

ALGORITMIA E ESTRUTURAS DE DADOS

Programação nas linguagens C e JAVA

O mais completo e actualizado livro português sobre algoritmos e estruturas de dados, lineares e não lineares, com dezenas de exemplos em pseudocódigo e nas linguagens de programação C e Java. Destinado quer a estudantes quer a profissionais do sector das Tecnologias da Informação.



JOSÉ BRAGA DE VASCONCELOS

JOÃO VIDAL DE CARVALHO

Algoritmia e Estruturas de Dados

Programação nas linguagens C e JAVA



CENTRO **ATLANTICO**.PT

Portugal/2005

Reservados todos os direitos por Centro Atlântico, Lda.

Qualquer reprodução, incluindo fotocópia, só pode ser feita com autorização expressa dos editores da obra.

ALGORITMIA E ESTRUTURAS DE DADOS - PROGRAMAÇÃO NAS LINGUAGENS C E JAVA

Colecção: Tecnologias

Autores: *José Braga de Vasconcelos*

João Vidal de Carvalho

Direcção gráfica: Centro Atlântico

Revisão técnica: Vitor Pereira
(Prof. Auxiliar, Univ. Lusíada)

Capa: Paulo Buchinho

© Centro Atlântico, Lda., 2005

Av. Dr. Carlos Bacelar, 968 – Escr. 1 – A
4764-901 V. N. Famalicão

Rua da Misericórdia, 76 – 1200-273 Lisboa

Portugal

Tel. 808 20 22 21

geral@centroatlantico.pt

www.centroatlantico.pt

Impressão e acabamento: Inova

1ª edição: Setembro de 2005

ISBN: 989-615-012-5

Depósito legal: /05

Marcas registadas: Todos os termos mencionados neste livro conhecidos como sendo marcas registadas de produtos e serviços foram apropriadamente capitalizados. A utilização de um termo neste livro não deve ser encarada como afectando a validade de alguma marca registada de produto ou serviço.

O Editor e os Autores não se responsabilizam por possíveis danos morais ou físicos causados pelas instruções contidas no livro nem por endereços Internet que não correspondam às *Home-Pages* pretendidas.

Aos nossos Pais

Índice

ÍNDICE DE FIGURAS	13
ÍNDICE DE TABELAS	17
1. ALGORITMIA E A MODELAÇÃO DE PROBLEMAS	19
1.1 A noção formal de algoritmo	20
1.2 A Algoritmia como Ciência da Computação	21
1.3 Algoritmos, computadores e programação	23
1.4 Características de um Algoritmo.....	24
1.5 Modelação algorítmica e a resolução de problemas	26
1.5.1 A Algoritmia e a Engenharia de Software.....	28
1.5.2 Aproximação descendente	29
1.6 Componentes de um algoritmo.....	32
1.7 Linguagens de representação algorítmica.....	34
1.7.1 Linguagem natural	35
1.7.2 Pseudocódigo	36
1.7.3 Notação gráfica.....	37
1.8 Métodos de concepção algorítmica	37
1.8.1 Método iterativo	38
1.8.2 Método recursivo	39
1.8.3 Iteração vs recursão	40
1.9 Análise da complexidade algorítmica	42
1.9.1 Objectivos da análise algorítmica.....	42
1.9.2 Métodos de avaliação algorítmica	43
1.9.3 Avaliação da complexidade de um algoritmo	44

