Algoritmia e Programação

Engenharia Informática
1º ano - 1º semestre
Ano Lectivo 2004/2005

Ana Maria Madureira

Apresentação

- Programa da Disciplina
- Metodologia
- Método de Avaliação
- Bibliografia

Introdução à Informática

- **Informação**
  - **Informação Binária -** bit, byte, word
- **Informática** = Informação + Automática
- **Computador**
  - **conjunto de circuitos eléctricos e electrónicos** capazes de realizar tarefas de modo autónomo **por obediência a um programa**

Conceitos

- **Hardware (Componente Física)** – o conjunto de componentes mecânicos, eléctricos, magnéticos e electrónicos de um sistema informático
  - executar um determinado tipo de instruções a determinada velocidade
  - armazenar um número de bytes
  - comunicar com um conjunto de periféricos
- **Software (Componente Lógica)** que é o conjunto de programas utilizados em determinado sistema informático

Sistema de Computação

- Hardware
- Software
  - Software de Sistema
    - Sistema Operativo
    - Utilitários (editores, compiladores,...)
  - Software de Desenvolvimento
    - Ambientes de programação
    - SGBD- Sistemas de Gestão de Bases de Dados
    - Folhas Cálculo,...
  - Aplicações
    - "Packages"
    - Aplicações por medida

Sistema Operativo

- É um programa (ou conjunto de programas) que dirige o processador no
  - controlo dos recursos do sistema de computação
  - controlo da execução dos programas de aplicação dos utilizadores do sistema de computação
- Funciona como uma interface entre o utilizador e o hardware, escondendo os detalhes de hardware, fornecendo assim uma máquina virtual de mais fácil utilização
- Apresenta ao programador serviços de alto nível para uma exploração fácil e eficiente do hardware.

Desenvolvimento de Software

- Desenho e Programação Estruturados
  - Desenho Modular
  - Refinamento gradual do topo para a base
  - Programação Estruturada

Paradigmas da Programação

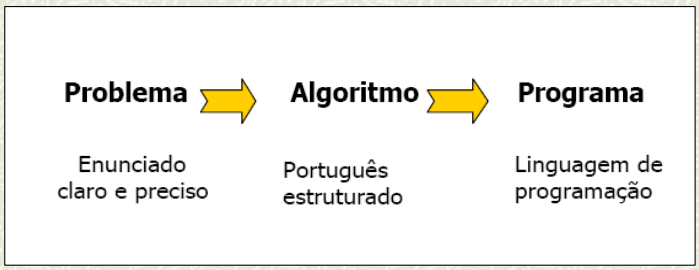
- Programação Imperativa (Procedimental)
  - Programação orientada à Acção

    PROGRAMA = Algoritmo + Estrutura de Dados.
- Programação Orientada a Objectos
- Programação Funcional
- Programação Declarativa

Resolução de Problemas

- **Analisar o problema**
  - Conhecer o bem o problema
  - Descrever o problema: subdividir, detalhar
- **Resolver o problema passo a passo,**
  - verificar se não há ambiguidade na solução apresentada, ou seja, descrever o algoritmo.
- **Implementar a solução numa linguagem de programação**

Resolução de Problemas



Algoritmo

- **Algoritmo -** sequência de passos lógicos para a realização de uma tarefa
- **Propriedades**
  - Entrada e Saída
  - Finito
  - Definido
  - Eficaz
- **Descrição de Algoritmos** - com base na Lógica da Programação e utilizando
  - Pseudo- Código
  - Fluxograma
- **Análise de Algoritmos**
  - "Tracing", Implementação e teste

Conceitos

- **Tipo de Dados**
  - Numéricos:
    - Inteiros (12, 1, 1908654, ...)
    - Reais (-12.4, 0.0000765, ...)
  - Cadeias de caracteres
    - Alfabéticos, algarismos (" Carlos", "Semestre2," ...)
    - Outros símbolos (!, #, @,£, ...)
  - As cadeias de caracteres são representadas entre aspas.
- **Estruturas de Dados**
  - (modo como os dados estão organizados e como são acedidos e alterados ...)
  - variáveis simples, arrays mono e multi-dimensionais, listas, filas, árvores, grafos,

Conceitos

- **Noção de Variável -** posição de memória
  - **Nome -** **sugestivo e curto, iniciado sempre por uma letra**
  - **Endereço**
  - **valor** (conteúdo)
- **Tipo de Dados**
  - tipo de valores
  - tipo de operações
- **Estrutura de Dados**
  - Atribuir valor a variável
  - Aceder ao valor de uma variável

