Conceitos de Programação

J. Barbosa

Conceitos básicos de programação

- **Algoritmo**
  - Conjunto finito de regras sobre as quais se pode dar execução a um dado processo (Knuth73v1)
    - Ex: ordenação de um conjunto, pesquisa numa base de dados.
  - Atributos que deve possuir:
    - Ser finito, inteligível, exequível, caracterizável.
  - Formas de representação :
    - Narrativa, Fluxograma, Pseudo código, Linguagens de programação

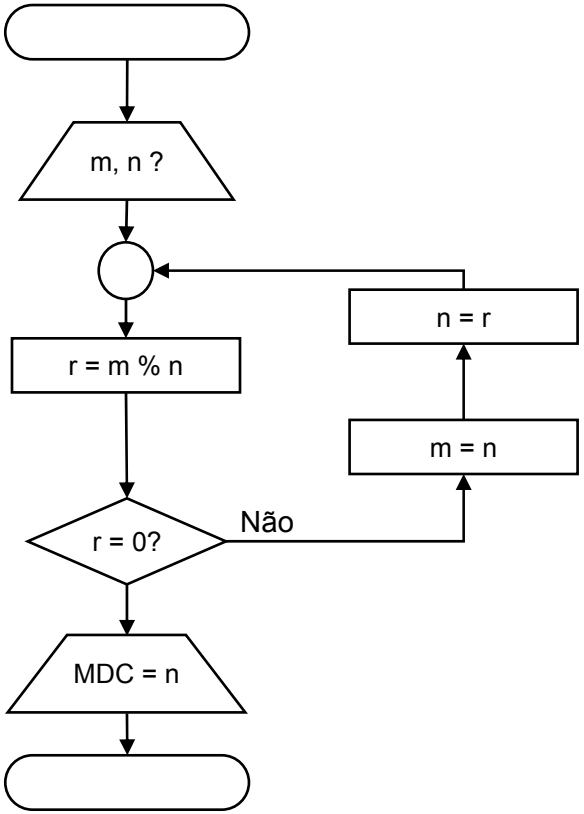
Conceitos básicos de programação
Exemplo: Algoritmo de Euclides

- **Enunciado**:
  - Dados dois inteiros **m** e **n**, encontrar o maior inteiro que os divida a ambos exactamente (máximo divisor comum).

- **Descrição narrativa:**
  - **Algoritmo mdc** (Algoritmo de Euclides)
    - 1º- (Encontrar o resto) - Dividir **m** por **n** e afectar **r** com o resto (**0 <= r < n**)
    - 2º- (O resto é zero?) - Se **r=0**, o algoritmo termina; **n** é o valor procurado.
    - 3 º- (Substituir) - Afectar **m** com **n** e **n** com **r**, voltando ao passo 1.

Conceitos básicos de programação
Exemplo: Algoritmo de Euclides

Descrição em fluxograma



- **Pseudocódigo:**
  - descrição do algoritmo próxima da linguagem de programação mas escrita em linguagem corrente.

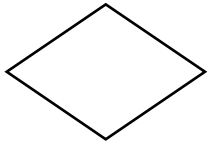
```
Função mdc
  Leia m
  Leia n
  r = mod(m,n)
  Enquanto (r ~= 0)
     m=n
     n=r
     r = mod(m,n)
  Escreve n
```

Conceitos básicos de programação: Fluxograma

Início e fim

Entrada e saída de dados

Intruções de atribuição/execução



Intruções condicionais

Liga pontos distintos do algoritmo

Conceitos básicos de programação

Descrição em Matlab:

```
function n=mdc
   m=input('Valor m?')
   n=input('Valor n?')
   r = mod(m,n);
   while ( r ~= 0)
      m=n;
      n=r;
      r= mod(m,n);
   end
```

Descrição em linguagem C:

```
int mdc(int m, int n) {
   int r;
   while ( (r= m % n) != 0) {
            m=n;
            n=r;
   }
   return n;
}
```

Variáveis

As **variáveis** representam a memória do computador onde se podem guardar dados de entrada e resultados. Facilitam a escrita dos programas ao permitirem identificar a memória através de nomes escolhidos pelo utilizador.

Nome da variável: letras, números e _

- Primeiro caracter tem de ser uma letra
- Distingue entre maiúsculas e minúsculas

ans: variável que fica com o resultado de uma operação quando não é especificada pelo utilizador uma variável para guardar esse resultado.

