

# INDEX

The page numbers relate to a version of the text typeset for A4 paper which is down-loadable from <http://scifac.ru.ac.za/compilers/longpcl.zip> as a set of PCL files suitable for direct printing to a LaserJet® compatible printer.

- 8-bit microprocessor 229
- Accumulator 25, 30
- Absolute addressing 28, 46
- Abstract machine 7
- Abstract syntax tree 10, 101, 181, 232, 270
- Accumulator machine 27
- Activation record 246, 263
- Actual parameter 93, 259, 263
- Adding machine 162
- Address 71
  - absolute 28
  - field expressions 87
  - fields 72
  - immediate 28
  - indexed 29
  - indirect 29
  - mode 27
  - relative 29, 46
  - return 246
  - run-time 247, 250, 252, 267
- Alex scanner generator 176
- Alias 259
- Algol 60 report 49, 54, 106
- Alphabet 50
- ALU 30, 39
- Ambiguity 49, 133
- Ambiguous grammar 104, 125, 133
- Analytic phase 8
- Anonymous type 215
- Antecedent 70
- ANTLR parser generator 148
- Applied occurrence 71, 191, 208
- Argument 259
- Arithmetic expression 100, 151, 190
- Arithmetic logic unit 30, 39
- ARM (Annotated Reference Manual) 70
- Array handling 212, 216, 219, 228, 268, 275
- ASCII 197, 202
- ASSEMBLER
  - grammar 72
  - language 14, 20, 71
- Assembly 7
  - assembler class 75
  - code generation 75, 82
  - conditional 95
  - lexical analysis 77
  - macro processing 92
  - minimal 37
  - one-pass 74, 83
  - semantic analysis 81
  - source handling 76
  - syntax analysis 79
  - two-pass 73, 80
- AST 10, 101, 181, 232, 270
- AST code generation 181, 232, 270
- Asynchronous input 297
- Attribute grammar 69, 110, 151, 157
- Augmented grammar 123
  - driver program 174
  - execution 162
  - frame file 147, 161, 175
  - ftp sites Appendix A
  - installation 161, Appendix A
  - makefile 162
  - parser interface 173
  - scanner interface 166
  - support modules 172
- Cocol 61, 72, 161
- Automata theory 110, 143
- Axiomatic semantics 69
- Back end 8, 182, 217
- Backpatching 74, 219, 224, 262
- Backtracking 108, 117
- Backus-Naur form 49, 54
- Bailes 205
- Base class 183, 201, 213, 232, 270
- Base pointer 39, 48, 247
- BASIC PRINT statement 109
- Beacon symbol 137, 211
- Binary search 201, 214
- Binary tree 82, 214, 244
- Block structure 241
- BNF 49, 54
  - notation 54
  - extensions 59
- Boolean operator 12, 46, 65, 180, 190
- Bootstrap 2, 20, 22, 62
- Bottom-up parsing 143
- Bound checks 46, 228
- Bounded buffer 286
- break statement 204
- Brinch Hansen 205, 216, 251, 263, 275
- British Standard for EBNF 61
- Bus lines 25
- Busy-wait 295
- C declarations 177
- Call-by-reference 259, 274
- Call-by-value 259, 274
- Canonical derivation 56, 104
- CASE statement 204, 225
- Character handler 8
- Character set 63, 164
- Chomsky hierarchy 107
- Cichelli 202
- Clang
  - Cocol specification 110, 220, Appendix C
  - code generator interface 212, 217
  - constraint analyser 208
  - error handling 198, 206
  - hand-crafted parser 203, 206
  - hand-crafted scanner 199
  - hand-crafted source handler 196
  - report (on Diskette)
  - syntax 111
- Clang-Topsy translator 192
- Closure 50
  - Kleene 50, 59, 65
  - positive 50
- COBEGIN ... COEND 282, 293
- Cocktail 147
- Coco-2 176
- Coco/R 61, 123, 146, 161, 189
  - Clang error handling 208
  - Clang scanner 201
  - Clang specification 110, 220, Appendix C
- Concrete syntax 10
- Concrete syntax tree 10
- Concurrent processes 281
- Concurrent programming 281
- Condition flag 30, 36, 46
- Conditional assembly 95
- Consequence rule 69
- Consequent 70
- Constant declarations 124
- Constant expressions 236

