1COP029 - Trabalho T1: DCMAT

Softwares de computação matemática são ferramentas que podem possuir uma interface em modo texto ou gráfico que auxiliam na resolução de uma grande gama de tarefas de cálculo, como manipular matrizes, gerar gráficos de funções, manipulação simbólica, etc. Alguns exemplos desses softwares incluem o GNU Octave, Maple e Matlab.

Através do emprego de técnicas de construção de compiladores, o objetivo deste trabalho é desenvolver um simples software matemático denominado **DCMAT**, que seja capaz de gerar gráficos, resolver expressões, calcular integrais e somatórios além de realizar operações básicas com matrizes. O dcmat utiliza uma interface em modo texto para receber comandos e apresentar os resultados.

No desenvolvimento deste software você pode utilizar as ferramentas `flex` e `bison` se assim o desejar. O software deve ser desenvolvido em `C` ou `C++`.

O dcmat reconhece um conjunto de operadores, delimitadores, identificadores e funções na construção das expressões matemáticas que é formado por:

Operadores binários: `+ - * / ^ %`
Operadores unários: `+ -`
Delimitadores: `( )`
Funções: `sen cos tan abs`
Números inteiros
Números reais
Variável `x`
Identificadores
Constante π
Constante e

Números inteiros são formados pela seguinte expressão regular:

```
[0-9]+
```

Números reais são formados pela seguinte expressão regular:

```
[0-9]+"."[0-9]+
```

Identificadores são formados pela seguinte expressão regular:

```
[a-zA-Z]+[_0-9a-zA-Z]*
```

Versão deste texto: V. 2024.01

Para a inserção de comandos o dcmat utiliza um conjunto de palavras reservadas e delimitadores. As palavras reservadas do dcmat são:

`about`	`float`	`pi`	`settings`
`abs`	`h_view`	`plot`	`show`
`axis`	`integral_steps`	`precision`	`solve`
`connect_dots`	`integrate`	`quit`	`sum`
`cos`	`linear_system`	`reset`	`symbols`
`determinant`	`matrix`	`rpn`	`tan`
`e`	`off`	`sen`	`v_view`
`erase`	`on`	`set`	`x`

As palavras reservadas distinguem entre maiúsculas e minúsculas, desta forma o identificador `About` é diferente da palavra reservada `about`. As únicas exceções são as palavras reservadas `e`, `pi`, `x`, as quais não distinguem entre maiúsculas e minúsculas; desta forma as variações `PI`, `Pi`, `pI` e `pi` todas correspondem a palavra reservada `pi` que serve para designar a constante π.

As palavras reservadas `sen`, `cos`, `tan`, `abs` correspondem respectivamente as funções seno, cosseno, tangente e valor absoluto. Em relação a constantes numéricas, o dcmat reconhece duas constantes:

Constante	**Palavra Reservada no dcmat**	**Valor**
π	`pi`, `pI`, `Pi`, `PI`	3,14159265
e	`e`, `E`	2,71828182

Na construção de expressões matemáticas e comandos, o dcmat utiliza um conjunto de delimitadores apresentados a seguir:

Delimitador	**Significado/Função**
`+`	adição
`-`	subtração
`*`	multiplicação
`/`	divisão
`^`	potenciação
`%`	resto da divisão
`(`	abre parênteses
`)`	fecha parênteses
`:`	intervalo
`=`	igual
`:=`	atribuição
`[`	abre colchetes
`]`	fecha colchetes
`;`	ponto e vírgula
`,`	vírgula

No dcmat alguns exemplos de expressões válidas são:

```
sen(x)
-x*sen(x)
-10*45^2
+(1+3) * 78
-17.78 + 3 + identificador - minha_variavel
-X * sen(x) + cos(x) * (tan(x+3.78)/4.2)
abs(-666)-666
+666-pi+e
```

Ao desenvolver o dcmat você deve criar a sua gramática de forma que os operadores e funções tenham as devidas precedências respeitadas. O programa quando iniciado apresenta um `prompt` de comando através do símbolo:

```
>
```

Ao se pressionar a tecla `ENTER` em uma linha sem nenhum comando, uma nova linha do prompt de comando com o símbolo `>` deve ser criada, sem que nenhum erro seja gerado. Neste `prompt` devem ser digitados os comandos, os quais são finalizados através do símbolo de ponto-e-vírgula, com exceção do comando `quit` e na avaliação de expressões. O `dcmat` fica aguardando comandos até que o comando `quit` seja digitado.

As próximas páginas irão detalhar os comandos do dcmat bem como a sua sintaxe detalhada.

1 Comandos do `dcmat`

1.1 show settings

Sintaxe: `show settings;`

Este comando mostra o conteúdo das variáveis internas do programa. Exemplo de utilização:

```
>show settings;

h_view_lo: -6.500000
h_view_hi: 6.500000
v_view_lo: -3.500000
v_view_hi: 3.500000
float precision: 6
integral_steps: 1000

Draw Axis: ON
Erase Plot: ON
Connect Dots: OFF

>
```

1.2 reset settings

Sintaxe: `reset settings;`

Este comando restaura os valores padrão das variáveis internas do programa. Exemplo de utilização:

```
>reset settings;
>
```

1.3 quit

Sintaxe: `quit`

Sai do programa `dcmat`. Exemplo de utilização:

```
>quit
```

1.4 set h_view

Sintaxe: `set h_view` *[valor float]* `:` *[valor float]* `;`

Este comando seta os parâmetros de visualização horizontal do eixo `X` para a geração de gráficos de funções. Exemplo de utilização:

```
>set h_view -10.5:7.2;
```

1.5 set v_view

Sintaxe: `set v_view` *[valor float]* `:` *[valor float]* `;`

Este comando seta os parâmetros de visualização vertical do eixo `Y` para a geração de gráficos de funções. Exemplo de utilização:

```
>set v_view -5:+10.2;
```

1.6 set axis on

Sintaxe: `set axis on;`

Este comando faz com que os eixos `X` e `Y` sejam desenhados ao se plotar um gráfico de função. Exemplo de utilização:

```
>set axis on;
>
```

1.7 set axis off

Sintaxe: `set axis off;`

Este comando faz com que os eixos `X` e `Y` **NÃO** sejam desenhados ao se plotar um gráfico de função. Exemplo de utilização:

```
>set axis off;
>
```

1.8 plot

Sintaxe: `plot;`

Este comando plota a última função que foi inserida. No caso de nenhuma função haver sido inserida ainda, o comando mostra a mensagem: `No function defined!` Exemplo de utilização quando nenhuma função foi inserida:

```
>plot;

No Function defined!

>
```

IMPORTANTE: Mudanças nas variáveis internas do dcmat devem ser refletidas ao se utilizar o comando `show settings`. Por exemplo, após a utilização dos comandos `set h_view`, `set v_view` e `set axis off` como mostrado nos exemplos anteriores, seguida da utilização do comando `show settings` irá apresentar o seguinte resultado:

```
>set h_view -10.5:7.2;
>set v_view -5:+10.2;
>set axis off;
>show settings;

h_view_lo: -10.500000
h_view_hi: 7.200000
v_view_lo: -5.000000
v_view_hi: 10.200000
float precision: 6
integral_steps: 1000

Draw Axis: OFF
Erase Plot: ON
Connect Dots: OFF

>
```

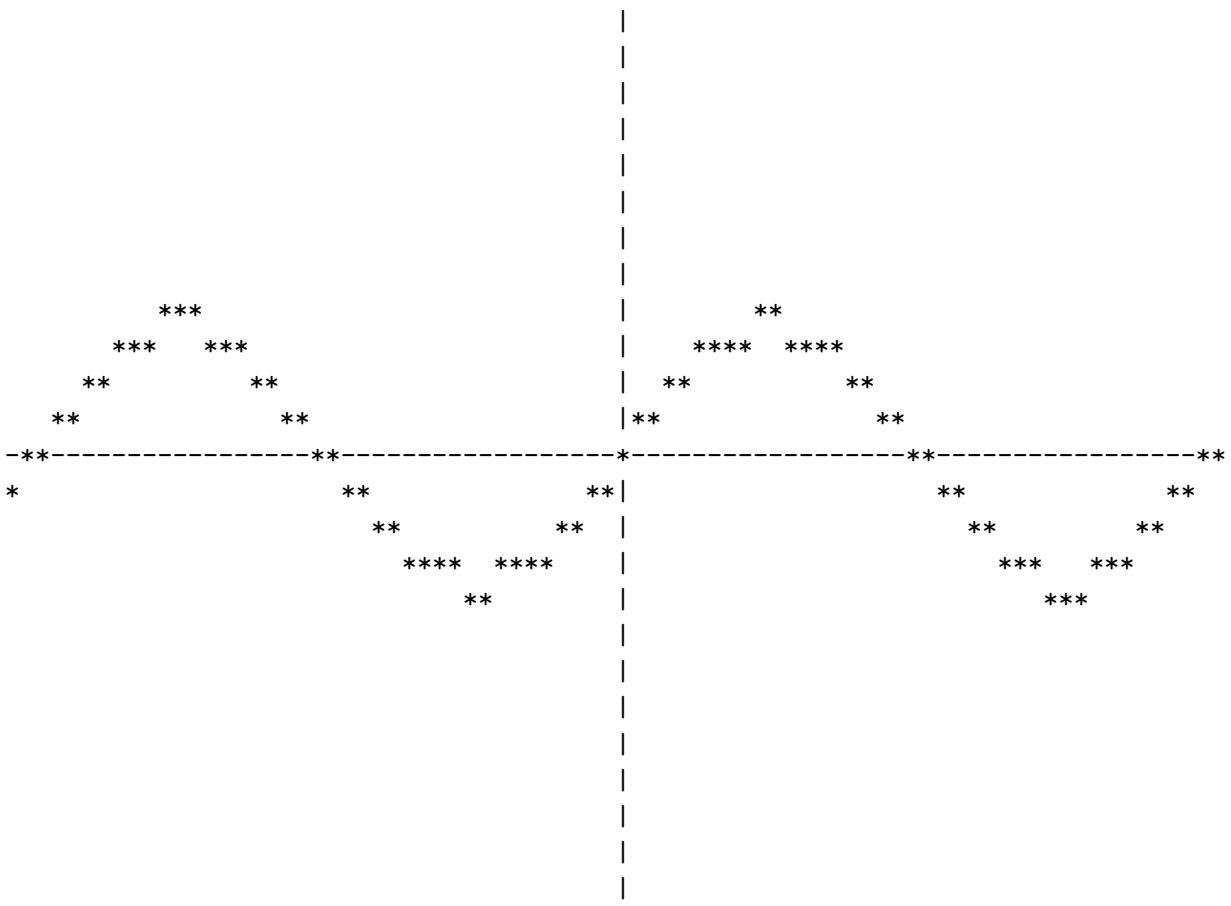
Desta forma o comando `show settings` serve para se visualizar todas as alterações que são feitas nas variáveis internas do dcmat.