Palavras reservadas: **iskeyword**

Variáveis definidas: **whos**

Operadores

- Operadores aritméticos: +, -, *, /
- Operadores de relação: <, <=, ==, ~=, >, >=
- Operadores Lógicos: & (e), | (ou), ~ (negação), xor (ou exclusivo)

Matlab

Alguns operadores:

+	Addition
-	Subtraction
*	Multiplication
/	Division
:	Colon operator
^	Power
.‘	Transpose
‘	Complex conjugate transpose
	Matrix multiplication/Matrix right division

Precedência dos operadores

$$\frac{-b+\sqrt{b^2-4ac}}{2a}$$ (-b+sqrt(b^2 – 4*a*c)) / (2*a)

Ordem:

- (…) conteúdo dos parêntesis
- ^ expoente
- ~
- *, /
- +, –
- Operadores de relação
- &
- |

Tipos de dados

inteiro	Números sem parte decimal, como 12562, -25
real com vírgula fixa	Números com parte decimal, como 35.1256, 0.65141
real com vírgula flutuante	Números com parte decimal em notação científica, como 0.351256e2, 0.65141
complexos	Números com parte real e imaginária
string	Sequências de caracteres da tabela ASCII

Funções para entrada e saída de dados

- Entrada de dados pelo teclado

valor numérico:

```
a = input('Introduza um valor?')
```

para ler uma string:

```
a = input('Introduza uma string?','s')
```

Funções para entrada e saída de dados

- Escrita de dados para o ecrã: ***fprintf (formato, valores)***

  Podemos escrever qualquer tipo de variável e na mesma instrução.

  ex: fprintf(‘O resultado é : %d’, x);

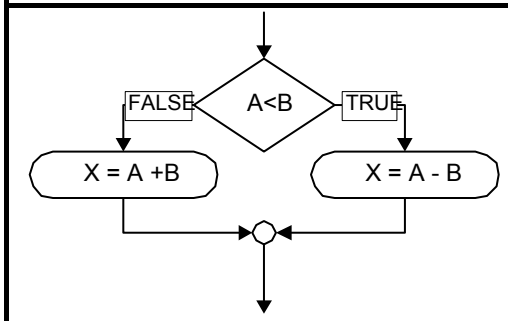
  %d apresenta os valores como inteiros
  %e apresenta os valores em formato exponencial
  %f apresenta os valores em vírgula flutuante
  %s escreve uma string
  \n muda de linha
  \t ‘tab’, permite organizar a escrita de dados

Instruções de controlo de execução

- Decisão binária - *if*
- Decisão múltipla – *switch*
- Repetição condicional - *while, for*

Decisão binária - *if*

Fluxograma	Em Matlab

	`if ( A < B )`
	`  x= A + B`
	`else`
	`  x= A – B`
	`end`

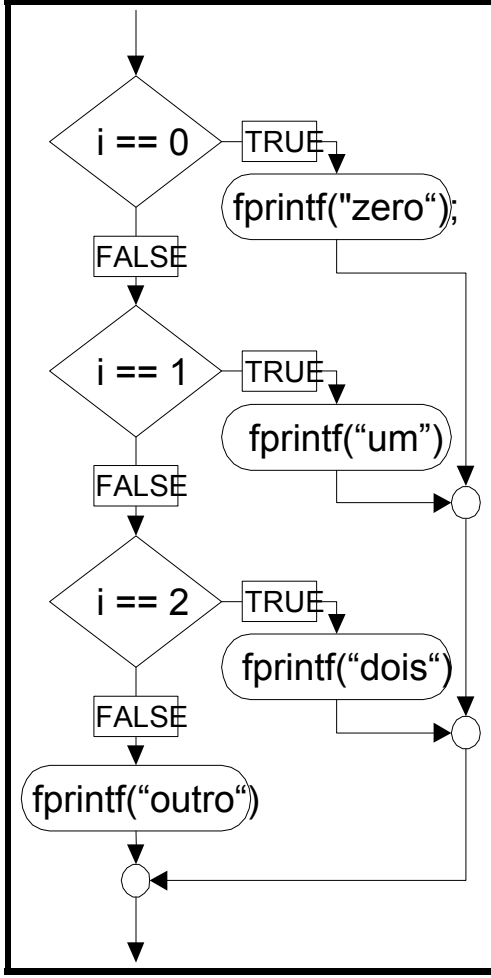

```
If exp1
        instruções 1
elseif exp2
        instruções 2
elseif exp3
        instruções 3
else
        instruções 4
end
```

Exemplo: Determinar se o ano é bissexto

```
function bissexto(ano)
   if (mod(ano,400) == 0 | mod(ano,4) == 0 & mod(year,100) ~= 0)
         fprintf('sim')
   else
         fprintf('Nao e')
   end
   fprintf(' um ano bissexto.')
}
```