- actions 168
- ANY 164, 168
- CHARACTERS 164
- COMMENTS 164
- CONTEXT clause 165
- EOF 170
- formal attributes 167
- grammar checks 171
- IGNORE 165
- LexString 173
- NAMES 166
- parser specification 167
- Pragmas 162, 166
- PRAGMAS 166
- PRODUCTIONS 167
- scanner interface 166
- scanner specification 163
- semantic errors 172
- SemError() 173
- Successful() 173
- SynError() 173
- SYNC 168, 170, 208
- synchronization sets 170, 207
- syntax error recovery 162, 169
- TOKENS 165
- user names 166
- WEAK 168, 208
- weak separators 171
- weak terminal 168, 170
- Code generation 12
  - ASSEMBLER 82
  - assembler code 238
  - from an AST 181, 232, 270
  - generator interface 212, 217
  - native 231
  - on-the-fly 179, 184, 226, 250, 255, 266
  - one-address code 179, 181
  - optimization 12, 181, 235
  - reentrant code 246
  - stack machine code 226
- Collision 91
- Comments 51, 71, 109, 164
  - pragmatic 201
- Communication 284
- Compile-time 2, 71
- Compiler 2, 7
  - load-and-go 7, 37, 219
  - multi-pass 14
  - one-pass 14
  - structure 8, 195
- Compiler generator 4, 146
- Dynamic link 247, 253
- Dynamic semantics 4, 68
- ε-free grammar 103
- ε-productions 58, 107, 109
- EBNF 59
  - British standard 61
  - expressions 60
- Edison 70, 216, 225, 284
- Effective address 28, 30, 39
- Electronic mail 193
- Empty set 51
- Empty statement 128
- Empty string 51, 186
- Emulation 25
- Emulator 16
  - single-accumulator 35, Appendix D
  - stack machine 42, Appendix B
- Environment 156
- Equivalent grammar 100, 125, 133
- Erasure 59, 107
- Error checking code 13, 275
- Error
  - context free 136, 169, 206
  - context-sensitive 172, 208
  - correction 13, 80, 136
  - detection 80, 86, 136
  - handling 13
  - recovery 13, 136, 206
  - reporting 13, 87, 198
  - semantic 172
  - spurious 136, 138
- Constant folding 184, 235
- Constraint analyser 10, 73, 208
- CONTEXT clause 165
- Context condition 157
- Context-free grammar 109, 135
- Context-sensitive 73, 106, 262
- Context switch 291
- Contextual constraint analyser 10
- continue statement 204
- Control statement 219
- Conway's problem 286
- Critical region 284, 288
- Cross compiler 2, 7, 17, 22, 75
- Cross reference generator 191
- Cycle-free grammar 104
- DAG (directed acyclic graph) 237
- Dangling ELSE 105, 126, 133, 204, 223
- Data register 25
- Deadlock 285, 295
- Debugger 239
- Declaration level 243
- Declaration order 278
- Declarations 81, 138, 208, 241, 245, 260, 262
- Decompiler 7
- Decorated tree 10, 152
- Defining occurrence 71, 208
- Delphi 3
- Denotational semantics 69
- Dereferencing 41
- Derived from 54
- Descriptor ring 291
- Designator 134, 209, 212, 252, 259, 273
- Deterministic finite automaton (DFA) 142
- Deterministic parsing 116
- DFA (deterministic finite automaton) 142
- Dijkstra 284
- Direct addressing 28
- Directed acyclic graph (DAG) 237
- Directive
  - assembler 71
  - compiler 201
  - table 74
- Directly produces 54
- Director set 123
- Disassembler 7
- Display 252, 257
- Display copy 253
- Distributed processing 290
- DLG 148
- Driver program 174, 196
- Free Software Foundation 19, 202
- Front end 8, 182, 195
- FSA (Finite state automata) 110, 139, 201
- Function 259
  - assignment 264
  - reference 260
  - return value 265, 277
- GNU project 19, 202
- Goal symbol 53, 55
- Goods trains 65
- GOTO statement 225, 258
- Gough 202
- Global variables 163, 242, 292
- gperf 202
- Grammar 53
  - ambiguous 104, 125, 133
  - attribute 69, 110, 151, 157
  - augmented 123
  - checks 171
  - context-free 109, 135
  - context-sensitive 106
  - cycle free 104
  - ε-free 103
  - equivalent 100, 125, 133
  - expression 57, 100, 149, 151, 179, 190
  - graph representation 185
  - hierarchy 107
  - L-attributed 157
  - left-linear 110
  - null free 103
  - reduced 103

