`imag(x1)` → retorna a parte imaginária.**

**Exemplos de números complexos:**

**`z1 = 3 + 4 * %i;`**

**`z2 = 1 - %i;`**

**`z3 = z1 + z2;`**

**`z4 = z1 * z2;`**

# Implementação

```
if (delta >= 0) then
    printf("\nX1 = %g", x1);
    printf("\nX2 = %g", x2);
else
    printf("PRIMEIRA RAIZ\n")
    printf("%g + %g.i", real(x1), imag(x1));
    printf("\nSEGUNDA RAIZ\n")
    printf("%g + %g.i", real(x2), imag(x2));
end
```

- quando delta for maior ou igual a zero, a expressão relacional resultará em **%t** e, portanto, somente o **bloco do então** (then) será executado;
- quando delta for menor que zero, a expressão relacional resultará em **%f** e, portanto, somente o **bloco do senão** (else) será executado.

# Implementação completa

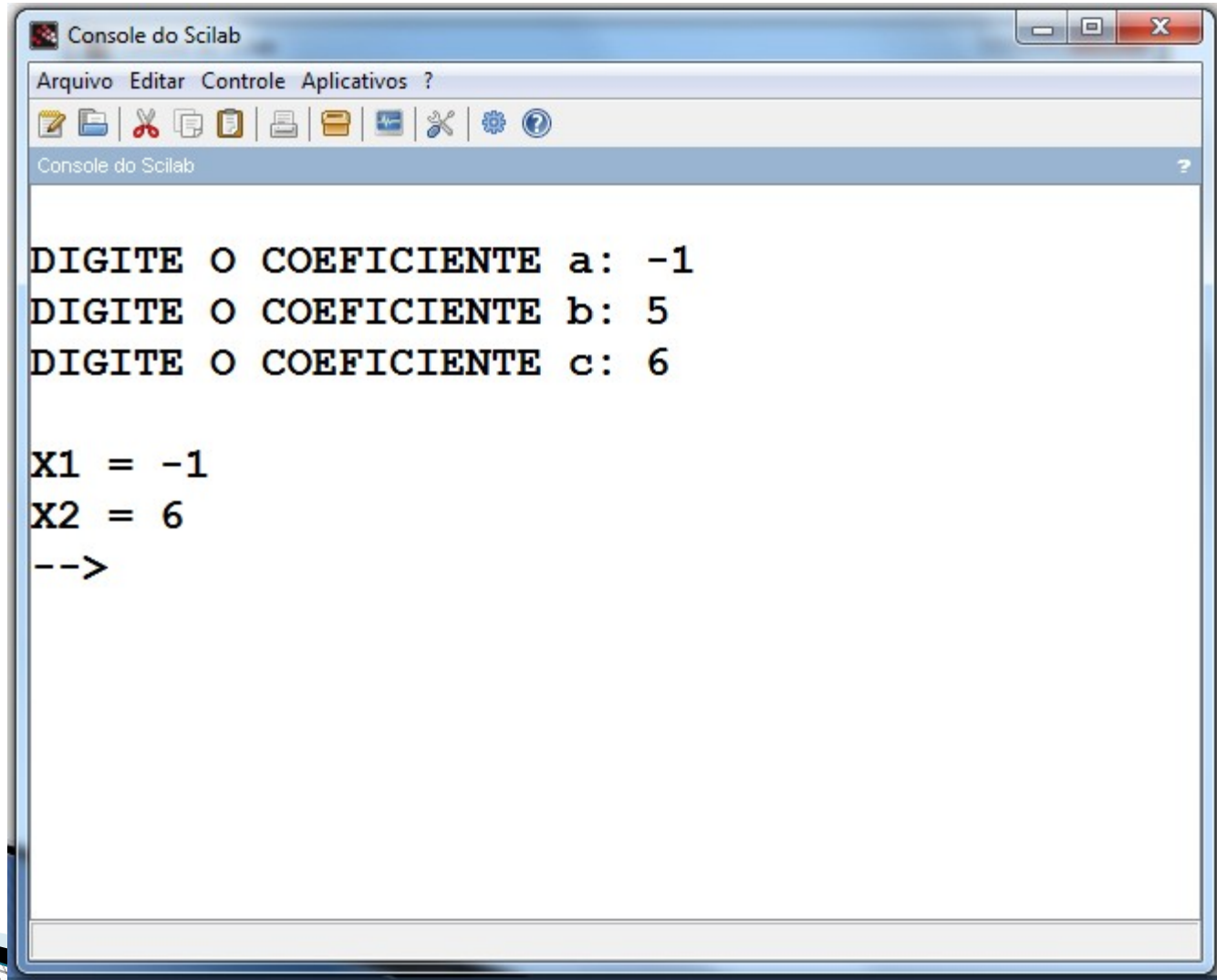
```
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
Arquivo  Editar  Formatar  Opções  Janela  Executar  ?
[Icons]
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
eq2g.sce [X]

1  clc; clear;
2  a = input("DIGITE O COEFICIENTE a: ");
3  if a == 0 then
4      printf("ATENÇÃO, a NÃO PODE SER ZERO");
5      printf("\nFIM DA EXECUÇÃO!");
6  else
7      b = input("DIGITE O COEFICIENTE b: ");
8      c = input("DIGITE O COEFICIENTE c: ");
9      delta = b^2 - 4*a*c;
10     x1 = (-b + sqrt(delta)) / (2*a);
11     x2 = (-b - sqrt(delta)) / (2*a);
12     if (delta >= 0) then
13         printf("\nX1 = %g", x1);
14         printf("\nX2 = %g", x2);
15     else
16         printf("PRIMEIRA RAIZ\n")
17         printf("%g + %g.i", real(x1), imag(x1));
18         printf("\nSEGUNDA RAIZ\n")
19         printf("%g + %g.i", real(x2), imag(x2));
20     end
21 end
```



