Appendices

Bibliography

This book is a little like the previews of coming attractions at the movies; it's meant to whet your appetite in several directions, without giving you the complete story about anything. To find out more, you'll have to consult more specialized books on each topic.

There are a lot of books on computer programming and computer science, and whichever I chose to list here would be out of date by the time you read this. Instead of trying to give current references in every area, in this edition I'm listing only the few most important and timeless books, plus an indication of the sources I used for each chapter.

Computer science is a fast-changing field; if you want to know what the current hot issues are, you have to read the journals. The way to start is to join the Association for Computing Machinery, 1515 Broadway New York, NY 10036. If you are a full-time student you are eligible for a special rate for dues, which as I write this is $25 per year. (But you should write for a membership application with the current rates.) The Association publishes about 20 monthly or quarterly periodicals, plus the newsletters of about 40 Special Interest Groups in particular fields.

Read These!

If you read no other books about computer science, you must read these two. One is an introductory text for college computer science students; the other is intended for a nonspecialist audience.

Abelson, Harold, and Gerald Jay Sussman with Julie Sussman, *Structure and Interpretation of Computer Programs*, MIT Press, Second Edition, 1996.

> The introductory computer science text at MIT, this book uses Lisp as the vehicle for an intense study of everything from data structures to machine architecture. Although it's not a book about artificial intelligence as such, this is the definitive presentation of the artificial intelligence view of what computer science in general is about, and the best computer science book ever written.

Hofstadter, Douglas R., *Gödel, Escher, Bach: an Eternal Golden Braid*, Basic Books, 1979.

> This book won the Pulitzer Prize for its success in explaining to readers who aren't computer scientists some of the deepest ideas of computer science, and it makes a strong case for the view that those ideas also have a lot to teach us about human intelligence.

Chapter 1: Automata Theory

The reference I used in thinking about this chapter was

Minsky, Marvin, *Computation Finite and Infinite Machines*, Prentice-Hall, 1967.

> Part of the interest of this particular text is that its author is a leading figure in artificial intelligence research, and so the question of whether the insights of automata theory apply also to human intelligence is always visible as a motivating force in the presentation of the theory. Minsky's bibliography will refer you to the original papers by Turing, Kleene, Church, and so on as well as some left-field references to biological information processing from people like Lettvin and McCulloch.

Chapter 2: Discrete Mathematics

This chapter touches on several topics. An overall introduction for computer scientists is

Liu, Chung Laung, Elements of Discrete Mathematics, McGraw-Hill, Second Edition, 1985.

> This book requires no advanced mathematical background, but it does require that the reader feel comfortable with mathematical notation and the notion of formal proof. The topics include both purely mathematical ones, like set theory, combinatorics, and modern algebra, and related computer science ones like computability, formal languages, automata theory, analysis of algorithms, and recursion. This list is not unlike the one in the book you're reading now, and in fact Professor Liu expresses a goal similar to mine: to show computer science undergraduates the relevance of mathematics to their work. The difference is that I use actual programs to illustrate the ideas whenever possible, whereas his is a "straight" math book. (Of course another difference is that he treats all these topics in much more depth. But don't be scared away; he starts simply.)

On the topic of mathematical logic, there is a range of books that vary in accessibility Among the most pleasant are

Smullyan, Raymond, What Is the Name of This Book? Prentice-Hall, 1978
———, The Lady or the Tiger? Knopf, 1982
———, 5000 B.C. and Other Philosophical Fantasies, St. Martin's, 1984
———, Alice in Puzzle-Land, Penguin, 1984.

> These are books of puzzles based on logic, but they go beyond the simple propositional inference puzzles like the one in the text. Smullyan starts with some of the classic puzzle categories, like the Liars and Truth-Tellers puzzle, and builds up to an exposition in puzzle form of topics like self-reference, modal logic, and Gödel's Theorem.

"Logic programming" is the use of mathematical logic formalisms as a programming language. It is also called "declarative programming" because instead of issuing commands to the computer, the programmer makes statements about things known to be true. The algorithm by which the programming system makes inferences from these statements is not explicitly provided by the programmer, but is built into the language.