2. TIPOS E ESTRUTURAS DE DADOS.....	47
2.1 Estruturas de Dados.....	48
2.2 Tipos de dados e estruturas de dados	49
2.2.1 Tipo de dados booleano	50
2.2.2 Tipo de dados numérico	50
2.2.3 Tipo de dados alfanumérico	50
2.2.4 Ponteiros.....	54
2.3 Representação dos dados em memória	55
2.3.1 Representação Simbólica.....	55
2.3.2 Representação Numérica	56
2.3.3 Informação Contextual.....	56
2.4 Estruturas de dados complexas	56
2.4.1 Vectores.....	57
2.4.2 Matrizes	58
2.5 Conjuntos e Dicionários	59
2.6 Tipos de Dados Abstractos	59
3. REPRESENTAÇÃO E NOTAÇÃO ALGORÍTMICA	65
3.1 Pseudocódigo e programação estruturada	66
3.1.1 Desenvolvimento de um algoritmo	66
3.2 Notação algorítmica	68
3.2.1 Identificadores	69
3.2.2 Variáveis	69
3.2.3 Constantes.....	70
3.2.4 Instrução de atribuição	70
3.2.5 Leitura e escrita de dados	71
3.2.6 Operações e Expressões Aritméticas.....	72
3.2.7 Operadores e operações relacionais.....	73
3.2.8 Operadores e operações lógicas.....	74
3.2.9 Finalização do Algoritmo	74
3.3 Estruturas Lógicas de um Algoritmo	75
3.3.1 Estrutura sequencial	75
3.3.2 Estrutura condicional	76
3.3.3 Estruturas de repetição.....	78
3.4 Modularização - Definição de subalgoritmos	81
3.4.1 Funções.....	83

3.4.2 Procedimentos.....	84
3.4.3 Parâmetros de entrada e saída	85
3.5 Notação Gráfica (Fluxogramas).....	85
3.5.1 Fluxogramas vs. Estruturas de controlo	87
3.6 Prova e Teste de Algoritmos	91
3.7 Algoritmos propostos	94
4. ORDENAÇÃO E PESQUISA.....	107
4.1 Ordenação	108
4.1.1 Ordenação <i>Bubble sort</i>	109
4.1.2 Ordenação <i>Quick Sort</i>	114
4.1.3 Ordenação <i>Selection Sort</i>	118
4.1.4 Ordenação <i>insertion Sort</i>	121
4.1.5 Análise comparativa de desempenho.....	123
4.2 Pesquisa	126
4.2.1 Pesquisa Sequencial (linear)	127
4.2.2 Pesquisa Binária.....	130
4.2.3 Análise comparativa de desempenho.....	133
4.3 Algoritmos propostos	134
5. PILHAS E FILAS	139
5.1 Pilhas	140
5.1.1 Campos de aplicação das pilhas.....	141
5.1.2 Implementação da pilha com vector	142
5.1.3 Operações básicas das pilhas.....	142
5.2 Filas	147
5.2.1 Campos de aplicação das filas	148
5.2.2 Implementação de fila com vector.....	149
5.2.3 Operações básicas das filas.....	150
5.2.4 Fila circular	154
5.2.5 Fila dupla	158
5.3 Algoritmos propostos	161

6. REPRESENTAÇÃO ENCADEADA DE ESTRUTURAS DE DADOS LINEARES	169
6.1 Estruturas de dados encadeadas.....	170
6.1.1 Tipos de listas	171
6.1.2 Variáveis ponteiro	172
6.1.3 Pilha de nodos disponíveis	172
6.2 Listas lineares simplesmente encadeadas.....	174
6.3 Listas lineares circulares simplesmente encadeadas	181
6.4 Listas lineares duplamente encadeadas	187
6.5 Listas lineares duplamente encadeadas circulares.....	193
6.6 Algoritmos propostos.....	196
7. ESTRUTURAS DE DADOS NÃO LINEARES	199
7.1 Árvores e Grafos	200
7.1.1 Definição recursiva de árvore	200
7.1.2 Notação, terminologia e propriedades	201
7.1.3 Formas de representação de árvores M-árias.....	203
7.2 Árvores Binárias	205
7.2.1 Definição e características.....	205
7.2.2 Conversão de árvores m-árias em árvores binárias	206
7.2.3 Representação encadeada de árvores binárias	208
7.2.4 Travessia de árvores binárias.....	209
7.2.5 travessia com Algoritmos iterativos	210
7.2.6 travessia com Algoritmos recursivos	215
7.2.7 Inserção ordenada.....	217
7.2.8 Árvores binárias de pesquisa	221
7.3 Grafos.....	229
7.3.1 Tipos e características de um grafo.....	229
7.3.2 Representação de grafos	232
7.3.3 Algoritmos para travessia de um grafo	235
7.4 Algoritmos propostos.....	242