# Testando

## Exemplo 1 de execução do programa:



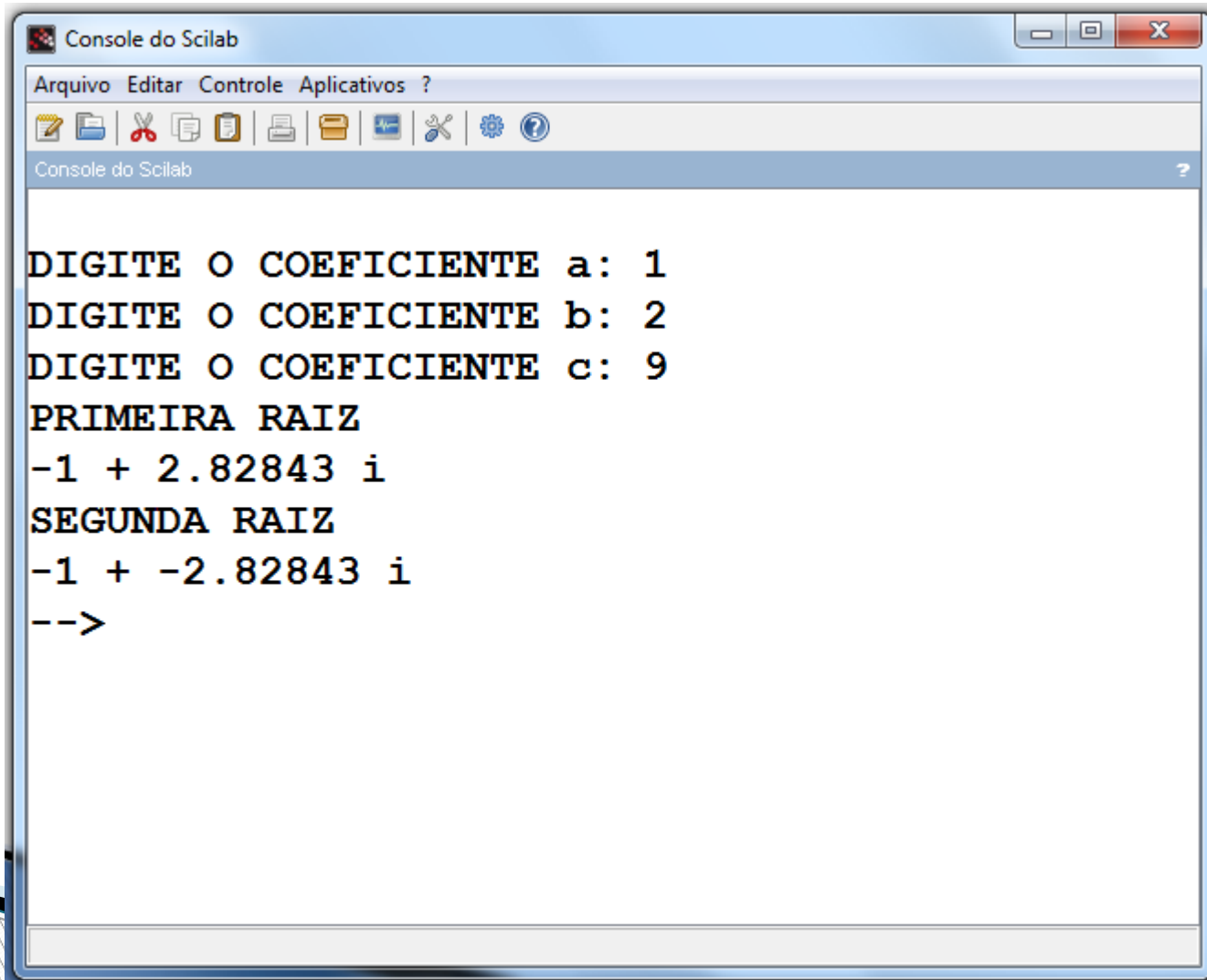
The image shows a screenshot of the Scilab Console window. The window has a title bar that says "Console do Scilab" and standard Windows window controls (minimize, maximize, close). Below the title bar is a menu bar with "Arquivo", "Editar", "Controle", and "Aplicativos ?". Underneath the menu bar is a toolbar with various icons for file operations and editing. The main area of the window is a text editor displaying the following text:

```
DIGITE O COEFICIENTE a: -1
DIGITE O COEFICIENTE b: 5
DIGITE O COEFICIENTE c: 6

X1 = -1
X2 = 6
-->
```

