Programação em MatLab - introdução

1. **Programas (script) -** um conjunto de comandos e instruções em MatLab armazenadas num M-Ficheiro

% <program-name>
{<specification-statements>}
{<executable-statements>}

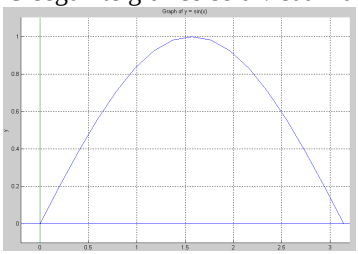
Exemplo 1: Escreva um programa em MatLab que construa o gráfico da função y=sin(x), para x pertencente ao intervalo [0, π] com passo π /8 . Grave o programa num ficheiro com o nome PlotSin.m
O Matlab permite rapidamente avaliar funções num largo conjunto de dados. Por exemplo, se X=[-1 0 1], então **sin(X)** produz o resultado **[sin(-1) sin(0) sin(1)]**.
Do mesmo modo se X= [0:pi/8:pi] então **Y=sin(X)** produz um vector **Y** com a mesma dimensão que **X e** com os correspondentes valores de sen(x).

```
% PlotSin – constroi o gráfico da função y=sin(x)
X = 0:pi/16:pi;
Y = sin(X);
figure(1); clf;
hold on;
axis([-0.2 3.2 -0.1 1.1]); % define os eixos de oordenadas
plot([-0.2 3.2],[0,0],[0,0],[-0.1 1.1]); % desenha os eixos
plot(X,Y); % desenha o gráfico
xlabel('x');
ylabel('y');
title('Graph of y = sin(x)');
grid;
hold off;
```

Execute o programa **PlotSin** na janela de comandos do MATLAB.

>> **PlotSin**

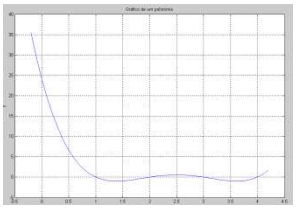
O seguinte gráfico será visualizado:



Comentário: O comando **plot** requer dois vectores de igual comprimento.

Exemplo 2: Escreva um programa em MatLab que construa o gráfico do polinómio $p(x) = x^5 - 10\,x^4 + 35\,x^3 - 50\,x^2 + 24$.
Grave o programa num ficheiro com o nome PlotPoly.m

```
% PlotPoly  constroi o gráfico do polinómio
figure(1);clf;
C = [1 -10 35 -50 24]; % define um vector com os coeficientes do polinómio
X=[-0.2:0.1:4.2]; % define as abcissas
Y=polyval(C,X); % polyval - função que avalia um polinómio
axis([-0.2 4.2 -2.3 4.3]); % define os eixos
plot([-0.2 4.2],[0,0],[0,0],[-2.3 4.3]); % desenha os eixos
plot(X,Y); % desenha o gráfico
hold on;
xlabel('x');
ylabel('y');
title('Gráfico de um polinómio');
grid;
```

Execute o programa **PlotPoly**. O seguinte gráfico será visualizado: 

Exemplo 3: Escreva um programa em MatLab onde o utilizador pode introduzir uma função para construir o gráfico.
Sugestão: A função f pode ser definida como texto, por exemplo, se f='sin(x)', então pode ser usado o comando `fplot(fun,limits) ou fplot(fun,limits,LineSpec)`.

```
% PlotFun constroi o gráfico de uma função
f=input('f(x)= ','s');
a= input('limite inferior para x  ');
b= input('limite superior para x  ');
figure(1);clf;
fplot(f,[a b]); % desenha o gráfico
hold on;
xlabel('x');
ylabel('y');
title(f);
grid;
```

2. Funções em MatLab

function [lista de parâmetros de saída] = **nome** (lista de parâmetros de entrada)
{<instruções para declaração de variáveis>}
{<instruções de execução>}

Exemplo 4 : Implemente a função **sraiz** para determinar a raíz quadrada positiva de um número positivo pelo método de Newton. Grave esta função no M-ficheiro **sraiz.m.**

```
function [r]=sraiz(A)
% Método de Newton para determinar a raiz quadrada de A
p0 = 1; % valor inicial
for k=1:50,
p1 = (p0+A/p0)/2;
disp(p1);
if abs(p1-p0)/p1 < eps, break, end;
p0 = p1;
end
r=p1;
```

Executando a função **sraiz** na janela de comandos do MATLAB, obtém-se:

```
>> sraiz(9)
    5
  3.4000
  3.0235
  3.0001
  3.0000
   3
   3
```

Exemplo 5 : Implemente em Matlab a função **f2ramos** definida por:

$$f(x) = \begin{cases} e^{-\theta} & se\ \theta < 0 \\ 1 + arctg(\theta) & se\ \theta \geq 0 \end{cases}$$

Grave esta função no M-ficheiro **f2ramos.m**

```
function [y]=f2ramos(t)
if t<0
    y=exp(-t);
else
    y=1+atan(t);
end
```

Agora executando a função **f2ramos** na janela de comandos do MATLAB, obtém-se por exemplo:

```
>> t=-0.5; fprintf('Valor de f(%f)=%f \n',t,f2ramos(t))
Valor de f(-0.500000)=1.648721
>> t=0.5; fprintf('Valor de f(%g)=%g \n',t,f2ramos(t))
Valor de f(0.5)=1.46365
```

Aqui foi utilizada a função **fprintf(***formato, resultados***)** em que format é um string.

