Contents

Chapter 1 Lexical Analysis Using JFLex

Tokens. Lexical Errors. Regular Expressions. Overview of JFlex. An Example. Lexical Structure of JFlex. The Syntax of JFlex. Directives and macros. Rules. Regular expressions. Character sets. Adding Java Code to the Lexical analyser. Directives in JFlex. Macro definitions. Built in fields and methods. The structure of the Java Source Generated by JFlex. Running Java, Javac and Jar. Running JFlex and CUP. Using JFlex and CUP to implement an interpreter or compiler. Using Cygwin in the Laboratory. Running Windows programs from Cygwin. Matching comments using Jflex. Matching Identifiers and Reserved Words and Interfacing JFlex with CUP.

Chapter 2 Context Free Grammars

Language Definition. Parse trees and abstract syntax trees. Bottom Up (Shift-Reduce) Parsing. LALR(1) Parsing. States. The relationship between the LR classes of grammars. An Example.

Chapter 3 Designing a Grammar

Sequences. Operators.

Chapter 4 Parsing Using Java CUP

An Example. CUP Specification Syntax. Package and Import Specifications. User Code Components. Symbol Lists. Precedence and Associativity declarations. The Grammar. Command Line Options When Running CUP. The structure of the Java Source Generated by CUP.

Chapter 5 Using Ambiguous Grammars

Using Ambiguous Grammars and Precedences. Conflict resolution in CUP. Assigning precedences to terminals and rules. Generating Abstract Syntax Trees and Using Precedences. The effects of omitting the precedence declarations.

Chapter 6 Error Recovery in CUP

The error symbol. The error recovery algorithm.

Chapter 7 Interpreting a Program by Performing a Treewalk

Interpreting Control Structures.

Chapter 8 Treewalking Passes and Attribute Evaluation

Division into passes. Representation of the Abstract Syntax Tree. Treewalking. Evaluation of Attributes. inherited, synthesized and threaded attributes.

Chapter 9 The Design of a Full Computer Language

The Lexical Analyser. The Grammar. Global Declarations and Statements. Types and Values. Class type declarations. Method Declarations. Variable Declarations. Statements. Operators. Primaries. Literal values. Variables. Method names. The Library. The B-- Directory Structure. Division into passes. The main program.

Chapter 10 Some sample B-- programs

Chapter 11 Compile-Time Data Structures

Block structured languages. Compile-Time Environments. Searching the compile-time environment. Declaration Lists. Declarations. Types. Cyclic Dependencies.

Chapter 12 Run-Time Data Structures

Memory Allocation and Deallocation. The Run-time Environment. Run-time values. Primitive values. Pointer Values. In-line storage of arrays and class instances. Representation of class instances. Calculation of Offsets.

Chapter 13 A Brief Comparison with Java

Some Java features. Overloading. Access control. The class Class. Order of initialisation. Type checking is a compile time action, based on the declared types of variables. Methods are overridden, fields are not. Private methods are not overridden. A more complex example.

Chapter 14 Tree Walking and the Interpreter

Representation of Nodes, and Treewalking. Reprinting of Nodes of the Tree. Declaration/Statement Lists. Types. Declarations. Control Statements. Simple Statements. Expressions. Variables.

Chapter 15 Method Declarations and Invocations

Method Invocations. Evaluation of the Method Name. Evaluation of the Parameters. Return Statements. Method Declarations.

Chapter 16 Code Generation

Registers and Run-time Data Structures. Method invocations. Evaluation of Expressions. Accessing Global Variables and Methods. Accessing Local Variables. Accessing Fields. Generating and Accessing the Method Table. Control Statements and Boolean Expressions.

Appendices

Appendix 1 An Algorithm for Performing Error Detection When Parsing Computer Programs

Appendix 2 A Grammar for Regular Expressions

Appendix 3 A Grammer for Grammers

Appendix 4 The Grammar For Java