1.9 plot

Sintaxe: `plot(` *[função]* `);`

Este comando é utilizado para se plotar funções matemáticas. Exemplo de utilização do comando `plot`, quando os valores padrão de plotagem horizontal e vertical são utilizados e com o desenho dos eixos `X` e `Y` habilitado:

```
>plot(sen(x));
```



```
>
```

O programa plota gráficos com resolução padrão de 25 linhas por 80 colunas em modo texto, sendo que o caractere `*` é utilizado como ***pixel***. Os valores armazenados nas variáveis internas `h_view_lo` e `h_view_hi` definem o intervalo inferior e superior do eixo `X`. Os valores armazenados nas variáveis internas `v_view_lo` e `v_view_hi` definem o intervalor inferior e superior do eixo `Y`.

Exemplo de utilização do comando `plot`, quando os valores padrão de plotagem horizontal e vertical são utilizados e com o desenho dos eixos `X` e `Y` **desabilitado**:

```
>set axis off;
>plot(sen(x));
```

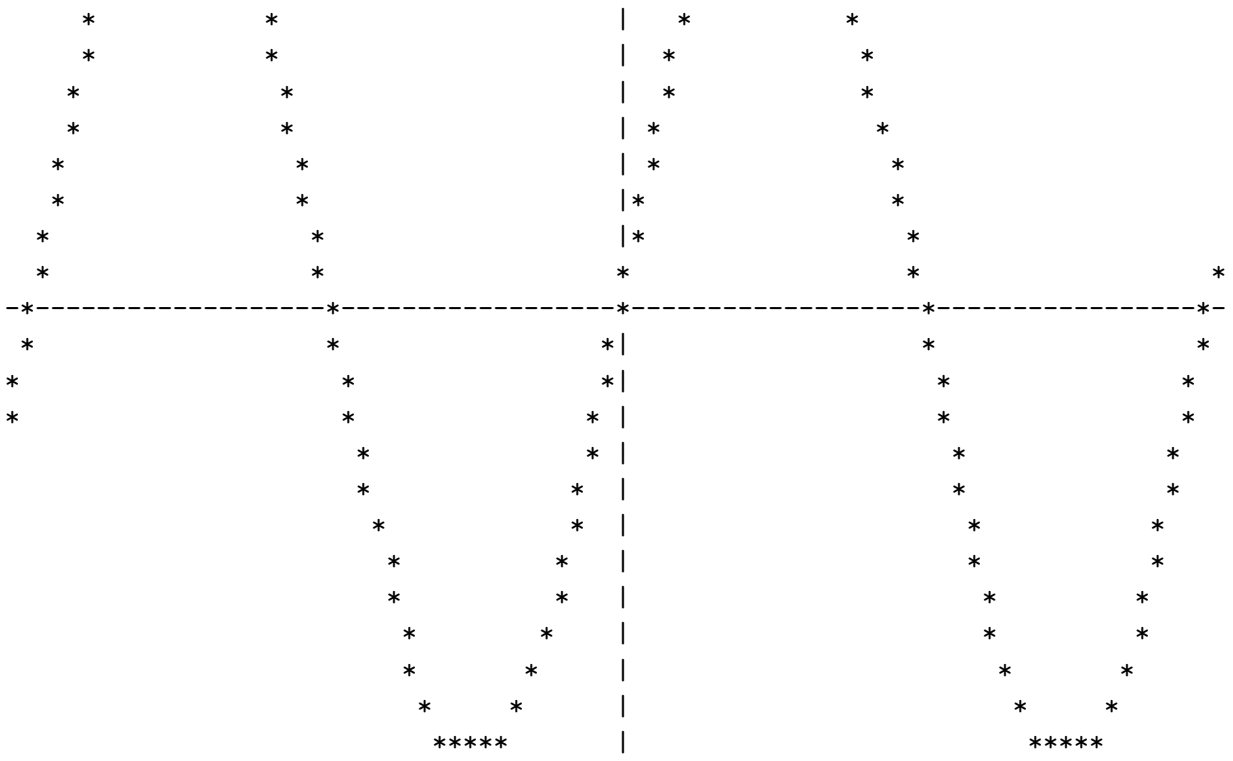
```
>
```

1.10 set erase plot off

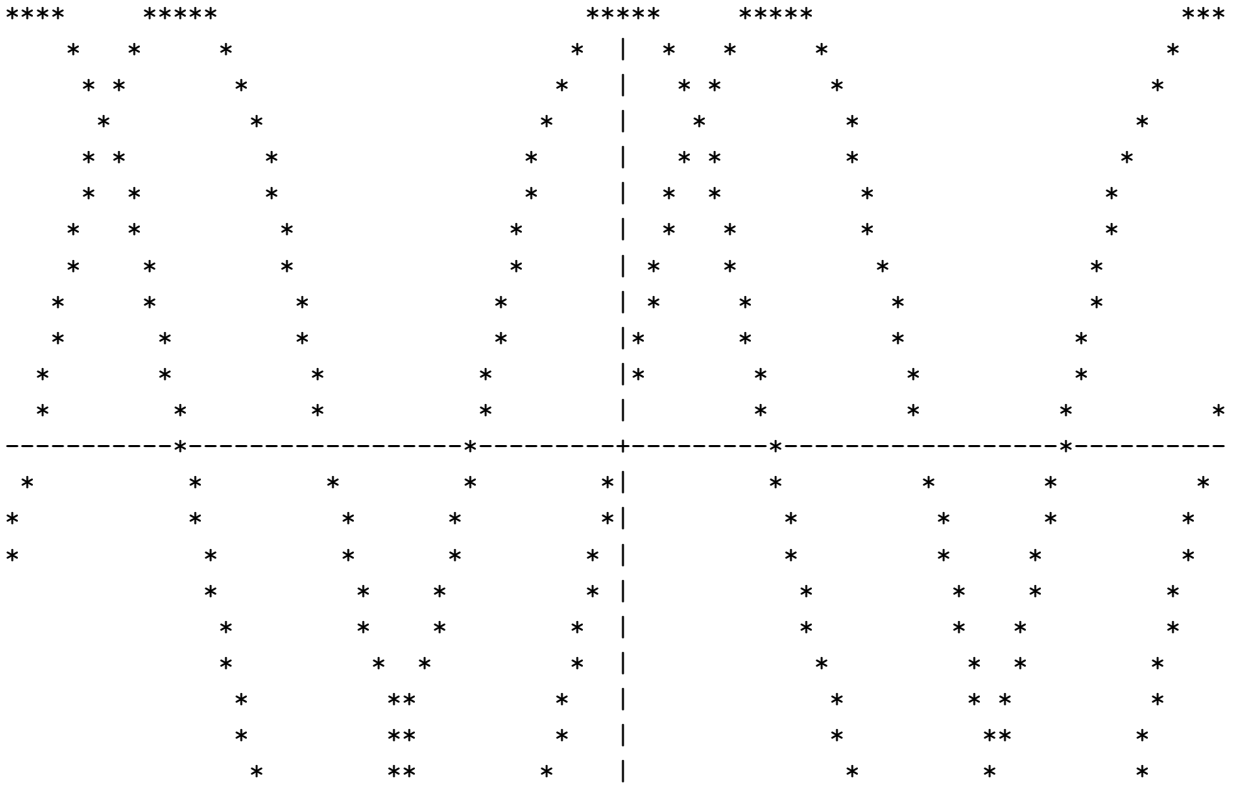
Sintaxe: `set erase plot off;`

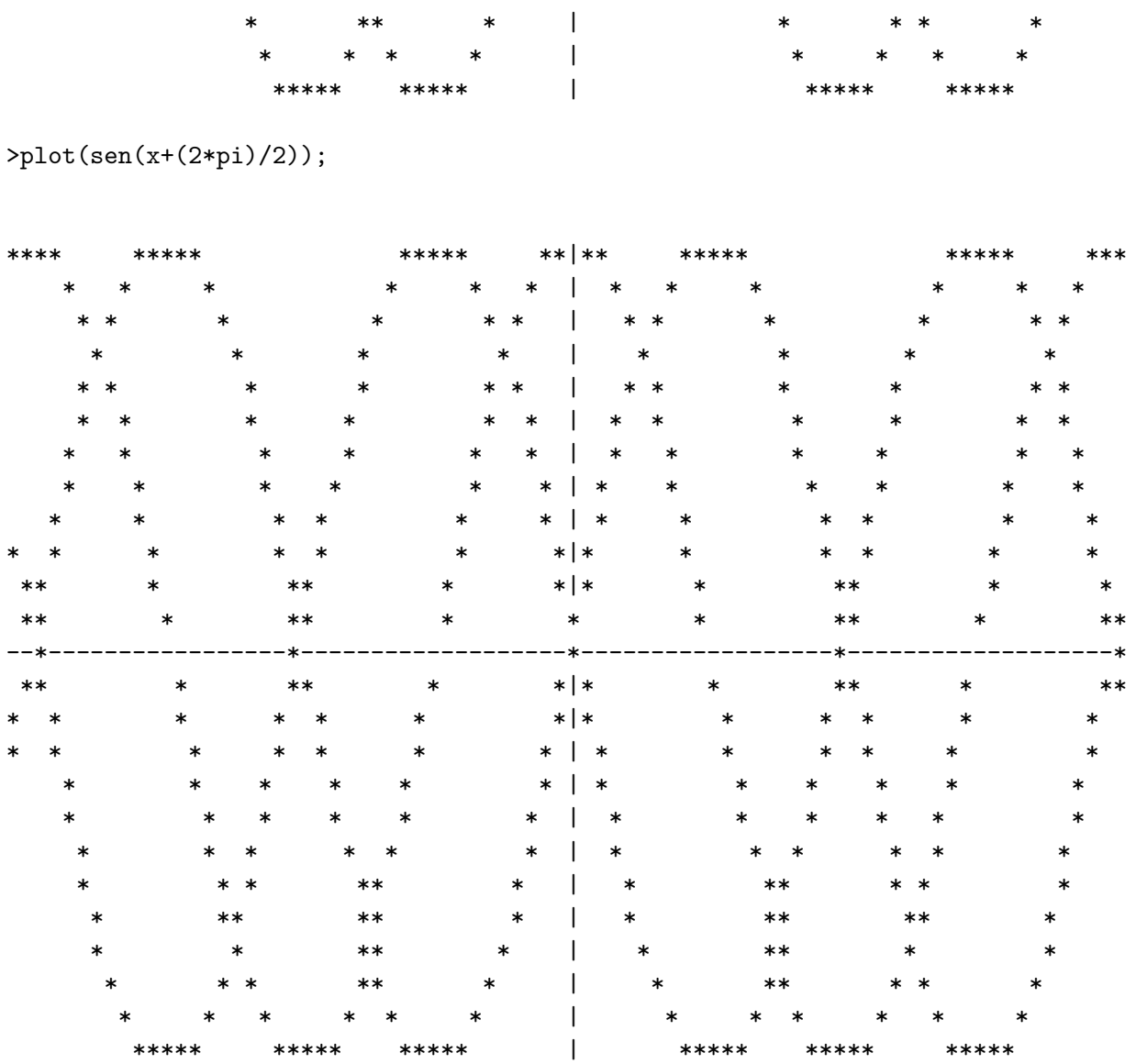
Este comando faz com ao se plotar um novo gráfico o anterior não seja apagado, se constituindo em uma forma de se visualizar mais de uma função ao mesmo tempo. Exemplo de utilização:

```
>plot(sen(x));
```



```
>set erase plot off;
>plot(cos(x));
```





```
>plot(sen(x+(2*pi)/2));
```



```
>
```

1.11 set erase plot on

Sintaxe: `set erase plot on;`

Este comando faz com que ao se plotar um novo gráfico o anterior seja apagado. Em sua configuração padrão o dcmat sempre irá apagar o gráfico anterior antes de plotar um novo. Exemplo de utilização:

```
>set erase plot on;
>
```

1.12 Impressão em RPN

Sintaxe: `rpn(` *[expressão]* `);`

Este comando é utilizado para mostrar a forma pós-fixa de uma expressão matemática qualquer. Qualquer identificador pode ser utilizado, mesmo aqueles que ainda não estão definidos (mais informações sobre isso irão aparecer à frente). Exemplo de utilização:

```
>rpn(sen(x)*(1+x)+chuchu*pi);

Expression in RPN format:

x SEN 1.000000 x + * chuchu 3.141593 * +

>
```

IMPORTANTE: A expressão introduzida para ser impressa na forma RPN não é armazenada e não altera a função previamente armazenada, caso exista, pelo comando `plot`.

1.13 set integral steps

Sintaxe: `set integral_steps` *[valor inteiro]* `;`

Este comando ajusta a quantidade de passos a serem utilizandos ao se calcular o valor numérico de uma integral utilizando-se a *Soma de Riemann*. Exemplo de utilização:

```
>set integral_steps 350;
>
```

1.14 integrate

Sintaxe: `integrate(` *[limite inferior]* `:` *[limite superior]* `,` *[função]* `);`

Este comando é utilizado para se calcular o valor numérico de uma integral entre o intervalo definido pelos valores *limite inferior* e *limite superior* para uma determinada função. Exemplo de utilização:

```
>integrate(0:6.28,sen(x));

0.000010

>
```

No exemplo apresentado é calculado o valor numérico da integral:

$$\int_0^{6,28} sen(x)\ \mathrm{d}x = -cos(x)\Big|_0^{6,28}$$

