Programação e Software

Aula 2
Métodos Numéricos Aplicados à Engenharia

Objectivo

- Como usar o computador como ferramenta para a obtenção de soluções numéricas de problemas de Engenharia
- Podemos utilizar os computadores de duas maneiras:
  - Utilizando software/programas disponíveis
  - Ou, escrever programas de computador que nos permitam aumentar as capacidades do software disponível (por exemplo do EXCEL)
- Um engenheiro não deverá estar limitado pelas ferramentas existentes!
  - Deverá ser capaz de utilizar os programas existentes e programar no caso da ferramenta adequada não existir.

- Programas de computador
  - Conjunto de instruções que indicam ao computador o modo de realizar uma dada tarefa
- De modo a serem capazes de efectuar cálculos numéricos no computador é necessária familiaridade com os seguintes tópicos:
  - Representação de informação simples (declaração de constantes e variáveis)
  - Representação de informação mais complexa
    - estrutura de dados, vectores/matrizes e Registos (records)
  - Formulas matemáticas
    - atribuição, regras de prioridade e funções intrínsecas
  - Entrada/ Saída (Input/Output)
  - Representação (fluxo) lógico
    - sequencia, selecção e repetição
  - Programação modular (funções e sub rotinas)
- Relembrar o que aprenderam em disciplinas anteriores!

Programação Estruturada

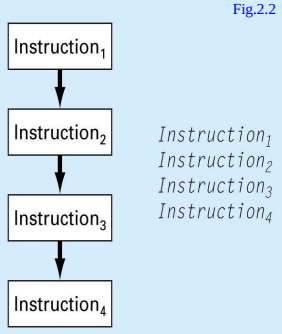
- *A programação estruturada* baseia-se num conjunto de regras que possibilitam ao programador desenvolver bons hábitos de programação...
  - Código organizado e bem estruturado
  - Código partilhável
  - Facilidade em testar e encontrar erros
  - Requires shorter time to develop, test, and update
- Qualquer algoritmo numérico pode ser programado utilizando três tipos de estruturas básicas
  - Sequencia, Selecção e Repetição

Fig. 2.1

SYMBOL	NAME	FUNCTION
	Terminal	Represents the beginning or end of a program.
	Flowlines	Represents the flow of logic. The humps on the horizontal arrow indicate that it passes over and does not connect with the vertical flowlines.
	Process	Represents calculations or data manipulations.
	Input/output	Represents inputs or outputs of data and information.
	Decision	Represents a comparison, question, or decision that determines alternative paths to be followed.
	Junction	Represents the confluence of flowlines.
	Off-page connector	Represents a break that is continued on another page.
	Count-controlled loop	Used for loops which repeat a prespecified number of iterations.

- Sequencia.
- O código é programado, e corre em sequência – uma instrução de cada vez.
- A estrutura pode ser representada utilizando um fluxograma ou pseudo-código

Fig.2.2

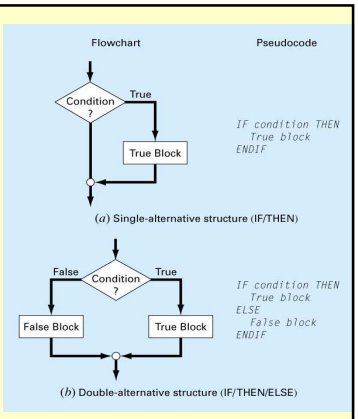


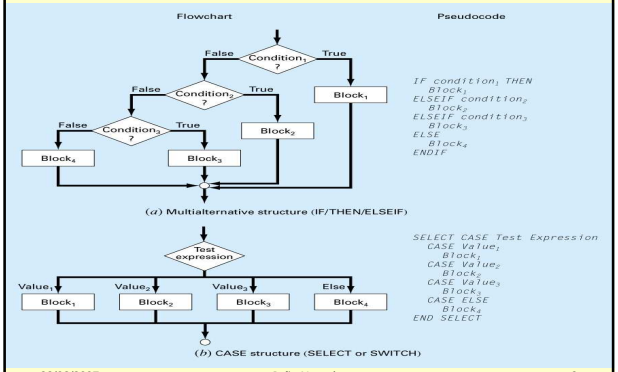
(a) Flowchart

```
Instruction1
Instruction2
Instruction3
Instruction4
```

