

Table of Contents

Preface	iii
Chapter 1: Introduction to Compiler	1
1.1 Overview of Translation Process	2
1.2 A Simple Compiler	3
1.3 Difference between Interpreter, Assembler, and Compiler	4
1.4 Overview and Use of Linker and Loader	7
1.5 Types of Compilers	7
Single-pass compilers	7
Multi-pass compilers	8
Load and go compiler	9
Link and load compiler	9
Optimizing compilers	9
1.6 Analysis of the Source Program	10
1.7 The Phases of a Compiler	11
Lexical Analysis	11
Syntax Analysis	12
Semantic Analysis	13
Intermediate Code Generation	14
Code Optimization	14
Code Generation	15
Symbol Table Management	16
Error Handling (Detection and Reporting)	16
1.8 Cousins of the Compiler	17
Preprocessors	17
Assembler	19
Loader and Linkers	20
1.9 The Grouping of Phases: Front End and Back End of Compiler	21
Front End of Compiler: The Analysis Phase	21
Back End of Compiler: The Synthesis Phase	22
1.10 The Pass Structure of Compiler	23

1.11	Compiler Writing Tools.....	24
	Summary.....	25
	Review Questions.....	26
	Multiple Choice Questions.....	28
Chapter 2:	Lexical Analyzer.....	29
2.1	Introduction to Lexical Analyzer.....	29
2.2	Input Buffering.....	30
2.3	Specification of Tokens.....	31
	Tokens, Patterns, and Lexemes.....	32
	Elements of Token Design.....	33
	Chomsky Classification of Grammar.....	35
	Regular Expression.....	37
	Arden's Rule.....	38
2.4	Recognition of Tokens.....	39
	Transition Diagrams.....	41
2.5	A Language for Specifying Lexical Analyzer.....	42
2.6	Finite-State Automata.....	45
2.7	Finite Automata from Regular Expressions.....	47
	Converting NFA to DFA.....	49
	Building DFAs.....	53
2.8	Designing a Lexical Analyzer Generator.....	53
	Recognizer for Identifiers.....	54
	Pattern Matching Using NFA.....	55
	DFA's for Lexical Analyzers.....	56
2.9	Optimizing of DFA tion of DFA.....	57
	Summary.....	61
	Review Questions.....	62
	Multiple Choice Questions.....	64
Chapter 3:	Parsing Theory.....	65
3.1	Types of Parsing: Top-Down and Bottom-Up Parsing Algorithms.....	66
3.2	Top-Down Parsing-Naïve Approach.....	67
	Top-Down Parsing Based on Prediction and Backtracking.....	68
	Recursive Productions.....	69
	Left Factoring.....	70

	Recursive Descent and Predictive Parser	72
	LL Parsers.....	72
3.3	Bottom-Up Parsing-Naïve Approach.....	78
	Shift-Reduce Parsers.....	78
	Precedence Parsers.....	80
	Simple Precedence Grammar and Parsing	81
	Operator Precedence Parsing	87
	Precedence Functions	94
3.4	LR Parsers.....	100
	LR Configuration and Tracing.....	102
	LR Parsing Algorithm.....	102
	Types of LR Parsers.....	104
	LR(0) Parsing Table	105
	SLR(1) Parsers.....	110
	LR(1) Parsers.....	113
	LALR(1) Parser	117
3.5	Parser Generators.....	120
3.6	Using Ambiguous Grammar	123
	Summary.....	127
	Review Questions	128
	Multiple Choice Questions	132
Chapter 4: Syntax-Directed Translation and Error Recovery.....		133
4.1	Syntax-Directed Program Evaluation	134
4.2	Syntax-Directed Definitions	134
	Analysis of SDD	138
4.3	Construction of Syntax Trees.....	140
4.4	Bottom-Up Evaluation of S-Inherited Attributes	143
4.5	L-Attributed Definitions	143
4.6	Syntax-Directed Translation Scheme.....	145
4.7	Error Detection and Recovery	147
	Error Detection	147
	Error Reporting	147
4.8	Adhoc and Systematic Methods.....	148
	Adhoc Methods for LR Parsers.....	148
	Systematic Methods.....	150