Decisão múltipla – *switch*

Fluxograma



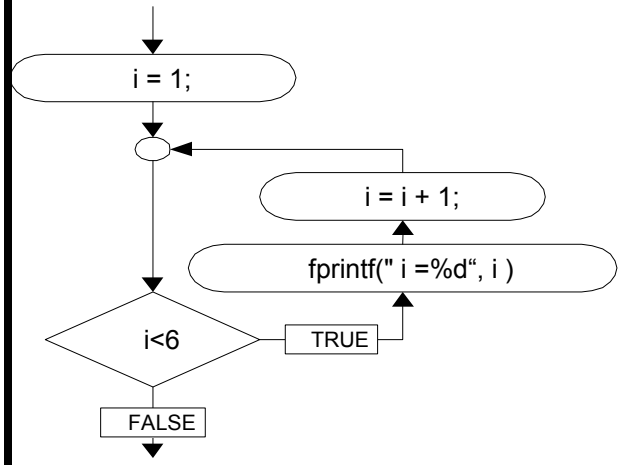
Em Matlab

```
switch i
   case 0
      fprintf('zero')
   case 1
      fprintf('um')
   case 2
      fprintf('dois')
   otherwise
      fprintf('outro')
end
```

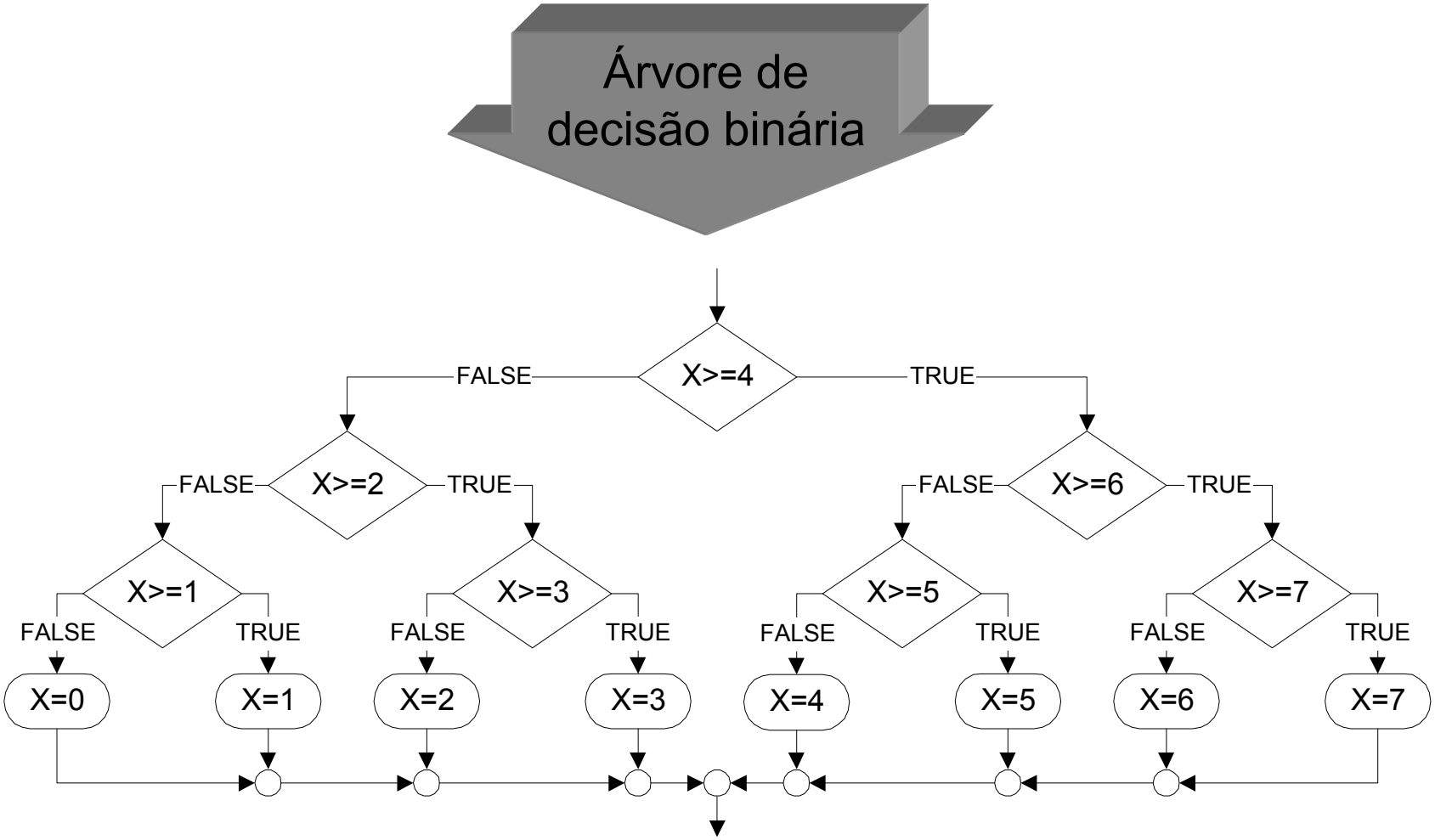
Pode escrever-se:
case {exp1, exp2, exp3,…}

