

# Contents

<b>1</b>	<b>Introduction</b>	<i>page</i> 1
1.1	Objective	1
1.2	Presentation	1
1.3	Programming languages	2
1.4	Language standards	3
1.5	Chapter summary	4
1.6	How to use this text	4
1.7	Additional and alternative software packages	4
<b>2</b>	<b>Octave programming</b>	5
2.1	Obtaining octave	5
2.2	Command summary	5
2.3	Logistic map	14
<b>3</b>	<b>Installing and running the Dev-C++ programming environment</b>	15
3.1	Compiling and running a first program	15
3.2	The Dev-C++ debugger	17
3.3	Installing DISLIN	19
3.4	A first graphics program	19
3.5	The help system	20
3.6	Example	20
<b>4</b>	<b>Introduction to computer and software architecture</b>	22
4.1	Computational methods	22
4.2	Hardware architecture	23
4.3	Software architecture	24
4.4	The operating system and application software	25
<b>5</b>	<b>Fundamental concepts</b>	26
5.1	Overview of program structure	26
5.2	Tokens, names and keywords	26

5.3	Expressions and statements	27
5.4	Constants, variables and identifiers	27
5.5	Constant and variable types	27
5.6	Block structure	30
5.7	Declarations, definitions and scope	31
5.8	rvalues and lvalues	31
5.9	Operators – precedence and associativity	31
5.10	The <b>const</b> keyword	32
5.11	Formatting conventions	33
5.12	Comments	33
<b>6</b>	<b>Procedural programming basics</b>	<b>35</b>
6.1	Scientific software development	35
6.2	The <b>main( )</b> function	36
6.3	Namespaces	37
6.4	Preprocessor directives and <b>#include</b> statements	38
6.5	Arithmetic and logical operators	39
6.6	The <b>bool</b> and <b>enum</b> types	40
6.7	Control flow, <b>if</b> statements and implicit blocks	40
6.8	The <b>for</b> statement	42
6.9	<b>while</b> and <b>do . . . while</b> statements	42
6.10	The <b>break</b> , <b>continue</b> and <b>exit( )</b> statements	43
6.11	The <b>typedef</b> keyword	44
6.12	Input and output streams	44
6.13	File streams	45
6.14	Casts	45
6.15	Functions	46
6.16	Principles of function operation	46
6.17	Function declarations and prototypes	48
6.18	Enumerators and functions	48
6.19	Overloading and argument conversion	48
6.20	Built-in functions and header files	49
6.21	The <b>assert</b> statement and <b>try</b> and <b>catch</b> blocks	50
6.22	Multiple return statements	51
6.23	Default parameters	52
6.24	Functions and global variables	52
6.25	Inline functions	53
6.26	Recursive functions	53
6.27	Modular programming	54

6.28	Arrays	54
6.29	Program errors	55
6.30	Numerical errors with floating-point types	56
<b>7</b>	<b>An introduction to object-oriented analysis</b>	<b>58</b>
7.1	Procedural versus object-oriented programming	58
7.2	Problem definition	60
7.3	Requirements specification	61
7.4	UML diagrams	61
7.5	Classes and objects	61
7.6	Object discovery	63
7.7	Inheritance	65
<b>8</b>	<b>C++ object-oriented programming syntax</b>	<b>66</b>
8.1	Class declaration	66
8.2	Class definition and member functions	66
8.3	Object creation and polymorphism	68
8.4	Information hiding	70
8.5	Constructors	71
8.6	Examples	73
8.7	Wrapping legacy code	75
8.8	Inheritance	76
8.9	The “protected” keyword	78
8.10	Multifile programs	78
8.11	<b>const</b> member functions	81
<b>9</b>	<b>Arrays and matrices</b>	<b>83</b>
9.1	Data structures and arrays	83
9.2	Array definition and initialization	83
9.3	Array manipulation and memory access	84
9.4	Arrays as function parameters	86
9.5	Returning arrays as objects and object arrays	87
9.6	<b>const</b> arrays	88
9.7	Multidimensional arrays	89
9.8	Multidimensional array storage and loop order	90
9.9	Multidimensional arrays as function arguments	91
<b>10</b>	<b>Input and output streams</b>	<b>93</b>
10.1	The <b>iostream</b> class and stream manipulators	93
10.2	File streams	95
10.3	The <b>string</b> class and <b>string</b> streams	97
10.4	The <b>toString( )</b> class member	98
10.5	The <b>printf</b> function	99

