

# Contents

	vii
<i>Preface</i>	xvii
<i>Foreword</i>	xix
<i>Technical note</i>	xxv
0. Programming to compute	1
<b>Basic Programming in C language</b>	<b>5</b>
1. Numbers and non-numbers	7
1.1 Numeral systems . . . . .	7
1.2 Positional systems . . . . .	8
1.2.1 The binary system . . . . .	10
1.2.2 The hexadecimal system . . . . .	14
1.3 Representation systems . . . . .	16
1.3.1 Representing negative numbers . . . . .	19
1.3.2 Complement representation . . . . .	19
1.3.3 Excess- $N$ representation . . . . .	21
1.3.4 Rational number representation . . . . .	22
1.4 The approximation problem . . . . .	25
1.5 Non-numbers on computers . . . . .	27
1.6 Logical value representation . . . . .	29
1.6.1 Logical Operators . . . . .	30
1.7 Character representation . . . . .	31

x	<i>Scientific Programming: C-Language, algorithms and models in science</i>	
	1.7.1 Character strings . . . . .	32
	1.7.2 The ASCII code . . . . .	32
	1.7.3 UNICODE . . . . .	33
	1.8 Representing other information . . . . .	34
2.	Programming languages	37
2.1	The necessity of a programming language . . . . .	37
2.2	High-level languages and elementary statements . . . . .	43
2.2.1	The assembly language . . . . .	44
2.3	The role of the compiler . . . . .	46
2.3.1	Interpreters and compilers . . . . .	47
2.4	The linker . . . . .	49
2.5	Procedural and object-oriented languages . . . . .	52
2.6	Why C? . . . . .	55
2.7	History and characteristics of the C language . . . . .	57
2.8	C compilers in their environment . . . . .	58
2.8.1	Linux and Windows . . . . .	60
2.8.2	A first example, in Linux: gcc . . . . .	61
2.8.3	Compiling C in Windows . . . . .	64
3.	Basics of C Programs	65
3.1	Starting to program . . . . .	66
3.2	Statements, variables, types . . . . .	67
3.3	Operators . . . . .	71
3.3.1	Arithmetic Operators . . . . .	72
3.3.2	Logical operators . . . . .	77
3.3.3	Other operators . . . . .	78
3.4	Input/Output for beginners . . . . .	80
3.5	Preprocessor directives . . . . .	84
3.6	Notes on library functions . . . . .	87
3.7	First applications . . . . .	89
4.	Logic management	93
4.1	Flow control . . . . .	94
4.1.1	Nonlinear flows . . . . .	96
4.2	Taking a decision . . . . .	97
4.2.1	if/else . . . . .	99
4.2.2	The selection operator . . . . .	102

*Contents*

xi

4.3	Iterations . . . . .	102
4.3.1	The factorial . . . . .	108
4.3.2	Solving equations . . . . .	109
4.3.3	Searching prime numbers . . . . .	113
4.4	Deprecated statements . . . . .	114
4.5	A rounding problem . . . . .	116
5.	Fundamental data structures	123
5.1	One-dimensional arrays . . . . .	124
5.2	Algorithms with arrays . . . . .	126
5.2.1	Sorting: Bubblesort . . . . .	128
5.2.2	Binary search . . . . .	130
5.3	Multidimensional arrays . . . . .	132
5.4	Solving systems of linear equations . . . . .	133
5.5	Generating random numbers . . . . .	137
5.6	Character strings . . . . .	140
5.6.1	C string syntax . . . . .	140
5.6.2	I/O of character strings . . . . .	141
5.6.3	Multidimensional strings and arrays . . . . .	143
6.	Pointers	149
6.1	Pointers and pointed variables . . . . .	149
6.2	Arrays and pointers in C . . . . .	153
6.2.1	The const qualifier . . . . .	156
6.2.2	String pointers . . . . .	158
6.3	Pointer arithmetic . . . . .	159
6.4	Efficiency issues . . . . .	162
6.5	Multidimensional arrays and pointers . . . . .	163
6.5.1	Pointer Arrays . . . . .	165
6.6	Pointers to pointers . . . . .	167
6.7	Input/Output with files . . . . .	168
6.7.1	Binary files . . . . .	174
7.	Functions	177
7.1	Declaration and definition . . . . .	177
7.1.1	Scope . . . . .	182
7.2	Formal parameters . . . . .	184
7.3	Pointers and array parameters . . . . .	189