1.15 sum

Sintaxe: `sum(` *[variável]* `,` *[limite inferior]* `:` *[limite superior]* `,` *[expressão]* `);`

Este comando é utilizado para se calcular o somário de uma expressão matemática que contenha uma determinada variável entre o intervalo definido pelos valores *limite inferior* e *limite superior*. Tais limites são **valores inteiros**, pois o índice do somatório é incrementado ou decrementado em uma unidade a cada iteração. Exemplo de utilização:

```
>sum(i,1:37,i-1);

666.000000

>
```

No exemplo apresentado é calculado o valor do somatório:

$$\sum_{i=1}^{37}(i-1) = 666$$

1.16 matrix

Sintaxe: `matrix = [ [` *valor* {`,` *valor*}* `] {,[` *valor* {`,` *valor*}* `] }* ];`

* Indica uma repetição de zero ou mais vezes do termo que este entre { e }.
valor indica um número real.

Este comando é utilizado para se realizar a entrada de uma matriz no sistema. Exemplo de utilização:

```
>matrix = [[1]];
>
```

No exemplo apresentado foi inserida a matriz:

$$\begin{bmatrix}1\end{bmatrix}$$

Outro exemplo de utilização:

```
>matrix = [ [1,2], [3,4,5], [6] ];
>
```

No exemplo apresentado, foi inserida a matriz:

$$\begin{bmatrix}1 & 2 & 0\\3 & 4 & 5\\6 & 0 & 0\end{bmatrix}$$

Observe que o número de colunas será determinado pela linha que contiver o maior número de elementos, sendo que as demais linhas terão o seu tamanho completado inserindo-se valores iguais a zero.

1.17 show matrix

Sintaxe: `show matrix;`

Este comando mostra a matriz que foi previamente inserida no sistema. Exemplo de utilização:

```
>show matrix;

+-                           -+
| 1.000000 2.000000 0.000000 |
| 3.000000 4.000000 5.000000 |
| 6.000000 0.000000 0.000000 |
+-                           -+

>
```

Considere que a seguinte matriz foi inserida:

$$\begin{bmatrix} -1 & 2 \\ 3 & -4 \end{bmatrix}$$

Observe que a matriz possui números negativos. Se a matriz for impressa, os números devem ser impressos de forma alinhada. Exemplo:

```
>matrix = [[-1,2],[3,-4]];
>show matrix;

+-                    -+
| -1.000000  2.000000 |
|  3.000000 -4.000000 |
+-                    -+

>
```

Se não houver nenhuma matriz inserida, o comando mostra a mensagem: `No Matrix defined!` Exemplo de utilização quando nenhuma matriz foi inserida:

```
>show matrix;

No Matrix defined!

>
```

1.18 solve determinant

Sintaxe: `solve determinant;`

Este comando calcula o determinante de uma matriz quadrada n por n.
Suponha que se deseje computar o determinante da seguinte matriz:

$$\begin{bmatrix} 10 & 2 & 1 \\ 1 & 5 & 1 \\ 2 & 3 & 10 \end{bmatrix}$$

A computação do determinante da matriz apresentada seria da seguinte forma:

```
>matrix = [[10,2,1],[1,5,1],[2,3,10]];
>solve determinant;

447.000000

>
```

Se não houver nenhuma matriz inserida, o comando mostra a mensagem: `No Matrix defined!`

1.19 solve linear_system

Sintaxe: `solve linear_system;`

Este comando é utilizado para se resolver sistemas de equações lineares que são representados através de matrizes aumentadas.