The most widely known logic programming language, although not the only one, is Prolog. An accessible introductory text is

Ennals, Richard, *Beginning Micro-Prolog*, Harper & Row, Second Edition, 1984.

> I list this book here because it's a Prolog text and therefore relevant to mathematical logic, but for me the main interest of the book is that it argues for the use of Prolog in teaching kids, as an alternative to Logo. The book gives examples of logic programming at work in various curriculum areas.

Chapter 3: Algorithms and Data Structures

To a software engineer, the issues in this chapter are among the central ones in computer science. That's not my own way of thinking, so it's possible that my presentation doesn't give the field all the pizazz that an enthusiast would manage. To compensate for that, you should read

Bentley, Jon, *Programming Pearls*, Addison-Wesley, 1986.

> This is a collection of monthly articles written by Bentley for the Communications of the Association for Computing Machinery. It requires virtually no formal mathematics and is extremely readable. If the book has a moral, it's "Think first, program later." It makes its case with a number of true-to-life examples of projects in which orders of magnitude were saved in the execution time of a program by rethinking its fundamental structure.

Chapter 4: Programming Language Design

There are textbooks in "comparative programming languages", but I'm going to stick to the strategy of the chapter by using Pascal as the example. Structure and Interpretation of Computer Programs, one of my must-reads, will be useful as a contrast here, giving the Lisp point of view.

Jensen, Kathleen, and Niklaus Wirth, Pascal User Manual and Report, Springer-Verlag, Third Edition, 1985.

> This is the official report of the international committee responsible for the design of the language. The book has two parts, a reference manual and the committee report itself. The latter includes some explicit discussion of the design decisions in the language.

Chapter 5: Programming Language Implementation

I really didn't have a reference for this chapter; I just sort of forged ahead on my own! But here's the book I should have read first:

Friedman, Daniel P., Mitchell Wand, and Christopher T. Haynes, Essentials of Programming Languages, MIT Press, 1992.

> This book uses the Scheme dialect of Lisp as the basis for a study of programming language interpreters. It's harder reading than most of the books in this bibliography, but it encourages the reader to think very deeply about how programming languages work.

Chapter 6: Artificial Intelligence

I'll list two references here; one on language understanding in general and one that contains a paper about the Student program specifically.

Winograd, Terry, Language as a Cognitive Process, Volume 1: Syntax, Addison-Wesley, 1983.

> A planned second volume on semantics was not published. This is a technically meaty book, but considering its depth it is quite readable.

The book strikes a good balance among technical programming issues, psychological issues, and the ideas of mainstream linguistics. It includes an extensive bibliography. When I attended Terry's course at Stanford in which he first presented the material that became this book, it was the first time I experienced a course that ended with a standing ovation for the instructor. The book shows the same clarity of explanation and the same enthusiasm.

Minsky, Marvin L., Semantic Information Processing, MIT Press, 1969.

This is a collection of early research reports. I include it here because one of its chapters is a paper by Bobrow on STUDENT, and you won't be able to find the more complete description in Bobrow's unpublished thesis. Other chapters describe similar microworld-strategy projects of the same vintage.

Computers and People

Last but far from least, some of the most fascinating reading connected with computer science is found outside of the technical literature, in the reactions of psychologists, philosophers, and sociologists to the computer as a social force. You owe it to yourself to understand the human context of your work; you owe it to everyone else to be strongly aware of the social implications of what you do.

Turkle, Sherry, The Second Self: Computers and the Human Spirit, Simon and Schuster, 1984.

A sociologist's view of the computer culture, this book explores both the psychology of computer experts and the ways in which a computer-rich environment has changed the thinking of non-experts not only about technology but about what it means to be human.

Weizenbaum, Joseph, Computer Power and Human Reason: From Judgment to Calculation, W. H. Freeman, 1976.

