Universidade Federal de Ouro Preto - UFOP
Departamento de Computação - DECOM
Comissão da Disciplina Programação de Computadores I – CDPCI
Programação de Computadores I – BCC701
www.decom.ufop.br/bcc701

Aula Teórica 02

Material Didático Unificado.

Conteúdos da Aula

- Variáveis
- Instrução de Atribuição
- Expressão da Linguagem
- Operadores Aritméticos
- Funções Elementares
- Valores Predefinidos
- Precedência e Associatividade de Operadores
- Instruções de Entrada de Dados e Saída de Dados
- Exercícios

Variáveis

Definição

- Variáveis correspondem a nomes para endereços de memória que são gerenciados pelo Scilab.

- Os endereços indicam a localização do local de armazenamento das informações na memória.

- O programador não precisa ter qualquer ideia de como tal gerência é realizada.

Nomes de Variáveis

- Para dar nomes a variáveis, algumas regras deve ser seguidas:
  - **Não podem conter acentos e nem espaços em branco;**
  - **Não podem iniciar com números;**
  - **Além das letras e caracteres alpha numéricos, pode conter os seguintes caracteres: # $ _ ? ! (*Nesta linguagem*)**
- É recomendado que variáveis tenham nomes significativos.
- Scilab é sensível a maiúsculas e minúsculas, ou seja: Nome ≠ nome ≠ NOME

Nomes de Variáveis

- A escolha de nomes significativos para as variáveis ajuda ao programador entender o que o programa faz e a prevenir erros.
- Nomes válidos:

  a

  total_de_alunos

  #funcionarios

  %valor

- Nomes inválidos

  1Aluno (o primeiro caractere é um algarismo)

  total de alunos (tem espaços)

  José (é acentuado)

Instruções de atribuição

Instrução de Atribuição

- Uma instrução de atribuição armazena um valor na memória. Sua forma é:


```
x = 2*12 + 6;
```

- **<variável>**: se não existia, passa a existir na memória; se existia, o antigo valor será substituído pelo valor corrente definido pela expressão.