xii *Scientific Programming: C-Language, algorithms and models in science*

7.3.1	Array in input and output . . . . .	190
7.3.2	Passing multidimensional arrays . . . . .	194
7.3.3	Global and local variables . . . . .	197
7.4	Applications . . . . .	198
7.4.1	Histograms . . . . .	198
7.4.2	Computing the $\chi^2$ of a distribution . . . . .	203
7.4.3	Programming style and reusability . . . . .	205
7.5	Pointers to functions . . . . .	206
7.6	Functions of functions . . . . .	207
8.	Numerical interpolation and integration . . . . .	211
8.1	Interpolation . . . . .	211
8.1.1	Determining the parameters of a function . . . . .	213
8.1.2	Interpolation with Lagrange polynomials . . . . .	217
8.2	Numerical integration . . . . .	221
8.2.1	The rectangle rule . . . . .	223
8.2.2	The midpoint method . . . . .	225
8.2.3	The trapezoid rule . . . . .	226
8.2.4	Other integration methods . . . . .	229
8.3	The Monte Carlo method . . . . .	234
8.3.1	Multiple integrals . . . . .	236
<b>Advanced programming and simple algorithms</b>		<b>239</b>
9.	Integrating differential equations . . . . .	241
9.1	The harmonic oscillator . . . . .	241
9.2	Integration algorithms . . . . .	245
9.2.1	Euler and Euler-Cromer: convergence and stability . . . . .	247
9.2.2	Other methods: midpoint and leapfrog . . . . .	252
9.2.3	Physical results of a simple integration . . . . .	254
9.3	The struct . . . . .	259
9.4	A complete program as a simple example . . . . .	262
10.	In-depth examination of differential equations . . . . .	271
10.1	More algorithms . . . . .	271
10.1.1	Verlet and self-starting Verlet . . . . .	272
10.1.2	Predictor-corrector and Runge-Kutta . . . . .	274
10.1.3	Stability analysis . . . . .	276

*Contents*

xiii

10.2	system, malloc and typedef . . . . .	279
10.2.1	The interaction with the operating system: the function system . . . . .	279
10.2.2	Dynamic memory allocation: malloc and calloc . . . . .	280
10.2.3	Defining derived types: typedef . . . . .	284
10.3	The simple pendulum . . . . .	286
10.4	Two planets revolving around the sun . . . . .	289
10.5	Software tools: gnuplot . . . . .	292
10.6	A falling body . . . . .	294
10.7	The pendulum and chaos . . . . .	296
11.	(Pseudo)random numbers . . . . .	301
11.1	Generating random sequences . . . . .	302
11.2	Linear congruence . . . . .	304
11.2.1	Choosing the modulo . . . . .	308
11.2.2	Choosing the multiplier . . . . .	309
11.2.3	Purely multiplicative generators . . . . .	311
11.3	Some examples of generators and a first test . . . . .	313
11.4	Statistical tests . . . . .	316
11.4.1	$\chi^2$ test . . . . .	317
11.4.2	The Kolmogorov-Smirnov test . . . . .	322
11.5	Shift registers . . . . .	325
11.6	Generating nonuniform distributions . . . . .	330
12.	Random walks . . . . .	333
12.1	A first simple crucial argument . . . . .	334
12.2	More than one dimension: the generating function . . . . .	341
12.3	A lattice gas . . . . .	353
12.4	Random walks in random environments . . . . .	364
13.	Lists, dictionaries and percolation . . . . .	369
13.1	The unions . . . . .	369
13.1.1	A virtual experiment . . . . .	372
13.2	Linked lists . . . . .	374
13.2.1	Lists, strings and a dictionary . . . . .	375
13.2.2	Recursive functions: computing the factorial . . . . .	382
13.2.3	Binary trees and dictionaries . . . . .	383
13.3	Connected clusters . . . . .	388