Suponha que se deseje resolver o seguinte sistema de equações lineares:

$$\begin{cases} 10x_1 + 2x_2 + 1x_3 = 7 \\ 1x_1 + 5x_2 + 1x_3 = -8 \\ 2x_1 + 3x_2 + 10x_3 = 6 \end{cases}$$

Tal sistema de equações lineares pode ser representado pela seguinte equação matricial:

$$\begin{bmatrix} 10 & 2 & 1 \\ 1 & 5 & 1 \\ 2 & 3 & 10 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ -8 \\ 6 \end{bmatrix}$$

Tal equação matricial pode ser representada pela seguinte matriz aumentada:

$$\begin{bmatrix} 10 & 2 & 1 & 7 \\ 1 & 5 & 1 & -8 \\ 2 & 3 & 10 & 6 \end{bmatrix}$$

A matriz aumentada pode ser inserida no `dcmat` e o sistema linear resolvido através do comando `solve linear_system` utilizando, por exemplo, o método da fatoração LU. A resolução do sistema linear apresentado seria:

```
>matrix = [[10,2,1,7],[1,5,1,-8],[2,3,10,6]];
>solve linear_system;

Matrix x:

1.000000
-2.000000
1.000000

>
```

No exemplo apresentado a matriz x contém a solução do sistema linear, ou seja:

$$x_1 = 1$$
$$x_2 = -2$$
$$x_3 = 1$$

Os sistemas lineares são classificados em 3 formas:

- **SPD** - Sistema Possível e Determinado
- **SPI** - Sistema Possível e Indeterminado
- **SI** - Sistema Impossível

Quando o sistema é possível e determinado, o sistema será resolvido e sua resposta mostrada, como no exemplo anterior. Se o sistema for possível e indeterminado, isto é, possui infinitas soluções, o programa deve mostrar a mensagem `SPI - The Linear System has infinitely many solutions`, como no exemplo a seguir:

```
>matrix=[[1,1,8],[2,2,16]];
>solve linear_system;

SPI - The Linear System has infinitely many solutions

>
```

Se o sistema for impossível, isto é, o sistema não possui nenhuma solução, o programa deve mostrar a mensagem `SI - The Linear System has no solution`, como no exemplo a seguir:

```
>matrix = [[3,2,6],[3,2,12]];
>solve linear_system;

SI - The Linear System has no solution

>
```

1.20 about

Sintaxe: `about;`

Este comando mostra informações sobre o desenvolvedor. Exemplo:

```
>about;

+-----------------------------------------------+
|                                               |
|           DCMAT - CopyRight DC-UEL            |
|                  V. 2024.01                   |
|                                               |
+-----------------------------------------------+

>
```

IMPORTANTE: Você deve mostrar o seu número de matrícula seguido do seu nome dentro da caixa de texto ao invés da mensagem aqui apresentada.

2 Manipulação de Símbolos

2.1 Atribuição de Valores a Símbolos

Sintaxe: *[variável]* `:=` *[expressão]* `;`

O dcmat permite que o usuário defina variáveis e atribua valores a elas para posterior utilização. Exemplo de utilização:

```
>minha_variavel := 123456;

123456.000000

>
```

No exemplo apresentado o usuário criou uma variável com o nome `minha_variavel` e atribuiu a ela o valor 123456. Se o usuário desejar mudar o valor associado a essa variável, basta fazer uma nova atribuição, como no exemplo a seguir:

```
>minha_variavel := 123456;

123456.000000

>minha_variavel := 666;

666.000000

>
```

No exemplo apresentado o usuário criou a variável `minha_variavel`, atribuiu a ela o valor 123456 e posteriormente atribuiu o valor 666 a essa mesma variável. Na atribuição de valores a variáveis é possível utilizar expressões matemáticas, como no exemplo a seguir:

```
>a := 222;

222.000000

>b := 3*a;

666.000000

>
```

No exemplo apresentado o usuário criou uma variável chamada `a` e atribuiu a ela o valor 222. Posteriormente o usuário criou uma variável chamada `b` e atribuiu a ela a expressão `3*a`. Como a variável `a` contém o valor 222, a variável `b` irá receber o valor 666, pois o dcmat avalia a expressão matemática para gerar o valor a ser atribuído.

Na atribuição de valores a variáveis, caso o usuário utilize símbolos que não estão definidos, o dcmat lista todos os símbolos sem valor associado e nenhuma atribuição é feita. Considere o exemplo a seguir:

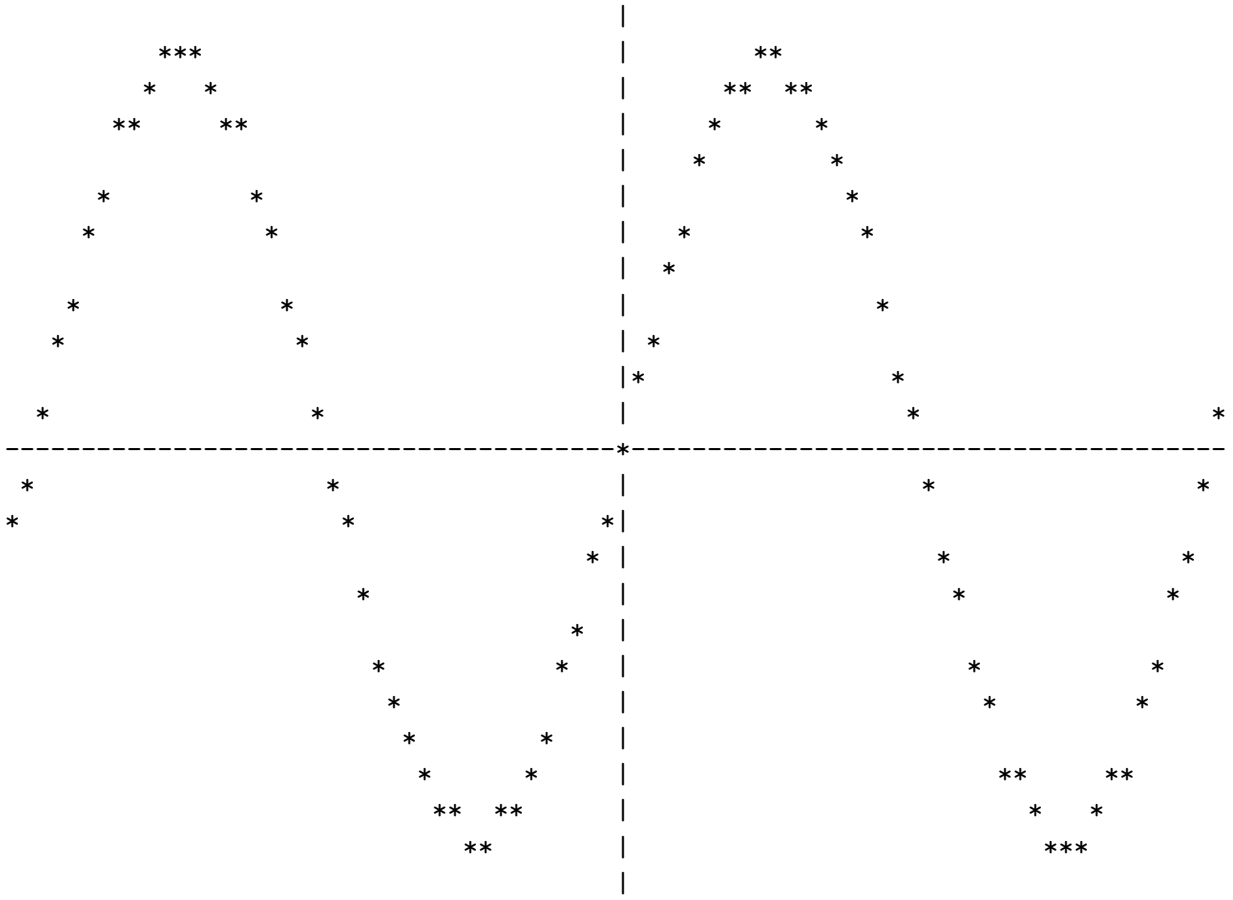
```
>a:=222;

222.000000

>b :=3*a;

666.000000

>b := a+b+c+d;

Undefined symbol [c]
Undefined symbol [d]

>
```

No exemplo apresentado o usuário definiu valores para as variáveis `a` e `b`. Em um segundo momento o usuário tentou gerar um novo valor para a variável `b`, mas a expressão que ele utilizou continha dois símbolos que não estavam definidos, isto é, `c` e `d`, os quais foram listados. Como a expressão matemática é inválida, a variável `b` continua com o valor 666 associado a si.