8. ALGORITMIA E PROGRAMAÇÃO	247
8.1 Estruturas de dados, algoritmos e programação.....	248
8.2 Linguagem de Programação C.....	249
8.2.1 Características da Linguagem C	249
8.2.2 Estrutura de um programa em C	250
8.2.3 Tipos primitivos de dados.....	251
8.2.4 Constantes.....	252
8.2.5 Variáveis.....	253
8.2.6 Expressões.....	255
8.2.7 Comandos básicos	256
8.2.8 Estruturas de controlo.....	259
8.2.9 Matrizes (<i>arrays</i>).....	262
8.2.10 Cadeias de caracteres (<i>strings</i>).....	263
8.2.11 Ponteiros e endereços.....	264
8.2.12 Subalgoritmos.....	265
8.2.13 Estruturas	266
8.2.14 Algoritmos de ordenação em C	270
8.2.15 Algoritmos de pesquisa em C.....	275
8.2.16 Algoritmos de manipulação de pilhas em C	278
8.2.17 Algoritmos de manipulação de filas em C	281
8.2.18 Algoritmos de manipulação de listas em C.....	285
8.2.19 outros Algoritmos.....	290
8.3 Linguagem de Programação JAVA	297
8.3.1 <i>JAVA Virtual MACHINE</i>	297
8.3.2 JAVA: Linguagem orientada aos objectos.....	298
8.3.3 Estrutura de um programa JAVA.....	300
8.3.4 Exemplos de codificação e implementação de classes JAVA ..	304
8.3.5 Especificação de tipos de dados abstractos.....	320
BIBLIOGRAFIA	327

Índice de Figuras

Figura 1.1: Algoritmos, estruturas de dados e programas.....	24
Figura 1.2: Problema, algoritmo e representação.....	25
Figura 1.3: Problema, algoritmo e representação.....	27
Figura 1.4: Abordagem para a resolução de problemas.....	30
Figura 1.5: Exemplo explicativo do método recursivo	41
Figura 2.1: Estruturas de dados.....	48
Figura 2.2: Estrutura de dados sob a forma de vector.....	56
Figura 2.3: Representação de um vector.....	57
Figura 2.4: Representação e manipulação de matrizes	58
Figura 2.5: Tipo de Dados Abstracto (ADT).....	61
Figura 3.1: Representação em fluxograma da estrutura sequencial	87
Figura 3.2 Representação em fluxograma das estruturas if .. then...else e if..then.....	88
Figura 3.3: Representação em fluxograma da estrutura de controlo Do For	89
Figura 3.4: Representação em fluxograma da estrutura de controlo Do While.....	90
Figura 3.5: Representação em fluxograma da estrutura de controlo Repeat ... Until.....	91
Figura 3.6: Representação parcial em fluxograma do algoritmo Lista_Notas	94
Figura 3.7: Representação em fluxograma do algoritmo Lista_Notas (continuação).....	95
Figura 5.1: Analogia de uma pilha com tubo de bolas de ténis	140
Figura 5.2: Funcionamento da estrutura Pilha adoptado no tubo de bolas de ténis	141
Figura 5.3: Implementação da pilha com vector	142
Figura 5.4: Funcionamento da pilha para as operações de inserção e eliminação.....	143
Figura 5.5: Analogia de uma fila com uma praça de táxis.....	148
Figura 5.6: Funcionamento da estrutura Fila adoptado numa praça de táxis	148
Figura 5.7: Implementação da fila com vector	149
Figura 5.8: Funcionamento conceptual de uma fila para as operações de inserção e eliminação.....	150
Figura 5.9: Fila original com 6 elementos inseridos.....	155
Figura 5.10: Fila após a eliminação de dois elementos.....	155
Figura 5.11: Fila com os princípios de funcionamento de uma fila circular	155
Figura 5.12: Fila com configuração circular	156

