Jon Sporring

Department of Computer Science,
University of Copenhagen

Learning to Program with F#

2023-09-18

Springer Nature

Preface

This book has been written as an introduction to programming for novice programmers. It is used in the first programming course at the University of Copenhagen's bachelor in computer science program. It has been typeset in LaTeX, and all programs have been developed and tested in dotnet version 6.0.101

Jon Sporring
Professor, Ph.d.
Department of Computer Science,
University of Copenhagen
2023-09-18

Contents

1 Introduction 1

1.1 How to Learn to Solve Problems by Programming 1

1.2 How to Solve Problems 2

1.3 Approaches to Programming 3

1.4 Why Use F# 4

1.5 How to Read This Book 6

Part I Getting started

2 Solving problems by writing a program 9

2.1 Executing F# programs on a computer 11

2.2 Values have types and types reduce the risk of programming errors . 13

2.3 Organizing often used code in functions 15

2.4 Asking the user for input 16

2.5 Conditionally execute code 17

2.6 Repeatedly execute code 18

2.7 Programming as a form of communication 20

2.8 Key Concepts and Terms in This Chapter 22

3 Using F# as a Calculator 23

3.1 Literals and Basic Types 25

3.2 Operators on Basic Types 30

3.3 Boolean Arithmetic 34

3.4 Integer Arithmetic 35

3.5 Floating Point Arithmetic 38

3.6 Char and String Arithmetic 39

3.7 Programming Intermezzo: Hand Conversion Between Decimal and Binary Numbers 41

3.8 Key Concepts and Terms in This Chapter 42

4 Values, Functions, and Statements 45

4.1 Value Bindings 49

4.2 Function Bindings 52

4.3 Do-Bindings 59

4.4 Conditional Expressions 59

4.5 Tracing code by hand 63

4.6 Key Concepts and Terms in This Chapter 66

5 Programming with Types 69

5.1 Type Products: Tuples 70

5.2 Type Sums: Discriminated Unions 73

5.3 Records 75

5.4 Type Abbreviations 77

5.5 Variable Types 78

5.6 Key Concepts and Terms in This Chapter 81

6 Making Programs and Documenting Them 83
6.1 The 8-step Guide to Writing Functions 84
6.2 Programming as a Communication Activity 86
6.3 Key Concepts and Terms in This Chapter 88

Part II Declarative Programming Paradigms

7 Lists 95
7.1 The List Module 101
7.2 Programming Intermezzo: Word Statistics 106
7.3 Key concepts and terms in this chapter 107

8 Recursion Revisited 109
8.1 Recursive Functions 111
8.2 The Call Stack and Tail Recursion 113
8.3 Mutually Recursive Functions 117
8.4 Recursive types 119
8.5 Tracing Recursive Programs 119
8.6 Key Concepts and Terms in This Chapter 122

9 Organising Code in Libraries and Application Programs 123
9.1 Dotnet projects: Libraries and applications 124
9.2 Libraries and applications 126
9.3 Specifying a Module's Interface with a Signature File 127
9.4 Programming Intermezzo: Postfix Arithmetic with a Stack 129
9.5 Generic Modules 132
9.6 Key Concepts and Terms in This Chapter 135

10 Higher-Order Functions 137

10.1 Functions as Values 138

10.2 The Function Composition Operator 139

10.3 Currying 141

10.4 Key concepts and terms in this chapter 142

11 Data Structures 143

11.1 Queues 145

11.2 Trees 145

11.3 Programming intermezzo: Sorting Integers with a Binary Tree 149

11.4 Sets 154

11.5 Maps 156

11.6 Key concepts and terms in this chapter 158

Part III Imperative Programming Paradigms

12 Working With Files 165

12.1 Command Line Arguments 167

12.2 Interacting With the Console 168

12.3 Exceptions 169

12.4 Storing and Retrieving Data From a File 176

12.5 Working With Files and Directories. 182

12.6 Programming intermezzo: Name of Existing File Dialogue 184

12.7 Resource Management 185

12.8 Key concepts and terms in this chapter 186

13 Imperative programming 189

13.1 Variables 190
13.2 While and For Loops 194
13.3 Programming Intermezzo: Imperative Fibonacci numbers 196
13.4 Arrays 198
13.4.1 Array Properties and Methods 202
13.4.2 The Array Module 202
13.4.3 Multidimensional Arrays 203
13.4.4 The Array2D Module 206
13.5 Tracing Imperative Programs 207
13.5.1 Tracing Loops 208
13.5.2 Tracing Mutable Values 211
13.6 Key Concepts and Terms in This Chapter 212

14 Testing Programs 215
14.1 Black-box Testing 218
14.2 White-box Testing 220
14.3 Key Concepts and Terms in This Chapter 223

15 Classes and Objects 227
15.1 Constructors and Members 228
15.2 Accessors 231
15.3 Objects are Reference Types 235
15.4 Static Classes 236
15.5 Recursive Members and Classes 237
15.6 Function and Operator Overloading 238
15.7 Additional Constructors 240

15.8 Programming Intermezzo: Two Dimensional Vectors 242

15.9 Key Concepts and Terms in This Chapter 244

16 Derived Classes 247

16.1 Inheritance 248

16.2 Interfacing with the `printf` Family 251

16.3 Abstract Classes 252

16.4 Interfaces 254

16.5 Programming Intermezzo: Chess 257

16.6 Key Concepts and Terms in This Chapter 269

17 Object-Oriented Design 271

17.1 Identification of Objects, Behaviors, and Interactions by Nouns-and-Verbs 273

17.2 Class Diagrams in the Unified Modelling Language 273

17.2.1 Associations 274

17.2.2 Inheritance-type relations 278

17.2.3 Packages 280

17.3 Programming Intermezzo: Designing a Racing Game 280

17.4 Key Concepts and Terms in This Chapter 284

Part IV Appendices

A The Console in Windows, MacOS X, and Linux 289

A.1 The Basics 289

A.2 Windows 290

A.3 MacOS X and Linux 294

B Number Systems on the Computer 299

B.1 Binary Numbers 299

B.2 IEEE 754 Floating Point Standard 300

C Commonly Used Character Sets 305

C.1 ASCII 305

C.2 ISO/IEC 8859 306

C.3 Unicode 307

D Common Language Infrastructure 311

References 313

Index 315