# Testando

## Exemplo 2 de execução do programa:

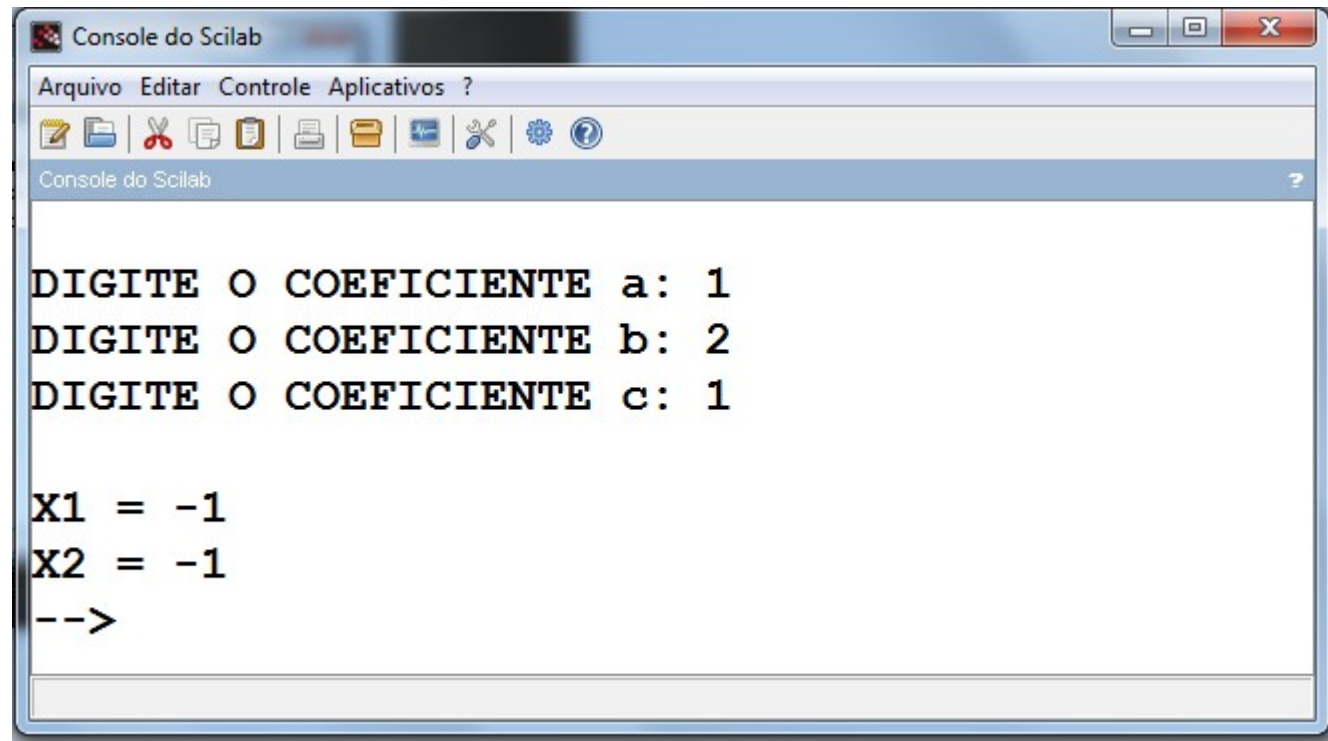


```
Console do Scilab
Arquivo  Editar  Control  Aplicativos ?
[Icons]
Console do Scilab ?

DIGITE O COEFICIENTE a: 1
DIGITE O COEFICIENTE b: 2
DIGITE O COEFICIENTE c: 9
PRIMEIRA RAIZ
-1 + 2.82843 i
SEGUNDA RAIZ
-1 + -2.82843 i
-->
```