Figura 5.13: Func. conceptual de uma fila dupla para as operações de inserção e eliminação	159
Figura 6.1: Estrutura de um Nodo de uma Lista	170
Figura 6.2: Exemplo de uma lista linear simplesmente encadeada.....	171
Figura 6.3: Estrutura de um nodo de uma lista linear duplamente encadeada.....	172
Figura 6.4: Processo de obtenção de um nodo da pilha de disponíveis.....	173
Figura 6.5: Processo de eliminação de um nodo e sua devolução à pilha de disponíveis.....	174
Figura 6.6: Inserção de um novo nodo no início de uma LLSE.	175
Figura 6.7: Inserção de um novo nodo no fim de uma LLSE.....	176
Figura 6.8: Inserção ordenada de um nodo.	177
Figura 6.9: Eliminação de um nodo de uma lista e sua restituição à pilha dos disponíveis	180
Figura 6.10: Representação gráfica de uma lista linear circular simplesmente encadeada.....	181
Figura 6.11: Lista circular vazia e com um nodo.....	182
Figura 6.12: Repres. gráfica de uma lista linear simplesmente encadeada circular encabeçada..	182
Figura 6.13: Representação gráfica de uma lista circular encabeçada vazia.....	183
Figura 6.14: Inserção de um novo nodo no início de uma LLSE circular encabeçada.....	184
Figura 6.15: Inserção de um novo nodo no fim de uma LLSE circular encabeçada.....	185
Figura 6.16: Inserção ordenada numa LLSE circular encabeçada.	186
Figura 6.17: Estrutura dos nodos de uma lista duplamente encadeada.....	187
Figura 6.18: Exemplo de uma lista linear duplamente encadeada	188
Figura 6.19: Pilha de disponíveis para listas lineares duplamente encadeadas.....	188
Figura 6.20: inserção de um nodo à esquerda de um nodo especificado	189
Figura 6.21: Eliminação de um nodo de uma lista linear duplamente encadeada.....	191
Figura 6.22: Representação gráfica de uma lista duplamente encadeada circular	193
Figura 6.23: Representação gráfica de uma lista duplamente encadeada circular encabeçada ...	194
Figura 6.24: Lista vazia	194
Figura 6.25: Inserção de um nodo no início da lista.....	194
Figura 7.1: Exemplo de uma árvore	200
Figura 7.2: Exemplo de árvore m-ária e respectivo vocabulário	202
Figura 7.3: Altura de uma árvore m-ária.....	202
Figura 7.4: Árvore m-ária ordenada	203
Figura 7.5: Representação em grafo.....	203
Figura 7.6: Representação através de um diagrama de Venn.....	204
Figura 7.7: Representação através de parêntesis imbricados.....	204
Figura 7.8: Representação em árvore de uma expressão matemática	204
Figura 7.9: Exemplo de uma árvore binária	205

Figura 7.10: Exemplo de uma árvore binária completa	206
Figura 7.11: Possíveis arranjos de uma árvore binária	206
Figura 7.12: Conversão de uma árvore m-ária numa árvore binária	207
Figura 7.13: Exemplo de uma árvore binária - Representação encadeada	208
Figura 7.14: Travessias de uma árvore binária	210
Figura 7.15: Inserção ordenada para os valores P, L, N, X, A, F, H, B e T	218
Figura 7.16: Diferentes passos desde o nodo raiz	219
Figura 7.17: Eliminação de um nodo de grau 0	224
Figura 7.18: Eliminação de um nodo de grau 1	224
Figura 7.19: Eliminação de um nodo com grau 2	225
Figura 7.20: Estrutura alternativa para a representação de um nodo de uma árvore binária	226
Figura 7.21: Exemplo de uma árvore binária com a estrutura alternativa de um nodo	226
Figura 7.22: Grafo orientado	230
Figura 7.23: Grafo não orientado	230
Figura 7.24: Grafo misto	230
Figura 7.25: Grafos completos	231
Figura 7.26: Grafo conexo	231
Figura 7.27: Grafo não conexo	231
Figura 7.28: Representação matricial de um grafo orientado	232
Figura 7.29: Representação encadeada de um grafo orientado	233
Figura 7.30: Representação encadeada de um grafo não orientado	233
Figura 7.31: Exemplo de um grafo com 9 nodos	237
Figura 7.32: Representação encadeada do grafo anterior	237
Figura 8.1: Compilação e execução do programa bubble_sort p/ a lista de valores 3,6,5,1,7,9	271
Figura 8.2: Compilação e execução do programa bubble_mod p/ a lista de valores 3,6,5,1,7,9	272
Figura 8.3: Compilação e execução do programa bubble_rec p/ a lista de valores 3,6,5,1,7,9	272
Figura 8.4: Compilação e execução do programa quick_sort p/ a lista de valores 3,6,5,1,7,9	273
Figura 8.5: Compilação e execução do programa selection_sort p/ a lista de valores 3,6,5,1,7,9	274
Figura 8.6: Compilação e execução do programa pesq_linear para a pesquisa de um elemento existente na lista	276
Figura 8.7: Compilação e execução do programa pesq_linear para a pesquisa de um elemento inexistente na lista	276
Figura 8.8: Compilação e execução do programa pesq_linear_ord para a pesquisa de um elemento existente na lista	276
Figura 8.9: Compilação e execução do programa pesq_linear_ord para a pesquisa de um elemento inexistente na lista	277