Sempre que o usuário utilizar um símbolo não definido em uma expressão matemática, o mesmo dever ser listado, como no exemplo anterior, e nenhuma computaçao será realizada.

A definição de símbolos é útil para se faciliar outras computações, pois em **TODOS** os comandos do dcmat onde é possível utilizar uma função ou expressão matemática, o usuário pode empregrar símbolos definidos por ele. Considere o seguinte exemplo:

```
>amplitude := 3;

3.000000

>integrate(0:3.14,amplitude*sen(x));

5.999921

>sum(n,0:10, n * amplitude);

165.000000

>rpn(amplitude * sen(x) + x);

Expression in RPN format:

amplitude x SEN * x +

>plot(amplitude*sen(x));
```



```
>
```

Como já dito, quando o usuário utiliza símbolos não definidos em expressões, estes são listados e nenhuma computação é realizada. A única exceção é o comando `rpn`, pois seu objetivo é apenas listar a ordem das computações e não realizar qualquer tipo de cálculo. Para ilustar isso, considere o exemplo a seguir:

```
>sum(n,0:10,n*abobrinha);

Undefined symbol [abobrinha]

>rpn(n*abobrinha);

Expression in RPN format:

n abobrinha *

>
```

2.2 Atribuição de Matrizes a Símbolos

Sintaxe: *[variável]* `:= [ [` *valor* {`,` *valor*}* `] {,[` *valor* {`,` *valor*}* `] }* ];`

* Indica uma repetição de zero ou mais vezes do termo que este entre { e }.
valor indica um número real.

Além de valores, o dcmat permite que o usuário defina variáveis e atribua matrizes a elas para posterior utilização. Exemplo de utilização:

```
>minha_matriz := [[1,2],[3,4,5],[6]];

+-                           -+
| 1.000000 2.000000 0.000000 |
| 3.000000 4.000000 5.000000 |
| 6.000000 0.000000 0.000000 |
+-                           -+

>
```

No exemplo apresentado o usuário criou uma variável chamada `minha_matriz` e atribuiu a ela a matriz:

$$\begin{bmatrix} 1 & 2 & 0 \\ 3 & 4 & 5 \\ 6 & 0 & 0 \end{bmatrix}$$

Observe que o número de colunas será determinado pela linha que contiver o maior número de elementos, sendo que as demais linhas terão o seu tamanho completado inserindo-se valores iguais a zero.

2.3 Mostrando o Valor de Símbolos

Sintaxe: *[variável]* `;`

Para visualizar o valor atribuído a um símbolo, basta entrar o seu nome seguido de ponto-e-vírgula na linha de comando. Se o símbolo não existir uma mensagem será apresentada. Considere o seguinte exemplo, onde as variáveis `amplitude` e `minha_matriz` já estão definidas, mas o símbolo `d` não está definido:

```
>amplitude;

amplitude = 3.000000

>minha_matriz;

+-                           -+
| 1.000000 2.000000 0.000000 |
| 3.000000 4.000000 5.000000 |
| 6.000000 0.000000 0.000000 |
+-                           -+
```

```
>d;

Undefined symbol

>
```

2.4 Listando Todos os Símbolos e seus Tipos

Sintaxe: `show symbols ;`

Este comando lista todos os símbolos definidos pelo usuário, bem como o seu tipo. Considere o seguinte exemplo:

```
>show symbols;

minha_variavel - FLOAT
minha_matriz - MATRIX [3][3]
a - FLOAT
b - FLOAT
i - FLOAT
n - FLOAT
amplitude - FLOAT

>
```

No exemplo apresentado o usuário havia definido um total de 7 símbolos. Quando o símbolo possui um valor numérico associado a si, o nome do mesmo é seguido do tipo `FLOAT`. Quando o símbolo possui uma matriz associada a si, o nome é seguido do tipo `MATRIX` e as dimensões da matriz, onde o primeiro valor indica o número de linhas e o segundo o número de colunas. Desta forma, no exemplo apresentado, o símbolo `minha_matriz` é uma matriz de tamanho $3x3$, isto é 3 linhas e 3 colunas. Todos os demais símbolos do exemplo possuem valores numéricos em ponto flutuante associados a si. A ordem em que os símbolos são apresentados é irrelevante.

3 Avaliação de Expressões

3.1 Avaliação de Expressões Matemáticas

Sintaxe: *[expressão]*

Uma importante função de qualquer software matemático é a realização de cálculos. No dcmat a avaliação de expressões é feita simplesmente se digitando a mesma na linha de comando. Considere o exemplo a seguir:

```
>1+3*5

16.000000

>a:=2;

2.000000

>b:=4;

4.000000

>a+b*pi

14.566371

>pi

3.141593

>a+b+c

Undefined symbol [c]

>abs(-666)

666.000000

>a/(-45+45)

inf

>x+5

The x variable cannot be present on expressions.

>
```

Como se observa no exemplo apresentado, não existe ponto-e-vírgula ao fim de uma expressão matemática. O dcmat é capaz de fazer contas simples onde o usuário pode empregar todos os operadores, funções e constantes

listados no início deste manual na construção do seu cálculo, além de que o usuário pode inclusive atribuir valores para símbolos e utilizar os mesmos na construção de seu cálculo, como mostra exemplo apresentado.

A exceção é a variável `x` que só pode ser utilizada nas expressões que serão plotadas ou terão a integral calculada. Outra observação importante é que quando um símbolo não definido for utilizado em uma expressão, o mesmo será listado e nenhum cálculo será realizado, como aconteceu na expressão `a+b+c`.

3.2 Operações com Matrizes

As expressões matemáticas também podem englobar matrizes, neste caso o dcmat precisa verificar se as operações são válidas. O dcmat é capaz de realizar as seguintes operações com matrizes: soma, subtração, multiplicação e multiplicação por escalar.

Soma e Subtração

As operações de soma e subtração requerem que ambas as matrizes tenham as mesmas dimensões. Considere o exemplo a seguir:

```
>a := [[1,1],[1,1]];

+-                 -+
| 1.000000 1.000000 |
| 1.000000 1.000000 |
+-                 -+

>b := [[2,2],[2,2]];

+-                 -+
| 2.000000 2.000000 |
| 2.000000 2.000000 |
+-                 -+

>a+b

+-                 -+
| 3.000000 3.000000 |
| 3.000000 3.000000 |
+-                 -+

>a-b

+-                   -+
| -1.000000 -1.000000 |
| -1.000000 -1.000000 |
+-                   -+

>
```

Quando as matrizes tiverem dimensões incompatíveis para a soma ou subtração o dcmat irá emitir uma mensagem de erro. O exemplo a seguir ilustra tal situação:

```
> a:= [[1,1]];

+-                  -+
| 1.000000 1.000000 |
+-                  -+

>b:=[[2,2,2]];

+-                           -+
| 2.000000 2.000000 2.000000 |
+-                           -+

>a+b

Incorrect dimensions for operator '+' - have MATRIX [1][2] and MATRIX [1][3]

>a-b

Incorrect dimensions for operator '-' - have MATRIX [1][2] and MATRIX [1][3]

>
```

Na mensagem de erro apresentada, o dcmat mostra o operador matemático que recebeu os tipos incompatíveis bem como a dimensão das matrizes conflitantes, como também ilustra o exemplo a seguir:

```
>a := [[1,1]];

+-                  -+
| 1.000000 1.000000 |
+-                  -+

>b := [[2,2]];

+-                  -+
| 2.000000 2.000000 |
+-                  -+

>c := [[1,2,3]];

+-                           -+
| 1.000000 2.000000 3.000000 |
+-                           -+

>a+b-c

Incorrect dimensions for operator '-' - have MATRIX [1][2] and MATRIX [1][3]
```

Se por acaso houver mais de um tipo de matriz incompatível na expressão, o dcmat informa somente a primeira ocorrência e finaliza o processo de avaliação, como mostra o exemplo a seguir:

```
>a:=[[1]];

+-        -+
| 1.000000 |
+-        -+

>b:=[[2,2]];

+-                 -+
| 2.000000 2.000000 |
+-                 -+

>c:=[[3,3,3]];

+-                           -+
| 3.000000 3.000000 3.000000 |
+-                           -+

>a+b+c

Incorrect dimensions for operator '+' - have MATRIX [1][1] and MATRIX [1][2]

>
```

Multiplicação

Na multiplicação entre duas matrizes é necessário que as matrizes envolvidas tenham dimensões $[n,m]$ e $[m,k]$, isto é, o número de colunas da primeira matriz deve ser igual ao número de linhas da segunda matriz. Se as matrizes envolvidas na multiplicação estiverem fora do padrão necessário, o dcmat apresenta uma mensagem de erro. Para ilustrar o processo considere o seguinte exemplo:

```
>a := [[1,2]];

+-                 -+
| 1.000000 2.000000 |
+-                 -+

>b:= [[3,4,5],[6,7,8]];

+-                           -+
| 3.000000 4.000000 5.000000 |
| 6.000000 7.000000 8.000000 |
+-                           -+
```

```
>a*b

+-                              -+
| 15.000000 18.000000 21.000000 |
+-                              -+

>b*a

Incorrect dimensions for operator '*' - have MATRIX [2][3] and MATRIX [1][2]

>
```

No exemplo apresentado a matriz `a` possui dimensão [1,2] e a matriz `b` possui dimensão [2,3]. A multiplicação `a*b` é possível pois `a` tem 2 colunas e `b` tem duas linhas. Já a multiplicação `b*a` não é possível pois **b** tem 3 colunas e `a` apenas uma linha.

