Cap. 1.3 - Algoritmia

Aprenda a Programar com C# 2018 — Edições Sílabo
http://www.silabo.pt/livros.asp?num=606

António Trigo, Jorge Henriques
{antonio.trigo,jmvhenriques}@gmail.com

17 de Abril de 2018

Conceito de variável, constante e expressão

- Constante
  - Objeto do algoritmo/programa que representa um valor imutável, isto é, cujo conteúdo não se altera durante a execução do programa;
  - Pode possuir como conteúdo um qualquer tipo de dado (inteiro, string, etc).
- Variável
  - Objeto situado na memória que representa um valor ou expressão.
  - Pode ver o seu conteúdo alterado durante a execução do programa, podendo armazenar valores diferentes em espaços de tempo diferentes.
- Expressão
  - Obtida compondo variáveis e constantes com operadores aritméticos ou lógicos, por forma a, por exemplo, permitir o cálculo pretendido.

Controlo de fluxo

- Estrutura de controlo é a unidade básica da lógica da programação;
- Em meados da década de 60, alguns matemáticos provaram que qualquer programa podia ser construído através da combinação de três estruturas básicas: sequência, seleção e repetição.

Controlo de fluxo

- Sequência
  - É o processamento de um conjunto de instruções em série, pela ordem pela qual as instruções são especificadas;
- Seleção
  - Ou expressão condicional, é uma estrutura de desvio do fluxo de processamento;
  - Permite ao algoritmo/programa adquirir poder de decisão.
- Repetição
  - É também uma estrutura de desvio de fluxo condicional, mas em que são repetidos um ou mais ações/instruções, dependendo se uma condição é verdadeira ou falsa.

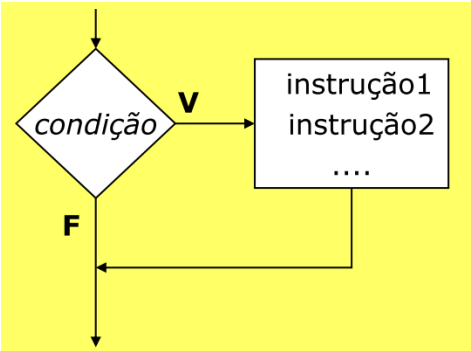
Controlo de fluxo - seleção

- Existem três variantes desta estrutura:
  - Seleção simples
    SE (condição) **ENTÃO** instrução **FIMSE**
  - Seleção em alternativa
    SE (condição) **ENTÃO** instrução **SENÃO** instrução **FIMSE**
  - Seleção múltipla
    CASO variável **SEJA**
    valor1: instrução1
    ...
    valorn: instruçãon
    FIMCASO

Seleção simples

- Permite decidir entre executar ou não um conjunto de instruções. A decisão é tomada em função do resultado da condição que é calculada no início da execução da estrutura;
- Se a condição for verdadeira então executa o conjunto de instruções, caso contrário continua para a próxima instrução a ser executada no algoritmo/programa.

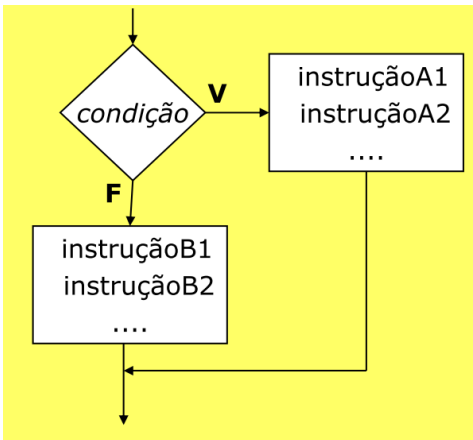
SE (condição) **ENTÃO**
    instrução1;
    instrução2;
    ...
FIMSE



Seleção alternativa

- Muitas vezes torna-se necessário escolher de entre dois conjuntos de instruções qual deles deve ser executado. Esta escolha também será feita em função do resultado de uma condição;
- Se a condição for verdadeira então executa um conjunto de instruções, senão executa o outro conjunto de instruções.

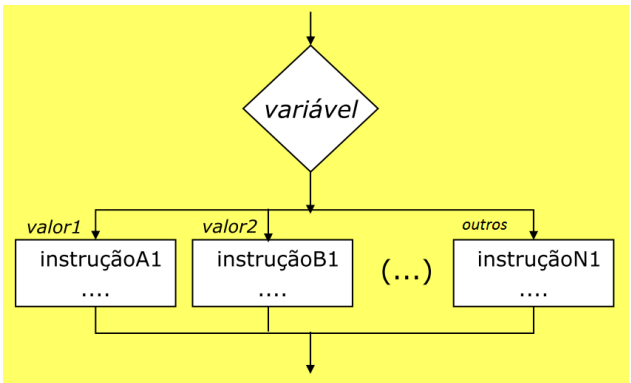
SE (condição) **ENTÃO**
    instruçãoA1;
    instruçãoA2;
    ...
SENÃO
    instruçãoB1;
    instruçãoB2;
    ...
FIMSE



Seleção múltipla

- Executar instruções consoante o critério de decisão;
- Também se pode usar para estes casos várias estruturas SE encadeadas, o que torna o algoritmo/programa difícil de ler.