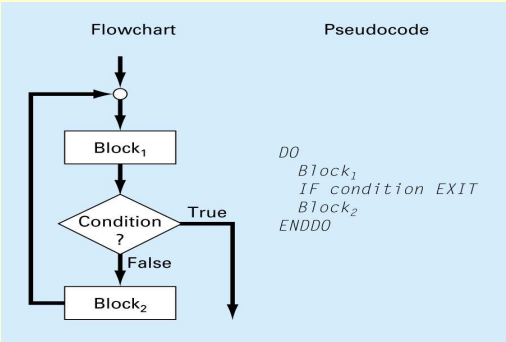
(b) Pseudocode

- Selecção.
- Permite dividir o fluxo em ramos
- A escolha do ramo é baseado no resultado de uma condição lógica

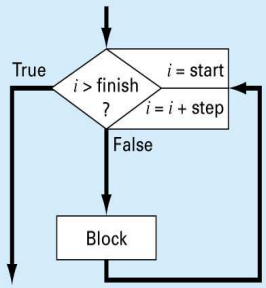




- Repetição . Permite a repetição (controlada) de um conjunto de instruções



Flowchart



Pseudocode

```
DOFOR i = start, finish, step
  Block
ENDDO
```


Programação Modular

- Os programas de computador podem (devem!) ser divididos em sub-programas (módulos) que possam ser desenvolvidos e testados separadamente
- Os módulos devem ser, na medida do possível, independentes e autónomos.
- As vantagens de uma organização modular:
  - É fácil de compreender a lógica existente em módulos pequenos
  - É mais fácil detectar erros e testar módulos pequenos
  - Facilita a manutenção e modificação
  - Permite a construção de uma biblioteca de módulos para uso em futuras aplicações


```
FUNCTION Euler(dt, ti, tf, yi)
t = ti
y = yi
h = dt
DO
  IF t + dt > tf THEN
    h = tf - t
  ENDIF
  dydt = dy(t, y)
  y = y + dydt * h
  t = t + h
  IF t ≥ tf EXIT
ENDDO
Euler = y
END
```

(a) Pseudocode	(b) Excel VBA
IF/THEN:	

IF condition THEN
  True block
ENDIF
	If b <> 0 Then
	  r1 = -c / b
	End If

IF/THEN/ELSE:	

IF condition THEN
  True block
ELSE
  False block
ENDIF
	If a < 0 Then
	  b = Sqr(Abs(a))
	Else
	  b = Sqr(a)
	End If

IF/THEN/ELSEIF:	

IF condition$_1$ THEN
  Block$_1$
ELSEIF condition$_2$
  Block$_2$
ELSEIF condition$_3$
  Block$_3$
ELSE
  Block$_4$
ENDIF
	If class = 1 Then
	  x = x + 8
	ElseIf class < 1 Then
	  x = x - 8
	ElseIf class < 10 Then
	  x = x - 32
	Else
	  x = x - 64
	End If

CASE:	

SELECT CASE Test Expression
  CASE Value$_1$
  Block$_1$
  CASE Value$_2$
  Block$_2$
  CASE Value$_3$
  Block$_3$
  CASE ELSE
  Block$_4$
END SELECT
	Select Case a + b
	  Case Is < -50
	  x = -5
	  Case Is < 0
	  x = -5 - (a + b) / 10
	  Case Is < 50
	  x = (a + b) / 10
	  Case Else
	  x = 5
	End Select

DOEXIT:	

DO
  Block$_1$
  IF condition EXIT
  Block$_2$
ENDIF
	Do
	  i = i + 1
	  If i >= 10 Then Exit Do
	  j = i*x
	Loop