Weizenbaum is a computer scientist, and this book is in part a technical argument about the limitations of what computers can do. But it is more importantly a call to computer scientists to take responsibility for the uses to which their inventions are put. Weizenbaum argues that there

are things we shouldn't do with computers, even if we can learn how to overcome the technical obstacles. Computer-based weapons of war are an obvious example, but Weizenbaum is also worried about things like automated psychotherapy, which was just a daydream when the book appeared but has since become a reality to a limited extent. Many computer scientists find this book offensive, and it is certainly possible to find flaws in the details. But the critics rarely present an alternative with an equally strong social conscience.

Dreyfus, Hubert L., What Computers Still Can't Do: A Critique of Artificial Reason, MIT Press, 1992.

Dreyfus is a philosopher who uses the phenomenological ideas of Heidegger and others to suggest a fundamental flaw in the assumptions AI researchers make about human intelligence. To try to sum it up in one sentence, the sort of thinking that people do in solving a puzzle is very different from the much more profound intelligence we use in carrying out our more customary activities. AI programs mimic the former but not the latter. This is a revision of an earlier book, taking into account more recent developments in AI research.

Weinberg, Gerald M., The Psychology of Computer Programming, Van Nostrand Reinhold, 1971.

This book studies programming as a social activity, programming as an individual activity, and the programming environment. In my opinion, its main contribution is the idea of "egoless programming," which means more or less that when your friend finds that impossible bug in your program for you, you should feel happy rather than threatened. Weinberg offers several good ideas for how to act as part of a programming community. On the other hand, I'm less enthusiastic about his manager's-eye view of the programmer as a cog in the machine, rather than as a creative artist. But overall I think this book is well worth reading; it's also entertainingly written.

Credits

Material on page xiv quoted from The Second Self: Computers and the Human Spirit by Sherry Turkle. Copyright © 1984 by Sherry Turkle. Reprinted by permission of Simon & Schuster, Inc.

Material on page 48 quoted from Mind Benders B-2 by Anita Harnadek. Copyright © 1978 by Midwest Publications (now called Critical Thinking Press, Box 448, Pacific Grove, CA 93950). Reprinted by permission of the publisher.

Material on page 53 by Diane C. Baldwin quoted from The Dell Book of Logic Problems #4. Copyright © 1989 by Dell Publishing, a division of Bantam Doubleday Dell Publishing Group, Inc., reprinted by permission of Dell Magazines.

Material on pages 279 and 294 quoted from Natural Language Input for a Computer Problem Solving Program by Daniel G. Bobrow (unpublished Ph.D. thesis). Copyright © 1964 by Daniel G. Bobrow. Reprinted by permission of the author.

Material on page 295 quoted from Mindstorms: Children, Computers, and Powerful Ideas by Seymour Papert. Copyright © 1984 by Basic Books, Inc., publishers. Reprinted by permission of the publisher.

The illustration on page 313 quoted from Language as a Cognitive Process, Volume 1: Syntax by Terry Winograd. Copyright © 1983 by Addison-Wesley Publishing Company, Inc. Reprinted by permission of the publisher.

Index of Defined Procedures

This index lists example procedures whose definitions are in the text. The general index lists technical terms and primitive procedures.

#gather 335
#test 335
#test2 335
&test 335
@test 336
@test2 336
@try.pred 336
^test 336

A

abs 331
accept 36
acceptpart 36
addchild 140, 141, 155
addnumbers 193
ageify 318
agepprob 317
agepprop 318
agesen 318
agewhen 318
always 336
anyof 336
anyof1 336
areacode 143, 157
arglist 257
array.save 42
arrays ize 256
arraytype 255
arrow.stub 43
arrowhead 37
arrows.from.start 39
arrowtail 37
arrowtext 37
article 332

B

balance 144, 158
balance1 158
bkt1 317
blank 36
blockbody 257
bracket 316

C

cap 333
cards 181
category 99
change.head 43
changeone 316
changes 315
changes1 316

chang es2 316
chang es3 316
check .n d 41
check .t ype 263
child re n 136, 141, 155
choos e 93, 104
citie s 133, 135, 137, 157
citie s1 157
city 143, 157
clean up 102
code 270
codel oad 265
codes to re 249, 265
combs 75, 76, 104
commai st 270
compound 258
copy. to .a cc ept s 40
copya rr ay 266
cub.r epor te r 57, 102