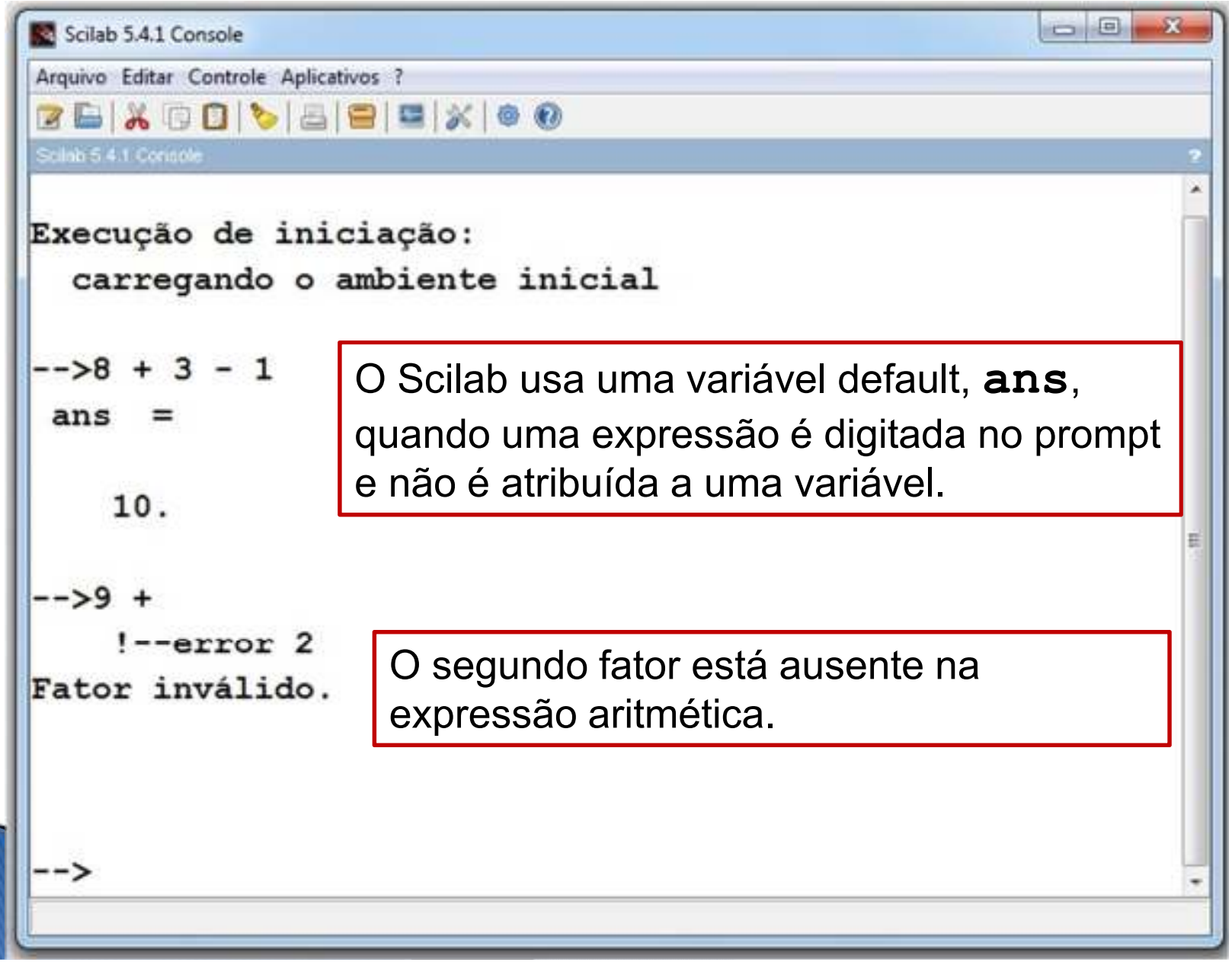
- na execução da instrução, a <expressão> é calculada e o resultado é atribuído à **<variável>**. Ou seja, *a variável recebe o valor resultante da expressão.*