Alguns caracteres de conversão, controlos de escrita do texto utilizados no *formato* são:

\t tabelamento (de 4 em 4 caracteres)	%d inteiro
%f real em vírgula fixa com 6 casas decimais	%s texto
%e real em vírgula flutuante com 6 casas decimais	\n muda de linha
%g real na forma mais compacta	0 os espaços são preenchidos por zeros

Exemplo 6 : Escreva as seguintes funções MatLab nos ficheiros f.m and G.m.

```
function y = f(x)
y = exp(-x./10) + sin(x);
```

```
function W = G(Z) % Z vector linha com 2 componentes
x = Z(1);
y = Z(2);
W = [x.^2-y.^2   2*x.*y];
```

Execute as funções f e G na janela de comandos. Obtém-se:

```
» f(pi/2)
ans =
1.85463599915323
```

```
» G([2 1])
ans =
     3     4
```

Exemplo 7 : Escreva as seguintes funções MatLab no ficheiro chamada.m.

```
function [y]=chamada(x)
y=fa(x)+fb(x);
return


function [y]=fa(t)
if t<0,
    y=exp(-t);
else
    y=1+atan(t);
end
return


function [y]=fb(t)
if t<0,
    y=exp(-2*t);
else
    y=1+atan(t^2);
end
return
```

Agora executando a função **chamada** na janela de comandos do MATLAB, obtém-se por exemplo:

```
>> z=chamada(0)+chamada(1)
z =
   5.5708
```

Se introduzir na janela de comandos a seguinte instrução **z=chamada(0)+fa(2)+fb(3)** obtem-se

```
>> z=chamada(0)+fa(2)+fb(3)
??? Undefined function or method 'fa' for input arguments of type 'double'.
```

Porquê?

3. Ficheiros de dados em MatLab

A instrução **save nome_ficheiro lista_variáveis opções** guarda variáveis do workspace num ficheiro. Para carregar para o espaço de trabalho as variáveis guardadas num ficheiro utiliza-se a instrução **load nome_ficheiro lista_variáveis**. Por exemplo:

```
>> clear variables
>> a=1;b=sin(a)+exp(a);load fich1 a b
```

```
>> clear variables
>> load fich1 a b;disp(a);disp(b)
    1
 3.5598
```

4. Instruções de Controlo para programação em MatLab

4.1 IF – ELSEIF – ELSE – END

if (<expresão-lógica#1>), {<instruções executáveis>} **elseif** (<expresão-lógica#2>), {<instruções executáveis>} …. **else** {<instruções executáveis>} **end**	**%Exemplo** **>> k=4;** **>> for m = 1:k** **for n = 1:k** **if m == n** **a(m,n) = 2;** **elseif abs(m-n) == 2** **a(m,n) = 1;** **else** **a(m,n) = 0;** **end** **end** **end** **>> a** **a =** **2 0 1 0** **0 2 0 1** **1 0 2 0** **0 1 0 2**

4.2 Ciclo FOR

for variável = expressão
{<instruções executáveis>}
end

Exemplo 7:	**Exemplo 8:**	**Exemplo 9:**

for k=1:100,
x=sqrt(k);
if x>5, **break**, **end**
end
	sum1 = 0;
	for k = 1:1:10000,
	sum1 = sum1 + 1/k;
	end
	» sum1
	sum1 =
	9.78760603604434
		sum2 = 0;
		for k = 10000:-1:1,
		sum2 = sum2 + 1/k;
		end
		» sum2
		sum2 =
		9.78760603604439

A instrução **break** interrompe o ciclo A instrução **Pause** gera uma pausa na execução do programa

4.3 Ciclo WHILE

while <expressão>
{ <instruções executáveis>}
end
	Exemplo 11:
	
	m = 10;
	k = 0;
	while k<=m
	x = k/10;
	disp([x, x^2, x^3]); % uma tabela de valores
	k = k+1;
	end