Operadores

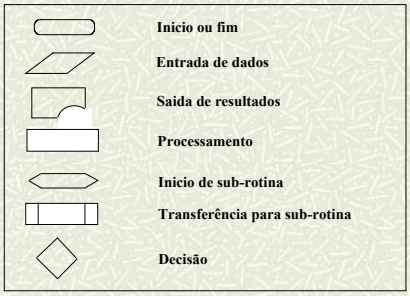
- **Aritméticos**
  - *** , / , ↑: multiplicação, divisão e exponenciação**
  - **+ , - : adição e subtracção**
- **Relacionais**
  - <= , >= , <> : menor ou igual, maior ou igual, diferente
- **Lógicos**
  - Negar : Negação
  - E : conjunção
  - Ou : disjunção

Programação Estruturada

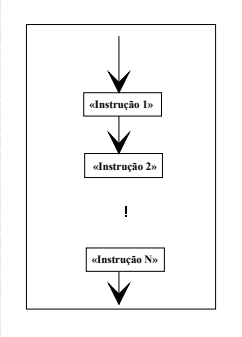
Qualquer programa pode ser descrito utilizando as três estruturas básicas de controlo:

- **Sequência** - permite a ordenação em série de instruções
- **Selecção/ Decisão** - permite a selecção em alternância de um ou outro conjunto de acções por avaliação de uma condição
- **Repetição** - permite a execução condicional em circuito fechado (ciclo) de um dado grupo de instruções. A condição é testada em cada iteração para decidir se sair ou não do ciclo.

Algoritmos

Linguagem formal	Fluxograma
INICIO ou FIM	
LER()	
ESCREVER()	
PROCEDIMENTO/FUNÇÃO	
SE...ENTÃO...SENÃO	
PARA...ATÉ...FAZER	
ENQUANTO...FAZER	
REPETIR...ATÉ	

Algoritmos - Sequência

Linguagem formal	Representação Gráfica
«Instrução 1» :::::::::::::::: «Instrução N»	

Sequência (Exemplo)

- Esboce o algoritmo que, sabendo um valor em metros, mostre o valor correspondente em centímetros.

```
ED: m, cm real
Algoritmo
Inicio
            escrever("Introduza o valor em metros:")
            ler(m)
            cm ← m*100
            escrever(m,"m =",cm,"cm")
Fim
```

Sequência (Exemplo)

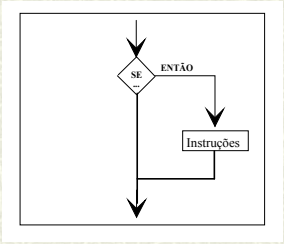
Esboce o algoritmo que permita determinar a área de um triângulo, dada a base e altura.

```
ED: real base, altura, area
Algoritmo
INICIO
    ler( base, altura)
    area <-- base* altura /2
    escrever(" Area do Triângulo=", area)
FIM
```

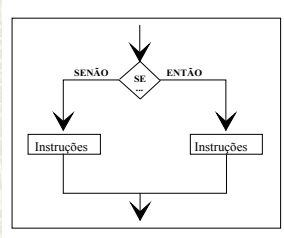
Traçagem

Decisão

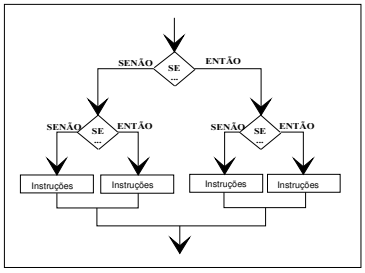
- **Decisão/ Selecção**
  - **Se (...) então ... FimSe**



  - **Se (...) então ... Senão ...FimSe**



Estruturas Encaixadas

Linguagem formal	Representação Gráfica
SE «condição 1» ENTÃO SE «Condição 2» ENTÃO «Instruções» SENÃO «Instruções» FIMSE SENÃO SE «Condição 3» ENTÃO «Instruções» SENÃO «Instruções» FIMSE FIMSE	

Decisão (Exemplo)

Desenvolver um programa que permita verificar se um dado número inteiro fornecido pelo utilizador é par ou é impar.

