1. Record Nr. UNINA9910974888203321

Autore	Clocksin William F
Titolo	Clause and Effect : Prolog Programming for the Working Programmer / / by William F. Clocksin
Pubbl/distr/stampa	Berlin, Heidelberg : , : Springer Berlin Heidelberg : , : Imprint : Springer, , 1997
ISBN	3-642-58274-5
Edizione	[1st ed. 1997.]
Descrizione fisica	1 online resource (IX, 143 p.)
Disciplina	005.13/3
Soggetti	Computer programming Software engineering Compilers (Computer programs) Artificial intelligence Programming Techniques Software Engineering Compilers and Interpreters Artificial Intelligence
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Bibliographic Level Mode of Issuance: Monograph
Nota di bibliografia	Includes bibliographical references at the end of each chapters and index.
Nota di contenuto	1. Getting Started -- 1.1 Syntax -- 1.2 Programs -- 1.3 Unification -- 1.4 Execution Model -- 2. Data Structures -- 2.1 Square Bracket Notation -- 2.2 Arithmetic -- 3. Mapping -- Worksheet 10: Full Maps -- Worksheet 11: Multiple Choices -- Worksheet 12: Partial Maps -- Worksheet 13: Removing Duplicates -- Worksheet 14: Partial Maps with a Parameter -- Worksheet 15: Multiple Disjoint Partial Maps -- Worksheet 16: Multiple Disjoint Partial Maps -- Worksheet 17: Full Maps with State -- Worksheet 18: Sequential Maps with State -- Worksheet 19: Scattered Maps with State -- 4. Choice and Commitment -- 4.1 The ‘Cut’ -- 4.2 A Disjoint Partial Map with Cut -- 4.3 Taming Cut -- 4.4 Cut and Negation-as-Failure -- 4.5 Negation-as-Failure Can Be Misleading -- 5. Difference Structures -- Worksheet 25: Concatenating Lists -- Worksheet 26: Rotations of a List -- Worksheet 27: Linearising -- 5.1 Difference Lists -- 5.2 Solution to Max Tree -- 6.

Case Study: Term Rewriting -- 6.1 Symbolic Differentiation -- 6.2 Matrix Products by Symbolic Algebra -- 6.3 The Simplifier -- 7. Case Study: Manipulation of Combinational Circuits -- 7.1 Representing Circuits -- 7.2 Simulation of Circuits -- 7.3 Sums and Products -- 7.4 Simplifying SOP Expressions -- 7.5 Alternative Representation -- 8. Case Study: Clocked Sequential Circuits -- 8.1 Divide-by-Two Pulse Divider -- 8.2 Sequential Parity Checker -- 8.3 Four-Stage Shift Register -- 8.4 Gray Code Counter -- 8.5 Specification of Cascaded Components -- 9. Case Study: A Compiler forThree Model Computers -- 9.1 The Register Machine -- 9.2 The Single-Accumulator Machine -- 9.3 The Stack Machine -- 9.4 Optimisation: Preprocessing the Syntax Tree -- 9.5 Peephole Optimisation -- 10. Case Study: The Fast Fourier Transform in Prolog -- 10.1 Introduction -- 10.2 Notation for Polynomials -- 10.3 The DFT -- 10.4Example: 8-point DFT -- 10.5 Naive Implementation of the DFT -- 10.6 From DFT to FFT -- 10.7 Merging Common Subexpressions -- 10.8 The Graph Generator -- 10.9 Example Run: 8-point FFT -- 10.10 Bibliographic Notes -- 11. Case Study: Higher-Order Functional Programming -- 11.1 Introduction -- 11.2 A Notation for Functions -- 11.3 The Evaluator -- 11.4 Using Higher-Order Functions -- 11.5 Discussion -- 11.6 Bibliographic Notes.

Sommario/riassunto

This book is for people who have done some programming, either in Prolog or in a language other than Prolog, and who can find their way around a reference manual. The emphasis of this book is on a simplified and disciplined methodology for discerning the mathematical structures related to a problem, and then turning these structures into Prolog programs. This book is therefore not concerned about the particular features of the language nor about Prolog programming skills or techniques in general. A relatively pure subset of Prolog is used, which includes the 'cut', but no input/output, no assert/retract, no syntactic extensions such as if- then-else and grammar rules, and hardly any built-in predicates apart from arithmetic operations. I trust that practitioners of Prolog program- ming who have a particular interest in the finer details of syntactic style and language features will understand my purposes in not discussing these matters. The presentation, which I believe is novel for a Prolog programming text, is in terms of an outline of basic concepts interleaved with worksheets. The idea is that worksheets are rather like musical exercises. Carefully graduated in scope, each worksheet introduces only a limited number of new ideas, and gives some guidance for practising them. The principles introduced in the worksheets are then applied to extended examples in the form of case studies.