O dcmat também suporta a multiplicação de matriz por um valor numérico, como mostra o próximo exemplo.

```
>a:=[[1,1]];

+-                 -+
| 1.000000 1.000000 |
+-                 -+

>a*666

+-                      -+
| 666.000000 666.000000 |
+-                      -+

>666*a

+-                      -+
| 666.000000 666.000000 |
+-                      -+

>-a

+-                    -+
| -1.000000 -1.000000 |
+-                    -+

>
```

No exemplo apresentado a expressão `-a` equivale a expressão `-1*a`.

3.3 Tipagem Dinâmica de Símbolos

O tipo de um símbolo é determinado pelo tipo do valor que é atribuído ao mesmo, desta forma, em um determinado momento um símbolo pode ser uma matriz e em outro instante este mesmo símbolo pode ser um valor numérico em ponto flutuante. Considere o exemplo a seguir:

```
>a := [[1,1]];

+-                 -+
| 1.000000 1.000000 |
+-                 -+

>b := [[2,2]];

+-                 -+
| 2.000000 2.000000 |
+-                 -+

>d := a+b;

+-                 -+
| 3.000000 3.000000 |
+-                 -+

>show symbols;

a - MATRIX [1][2]
b - MATRIX [1][2]
d - MATRIX [1][2]

>d := Pi;

3.141593

>show symbols;

a - MATRIX [1][2]
b - MATRIX [1][2]
d - FLOAT

>
```

No exemplo apresentado foram definidas duas matrizes: `a` e `b`. Em seguida o usuário criou a variável `d` a qual recebeu o resultado de uma soma de matrizes através da expressão `a+b`. Neste instante, a variável `d` é uma matriz, como o comando `show symbols` ilustra.

Como o tipo dos símbolos não é imutável, outro tipo de valor pode ser atribuído aos mesmos, como ocorre com a variável `d` logo após o comando show symbols, onde ela recebe um valor em ponto flutuante, no caso

a constante π. A nova execução do comando `show symbols` apresenta o novo tipo da variável `d`, que agora é float.

Nada impede que uma outra atribuição a `d` faça o seu tipo voltar a ser uma matriz. Desta forma o tipo dos símbolos definidos pelo usuário pode ser alterado por sucessivas atribuições ao mesmo, a depender do tipo gerado pela expressão sendo avaliada, como ilustra o exemplo a seguir:

```
>a:=666;

666.000000

>a:=[[1,1]];

+-                  -+
| 1.000000 1.000000 |
+-                  -+

>a:=Pi;

3.141593

>a:=[[1,2],[3,4]];

+-                  -+
| 1.000000 2.000000 |
| 3.000000 4.000000 |
+-                  -+

>
```

4 Apresentação de Resultados

No dcmat, por padrão, todos os valores em ponto flutuante são apresentados com 6 casas decimais, como definido na variável interna do sistema `float precision`. Tal definição afeta todos os resultados apresentados. O comando a seguir ilustra como alterar a quantidade de casas decimais que são impressas.

4.1 set float precision

Sintaxe: `set float precision` *[valor inteiro]* `;`

Este comando ajusta a quantidade de casas decimais que são impressas em valores em ponto flutuante. Os valores válidos são inteiros que vão de `0` até `8`. Exemplo de utilização:

```
>a := 3.45;

3.450000

>a;

a = 3.450000

>set float precision 0;
>a;

a = 3

>b := [[1,2,3],[4,5,6],[7,8,9]];

+-     -+
| 1 2 3 |
| 4 5 6 |
| 7 8 9 |
+-     -+

>show settings;

h_view_lo: -6.500000
h_view_hi: 6.500000
v_view_lo: -3.500000
v_view_hi: 3.500000
float precision: 0
integral_steps: 1000

Draw Axis: ON
Erase Plot: ON
Connect Dots: OFF

>
```

Como o exemplo mostrou, quando a quantidade de casas decimais a ser impressa foi definida como ZERO, nenhum número após a vírgula é apresentado no resultados. Tal definição afeta todos os comandos de cálculo do dcmat na apresentação de resultados. Internamente, os cálculos no dcmat estão sendo feitos utilizando os valores em ponto flutuante com todas as casas decimais, mas somente na apresentação dos resultados é que a quantidade de casas decimais é impressa de acordo com o que foi definido pelo usuário. A única exceção a essa regra é o comando `show settings`, que sempre utiliza 6 casas decimais após a vírgula para apresentar as variáveis padrão dos sitema, pois o usuário precisa saber, sem truncamento, os valores da região de plotagem dos gráficos.

Se o usuário tentar inserir uma valor inválido de casas decimais, o dcmat irá emitir uma mensagem de erro e nenhuma alteração é realizada.

```
>set float precision -1;

ERROR: float precision must be from 0 to 8

>set float precision 9;

ERROR: float precision must be from 0 to 8

>
```

5 Mensagens de Erro

5.1 Caracteres Inválidos

Quando um caractere inválido for inserido, o sistema deve apresentar a mensagem `Invalid Symbol:` e apresentar o caractere inválido. Exemplo do erro:

```
>@

Invalid Symbol: @

>
```

5.2 Erros de Sintaxe

Quando houver um erro de sintaxe, o sistema deve apresentar a mensagem `SYNTAX ERROR:` e apresentar entre colchetes o *token* que gerou o erro. Exemplo do erro:

```
>matrix = ;

SYNTAX ERROR: [;]

>
```

Quando o comando digitado estiver incompleto, sem um erro de sintaxe aparente, o sistema deve apresentar a mensagem `SYNTAX ERROR: Incomplete Command`. Exemplo do erro:

```
>plot

SYNTAX ERROR: Incomplete Command

>
```

5.3 Tamanho Inválido de Matriz

O `dcmat` suporta matrizes com tamanho máximo de 10x10, isto é 10 linhas por 10 colunas. Ao se tentar inserir uma matriz fora destes limites, a mensagem `ERROR: Matrix limits out of boundaries.` é apresentada pelo sistema e nenhuma matriz é inserida. Exemplo do erro:

```
>matrix = [[0,1,2,3,4,5,6,7,8,9,0]];

ERROR: Matrix limits out of boundaries.

>minha_matriz := [[0,1,2,3,4,5,6,7,8,9,0]];

ERROR: Matrix limits out of boundaries.

>
```

5.4 Ajuste da Área de Plotagem

Ao se ajustar os valores que serão utilizados na vizualização horizontal e vertical do gráfico, o primeiro valor deve ser menor que o segundo valor, caso contrário uma mensagem de erro deve ser exibida. Exemplo:

```
>set v_view 4:-1;

ERROR: v_view_lo must be smaller than v_view_hi

>
>set h_view 1:-3.4;

ERROR: h_view_lo must be smaller than h_view_hi

>
```

5.5 Integral: Passo no cálculo da Integral

Caso o método que calcula a integral numericamente utilize passos e se deseje mudar os passos do algoritmo, o valor passado deve ser positivo, caso contrário a seguinte mensagem é mostrada:

```
>set integral_steps -2;

ERROR: integral_steps must be a positive non-zero integer

>
```

5.6 Integral: Limite inferior e superior

Ao se calcular uma integral numericamente, o limite inferior deve ser menor ou igual que o limite superior, caso contrário a seguinte mensagem é mostrada:

```
>integrate(1:-1,cos(x));

ERROR: lower limit must be smaller than upper limit

>
```

Se o limite inferior for igual ao limite superior a integral deve retornar o valor zero.

5.7 Determinante: Matriz não quadrada

O determinante só pode ser calculado para matrizes quadradas. Ao se tentar calcular o determinante de uma matriz não quadrada, a seguinte mensagem é mostrada:

```
>matrix = [[1,2]];
>solve determinant;

Matrix format incorrect!

>
```

5.8 Sistema Linear

Na resolução de um sistema linear de n variáveis, a matriz deve possuir n linhas e $n+1$ colunas, caso contrário a seguinte mensagem é mostrada:

```
>matrix = [[1]];
>solve linear_system;

Matrix format incorrect!

>
```

5.9 Tipos Incompatíveis em Expressões com Matrizes

Como o dcmat permite que o usuário defina e utilize matrizes em expressões matemáticas, podem acontecer situações em que o usuário digitou uma expressão com tipos inválidos. Neste caso o dcmat emite a mensagem de error `Incorrect type for operator`, apresentando operador/função que recebeu o(s) tipo(s) inválido(s), bem como a combinação de tipos que gerou o erro. Para ilustar esse erro, considere o exemplo a seguir:

```
>set float precision 0;
>a := [[1,2],[3,4]];

+-   -+
| 1 2 |
| 3 4 |
+-   -+

>a-5

Incorrect type for operator '-' - have MATRIX and FLOAT

>666-a

Incorrect type for operator '-' - have FLOAT and MATRIX

>sen(a)

Incorrect type for operator 'SEN' - have MATRIX

>
```