5. Outras funções do MatLab

Nota: exemplos de utilização destas funções podem ser estudadas usando o HELP

Funções Matemáticas:

cos(x) → cosine (radians)	**fix(x)** → round towards zero
sin(x) → sine (radians)	**floor(x)** → round towards -ì
tan(x) → tangent (radians)	**ceil(x)** → round towards +ì
cosd(x) → cosine (degrees)	**sign(x)** → signum function
sind(x) → sine (degrees)	**cosh(x)** → hyperbolic cosine
tand(x) → tangent (degrees)	**sinh(x)** → hyperbolic sine
exp(x) → exponential exp(x)	**tanh(x)** → hyperbolic tangent
acos(x) → inverse cosine (radians)	**acosh(x)** → inverse hyperbolic cosine
asin(x) → inverse sine (radians)	**asinh(x)** → inverse hyperbolic sine
atan(x) → inverse tangent (radians)	**atanh(x)** → inverse hyperbolic tangent
log(x) → natural logarithm base e	**real(z)** → real part of complex number z
log10(x) → common logarithm base 10	**imag(z)** → imaginary part of complex number z
sqrt(x) → square root	**conj(z)** → complex conjugate of the complex number z
abs(x) → absolute value	**angle(z)** → argument of complex number z
round(x) → round to nearest integer	**rem(p,q)** → remainder when p is divided by q

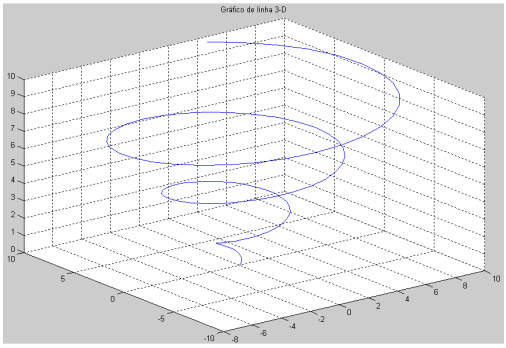
Funções para Análise de Dados:

max → maximum value	
min → minimum value	**cumsum** → cumulative sum of elements
mean → mean value	**cumprod** → cumulative product of elements
median → median value	**diff** → approximate derivatives (differences)
std → standard deviation	**hist** → histogram
sort → sorting	**corrcoef** → correlation coefficients
sum → sum the elements	**cov** → covariance matrix
prod → form product of the elements	

6. Gráficos 3-D

Exemplo de gráfico de linhas:

```
>> % Exemplo de utilização da função plot3(x,y,z)
>> z=[0:0.1:10];
>> x=z.*cos(2*z);y=z.*sin(2*z);plot3(x,y,z);
>> title('Gráfico de linha 3-D'); grid on
```



Exemplos de outros gráficos 3-D são os chamados gráficos de superfície e de contornos cujos exemplos podem ser consultados usando a facilidade HELP do MatLab.

Exercícios de introdução à Programação em MatLab

1. Uma equação do 2º grau tem a forma genérica $ax^2 + bx + c = 0$ com $a \neq 0$. Escreva um programa em MatLab que leia os valores dos parâmetros a, b e c e determine as duas raízes da equação.
   Nota: as raízes podem ser reais ou complexas.

2. Numa conta a prazo com capitalização automática o valor do juro é acumulado ao capital inicial no final de cada período. Assim, o capital acumulado ao fim de N períodos é dado pela equação
   $$Capital\ final = Capital\ inicial\ \times (1 + t)^N$$
   onde t é a taxa de juro aplicada $(0 < t < 1)$.
   Escreva um programa MatLab que tenha como dados o *Capital inicial*, a taxa de juro anual *t* e o número de anos *N* e que calcule o capital acumulado ao fim desses anos.

3. Escreva um programa em MatLab que mande desenhar um gráfico com várias funções sobrepostas. O utilizador deve poder escolher quantas e quais as funções a introduzir para construir o gráfico.

4. Escreva uma função em MatLab para procurar numericamente o limite
   $$\lim_{x \to a} f(x)$$
   Para procurar numericamente este limite pode avaliar *f(x)* para a seguinte sucessão: $a + \frac{h}{b}, a + \frac{h}{b^2}, \ldots, a + \frac{h}{b^n}, \ldots$ para valores de b e n dados sendo h escolhido convenientemente. Experimente para b=5, h=1 e n=10 calculando
   a) $\lim_{x \to 1} \left(\frac{\ln x}{x^2 - 1}\right)$ b) $\lim_{x \to 0} (sen\ x)^x$

5. A instrução 1+fix(6*rand(1,10)) simula o resultado de fazer dez lançamentos de um dado. Usando uma instrução semelhante, escreva um programa para:
   a) Contabilizar o número de vezes que em 50 lançamentos saíram valores pares;
   b) Contabilizar o número de vezes que nos 50 lançamentos, a soma do resultado de dois lançamentos consecutivos foi de exactamente 8.

TPC5: 2º e últimoTrabalho de LI
Trabalho em MatLab para entrega via Moodle:

Os exercícios 1 a 5 contituem o enunciado do TPC5, 2º e último trabalho em MatLab a ser entregue até à meia-noite de domingo, 7 de Fevereiro de 2010. Os alunos deverão submeter via Moodle uma única pasta *zipada* contendo ficheiros MatLab do tipo diary e do tipo <nome>.m com o trabalho desenvolvido.

Prova P3: Criação de uma página na internet e colocação de um portefólio com os trabalhos realizados ao longo do semestre (trabalhos obrigatórios e trabalhos opcionais): esta prova deve ser entregue até 12 de Fevereiro de 2010.