Repetição condicional - *while, for*

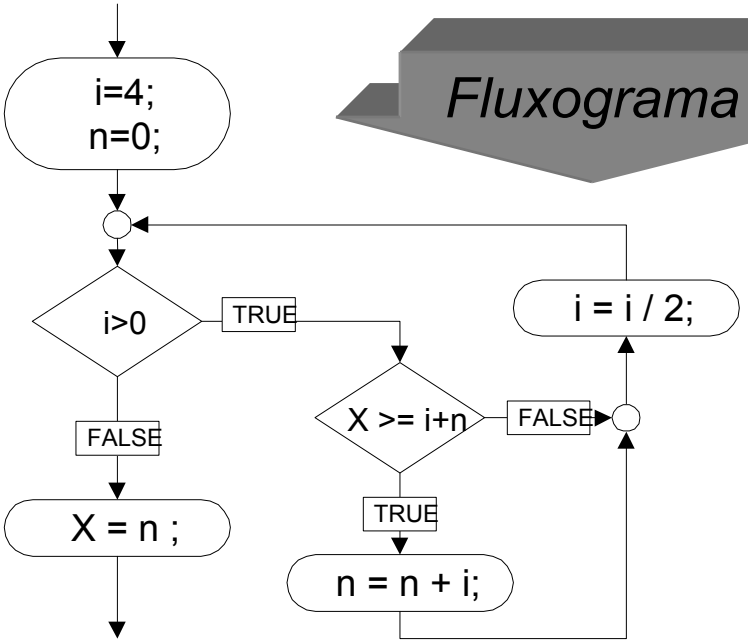
Fluxograma	Em Matlab

	`i = 1;`
	`while i<6`
	`  fprintf('i=%d',i)`
	`  i = i + 1 ;`
	`end`
	
	`for i=1:6`
	`  fprintf('i=%d',i)`
	`end`

Exemplo: Algoritmo de aproximações sucessivas
Quantas tentativas são necessárias para adivinhar um nº entre 0 e 7?



Algoritmo de aproximações sucessivas



Fluxograma

Exercício: passar o fluxograma para um programa em Matlab

Programa em Matlab

```
i = floor((MAXIMO+1)/2);
numero = 0;
while i>0
    fprintf('valor maior ou igual que %d ',i+numero);
    f=input('(S/N)?','s')
    if f == 's'
        numero = numero + i;
    end
    i = floor(i / 2);
end
fprintf('O numero que pensou e' %2.0d', numero);
```

Exercícios

- Escreva uma função para resolver equações quadráticas:

$ax^2+bx+c=0$

- O ganho de tensão de um amplificador é dado por

$$\mathbf{v} = [23/(23^2 + (0.5f)^2)^{1/2}]^n$$

onde f é a frequência de funcionamento em Hertz e n o número de etapas do amplificador.

Escreva uma função que calcule **v** em função dos valores **f** e **n.**

Vector

- Vector: é uma sequência de valores do mesmo tipo à qual é dada no programa um nome único. Os elementos do vector são acedidos pelo nome e pela posição ocupada na sequência.

  v = 2 variável escalar

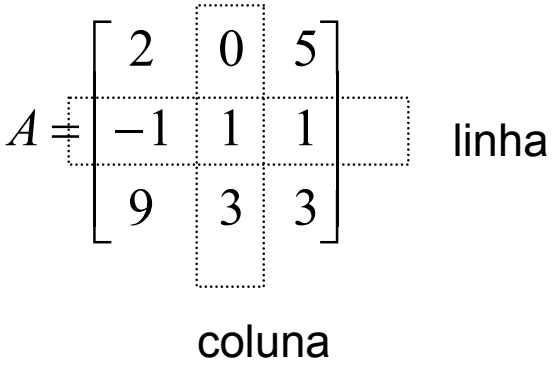
  m=[2 4 -3] vector com 3 elementos

  v = m(1) v ficará com o valor 2, um escalar.

  k = lenght(m) k ficará com o valor 3, ou seja, a dimensão do vector

Matriz

- Matriz: é um vector com duas dimensões



b = A(1,3) b ficará com o valor 5

Matriz

[1,2,3]	vector linha ou matriz 1x3
[1;2;3]	vector coluna ou matriz 3x1
[1,2;3,4;5,6]	matriz 3x2
[]	matriz vazia 0x0

Operador (:)

x = 0:0.2:1 equivale a x=[0,0.2,0.4,0.6,0.8,1]

Operador (‘) transposta:

A’ corresponde à transposta da matriz A, ou seja, transforma as linhas em colunas.