# Exercício

Considerando o programa que calcula a equação do segundo grau, observamos que quando as duas raízes são iguais, o programa calcula e imprime  $x_1$  e  $x_2$  com os mesmos valores.



```
Console do Scilab
Arquivo  Editar  Controle  Aplicativos  ?
[Icons]
Console do Scilab

DIGITE O COEFICIENTE a: 1
DIGITE O COEFICIENTE b: 2
DIGITE O COEFICIENTE c: 1

X1 = -1
X2 = -1
-->
```

Como poderíamos resolver isso?

# Solução (onde está o problema)

```
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
Arquivo  Editar  Formatar  Opções  Janela  Executar  ?
eq2g.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\eq2g.sce) - SciNotes
eq2g.sce
1  clc; clear;
2  a = input("DIGITE O COEFICIENTE a: ");
3  if a == 0 then
4      printf("ATENÇÃO, a NÃO PODE SER ZERO");
5      printf("\nFIM DA EXECUÇÃO!");
6  else
7      b = input("DIGITE O COEFICIENTE b: ");
8      c = input("DIGITE O COEFICIENTE c: ");
9      delta = b^2 - 4*a*c;
10     x1 = (-b + sqrt(delta)) / (2*a);
11     x2 = (-b - sqrt(delta)) / (2*a);
12     if (delta >= 0) then
13         printf("\nX1 = %g", x1);
14         printf("\nX2 = %g", x2);
15     else
16         printf("PRIMEIRA RAIZ\n");
17         printf("%g + %g.i", real(x1), imag(x1));
18         printf("\nSEGUNDA RAIZ\n");
19         printf("%g + %g.i", real(x2), imag(x2));
20     end
21 end
```

```
if delta == 0 then
    printf("AS RAÍZES SÃO IGUAIS: %g", x1);
else
    if (delta >= 0) then
        printf("\nX1 = %g", x1);
        printf("\nX2 = %g", x2);
    else
        printf("PRIMEIRA RAIZ\n")
        printf("%g + %g.i", real(x1), imag(x1));
        printf("\nSEGUNDA RAIZ\n")
        printf("%g + %g.i", real(x2), imag(x2));
    end
end
```

# Exemplo 2

- ▶ Faça um programa que:
  - Leia o nome do usuário
  - Leia o total de pontos feitos pelo usuário
  - Imprima, conforme o caso, a frase
    - ▢ <usuário>, com <pontos> você passou!
  - Ou
    - ▢ <usuário>, com <pontos> você não passou!
  - Ex.: **José, com 75 pontos você passou!**

# Programa PassouNaoPassou.sce

```
// Leitura do nome do usuário
Nome = input("Digite seu nome, por favor")

// Leitura da pontuação
Pontos = input(Nome + ", qual foi sua pontuação? ")

// Impressão do resultado
if Pontos >= 60 then
    printf("%s, com %g pontos você passou!", Nome, Pontos);
else
    printf("%s, com %g pontos você não passou : (", ...
    Nome, Pontos);
end
```

Comando continua  
na próxima linha

# Um jovem programador

Certa vez a mãe disse ao filho estudante de computação:

“Filho, por favor vá ao mercado e compre 1 caixa de leite. Se eles tiverem ovos, traga 6.