Expressão da Linguagem

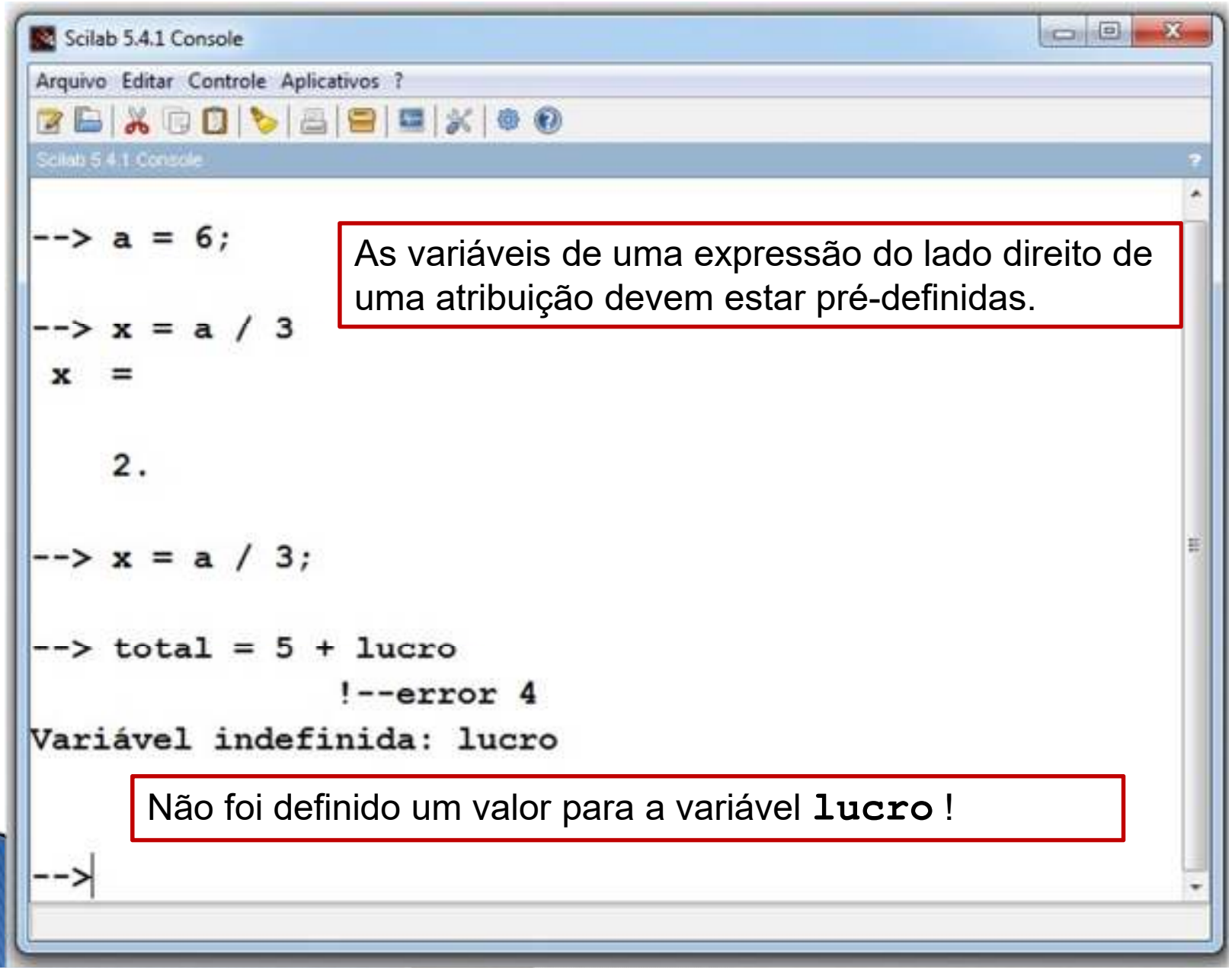
Uma expressão é:

- um valor numérico: 2 ou 2.7698 ou 0.00023
- uma função elementar do Scilab: sin, cos, etc.
- variáveis previamente definidas.
- uma expressão entre parênteses.
- uma expressão aritmética: composição de duas, ou mais, expressões e operadores aritméticos.