Matrizes

- **Funções disponiveis para inicializar matrizes:**

zeros(n)	matriz de zeros de nxn
zeros(m,n)	idem de mxn
ones(n)	matriz de uns de nxn
ones(m,n)	idem de mxn
eye(n)	matriz identidade de nxn
eye(m,n)	matriz de zeros e uns nas posições (i,i)
rand(n)	matriz aleatória de nxn
rand(m,n)	idem de mxn
magic(n)	matriz nxn onde a soma dos elementos de qualquer linha ou coluna é sempre igual

Matrizes

- **Exemplos:**

```
a=ones(3,2)
b=[a, zeros(3), a]
c=eye(size(b))
d = rand(2)
```

- Soma de matrizes

  C = A + B

- Produto de matrizes

  C = A*B o número de colunas da primeira tem de ser igual ao número de linhas da segunda matriz.

Matrizes

- Divisão matricial

  x=A\b é a solução de A*x=b

  x=A/b é a solução de x*A=b

Funções que operam por colunas/linhas

- max, min, mean, sort, sum, prod

Ex:

x = [1:5:20] A=[1:3;4:6]

a) a=sum(x) a=34

b) b=sum(A) b=[5 7 9]

c) c=sum(A,2) c=[6;15] soma os elementos de cada linha

d) d=sum(A(:)) d=21 soma todos os elementos da A

e) e=max(A) e=[4 9 6] maior elemento de cada coluna

f) [f1,f2]=max(A) f1=[4 9 6] f2=[2 1 2] f1 contém os maiores elementos em cada coluna e f2 o índice da linha

Gravar e ler dados de ficheiros

- ***save*** nome_ficheiro var1 var2 … -opcoes

  opções:

-mat	Formato binário (opção por defeito)
-append	Acrescenta os dados no fim do ficheiro (por defeito se o ficheiro já existir apaga o conteúdo anterior)
-ascii	Formato ascii, mantissa de 8 digitos
-ascii -double	Formato ascii, mantissa de 16 digitos
-compress	faz compressão de dados

Ex:

save fnome a b	guarda no ficheiro **fnome.mat** o conteúdo das variáveis **a** e **b**
save fnome	guarda todas as variáveis definidas

Gravar e ler dados de ficheiros

- ***load*** nome_ficheiro var1 var2 … -opcoes

load fnome s t	lê as variáveis s,t do ficheiro, que poderá ter mais variáveis
load fnome.dat	lê todas as variáveis do ficheiro Nota: se o ficheiro tiver uma extensão diferente de .MAT o Matlab considera que está em ASCII.

clear	limpa todas as variáveis definidas
clear a c	limpa apenas as variáveis **a** e **c**

Gráficos

Alguns tipos:

plot(x,y)	gráfico linear x-y
loglog(x,y)	ambas as escalas logaritmicas
semilogx(x,y)	escala de x logaritmica
semilogy(x,y)	escala de y logaritmica
bar(x,y)	gráfico de barras
barh(x,y)	gráfico barras horizontais
stairs(x,y)	gráfico em degraus
hist(x,n)	histograma
pie(x)	gráfico redondo de percentagens

Gráficos

Ex:

```
x = -2:0.2:2;
y = cos(x);
plot(x,y,'r-')   gráfico linear x-y com linha sólida vermelha
grid on
```

Representar duas funções:

```
z = sin(x)
plot(x,y,'r-',x,z,'g:')
ou:
plot(x,y,'r-')
hold on
plot(x,z,'g:')
```

Gráficos

Comandos sobre os gráficos:

```
title('Título')
xlabel('x')
ylabel('y')
text(x,y,'texto')
gtext('texto')          Posiciona o texto com o rato
grid on
grid off
axis([xmin xmax ymin ymax])
subplot(n,m,p)          coloca vários gráficos na mesma janela
```

Bibliografia

Prontuário do Matlab

de Fernando Gomes Martins

FEUP Edições