Ele retornou com 6 caixas de leite.

A mãe disse: "Porque diabos você comprou 6 caixas de leite?".

Ele disse: "PORQUE ELES TINHAM OVOS!".



# O raciocínio na ambiguidade

**se** tiverem ovos **então**  
traga 6 caixas de leite;  
**senão**  
traga 1 caixa de leite;



<http://www.hardware.com.br/artigos/computadores-receitas-bolo-aprendendo-comunicar-com-maquina/aprendendo-pensar-como-maquina.html>

## » Exercícios

# Exercício 1

**Codifique um programa que calcule o volume de uma pirâmide, em  $\text{cm}^3$ , através da fórmula:**

$$\text{Volume} = 1/3 * \text{ÁreaBase} * \text{altura}$$

**onde**

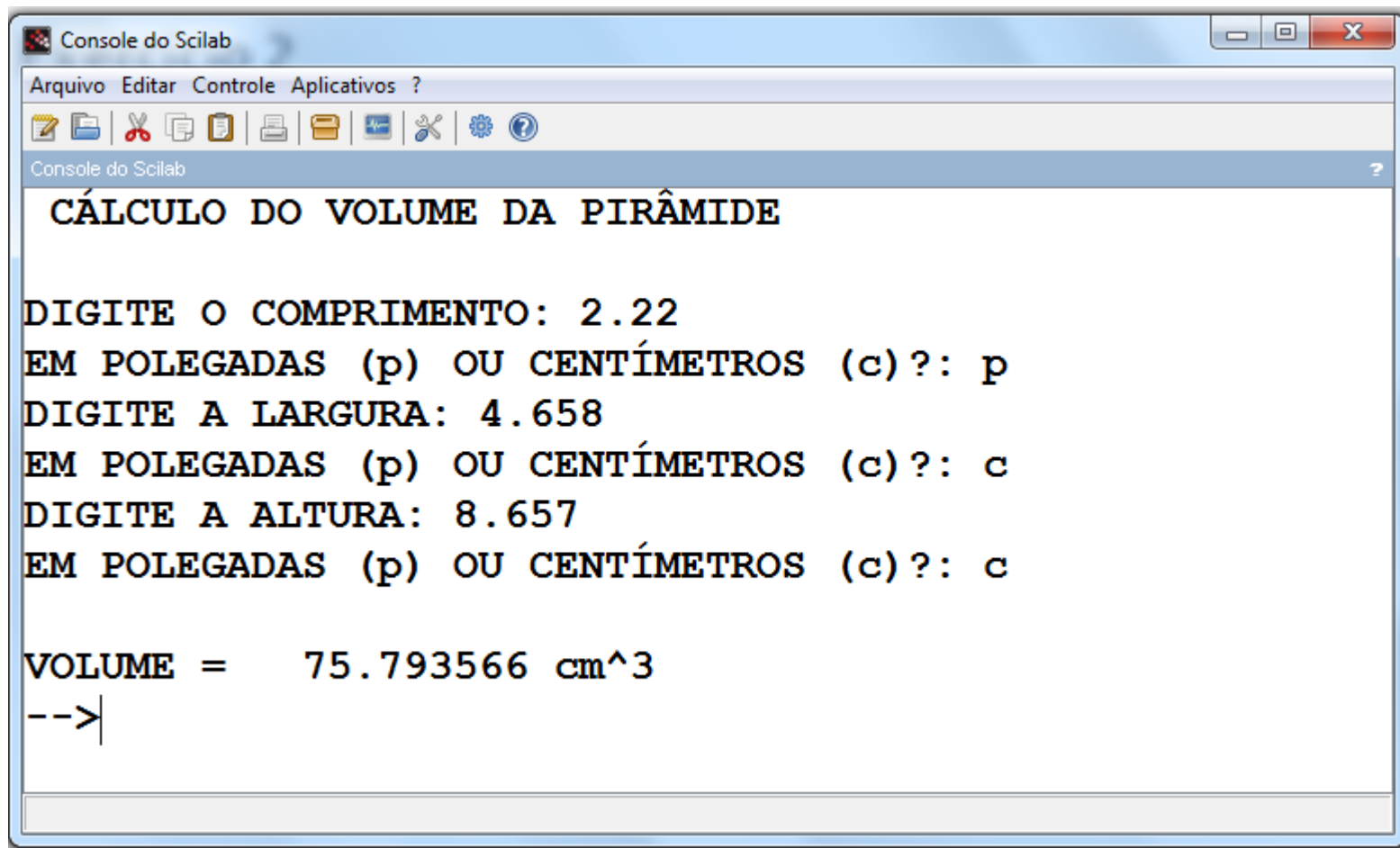
$$\text{ÁreaBase} = \text{comprimento} * \text{largura}$$