D

datum 136, 141, 155
deck 182
denom 325
depun ct 315
deter mi ne 40
diffe r 58, 100
diffe r1 100
displ ay 36
distr ib ti mes 325
distr ib x 326
divte rm 326
dlm 332

E

equiv 64, 101
exch 206
exit 276
exp.m ode 273
exp.t ype 273
exp.v al ue 273
expan d 78, 104
expre ssio n 224
expt 320

F

f 93, 105, 164
fact 79, 104, 162, 163
fact.s eq 162
factor 326
factor 1 326
factor 2 326
falses 100
falsif y 59, 99
female 63, 103
findda tu m138, 157
findde li m316
findde li m1316
findfa ls e 100
findke y 315
findtr ue 100
fix.ar ro ws 43
foote. fa mily 62, 103
frame. oute rf rame 274
frame. pr evfr ame 274
frame. re ta ddr 274
frame. sa ve .n ewf p 274
fsm 35
fsm1 35
fsmnex t 35
fsmtes t 36
functi on 241, 256

G

game 35
get 101
getcha r 215, 272
geteqn s 331
getid 273
gettyp e 273
guess. middle .va lu e 120

H

haltp 34
hanoi 170

haspair 112
highbranch 145, 159
howmany 116, 155

I

id 255
id.lname 273
id.pointer 273
id.type 273
idioms 315
ifbe 222
ifbeels e 222
implies 47, 61, 101
implies1 101
in 336
increment 197
insert 41
IntSquare 194

J

jal 276
jr 276
jump 275
jumpf 276
jumpt 275
justbefore 63, 100
justbefore1 101

L

last2 315
leaf 138, 156
leafp 137, 156
leaves 138, 156
lesthanp 116, 155
letter.join 42
letterp 272
linde x 264
linde x1 265
listcity 143, 158
lsay 333
lname 272
locate 133, 135, 137, 156
locate.in.forest 133, 135, 137, 157
locate1 157
lock 88, 105
lock1 105
lock2 105
lowbranch 145, 159

M

machine 37
make.arrow 37
make.machine 36
make.stub 43
male 63, 103
match 333
match! 335
match# 335
match& 335
match? 335
match@ 336
match^ 336
maybeadd 326
maybemul 327
median 206
memaddr 265
memsetup 266
minusin 327
movedisk 170
movepart 36
multi 195
mustbe 221, 270

N

nd.traverse 40
ndconcat 38
ndletter 38
ndmany 39
ndor 39
newarg 257
newline 276
newlname 272
newregister 270
newstate 40
newtail 40

newvar 256
nexti ndex 265
nextr ow 115, 154, 164
nmtes t 322
nocap 316
noimmedia te 246, 263
nonde t 22, 38
number 272
numty pe 269

O

occva r 327
offse t 265
op.in st r 273
op.na rg s 273
op0 332
op1 332
op2 333
opdif f 320
opera to rp 333
opfor m320
oprem 320
opset up 253
optes t 320
optim iz e 42
optim iz e. st ate 43
or.sp li ce 39

P

parra yassig n 264
parse .s peci al 334
passi gn 248, 262
passi gn1 248, 263
pbool ean 246, 263
pchar 263
pchar data 269
pchec kt ype 269
pclos e 229, 267
pdata 269
peers 100
perms 79, 104
perso np 332
pexpr 228, 267
pexpr1 267
pexpro p 229, 267
pfor 259
pfunca ll 269
pfunse t 262
pgetbi nary 268
pgetun ar y 267
pif 220, 222, 245, 258
pinteg er 263
plibra ry 254
plural 333
pnewty pe 268
popen 229, 267
posspr o 332
ppopop 229, 268
pprocc al l 259
pprocc al l1 260
prans 330
pransw er s 330
preal 263
premat ch 334
prepea t 258
proc1 241, 257
procar g 260
procar gs 260
procar ra yarg 261
proced ur e 241, 256
procva ra rg 261
progra m254
progra m1254
pronou n 332
prun 274
prun1 275
psort 120, 121, 123, 154, 205
psort1 155
pstrin gassig n 264
pstrin gassig n1 264
putch 276
putint 276
putrea l 276
putstr 276
puttf 276
pwhile 259
pwrite 261
pwrite 1 261