- Escape sequence 200
- EXIT statement 204, 224
- Exception 13
- Exclusion 284
- Expression
  - evaluation 151, 190
  - grammar 57, 100, 149, 151, 179, 190
  - parameter 259
- Extended BNF 59
- Extent of identifiers 242
- Fetch-execute cycle 26, 35, 43
- Finite state automata (FSA) 110, 139, 162
- FIRST function 118, 129, 136
- FOLLOW function 120, 129, 136, 206, 211
- Followers 137, 206
- FOR loop 204, 223, 279
- Formal
  - attributes 168
  - language 50
  - methods 4
  - parameter 93, 259
  - semantics 67
- Fortran 2, 9, 50, 60, 117, 241
- Forward
  - declaration 257
  - reference 73, 83, 85, 88, 159, 219, 238, 251
- Frame file 147, 161, 175
- Frame header 246, 263
- Indexed addressing 29
- Indivisible operation 284, 295
- Inductive expression 69
- Inference rule 69
- Inherent addressing 28
- Inherited attribute 157
- Instruction pointer 26
- Instruction register 25
- Instruction set 31, 40
- Intermediate addressing 247
- Intermediate code 11
- Interpreter 15
- Interpretive compiler 15, 22
- Interrupts 26
- IOTRANSFER 297
- Java 4
- Kernel 290
- Keywords 10, 64, 125, 142, 200, 201
- Kleene closure 50, 59, 65
- L-attributed grammar 157
- L-value 212
- Label 71, 81, 99
  - redefined 82, 85
  - undefined 82
- LALR parsing 146
- Lambda production 59
- Language 50
- Language design 5, 276, 280
- Left canonical derivation 56
- Left linear grammar 110
- Left recursion 54, 126, 152
- Level, declaration 243
- lex scanner generator 147
- Lexeme 142, 163, 199, 201
- Lexical analyser 8, 58, 77, 89
- Lexical structure 58, 61, 63
- Lexicon 58
- LexName 173
- LexString 173
- Lifetime of identifiers 242
- Link
  - dynamic 247, 253
  - static 248, 257
- Linkage area 246
- Linkage editor 7
- Linker 7
- Literal pool 39, 48
- Literal strings 115, 199, 228
- LL(1) conflict resolution 210, 223, 241, 245, 266, 269
- LL(1) restrictions 118, 125
- LL(k) parsing 117, 146
- regular 109
- restrictions 102, 118, 125
- right-linear 109
- S-attributed 157
- transformation 125
- type 0 107
- type 1 107
- type 2 109
- type 3 109
- Graph 185
- Half bootstrap 21
- Hand-crafted parser 203, 206
- Hand-crafted scanner 139, 199
- Hand-crafted source handler 196
- Hash table 90, 214, 245
- Hashing function 90
- Hexadecimal literals 78
- High-level translator 7, 14, 19, 192
- Highland Gathering 66
- Host language 2, 19
- Hypothetical stack machine 38, 217, 226
- IF ... THEN ... ELSE 70, 105, 125, 133, 204, 205
- Ignorable characters 63, 164
- Immediate addressing 28
- Implementation language 2
- Independent compilation 14
- Index register 30
- Loop
  - EXIT 224
  - FOR 204, 223, 279
  - REPEAT 125, 223
  - WHILE 10, 68, 70, 205, 219
- LR(k) parsing 143
- Machine
  - abstract 7
  - code 231
  - dependencies 8
  - instructions 25
  - single-accumulator 30
  - stack 38
- Macro 92
  - assembler 7, 74, 92
  - handler class 93
  - expansion 92, 92
  - parameters 93
  - recursive 96
- Make file 66, 162
- Mark stack pointer 39, 248, 263
- Memory-indexed addressing 29
- Memory-indirect addressing 29
- Metalanguage 49
- Metasymbol 50, 59
- Minimal perfect hashing function 202
- Mnemonic 26, 71
- Modularity 4
- Multi-stage translator 14
- Music 193
- Mutual exclusion 284
- Mutually recursive procedures 257
- Native code 231
- Name equivalence 215
- NDFA (non-deterministic finite automaton) 142
- Nested blocks 242
- Non-deterministic finite automaton (NDFA) 142
- Non-reachable production 103
- Non-terminal 53, 60
- Non-terminating production 103
- Null
  - production 58
  - string 50
  - string problem 120
- Nullable 59, 119, 130, 134
- Oberon 2, 3, 216, 240, 278
- Object language 2
- Object orientation 3, 132, 183, 235
- occam 284
- Offset 226, 247, 263, 289
- On-the-fly code generator 226