**O usuário deve fornecer os valores do comprimento, da largura e da altura. Ao entrar um valor, ele também será solicitado a indicar se o valor digitado foi em polegadas ('p') ou em centímetros ('c'). Quando a entrada for em polegadas, o programa a converte automaticamente para centímetros.**

**Ao final, o programa imprime o volume calculado.**

**Observação: 3.54 cm = 1 polegada**

# Exercício 1 - Solução



Console do Scilab

Arquivo Editar Controle Aplicativos ?

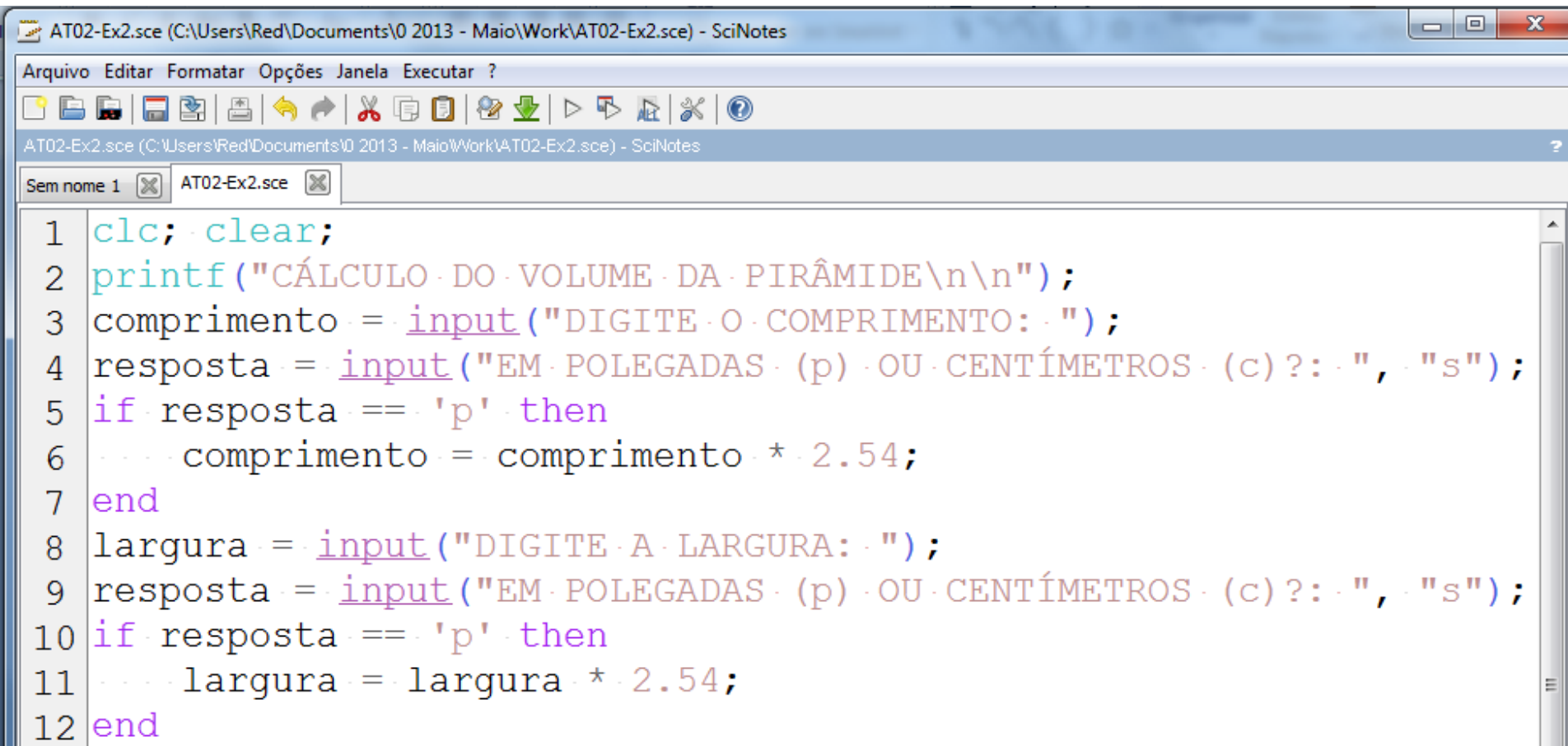
Console do Scilab

```
CÁLCULO DO VOLUME DA PIRÂMIDE

DIGITE O COMPRIMENTO: 2.22
EM POLEGADAS (p) OU CENTÍMETROS (c)?: p
DIGITE A LARGURA: 4.658
EM POLEGADAS (p) OU CENTÍMETROS (c)?: c
DIGITE A ALTURA: 8.657
EM POLEGADAS (p) OU CENTÍMETROS (c)?: c

VOLUME =    75.793566 cm^3
-->
```