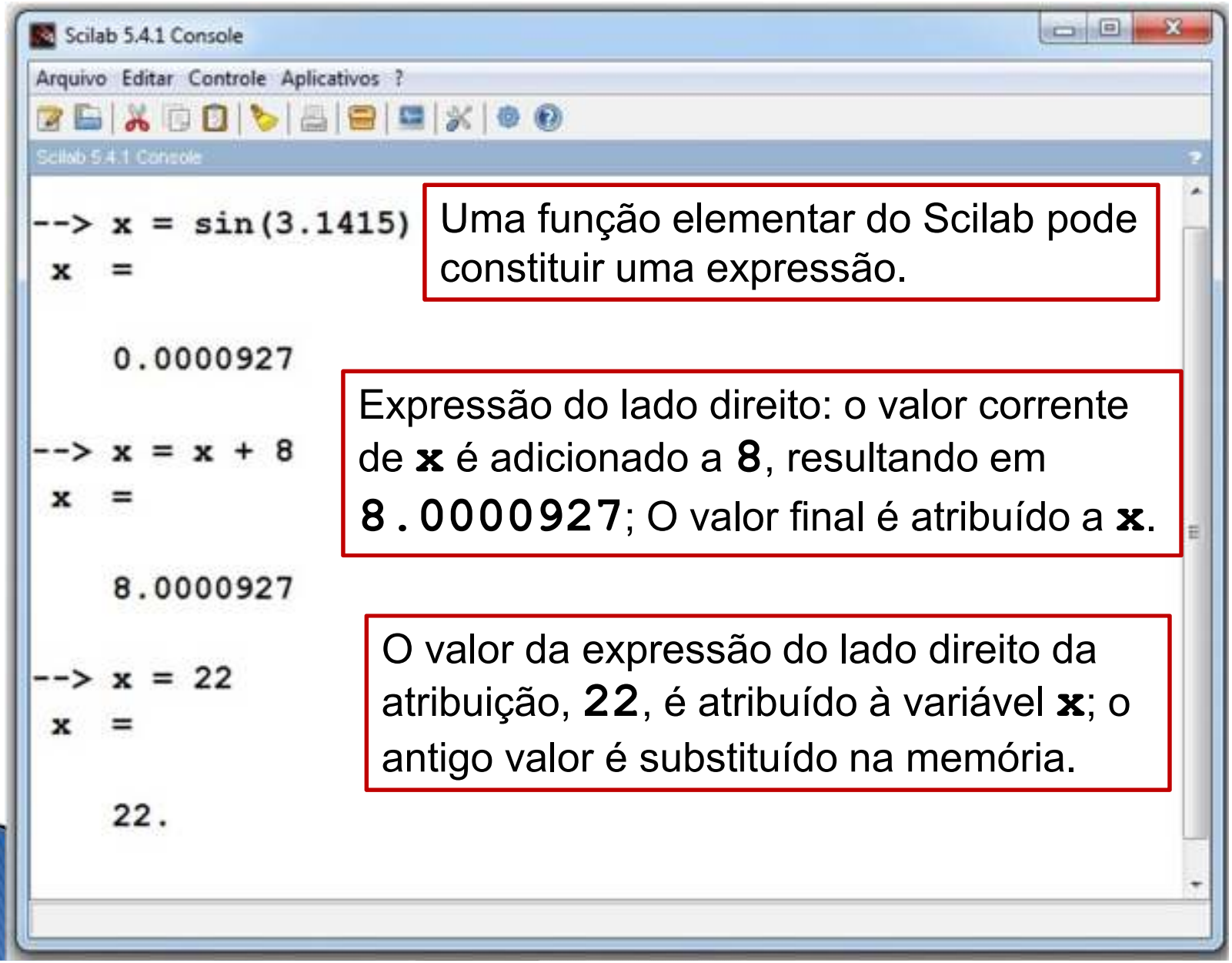
Expressão – Variável – Atribuição



Expressão – Variável – Atribuição



Expressão – Variável – Atribuição



Uma função elementar do Scilab pode constituir uma expressão.

Expressão do lado direito: o valor corrente de **x** é adicionado a **8**, resultando em **8.0000927**; O valor final é atribuído a **x**.

O valor da expressão do lado direito da atribuição, **22**, é atribuído à variável **x**; o antigo valor é substituído na memória.

Operadores aritméticos
Funções elementares
Valores Predefinidos

Operadores Aritméticos

- A linguagem SciLab possui os **operadores aritméticos**:

Operador Aritmético	Denotação em SciLab	Exemplo	Resultado
Soma	+	7 + 5	12
Subtração	-	10 - 9	1
Multiplicação	*	22 * 10	220
Divisão	/	50 / 2	25
Menos Unário Inversão do sinal	-	-26	-26
Exponenciação (potenciação)	^	8^2	64

Funções Elementares

- São exemplos de funções implementadas no SciLab:

Função	Denotação em SciLab	Exemplo	Resultado
Resto da Divisão Inteira	modulo	modulo(8, 3)	2
Raiz Quadrada	sqrt	sqrt(32)	5.6568542
		sqrt(16)	4.0
Valor Absoluto	abs	abs(-8)	8
Seno	sin	sin(%pi)	1.225D-16
Coseno	cos	cos(30)	0.1542514
Tangente	tan	tan(7.3456)	1.7945721
Inteiro	int	int(7.234)	7
Teto	ceil	ceil(3.2)	4
Piso	floor	floor(2.7)	2

OBS: Nas funções trigonométricas os ângulos devem ser usados em radianos.

Valores Pré-Definidos

- O SciLab possui alguns valores pré-definidos, alguns exemplos:

Denotação em Scilab	Valor
%pi	O número π.
%inf	Representa infinito ∞.
%i	$\sqrt{-1}$
%e	A base do logaritmo natural.
%t ou %T	Representa o valor booleano verdadeiro.
%f ou %F	Representa o valor booleano falso.