<b>11</b>	<b>References</b>	101
11.1	Basic properties	101
11.2	References as function arguments	102
11.3	Reference member variables	103
11.4	<b>const</b> reference variables	103
11.5	Reference return values	104
<b>12</b>	<b>Pointers and dynamic memory allocation</b>	106
12.1	Introduction to pointers	106
12.2	Initializing pointer variables	107
12.3	The address-of and dereferencing operators	107
12.4	Uninitialized pointer errors	108
12.5	The <b>const</b> keyword and pointers	109
12.6	Pointer arithmetic	110
12.7	Pointers and arrays	110
12.8	Pointer comparisons	111
12.9	Pointers to pointers and matrices	111
12.10	String manipulation	112
12.11	Static and dynamic memory allocation	113
12.12	Memory leaks	115
12.13	Dangling pointers	115
12.14	Pointers in function blocks	117
12.15	Dynamic memory allocation within functions	118
12.16	Dynamically allocated matrices	119
12.17	Dynamically allocated matrices as function arguments	120
12.18	Pointer data structures and linked lists	121
<b>13</b>	<b>Memory management</b>	123
13.1	The <b>this</b> pointer	123
13.2	The friend keyword	124
13.3	Operators	125
13.4	Destructors	127
13.5	Assignment operators	128
13.6	Copy constructors	130
<b>14</b>	<b>The static keyword, multiple and virtual inheritance, templates and the STL</b>	132
14.1	Static variables	132
14.2	Static class members	132
14.3	Virtual functions	134
14.4	Heterogeneous object collections and runtime type identification	135
14.5	Abstract base classes and interfaces	136

14.6	Multiple inheritance	137
14.7	Virtual inheritance	138
14.8	User-defined conversions	138
14.9	Function templates	139
14.10	Templates and classes	140
14.11	The <b>complex</b> class	142
14.12	The standard template library	143
14.13	Structures, unions and nested classes	147
14.14	Bit-fields and operators	148
14.15	Program optimization	149
<b>15</b>	<b>Creating a Java development environment</b>	<b>152</b>
15.1	Basic setup	152
15.2	Command-line operation	153
15.3	A first graphical Java program	155
15.4	DISLIN applet	155
15.5	Graphics applet	156
15.6	Packages	157
15.7	Static (instance) and class members	159
<b>16</b>	<b>Basic Java programming constructs</b>	<b>161</b>
16.1	Comments	161
16.2	Primitive types	161
16.3	Conversions	163
16.4	Operators	164
16.5	Control logic	165
16.6	Enumerations	167
<b>17</b>	<b>Java classes and objects</b>	<b>168</b>
17.1	Class definition	168
17.2	Inheritance	170
17.3	Java references and functions	171
17.4	Exceptions	173
17.5	Basic Java reference types	174
17.6	Input and output	177
17.7	File I/O	178
<b>18</b>	<b>Advanced Java features</b>	<b>179</b>
18.1	Dynamic method dispatch	179
18.2	Abstract classes	180
18.3	Interfaces	180
18.4	Java event handling	182
18.5	Multithreading	184

18.6	Serialization	186
18.7	Generic types	186
<b>19</b>	<b>Introductory numerical analysis</b>	<b>188</b>
19.1	The derivative operator	188
19.2	Error dependence	190
19.3	Graphical error analysis	190
19.4	Analytic error analysis – higher-order methods	192
19.5	Extrapolation	193
19.6	The derivative calculator class	193
19.7	Integration	194
19.8	Root-finding procedures	196
19.9	Minimization	198
<b>20</b>	<b>Linear algebra</b>	<b>200</b>
20.1	Matrices	200
20.2	Linear-equation solvers	200
20.3	Errors and condition numbers	204
20.4	Application: least-squares procedure	204
20.5	Eigenvalues and iterative eigenvalue solvers	206
<b>21</b>	<b>Fourier transforms</b>	<b>208</b>
<b>22</b>	<b>Differential equations</b>	<b>212</b>
22.1	Euler’s method	212
22.2	Error analysis	214
22.3	The Runge–Kutta procedure	216
<b>23</b>	<b>Monte Carlo methods</b>	<b>218</b>
23.1	Monte Carlo integration	218
23.2	Monte Carlo evaluation of distribution functions	219
23.3	Importance sampling	220
23.4	The Metropolis algorithm	221
23.5	Multicanonical methods	224
23.6	Particle simulations	225
23.7	The Ising model	227
<b>24</b>	<b>Partial differential equations</b>	<b>230</b>
24.1	Scientific applications	230
24.2	Direct solution methods	234
24.3	Hyperbolic differential equations and electromagnetics	235
24.4	Elliptic equations	240
24.5	Split-operator methods for parabolic differential equations	242

24.6	Symplectic evolution operators in classical mechanics	245
24.7	Fast Fourier transform methods in optics	246
24.8	The Crank–Nicholson method in quantum mechanics	252
24.9	Finite-difference and finite-element procedures	254
	<i>Index</i>	259