# Exercício 1 - Solução



AT02-Ex2.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\AT02-Ex2.sce) - SciNotes

Arquivo Editar Formatar Opções Janela Executar ?

AT02-Ex2.sce (C:\Users\Red\Documents\0 2013 - Maio\Work\AT02-Ex2.sce) - SciNotes

Sem nome 1 AT02-Ex2.sce

```
1 clc; clear;
2 printf("CÁLCULO DO VOLUME DA PIRÂMIDE\n\n");
3 comprimento = input("DIGITE O COMPRIMENTO: ");
4 resposta = input("EM POLEGADAS (p) OU CENTÍMETROS (c)?: ", 's');
5 if resposta == 'p' then
6     comprimento = comprimento * 2.54;
7 end
8 largura = input("DIGITE A LARGURA: ");
9 resposta = input("EM POLEGADAS (p) OU CENTÍMETROS (c)?: ", 's');
10 if resposta == 'p' then
11     largura = largura * 2.54;
12 end
```

# Exercício 1 - Solução

```
13 altura = input("DIGITE A ALTURA: ");
14 resposta = input("EM POLEGADAS (p) OU CENTÍMETROS (c)?: ", "s");
15 if resposta == 'p' then
16     altura = altura * 2.54;
17 end
18 areaBase = comprimento * largura;
19 volumePiramide = 1/3 * areaBase * altura;
20 printf("\nVOLUME = %11.6f cm^3", volumePiramide);
21
```

## Exercício 2

Na química, o pH de uma solução aquosa é medido por sua acidez.

A escala do pH varia entre 0 e 14, inclusive. Uma solução com pH igual a 7 é dita neutra; uma solução com o pH maior que 7 é dita básica; e uma solução com o pH menor que 7 é dita ácida.

Codifique um programa que tenha como entrada o pH de uma solução. O programa imprime se o pH é neutro, básico ou ácido.

## Exercício 3

**Codifique um programa que converta uma temperatura em graus celsius para graus kelvin, ou para graus fahrenheit.**

**Após o usuário fornecer a temperatura em celsius, o usuário deve responder 'f' para conversão em fahrenheit, ou 'k' para conversão em kelvin.**

**Fórmulas:**

$$F = \frac{9}{5}C + 32$$

$$K = C + 273.15$$



## Exercício 4

**Codifique um programa que gere um valor inteiro aleatório. A seguir o programa imprime a mensagem**

**"O NÚMERO GERADO É PAR"**

**caso o número gerado seja par;**

**caso contrário imprime a mensagem:**

**"O NÚMERO GERADO É ÍMPAR"**

**Dica:**

**`floor(rand() * 10)`**