6 Valores Padrão das Variáveis Internas

Os valores padrão das variáveis internas são:

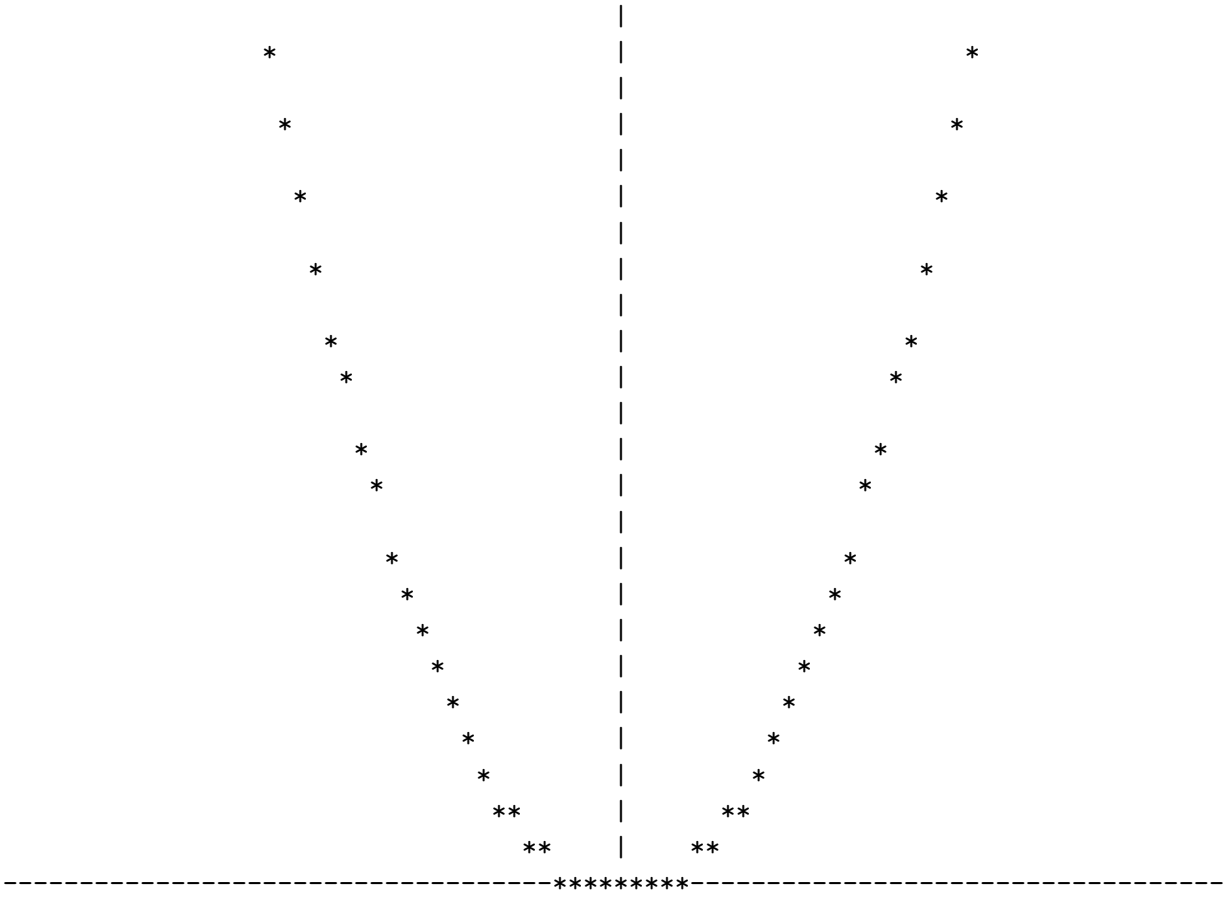
```
h_view_lo: -6.500000
h_view_hi: 6.500000
v_view_lo: -3.500000
v_view_hi: 3.500000
float precision: 6
integral_steps: 1000

Draw Axis: ON
Erase Plot: ON
Connect Dots: OFF
```

Os valores padrão podem ser modificados livremente pelo usuário. Ao se digitar o comando `reset settings`, as variáveis internas tem o seu valor restabelecido aos valores aqui apresentados.

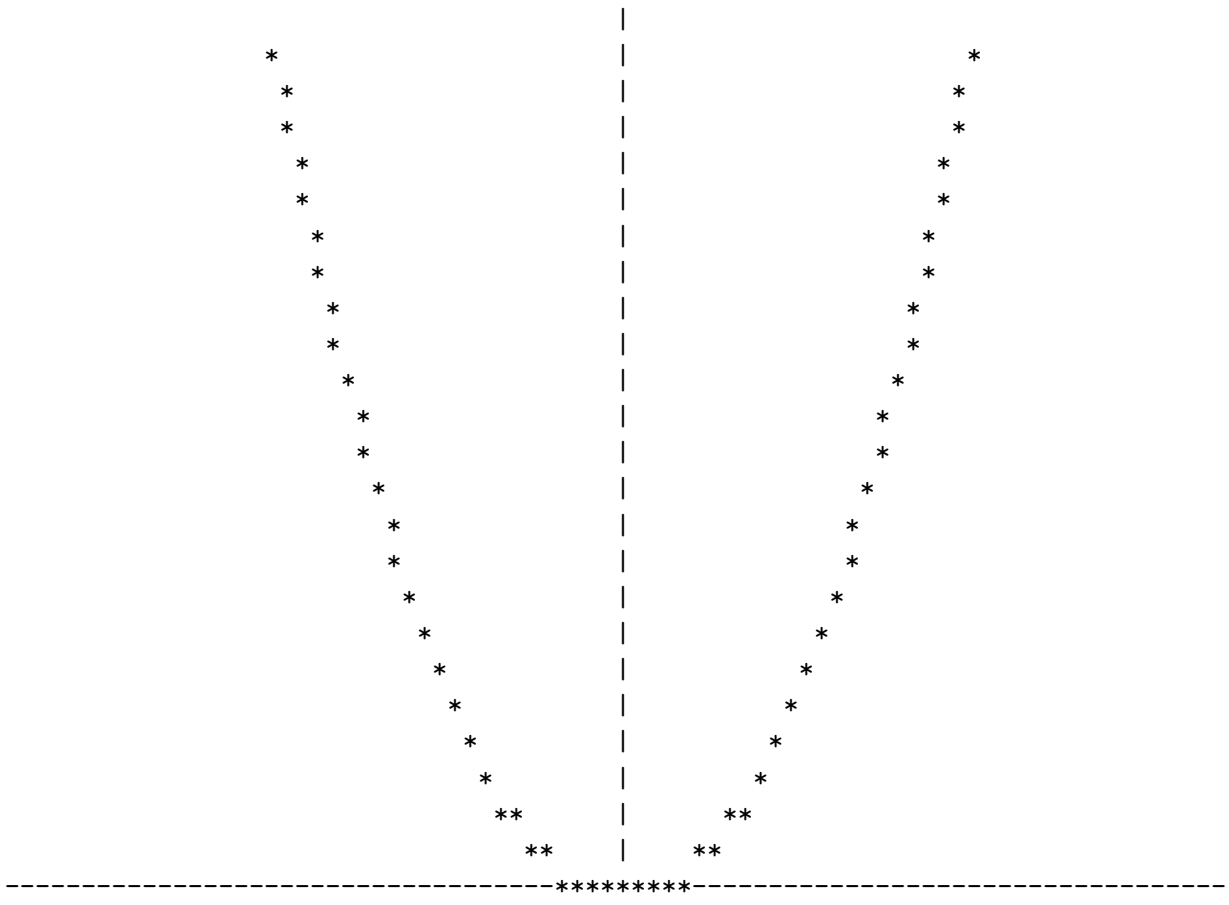
A variável interna `Connect Dots` se refere a plotagem de gráficos. Quando um gráfico é plotado, é necessário ligar cada dois "pixels" consecutivos por uma reta, caso contrário o gráfico pode se apresentar com regiões descontínuas. Essa variável interna determina se essa ligação através de retas é realizada ou não. Em geral, se utiliza o Algorito de Bresenham para se traçar tais retas.