Figura 8.10: Compilação e execução do programa pesq_binaria para a pesquisa de um elemento 5 existente na lista na posição 2	278
Figura 8.11: Compilação e execução do programa pesq_binaria para a pesquisa de um elemento inexistente	278
Figura 8.12: Compilação e execução do programa pilha para a inserção dos elementos 2,3,4,5 e eliminação dos elementos 3,4,5.....	281
Figura 8.13: Compilação e execução do programa fila para a inserção dos elementos 2,3,4 e eliminação dos elementos 2,3.....	284
Figura 8.14: Compilação e execução do programa lista_enc para a consulta do último elemento de uma lista vazia.....	289
Figura 8.15: Compilação e execução do programa lista_enc para a consulta do último elemento de uma lista composta pelos elementos 4,7,3,8,9,2.....	289
Figura 8.16: Compilação e execução do programa factorial para o número 6	290
Figura 8.17: Compilação e execução do programa maiúsculas para dois textos.....	291
Figura 8.18: Compilação e execução do programa capicua para números entre 10 e 500	292
Figura 8.19: Compilação e execução do programa primos para 4 números diferentes	293
Figura 8.20: Compilação e execução do programa tabela Fahrenheit-celcius	294
Figura 8.21: Compilação e execução do programa totoloto para a geração aleatória de 8 números.....	296
Figura 8.22: Compilação e execução do programa Inversor	296
Figura 8.23: Ambiente de desenvolvimento JAVA	298
Figura 8.24: Compilação e execução da classe (programa) Factorial.....	301
Figura 8.25: Parte da hierarquia de classes Java	304
Figura 8.26: Package Formas que agrupa um conjunto de ficheiros Java	304
Figura 8.27: Compilação e execução da classe DetArea	306
Figura 8.28: Compilação e execução da classe (programa) sequência de Fibonacci.....	309
Figura 8.29: Compilação e execução da classe (programa) Primo	311
Figura 8.30: Compilação e execução da classe (programa) de ordenação por selecção	314
Figura 8.31: Compilação e execução da classe ordenação por selecção (geração aleatória).....	315
Figura 8.32: Compilação e execução da classe ordenação quicksort	317
Figura 8.33: Compilação e execução da classe (programa) Pesquisa Binária.....	319
Figura 8.34: Execução da classe (programa) Verifica expressão.....	324