- LLGen parser generator 147
- Load-and-go translator 7, 219
- Loader 37
- Local variables 220, 245
- Location counter 75, 212
- Logical operator 180
- LOOP ... EXIT ... END statements 204, 224
- Optimization 12, 181, 235, 251
  - Peephole 12, 228, 231, 257
- Orthogonality 4
- Overflow 36, 46, 185
- P-code assembler 23
- P-machine 17
- Parameters
  - actual 93, 259, 263, 267, 271
  - array 268, 275
  - formal 259, 262
  - passing mechanisms 259, 265, 268
  - procedural 275
- Parameterless procedure 241
- Parse stack 144
- Parse tree 101, 152
- Parser 10, 58
  - bottom-up 143
  - construction 129
  - deterministic 116
  - generator 146, 185
  - interface 173
  - LALR 146
  - LL(k) 117
  - LR(k) 143
  - recursive descent 116, 129, 132, 149, 195
  - SLR 146
  - specification 167
  - table driven 141, 145
  - top-down 116, 143
- Pascal standard 70
- Pascal-FC 284
- Pascal-P compiler 17, 22, 203, 225
- Pascal-Plus 284
- Pascal-S 239, 257, 298
- Passes 14, 195
- PCCTS compiler generator 148
- Peephole optimization 12, 228, 231, 257
- Perfect hashing function 202
- Pervasive identifier 244, 275
- Phases 8, 195
- Phrase 50
- Phrase structure 56, 58, 61, 63
- Phrase structure tree 56, 101, 151
- Pointer types 216
- Portability 3, 15, 20
- Portable interpretive compilers 17, 22
- Porting a compiler 20
- Post-mortem dump 275
- Postfix notation 150
- Pragma 162, 165, 201
- Pragmatic considerations 50, 73
- Precedence 51, 127, 180
- Precedence graph 282
- Predeclared identifier 275
- Preprocessor 7
- Pretty printer 192
- Priority queue 297
- Procedural parameter 275
- Procedure
  - activation 248, 263
  - grammars 102, 118
  - LL(1) 118, 125
- Return address 246, 249, 253
- RETURN statement 260, 265, 268, 277
- Return value 265
- Reverse Polish 65, 150
- Rewrite rule 54
- Right canonical derivation 56
- Right linear grammar 109
- Right recursion 54, 126
- ROM BIOS 33
- Round-robin scheduler 296
- Rule of inference 69
- Rule of consequence 69
- Run-time 2, 71
- One-address code 27
- One and a half address code 27
- One-pass assembler 74, 83
- Opcode 27, 71
- Open array 260, 275
- Operating system 2
- Operational semantics 33, 68
  - calling 249, 252, 255, 266
  - declaration 245
  - nested 242
  - parameters 259
  - regular 241
  - return 246, 250, 253
- Process
  - concurrent 281
  - descriptor 291
  - parallel 282
  - priority 297
  - sequential 281
- Producer-consumer problem 285
- Produces 54
- Produces directly 54
- Production rule 53
- Productions 53, 62
  - Cocol 169
  - null 58, 103
  - single 104, 129
  - unreachable 107
  - useless 103, 128
- Program counter 26, 30, 39
- Program proving 69
- Pseudo-code 16
- Push-down automata 143
- Qualified identifier 135
- R-value 212
- Range checks 229
- Rational Pascal 205
- Real-time 282
- Record types 216
- Recursion
  - in BNF 57, 59
  - in grammars 54
  - in procedures 254, 257, 276
  - left 54, 126, 152
  - right 54, 126
- Recursive descent parsing 116, 129, 132, 149,
- Reduce action 144
- Reduced grammar 103
- Reductions 143
- Redundant code 236
- Register 25
  - allocation 180
  - indexed addressing 29
  - indirect addressing 29
  - status 30
- Regular
  - expression 50, 139
  - grammar 109
  - procedure 168, 241
- Rehashing 91
- Relative addressing 29, 46
- Relocatable code 7, 97
- Removal of redundant code 236
- REPEAT loop 125, 204, 223
- Reserved keywords 10, 64, 125, 142, 200
- Restrictions
  - Side-effect 70, 84, 279
  - Signal 284, 287, 295
- Single-accumulator machine 30
  - assembler 37
  - emulator 35
  - opcodes 31
- Single pass compilation 14
- Single production 104, 129
- SKIP statement 205
- Source handling 76, 196, 198
- Source language 2
- Spreadsheet 15, 190
- Spurious error 136, 138
- Stack frame 39, 246, 253, 263
- Stack machine 38, 217, 226, 230