Summary.....	150
Review Question.....	151
Multiple Choice Questions	152
Chapter 5: Intermediate Code Generation	153
5.1 Different Intermediate Forms	153
High-Level IRs	154
Low-Level IRs	159
5.2 Syntax-Directed Translation Mechanism and Attributed Mechanisms.....	163
Three-Address Code Generation for Assignment Statements.....	163
Three-Address Code Generation for Arrays	165
Three-Address Code Generation for Boolean Expressions.....	166
Translation of Conditional and Iterative Control Flow	170
Computing l-and r-Values	172
Code Generation for Function Calls	173
Translation of Mixed-Mode Expressions.....	174
Summary.....	175
Review Questions	176
Multiple Choice Questions	178
Chapter 6: Runtime Memory Management.....	179
6.1 Source Language Issues.....	180
6.2 Storage Organization	180
The Runtime Storage Organization.....	181
Allocation of Simple Data Objects	181
Activation and Activation Trees	182
Activation Records	184
Activation Record for Procedure Calls	182
6.3 Storage Allocation Strategies.....	185
Static Allocation	186
Stack Allocation.....	186
Heap Allocation.....	187
Handling Recursive Calls	187
6.4 Management of Variable-Length Blocks.....	189
Runtime Allocation Using Stacks.....	189
Runtime Allocation Using Heaps	190

6.5	Passing Parameters	191
6.6	Symbol Table.....	191
	Uses of Symbol Table.....	192
	Content of the Symbol Table	192
	Data Structures for Symbol Table.....	193
6.7	Language Facilities for Dynamic Allocation	195
	Dangling References.....	195
	Garbage.....	195
	Garbage Collection Algorithms	196
	Compacting Collectors	196
	Generational Collectors	198
	Advantages and Disadvantages of Garbage Collection	198
6.8	Dynamic Storage Allocation Techniques	199
	Managing Implicit Deallocation	199
	Explicit Allocation of Fixed-Sized Blocks	199
	Explicit Allocations of Variable-Sized Blocks	199
	Summary.....	199
	Review Questions	201
	Multiple Choice Questions	202
Chapter 7: Code Optimization		203
7.1	Classification of Compiler Optimization	203
7.2	Important Optimizing Transformations	204
	Compile-Time Evaluation.....	204
	Common Sub-Expression Elimination.....	205
	Code Movement.....	206
	Dead Code Elimination.....	207
	Loop Optimization.....	207
	Strength Reduction	210
7.3	Global Data Flow Analysis.....	210
	Available Expressions.....	211
	Reaching Definitions	211
	Data-Flow Equations	212
	Use-Definition (U-D) Chaining	212
7.4	Optimizations Based on Data-Flow Analysis	212
	Loop Detection	213

	Invariant Code Motion.....	213
	Common Sub-Expression Elimination for Data-Flow Analysis	214
	Dead Code Elimination.....	215
	Loop Unrolling	215
	Induction Variable Elimination.....	215
	Busy Expressions.....	216
	Summary.....	217
	Review Questions	218
	Multiple Choice Questions	220
Chapter 8:	Code Generation	221
8.1	Major Tasks in Code Generation	221
8.2	Design Issues of a Code Generator	223
	Input for Code Generators	223
	Output from the Code Generator	223
	Memory and Reference Management	224
	Choice of Instruction to Generate	224
	Register Allocation	227
	Choice of Evaluation Order	227
8.3	The Target Machine.....	227
	Addressing Modes	227
	Program and Instruction Cost	228
8.4	Runtime Storage Management.....	229
	Stack Allocation.....	229
	Caller Calling Callee.....	229
	Callee Returning	230
8.5	Basic Blocks and Flow Graphs	230
	Basic Blocks	230
	Flow Graphs	231
8.6	Next-Use Information	233
8.7	A Simple Code Generator	235
	Code Generation Algorithm.....	235
	Register Allocation and Assignments	235
	Machines Implementation of Conditional Jumps	236
8.8	The DAG Representation of Basic Blocks	237

8.9	Peephole Optimization.....	238
	Eliminating Redundant Loads and Stores.....	239
	Identifying Unreachable Instructions.....	240
	Simplify Algebraic Expressions	240
	Strength Reduction	241
	Use Faster Machine Instructions.....	241
	Elimination of Multiple Jumps	242
8.10	Generating Code from DAGs	242
	Rearranging Order of the Code.....	242
	A More Practical Code Generation from DAG.....	244
8.11	Dynamic Programming Code-Generation Algorithm	246
8.12	Code-Generator Generators	247
	Summary.....	247
	Review Questions	249
	Multiple Choice Questions	250
Index	251