pwrite2 262
pwrite3 262
pwriteln 262

Q

qset 319
quadratic 108, 109, 153
quoted 336
qword 332

R

range 255
range1 256
rc1 272
readnumber 193
RealSquare 194
realt 113, 153, 195
reg.globalptr 274
reg.retaddr 274
reg.stackptr 274
reg.zero 274
regfree 270
reject 36
remfactor 327
remfactor1 327
remop 327
removeonce 118
reservedp 271
reverse 110
rload 275
rmatch 334
roundoff 331
runsetup 275

S

say 333
says 60, 103
senform 318
senform1 319
set.in 334
set.special 334
setindex 264
setminus 333
settruth 59, 99
settruth1 100
showdata 205
showdeck 181
shuffle 182
simdiv 328
simone 328
simp 93, 106
simplex 93, 105, 115, 154, 164, 166
simplex.seq 165
simplus 328
simplus1 328
simplus2 329
simtimes 329
simtimes1 329
singular 333
skipcomment 271
slowsort 127
socks 76, 77, 104
sockets 84, 105
solution 102
solve 322
solve.reduce 323
solve1 102, 323
solveq 324
solveq.minus 325
solveq.product 325
solveq.product1 325
solveq.quotient 325
solveq.rplus 324
solveq.sum 324
solveq.sum1 324
solveq.sum2 324
solveq1 324
solveqboth 324
solver 323
sort 41, 206
spaces 276
special 334
splice 39
square 194, 321
ssort 118, 154
ssort1 154
startpart 36

state ment 220, 258
store 101, 275
strin g 38, 271
strin ga 38
strin gl ose 264
stub. add 42
stub. ar ro w 44
stub. head 44
stub. te xt 43
stude nt 1 314
stude nt 2 314
subor d 329
subor d1 329
subst op 330
subte rm 330
sumpr ods 115, 154, 164

T

t 97, 106, 112, 113, 153, 195
targe ta ddr 265
this 322
toboo l 275
token 216, 271
token 1 272
tower 170
tree 137, 141, 155
treec it y 145, 158
treec it y1 158
try 34
try.p re d 336
tryid io m 332
trypr oc part 256
tryso lv e 322
tst.d if fe re nce 321
tst.d iv by 320
tst.l es st han 320
tst.m in us 321
tst.m in us s 321
tst.n umof 320
tst.p er 320
tst.p er ce nt 321
tst.p er le ss 322
tst.p lu s 321
tst.p lu ss 321
tst.sq uare 321
tst.sq uare d 321
tst.su m 321
tst.ti mes 322
tst.to th epower 320
twocha r 217, 271
twoto 89, 105
typech eck 256

U

unitst ri ng 331

V

varequ al 332
vargro up 255
varkey 331
varpar t 255
varter ms 330
vartes t 331
vartes t1 331
verb 332
verify 58, 99

W

worldt re e 132, 135, 138, 139, 156

X

xor 60, 101

General Index

This index lists technical terms and primitive procedures. There is also an index of defined procedures, which lists procedures whose definitions are in the text.