Algoritmo 1:

1º passo – ler o valor introduzido no teclado;
2º passo – verificar se o valor é par ou impar;
3º passo – enviar para o monitor a informação;

Algoritmo 2:

```
Inicio
    Ler(valor)
    Se valor fôr par então
        Escrever ("o valor é par")
    Senão
        Escrever ("o valor é impar")
    FimSe
Fim
```

Decisão (Exemplo)

Descrever um algoritmo para ler dois valores diferentes e determinar qual o maior.

```
ED :  A e B variáveis inteiras
Inicio
   ler(A,B)
   Se A > B Então
              MAX←A
        Senão
              MAX←B
   FimSe
   Escrever(MAX)
Fim
```

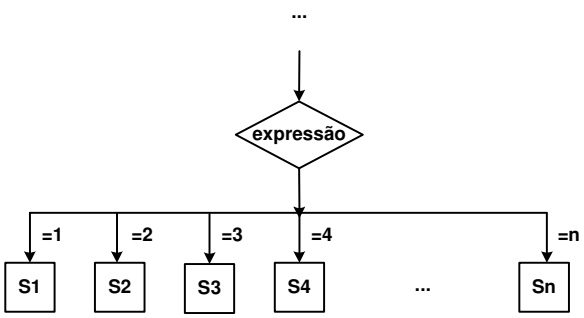

Decisão – Estruturas Encaixadas (Exemplo)

Descrever um algoritmo para ler três valores, e imprimir o maior valor, assumindo que são valores diferentes.

```
ED: 3 variáveis inteiras
INICIO
   LER(A,B,C)
   SE A>B ENTÃO
                  SE A>C   ENTÃO
                                    MAX ← A
                           SENÃO
                                    MAX ← C
                  FIMSE
         SENÃO
                  SE B>C   ENTÃO
                                    MAX ← B
                           SENÃO
                                    MAX ← C
                  FIMSE
   FIMSE
   ESCREVER(MAX)
FIM
```

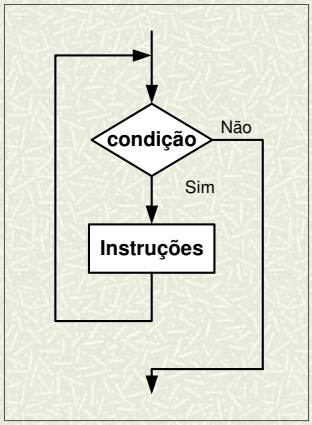
Estrutura de Decisão Múltipla
Caso ...

```
...
Caso expressão
=1: <instruções1>
=2: <instruções2>
=3: <instruções3>
...
=n: <instruçõesn>
FimCaso
```



Repetição - enquanto (...) fazer ...

Enquanto «condição» fazer
«Instruções»
FimEnquanto



Repetição Enquanto (...) fazer ...
Exemplo

Exemplo

Ler uma sequência de números terminada por zero, e mostrá-la.

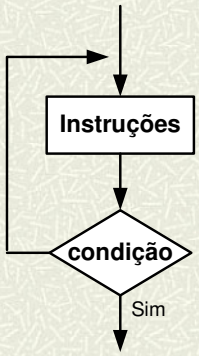
```
ED: 1 variável real
INICIO
   LER(NUM)
   ENQUANTO NUM <> 0 FAZER
                ESCREVER(NUM)
                LER(NUM)
   FIMENQUANTO
FIM
```


Repetição Repetir ... até (...)

Repetir

«Instruções»

Enquanto «condição»



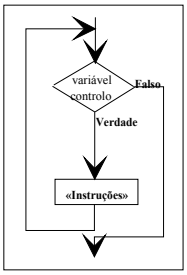
Repetição Repetir ... até (...)

Exemplo

- Ler uma sequência de números terminada por zero, e mostrá-la.

```
ED: 1 variável real
INICIO
   REPETIR
        LER(NUM)
        ESCREVER(NUM)
   ATÉ NUM = 0
FIM
```


Repetição para (...) fazer

Linguagem formal	Representação Gráfica
PARA «variável de controlo »= «valor1» ATÉ «valor 2» FAZER «Instruções» FIMPARA	

Repetição para (...) fazer
Exemplo

Ler uma sequência de 10 números reais, e mostrá-la.

```
ED: 1 variável real, 1 variável inteira
INICIO
   PARA I ← 1 ATÉ 10  FAZER
              LER(NUM)
              ESCREVER(NUM)
   FIMPARA
FIM
```


Repetição para (...) fazer
Exemplo

Desenvolver um programa que permita calcular o valor da potência de base x e expoente inteiro y - x^y.

```
Inicio
  Ler (x,y)
  res ← 1
  Para a←1 até y incrementando 1 fazer
        res ← res * x
  Fim Para
  Escrever (res)
Fim
```

Traçagem

- A traçagem consiste em testar um algoritmo para determinados valores de entrada, observando o seu comportamento. Todos os passos do algoritmo devem ser numerados: P1, P2 ...
- A forma mais simples é construir uma tabela, em que os títulos das colunas são constituídas por todas as operações efectuadas pelo algoritmo, desde atribuições, condições, repetições.
- Os títulos das linhas correspondem aos passos P1, P2, etc. O algoritmo pode a partir de agora ser executado manualmente. A ideia é determinar se a sua lógica é válida e corresponde aos objectivos que se pretendem atingir.