Índice de tabelas

Tabela 1.1: Enquadramento da algoritmia com outras disciplinas das ciências da computação.....	22
Tabela 1.2: Passos genéricos de uma aproximação descendente (top-down).....	30
Tabela 1.3: Especificação de uma tarefa numa aproximação descendente	31
Tabela 1.4: Passos de uma aproximação descendente	31
Tabela 1.5: Algoritmo (top-down) para o cálculo do máximo divisor comum	32
Tabela 2.1: Exemplos de dados alfanuméricos com os respectivos comprimentos	51
Tabela 3.1: Regras para definição dos identificadores	69
Tabela 3.2: Descrição dos operadores lógicos usados na construção de expressões	74
Tabela 3.3: Principais símbolos utilizados na construção de fluxogramas.....	86
Tabela 4.1: Tamanho do vector em cada repetição em que a parte considerada na pesquisa é dividida a metade.....	134
Tabela 5.1: Repercussões das diferentes operações numa pilha que armazena valores alfanuméricos (letras)	144
Tabela 5.2: Repercussões das diferentes operações numa fila que armazena valores alfanuméricos (letras)	151
Tabela 8.1: Tipos de dados e a respectiva gama de valores	252
Tabela 8.2: Operadores Aritméticos, Relacionais e Lógicos usados em C.....	255
Tabela 8.3: Abreviação das expressões aritméticas em C.....	256
Tabela 8.4: Especificador de formato e respectivo tipo de argumento para a entrada de dados ...	256
Tabela 8.5: Especificador e respectivo tipo de argumento para a saída de dados.....	258
Tabela 8.6: Associatividade do operador de atribuição	259
Tabela 8.7: Alguns exemplos de Packages Java standard	302

1. Algoritmia e a Modelação de Problemas

Este capítulo tem por objectivo efectuar um enquadramento da disciplina de algoritmia no contexto das ciências da computação. Mais concretamente, visa apresentar as principais características de um algoritmo, assim como a definição e modelação de problemas do mundo real e respectivas representações algorítmicas. Pretende-se também apresentar as diferentes notações algorítmicas existentes e efectuar uma transição genérica para as linguagens de programação.

1.1 A noção formal de algoritmo

A noção de algoritmo surgiu pela primeira vez através de um matemático árabe¹ que associou este termo à capacidade de um computador executar procedimentos para a resolução de problemas. Na área das ciências da computação, a palavra ‘algoritmo’ refere-se a um procedimento que pode ser implementado e executado por um programa de computador.

Um algoritmo representa uma sequência finita e não ambígua de instruções elementares bem definidas, conducente à solução de um determinado problema, cada uma das quais pode ser executada mecanicamente numa quantidade finita de tempo. Neste sentido, um algoritmo é um conjunto de instruções que podem ser mecanicamente executadas num período de tempo, de modo a resolver um determinado problema.

Um programa de computador envolve a definição de um algoritmo para a resolução de um problema. Um algoritmo é representado através de expressões simbólicas que utilizam estruturas de dados de modo a descrever e a encontrar a solução de problemas do mundo real. As estruturas de dados representam de modo simbólico entidades e objectos do mundo real e definem a parte estática de um algoritmo. A manipulação das estruturas de dados através de declarações e instruções precisas de controlo definem a parte dinâmica de um algoritmo. Este conjunto de estruturas de dados e de controlo constituem formalmente um algoritmo para a resolução de problemas.

Um algoritmo é constituído por um conjunto de expressões simbólicas que representam acções (escolher, atribuir, etc.), testes de condições (estruturas condicionais) e estruturas de controlo (ciclos na estrutura sequencial do algoritmo) de modo a especificar o problema e respectiva solução.

¹ al Khawarizmi

1.2 A Algoritmia como Ciência da Computação

A algoritmia é uma ciência da computação que estuda e investiga a sintaxe e a semântica de expressões e instruções simbólicas que, em conjunto com estruturas de dados que representam entidades do mundo real, permitem a resolução de problemas associados a diferentes domínios do nosso conhecimento.

A base da algoritmia assenta na lógica matemática e na álgebra linear. Esta ciência estuda o algoritmo como um processo discreto (sequência de acções indivisíveis) e determinístico (para cada passo da sequência e para cada conjunto válido de dados, corresponde uma e uma só acção) que termina quaisquer que sejam os dados iniciais (pertencentes a conjuntos predefinidos).

A ciência da computação pode ser interpretada como o conjunto de teorias, regras, práticas, métodos e ferramentas que permitem a análise, a representação e a implementação de processos sistemáticos que descrevem e transformam dados em informação. A ciência da computação envolve um conjunto significativo de disciplinas (tabela 1.1) que estudam e utilizam algoritmos e respectivas estruturas de dados. Um programa de computador é uma implementação (codificação) de um conjunto de algoritmos e respectivas estruturas de dados através de uma determinada linguagem de programação.