De forma a ilustrar a diferença que tal ajuste causa na plotagem de gráficos, considere o exemplo apresentado a seguir:

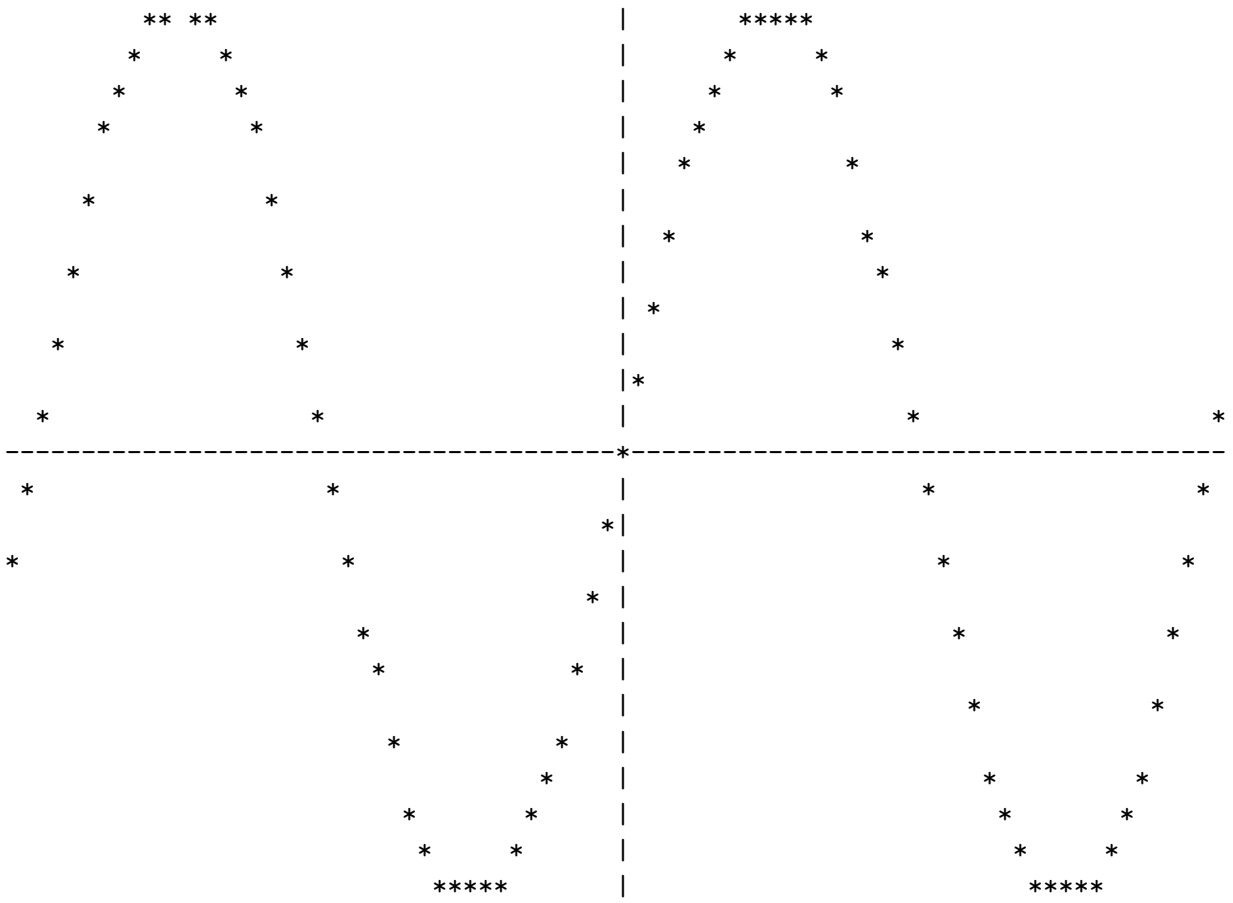
```
>set v_view 0:15;
>plot(x^2);
```



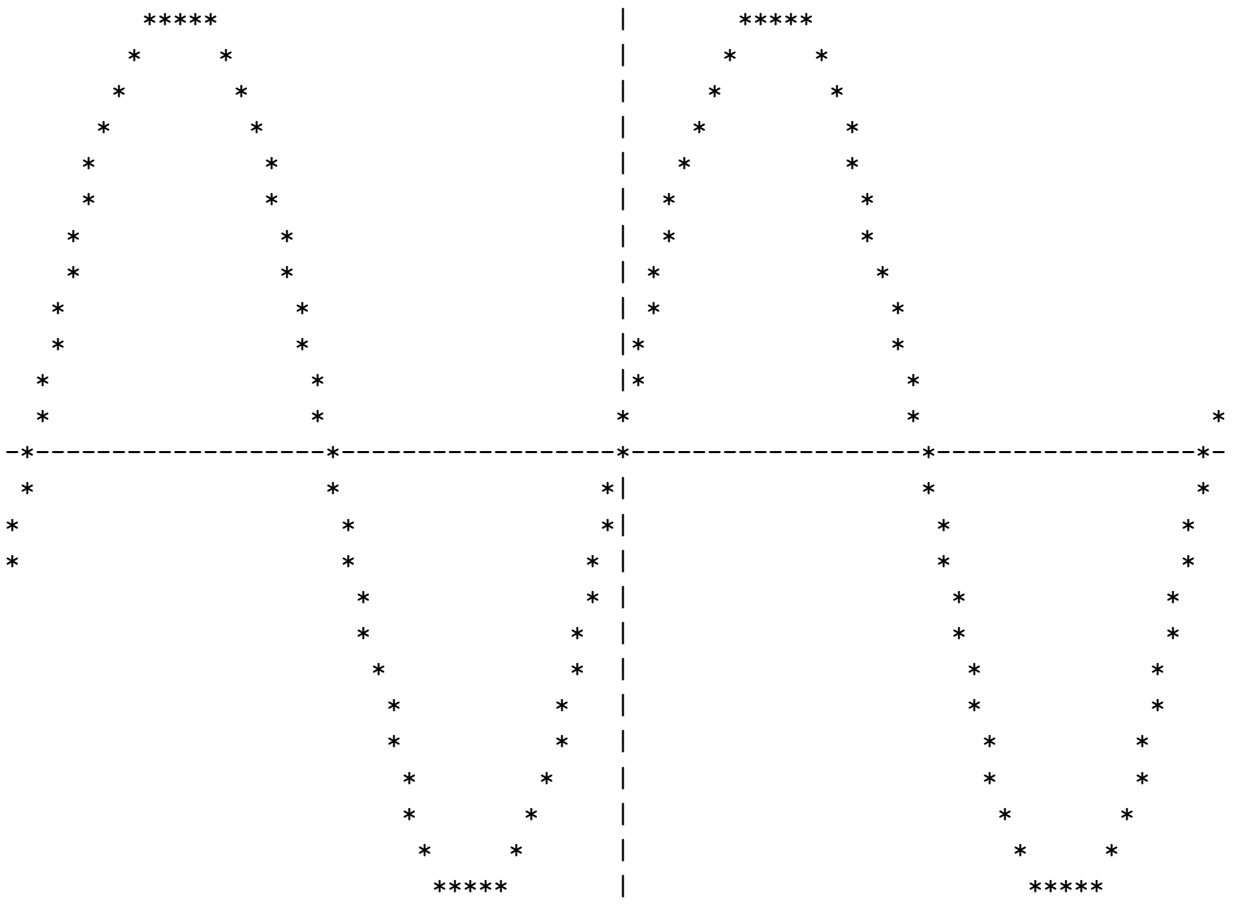
```
>set connect_dots on;
>plot;
```



```
>reset settings;
>set v_view -1:1;
>plot(sen(x));
```



```
>set connect_dots on;
>plot;
```



O comando `set connect_dots on;` habilita a geração de retas entre os pontos e o comando `set connect_dots off;` desabilita tal funcionalidade.

IMPORTANTE: a implementaçao dos comandos `set connect_dots on;` e `set connect_dots off;` **não é obrigatória**, ou seja, é **OPCIONAL**.

7 Critérios de Correção

Cada comando/mensagem de erro será testado 2 vezes, sendo a nota final proporcional a quantidade de comandos/mensagens que funcionaram corretamente. Os comandos serão inseridos continuamente dentro de uma mesma execução do programa, seguindo uma ordem aleatória.

DICA: ao testar seu programa, muitas vezes os comandos podem funcionar corretamente de forma isolada. O principal motivo de erro neste tipo de software são acessos inválidos à posições de memória e ponteiros perdidos. Muitas vezes um comando com problema, quando executado de forma isolada não causa problemas ao sistema, mas quando uma grande quantidade de comandos é realizada de forma sequencial e aleatória, os problemas de memória vão se acumulando e eventualmente causam `segmentation fault`. Por isso teste bastante o seu programa e se possível utilize ferramentas para a detecção de problemas de memória.

8 Especificações de Entrega

O trabalho deve ser entregue no *AVA* em um arquivo .zip com o nome `dcmat.zip`. A entrega deve ser feita exclusivamente no *AVA* até a data/hora especificada. Não serão aceitas entregas atrasadas ou por outro meio que não seja o *AVA*.

Este arquivo .zip deve conter somente os arquivos necessários à compilação, sendo que deve haver um `Makefile` para a geração do executável.

Observação: o arquivo .zip **não** deve conter pastas, para que quando descompactado, os fontes do trabalho apareçam no mesmo diretório do .zip. O nome do executável gerado pelo `Makefile` deve ser `dcmat`.

IMPORTANTE: Arquivos ou programas entregues fora do padrão receberão nota **ZERO**. Entende-se como arquivo fora do padrão aquele que tenha um nome diferente de `dcmat.zip`, que contenha pastas ou não seja um `.zip`, por exemplo. Entende-se como programa fora do padrão aquele que não contiver um `Makefile`, que apresentar erro de compilação ou o nome do executável for diferente de `dcmat`, por exemplo.

IMPORTANTE: Se ficou com alguma dúvida em relação a qualquer item deste texto, não hesite em falar com o professor da disciplina, pois ele está à disposição para sanar eventuais dúvidas, além do que, isso faz parte do trabalho dele.