

Programming in Java

A C Norman, Lent Term 2008

Part IA

Contents

1	Preface	7
1.1	What is programming about?	7
1.2	What about <i>good</i> programming?	8
1.3	Ways to save time and effort	9
1.3.1	Use existing resources	9
1.3.2	Avoid dead-ends	10
1.3.3	Create new re-usable resources	10
1.3.4	Documentation and Test trails	10
1.3.5	Do not make the same mistake twice	10
1.4	Where does Java fit in?	12
2	General advice for novices	13
3	Introduction	15
3.1	Introduction	15
3.1.1	Books	18
3.2	Practical work	21
3.2.1	Exercises	24
3.3	A Cook-book Kick-start	28
3.3.1	Code Layout	33
3.3.2	Emacs	34
3.3.3	Drawing to a window: JApplets	37
3.3.4	HTML and appletviewer	42
3.3.5	Exercises	43
4	Basic use of Java	49
4.1	Data types, constants and operations	49
4.1.1	Reserved Words	49
4.1.2	Basic Types	51
4.1.3	Exercises	65
4.2	Operators and expressions	71

4.2.1	Exercises	74
4.3	Control structures	77
4.3.1	Exercises	77
4.4	Control structures Part 2	82
4.4.1	Expression Statements	82
4.4.2	Blocks	82
4.4.3	Null statements	83
4.4.4	if	83
4.4.5	while, continue and break	84
4.4.6	do	84
4.4.7	for	85
4.4.8	switch, case and default	85
4.4.9	return	87
4.4.10	try, catch and throw, finally	87
4.4.11	assert	88
4.4.12	Variable declarations	88
4.4.13	Method definitions	89
4.4.14	Exercises	90
4.5	Java classes and packages	98
4.5.1	Exercises	108
4.6	Inheritance	115
4.6.1	Inheritance and the standard libraries	116
4.6.2	Name-spaces and classes	120
4.6.3	Program development with classes	125
4.7	Generics	129
4.7.1	Exercises	130
4.8	Important features of the class libraries	139
4.8.1	File input and output	140
4.8.2	Big integers	147
4.8.3	Collections	150
4.8.4	Simple use of Threads	150
4.8.5	Network access	153
4.8.6	Menus, scroll bars and dialog boxes	155
4.8.7	Exercises	160
5	Designing and testing programs in Java	167
5.1	Different sorts of programming tasks	171
5.2	Analysis and description of the objective	179
5.2.1	Important Questions	179
5.2.2	Informal specifications	180
5.2.3	Formal descriptions	181

5.2.4	Executable specifications	181
5.3	Ethical Considerations	182
5.4	How much of the work has been done already?	183
5.5	What skills and knowledge are available?	185
5.6	Design of methods to achieve a goal	186
5.6.1	Top-Down Design	186
5.6.2	Bottom-Up Implementation	189
5.6.3	Data Centred Programming	190
5.6.4	Iterative Refinement	190
5.6.5	Which of the above is best?	191
5.7	How do we know it will work?	191
5.8	While you are writing the program	194
5.9	Documenting a program or project	195
5.10	How do we know it does work?	197
5.11	Is it efficient?	200
5.12	Identifying errors	201
5.13	Corrections and other changes	204
5.14	Portability of software	205
5.15	Team-work	206
5.16	Lessons learned	207
5.17	Final Words	208
5.18	Challenging exercises	208
6	A representative application	219
6.1	A Lisp interpreter	219
6.1.1	Exercises	233
7	What you do NOT know yet	235
8	Model Examination Questions	237
8.1	Java vs ML	237
8.2	Matrix Class	238
8.3	Hash Tables	238
8.4	Compass Rose	239
8.5	Language Words	239
8.6	Exception abuse	240
8.7	Queues	240
8.8	Loops	240
8.9	Snap	240
8.10	Partitions	241
8.11	Laziness	241

[Click here to download full PDF material](#)