A algoritmia é uma disciplina basilar na área das ciências da computação, e o seu entendimento poderá ser muito relevante no desenvolvimento de competências noutras áreas do conhecimento.

Um programa de computador não existe sem um algoritmo associado. As aplicações de software são cada vez mais importantes em todas as áreas de conhecimento. Neste contexto, e tendo em conta a interdisciplinaridade do conhecimento actual, o estudo e desenvolvimento de algoritmos tem vindo a ultrapassar a dimensão inicial das ciências da computação, dado que a disciplina algorítmica tem sido aplicada noutras áreas do conhecimento.

Disciplina	Áreas de aplicação (algoritmia)
Análise de Sistemas	Análise e desenvolvimento de algoritmos para a representação de modelos de sistemas de informação organizacionais.
Sistemas de Informação	Utilização de algoritmos para o estudo e aplicação de notações formais de sistemas de informação.
Engenharia de Software	Análise, desenvolvimento e verificação de algoritmos em diferentes fases do ciclo de vida do software.
Linguagens de Programação	Codificação de algoritmos numa linguagem de programação (Basic, Fortran, Pascal, C, C++, C#, JAVA, etc.).
Inteligência Artificial	Especificação formal de software para a definição de modelos de representação de conhecimento.

Tabela 1.1: Enquadramento da algoritmia com outras disciplinas das ciências da computação

Muitos outros exemplos poderiam ser apresentados para fundamentar a importância da algoritmia para o desenvolvimento da ciência, tanto na área da ciência da computação como noutras áreas técnicas, humanas e sociais.

A noção de algoritmo não é exclusiva da ciência da computação e da programação de computadores. Contudo, a engenharia de software e a programação são actualmente o principal campo de aplicação de algoritmos. Um profissional bem preparado na área da ciência da computação sabe necessariamente como lidar com algoritmos, nomeadamente ao nível da sua análise, compreensão, construção e manipulação. Este conhecimento é essencial ao desenvolvimento de programas eficazes (que resolvem o problema) e eficientes (com um bom desempenho) para a resolução de problemas dos mais diversos domínios de conhecimento.

1.3 Algoritmos, computadores e programação

Os computadores são máquinas electrónico-digitais que simplesmente executam algoritmos tendo em conta os dados de entrada a serem processados. Para alguns cientistas, o algoritmo é o conceito basilar da ciência da computação. Os computadores executam algoritmos e manipulam dados. A execução de um algoritmo é também designada por processamento de dados e consiste em três partes ou fases de execução: entrada de dados, o processo (ou processamento dos dados) e uma saída de dados (ou resultados do processamento). Estas fases são amplamente discutidas na literatura: a entrada é composta por um conjunto de dados requisitados e necessários à execução das instruções do algoritmo, o processo (ou processamento) é composto por uma sequência finita de instruções que definem simbolicamente o algoritmo e a saída é interpretada como o resultado obtido com a execução do algoritmo para a entrada fornecida.

A implementação de um algoritmo para a resolução de um problema é designada por programação de computadores. Esta tarefa tem subjacente a inserção de um conjunto de instruções no computador através da conversão das instruções algorítmicas numa determinada linguagem de programação. O termo programa computacional refere-se à representação de um algoritmo numa linguagem de programação. Em última instância, não existe uma distinção efectiva entre um algoritmo e um programa no que concerne a linguagem em que foram especificados. Normalmente, a linguagem algorítmica (por exemplo, o pseudocódigo que se utiliza neste livro) é uma linguagem de alto nível independente da plataforma de implementação da aplicação em análise. Uma linguagem de programação introduz construtores adicionais que permitem facilitar o processo de especificação, desenvolvimento e manutenção da aplicação de software em curso.

A linguagem do computador, designada por linguagem máquina, é a única linguagem que o computador realmente conhece. A linguagem máquina é uma linguagem próxima (interpretável) do processador da máquina (ou computador) composto por sequências de código binário (0 e 1). No entanto, foram desenvolvidas as linguagens de programação (C, C++, C#, Java, etc.) independentes da máquina (e mais próximas da linguagem natural) de modo a facilitar o processo de desenvolvimento de software.