- Como o Scilab é sensível a maiúsculas e minúsculas, não será possível usar %PI, %Pi, %Inf, ou qualquer variação desta natureza.

Precedência e associatividade de operadores

Precedência de Operadores

- A precedência de operadores indica qual operador deverá ser executado primeiro.

- Assim, na expressão aritmética 2 + 3 * 6 , a subexpressão 3 * 6 é executada primeiro.

- Portanto, tem-se como resultado para a expressão o valor 20.

Precedência de Operadores

- Para a expressão:

  `2^3*4`

  o valor resultante será:

  `2^12 = 4096 ?`

  ou o valor será:

  `2^3 * 4 = 8 * 4 = 32 ?`

- Para respondermos esta pergunta, além do conhecimento da precedência (prioridade) dos operadores envolvidos, devemos saber também qual são as suas associatividades.

Precedência de Operadores

- A tabela abaixo define a precedência e a associatividade para alguns operadores:

Prioridade	Operação	Associatividade
1ª	^	Da direita para a esquerda.
2ª	* /	Da esquerda para a direita.
3ª	+ -	Da esquerda para a direita.

- Exemplos:
  - 2+10/5 → 10/5 é avaliada primeiro
  - A+B/C+D → B/C é avaliada primeiro
  - R*3+B^3/2+1 → B^3 é avaliada primeiro
  - 2^3^2 → 3^2 é avaliado primeiro

Associatividade de Operadores

- Associatividade é a regra usada quando os operadores têm a mesma prioridade.
- Por exemplo, para as operações de adição e subtração (que possuem mesma prioridade) a regra de associatividade diz que a operação mais a esquerda é avaliada primeiro:

  A−B+C+D → A−B é avaliada primeiro, pois está mais à esquerda.

O mesmo vale para multiplicação e divisão.

Associatividade de Operadores

- Mas, para potenciação, a regra da associatividade diz que a operação mais a direita deve ser avaliada primeiro:

A^B^C^D → C^D é avaliada primeiro, pois está mais à direita.

Quebra da Precedência

- A precedência de operadores pode ser alterada mediante o uso de parênteses. Ex:
  - `(A + 4) / 3`

    A + 4 é avaliada primeiro
  - `(A - B) / (C + D)`

    A - B é avaliada primeiro, depois a soma e por último a divisão
  - `R * 3 + B^(3 / 2) + 1`

    3 / 2 é avaliada primeiro

Instruções de entrada e saída de dados

Instrução de Entrada de Dados

- O comando de atribuição é a forma que o programador possui para armazenar valores e expressões numéricas na memória do computador.
- Outra possibilidade que dispõe o programador, é a utilização do comando de leitura de dados pelo teclado, `input` no Scinote.
- Este comando permite o armazenamento de valores diferentes para uma mesma variável, a cada execução do programa.
- A seguir, a sintaxe geral do comando `input`.

Instrução de Entrada de Dados

Sintaxe geral do comando input:

```
<variável> = input( <frase> )
idade = input("digite sua idade: ")
```

Onde:

`<variável>` é uma variável que representará o local da memória que armazenará o valor digitado e como acessá-lo.

`<frase>` é uma *string* que informa ao usuário qual o dado que ele deve digitar nesta interação. A *string* deve estar sempre entre aspas duplas.

Instrução de Entrada de Dados

Suponha que o programador deseje solicitar ao usuário a quantidade de alunos de uma sala de aula e armazená-la na variável qtd_alunos.