- S-attributed grammar 157
- SLR parsing 146
- SYNC 168, 170, 208
- Sale's algorithm 277
- Scanner 8, 52, 58, 77, 89, 129, 139
  - generator 146
  - interface 166
  - specification 163
- Scope
  - insecurities 276
  - node 243
  - rules 242
- Self-compiling compiler 20
- Self-embedding 53, 54, 143
- Self-resident translator 7
- Semantic
  - action 149, 151
  - attributes 152
  - driven parser 169, 210
  - error detection 172
  - overtones 61, 133
- Semantics 49
  - axiomatic 69
  - denotational 69
  - dynamic 4, 68
  - formal 67
  - operational 68
  - short-circuit 12, 180, 228, 239
  - static 4, 68, 81, 209
- Semaphore 284, 287, 291, 295
- SemError() 173
- Semicolon 128, 212
- Sentence 50, 53
- Sentential form 54
- Sentinel node 243
- Separate compilation 7, 14
- Sequential algorithms 281
- Sequential conjunction 12
- Sequential process
- Set class 87, 137
- Shakespeare 124
- Shared memory 290
- Shift action 144
- Shift-reduce conflict 146
- Short-circuit semantics 12, 14, 180, 228, 239
- Terminal 53, 60
  - start sets and symbols 118
  - successors 120
- Three-address code 27
- Time-slicing 290, 297
- Token 8, 50, 165
  - classes 163
- Tonic Solfa 193
- Top-down parsing 116, 143
- Topsy 114, 195
  - report (on Diskette)
- Transition diagram 141
- Transmitted attribute 157
- Transputer 290
- Tree-building actions 181
- Turbo Pascal 3, 7, 16, 35, 183, 198, 257
- Two-address code 27
- Two-pass assembler 73, 80
- Type 0 grammar 107
- Type 1 grammar 103
- Type 2 grammar 109
- Type 3 grammar 109
- Type checking 10, 214
- Type identifiers 205
- UCSD Pascal 17, 23, 99
- Umbriel 239, 278
- Undeclared identifier 82, 214
- Union 146, 183, 213, 216, 232
- UNIX 2, 66, 151
- Unrestricted grammar 107
- Useless production 103, 128
- User names 166
- assembler 47
- emulator 42
- opcodes 40
- Stack pointer 30, 39, 48, 230
- Start sets 118
- Start symbol 53, 55
- State diagram 290
- State variable 91
- Static
  - level 243, 247
  - link 248
  - semantic analyser 10, 81, 87, 208, 266
  - semantics 4, 68, 209, 266
- Status register 30
- Storage management 246
- String 50, 199, 228
  - table 74, 79
  - type 216
- Structural equivalence 215
- Subrange 25, 200, 216
- Subroutine 93
- Subset language 20
- Successful() 173
- switch statement 204
- Symbol 50
  - beacon 137, 211
  - goal 53
- Symbol table 13, 73, 80, 89, 135, 209, 212, 2
- Synchronization 136, 170, 206, 284
- SynError() 173
- Syntactic class 53
- Syntax 4
  - analyser 10, 79, 87
  - diagram 67
  - equation 54
  - error recovery 136, 162
- Syntax directed translation 149
- Synthesized attribute 154
- Synthetic phase 8
- Systems program 2
- T-diagram 6, 19
- Table-driven algorithm 141, 145
- Target language 2
- VDL 69
- VDM 69
- Value Designator 212, 232

Variable  
  declarations 138  
  designator 212, 259  
  offset 226, 247, 263, 289  
  redeclared 242  
  undeclared 82, 214  
Variable Designator 212, 232, 259  
Variant record 146, 183, 213, 216, 232  
Virtual machine 7, 16, 230  
Visibility 242  
Vocabulary 53  
Void function 129, 168, 241, 244, 257  
  
Wait 284, 287, 295  
Weak separator 171, 206  
Weak terminal 168, 170  
WHILE loop 10, 68, 70, 205, 219  
Wirth 3, 59, 189, 206, 240, 257, 277, 298  
WITH statement 68  
  
Yacc parser generator 146, 147, 151  
  
Z80 processor 229  
Zero-address instruction 40