1.4 Características de um Algoritmo

Um algoritmo deve ser caracterizado por um conjunto de adjectivos e substantivos universalmente consensuais que constam em várias definições de algoritmo:

Um algoritmo representa uma sequência *finita e não ambígua* de instruções de modo a obter a resolução do problema sob a forma de resultado (*saída de dados*) tendo por base uma *entrada* prévia de dados. Um algoritmo é representado através de uma linguagem com uma determinada *sintaxe* e *semântica* associada. Por fim, um algoritmo deve ser *eficaz* na resolução do problema subjacente assim como *eficiente* de modo a resolver o problema com o melhor desempenho (performance) possível.

A referência a instruções implica o facto de existir algo (ex.: computador) capaz de compreender e posteriormente executar as referidas instruções. Há algumas décadas atrás, um livro essencial nesta área [Aho et al. 1974], apresentou que a base estrutural das ciências da computação e da programação são os algoritmos e as estruturas de dados. Daqui também resultou a relação próxima e actual (figura 1.1) dos algoritmos com os programas de software.

Algoritmos + **Estruturas de Dados** = **Programas de software**

Figura 1.1: Algoritmos, estruturas de dados e programas

É consensual definir um algoritmo como uma sequência de passos e instruções com o objectivo de resolver um determinado problema. Esta definição tem associada um conjunto de propriedades que devem ser consideradas durante o processo de concepção algorítmica:

Entrada de dados

Leitura de dados de entrada representativos de valores, quantidades, ou atributos inicialmente especificados (por exemplo, através de instruções de leitura). Uma entrada de dados define uma instância do problema para a qual o algoritmo deverá dar uma resposta (saída de resultados). É essencial definir com precisão o âmbito (ou espaço de informação) de entrada de dados para um determinado algoritmo.

Este livro foi preparado e concebido para ser utilizado como objecto de estudo em diferentes áreas de conhecimento, dado que apresenta a disciplina de algoritmia como requisito fundamental à resolução de problemas do mundo real. Contudo, tem especial interesse para docentes, alunos e formandos nas áreas das ciências da computação, nomeadamente sistemas de informação, matemáticas aplicadas e cursos de engenharia.

Este livro procura dotar futuros profissionais com as componentes de raciocínio e abstracção necessárias à resolução de problemas. Explora a análise e as técnicas essenciais à concepção de algoritmos e à especificação formal de estruturas de dados lineares e não lineares. Neste sentido, os autores introduzem uma componente prática baseada na apresentação de um conjunto alargado de algoritmos resolvidos. A implementação dos algoritmos é efectuada através das linguagens de programação C e Java, também apresentadas neste livro.

José Braga de Vasconcelos é doutorado em Ciências da Computação pela Universidade de York (UK). Após o doutoramento produziu e apresentou diversas publicações em conferências e revistas científicas na área de Gestão de Conhecimento Organizacional. Actualmente é Professor Associado na Faculdade de Ciência e Tecnologia da Universidade Fernando Pessoa, e membro fundador do Grupo de Investigação e Desenvolvimento (I&D) em Informática Médica (GIMED). Prestou também consultoria em Sistemas de Informação e Engenharia de Software em empresas. Ultimamente, além da sua área de investigação em Gestão e Modelação de Competências Organizacionais, tem vindo a exercer I&D na área de Sistemas de Informação para a Saúde.

João Vidal de Carvalho é mestre em Informática de Gestão pela Escola de Engenharia da Universidade do Minho. Professor Adjunto no Instituto Politécnico do Porto (ISCAP), onde lecciona desde 1998. Responsável pelas disciplinas da área de Informática para a Gestão. Actualmente desenvolve investigação na área de Sistemas de Informação. É co-autor dos livros Desenho e Implementação de Bases de Dados com Microsoft Access e Microsoft Access 2003 igualmente editados pelo Centro Atlântico.

ISBN 989-615-012-5



9 789896 150129