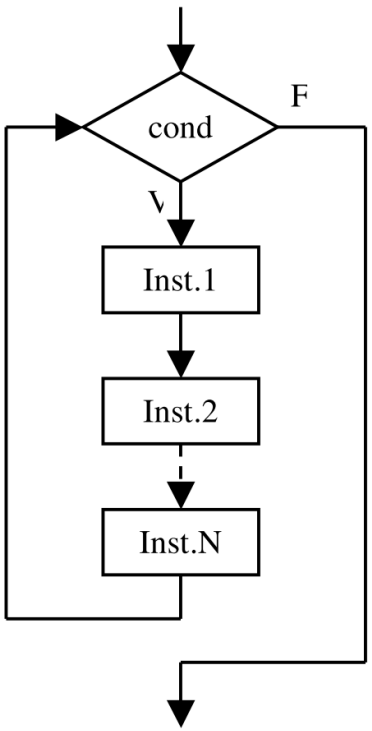
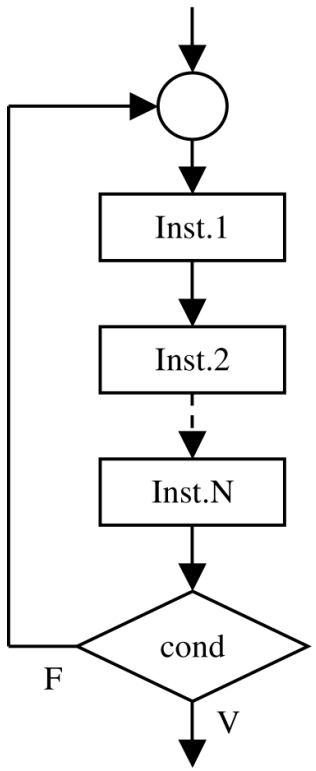
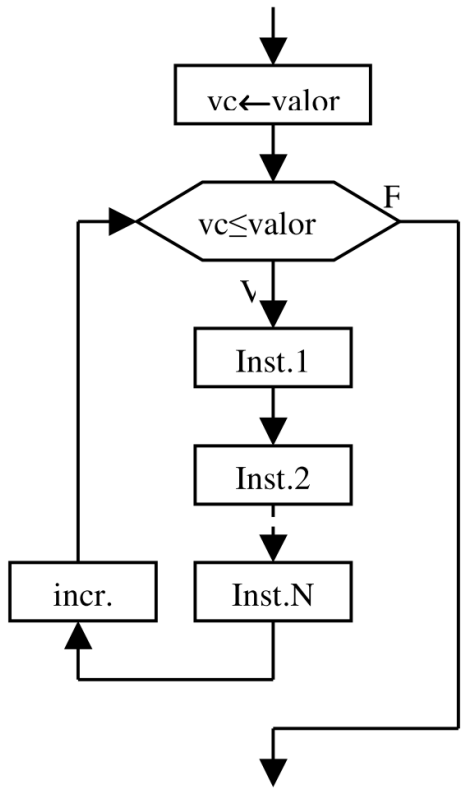
CASO variável **SEJA**
    valor1: instruçãoA1; ...
    valor2: instruçãoB1; ...
    ...
    outros: instruçãoN1; ...
FIMCASO



Controlo de fluxo - repetição

- As estruturas de repetição, também conhecidas por ciclos, permitem repetir um conjunto de ações/instruções de acordo com uma determinada condição.
- Existem três variantes desta estrutura:
  - Enquanto
    ENQUANTO (condição) **FAZER** instrução
    FIMENQUANTO
  - Repetir
    REPETIR instrução **ATÉ** (condição)
  - Para
    PARA var $\leftarrow$ valorinicial **ATÉ** valorfinal **fazer** instrução;
    FIMPARA

Controlo de fluxo - repetição

ENQUANTO [condição] **FAZER** [sequência_de_instruções] **FIMENQUANTO**	**REPETIR** [sequência_de_instruções] **ATÉ** [condição]	**PARA** [variável_de_controlo] ← [valor1] **ATÉ** [valorN] **FAZER** [sequência_de_instruções] **FIMPARA**
		

Introdução

- Uma das grandes dificuldades relacionadas com o desenvolvimento de programas mais elaborados é o controlo da sua complexidade, isto é, como limitar o nível de detalhe a considerar a cada momento;
- Como forma de ultrapassar esta dificuldade é usada, habitualmente, uma técnica que consiste em dividir o problema (complexo) em subproblemas de menor dificuldade;
- Esta abordagem é também designada por abordagem "Top-down design" (do topo para a base), uma vez que se parte do geral (topo) para o detalhe (base).

Módulos

- Cada um dos subproblemas identificados deve ser tratado num módulo diferente, assegurando a clareza do algoritmo e, posteriormente, do programa;
- Para cada módulo deve ser definido qual o seu estado inicial (dados necessários ao seu funcionamento), estado final (resultados que produz) e algoritmo;
- A construção modular de programas, à custa de subprogramas que executam tarefas bem definidas, é uma das estratégias mais profícuas em programação:
  - Facilita a elaboração do(s) algoritmo(s);
  - Aumenta a clareza dos programas;
  - Simplifica futuras modificações aos programas.

Exercício

- A partir de um conjunto de números, fornecidos sequencialmente, o utilizador deverá poder escolher entre as várias opções:
  - Calcular a sua média;
  - Determinar o maior deles;
  - Determinar o menor.