Isso pode ser realizado pela instrução:

```
qtd_alunos = input("Digite o total de alunos: ");

nome = input("Entre com um nome: ", "s"); ou
nome = input("Entre com um nome: ", "string");
```

Instrução de Saída de Dados

Após um valor ser obtido por cálculos e armazenado em uma variável, o mesmo pode ser exibido na tela do computador através do comando printf, o qual tem a seguinte sintaxe geral:

```
printf(<frase>, <lista de expressões>)
```

Considere a sequencia:

```
-->idade = input("Digite sua idade: ")
-->printf("A sua idade é %g anos",idade)
```

Exemplo de um input e printf

```
clc; // limpa a tela
clear; // limpa as variáveis da memória
// ENTRADA DE DADOS isto é um comentário
x = input("Entre com o primeiro valor: ");
y = input("Entre com o segundo valor: ");
// PROCESSAMENTO 1
z = x + y;
// SAÍDA DE DADOS
printf("primeiro valor:%g, segundo valor: %g,
soma %g", x, y,z);
den = input("entre com o denominador: ");
divisao = z/den; // PROCESSAMENTO 2
printf("Resultado da divisão: %g", divisao);
```

Instrução de Saída de Dados

Onde:

<frase> é a sentença que se quer imprimir na tela, e que pode estar entremeada por códigos de formato como %g.

- %g é um código de formato geral para expressões com valores numéricos (veremos em seguida expressões com outros tipos de valores).

- existem vários outros códigos de formato como %d, %f ou %s, que exploraremos em exercícios e em outros exemplos futuramente.

Instrução de Saída de Dados

Onde:

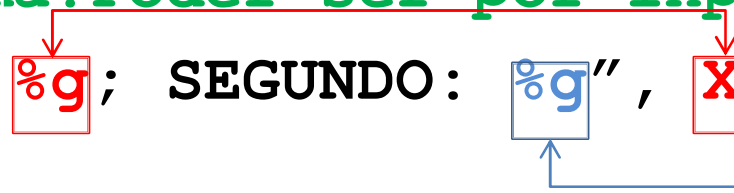
<lista de expressões> é uma lista de expressões separadas por vírgulas, que são calculadas no momento da execução do comando.

As expressões na lista são mapeadas uma a uma nos códigos de formato, na mesma sequência em que aparecem na **<frase>**, e a sentença impressa é obtida pela substituição do valor da expressão na posição marcada pelo código de formato.

Instrução de Saída de Dados Simples

Por exemplo, o código abaixo:

```
x = 30; // não faz leitura pelo teclado
Y = 60; // o valor é "cravado" ao escrever o
        // programa.Poder ser por input!
printf("PRIMEIRO: %g; SEGUNDO: %g", X, Y);
```



Vai ter como saída:

```
- -> PRIMEIRO: 30; SEGUNDO: 60
```

Instrução de Saída de Dados com pulo de linha

Por exemplo, o código abaixo:

```
x = 30;
Y = 60;
printf("PRIMEIRO: %g.\nSEGUNDO: %g", X, Y);
```

Ocorre um pulo de linha nesta posição da frase

Vai ter como saída:

- -> PRIMEIRO: 30.
- -> SEGUNDO: 60

Instrução de Saída de Dados com variável literal

Por exemplo, o código abaixo:

```
nome = "João"; //Atribuição de um literal
a partir de agora nome é um variável literal

idade = 20;
printf("O %s tem %g anos.", nome, idade);
```

Vai ter como saída:

```
- -> O João tem 20 anos.
```

Exercícios

Exercícios

- Codifique os programas a seguir na linguagem Scilab. Utilize comentários e mensagens textuais para o usuário

1. Codifique um programa que leia dois valores pelo teclado. O programa calcula a soma desses valores, armazenando-a em uma terceira variável. A seguir o programa imprime o resultado da soma armazenada na terceira variável.

2. Modifique o programa anterior, onde o resultado de (1) será o numerador de uma divisão. O denominador será um novo valor lido pelo teclado. O programa imprime o resultado final da divisão.

Exercícios

3. Crie um programa que imprima a hipotenusa de um triangulo retângulo após ler de seus dois catetos.

4. Crie um programa que leia do teclado um valor de temperatura em graus Celsius (°C), calcule e imprima essa temperatura em graus Farenheit (°F) e em graus Kelvin (°K).

   OBS.: $°F = °C \times 1.8 + 32$
   $°K = °C + 273.15$