" 67
Ù 46
Ø 46
Ú 46
Ä 68
S 91
® 46

A

Abelson, Harold ix, xvii, 342
accepting state 4
access, random 148
actual argument 176
adder 71
adder, finite-state 25
address 147, 231
age problem 281, 290
aggregate type 187
algebra word problems 278
algorithm 15, 107, 294
algorithm, two-stack 225
algorithms, analysis of 107
allocation, dynamic 148
alphabet rule 11, 16
alternatives rule 11, 19
ambiguous 217
analysis of algorithms 107
analysis, lexical 215
Anderson, Chris xvii
APL 224
apprenticeship xi
argument, actual 176
array 181, 187, 199
array, packed 188, 189
artificial intelligence 277
assembly language 231
assignment statement 178, 180, 195
Association for Computing Machinery 341
association list 287
ATN 312
augmented transition network 312
automata theory 1, 309

B

background 300
backtracking 65
balanced tree 146
balancing parentheses 13, 28
Baldwin, Diane C. 53
BASIC 196
Beatles 46
begin (Pascal) 175

Bentley, Jon 344
bibliography 341
binary computer 69
binary number 25, 88
binary operator 223
binary search algorithm 143
binary tree 132, 144
binding, call by 199
binomial coefficient 82
bit 26, 28, 71
block structure 166, 177
Bobrow, Daniel G. xvii, 278
Boolean (Pascal) 187
bottom-up 175
bound reference 184
branch node 132
byte 147, 188

C

C 225
CAI, intelligent 294
call by binding 199
call by reference 197, 198
call by value 197, 198
call, procedure 178
category 49
char (Pascal) 187
checking, compile-time 223
Chinese food 46
circuit, integrated 230
Clancy, Michael xvii
closed form definition 80, 98
code generation 211, 245
coefficient, binomial 82
coefficient, multinomial 94
cognitive science 278
Colby, Kenneth 31
combination 74, 79
combination lock 74, 86
combinatorics 72
common subexpression elimination 109
community xv
compile-time checking 223
compiler 172
compiler compiler 219, 224
compiler, incremental 174
compiler, optimizing 109
compiler, Pascal 209
complexity ix
composition of functions 162
compound proposition 46
compound statement 177
computer assisted instruction 294
computer center xv
computer hardware 69
computer logic 71
computer science ix
computer, binary 69
concatenation rule 11, 17
conditional statement 177
constant string 189
constructor 137
context, limited 278
context-free language 310
continuous function 45
contradiction, proof by 34
contrapositive rule 61
correctness 107
correspondence, one-to-one 89
counting problem 45

D

data structure 107, 129
data type 187
Davis, Jim xvii
declaration part 176
declarative knowledge 16
declarative programming 168
declarative programming languages 16
declarative representation x
definition, closed form 80, 98
definition, formal 211
definition, inductive 80, 91, 94, 112
definition, recursive 12
descent, recursive 220
deterministic grammar 219
directed graph 22
discrete mathematics 45

Dreyfus, Hubert L. 347
dyadic 224
dynamic allocation 148
dynamic environment 184, 241
dynamic programming 115
dynamic scope 184, 185, 197, 198, 242

E

economics xiv
editor, text 9, 169
education 294
effective procedure 31
efficiency 107
elementary function 80
elimination rule 50
embedding 310
end (Pascal) 175
engineering, knowledge 278
English 278
Ennals, Richard 344
enumerated type 191
environment, dynamic 184, 241
environment, lexical 184, 198, 241
equation 281, 285
equivalence relation 68
ethics xiv
exclusive or 68
expansion, multinomial 112
experimental method 83
expert system 278
exponential 129
expression 223
expression, regular 11, 15, 211
extensibility 181
external memory 30

F

factorial 79
fence 205
Fermat, Pierre de 82
finite-state adder 25
finite-state machine 3, 15, 213, 309
first 110
food, Chinese 46
forest 133
formal definition 211
formal parameter 176
formal thinking ix
for (Pascal) 178, 181, 183
frame pointer 236
frame, stack 235
free reference 184, 185
Friedman, Daniel P. 345
function, continuous 45
function, elementary 80
function, generating 81, 98
function, predicate 67
function, sine 81
function, truth-valued 46, 67
functional programming 162
functions, composition of 162
function (Pascal) 194

G

gate 69
general knowledge 300
generated symbol 141, 239, 293, 304
generating function 81, 98
generation, code 211, 245
global optimization 111
global pointer 235
Goldenberg, Paul xvii
grammar, deterministic 219
grammar, predictive 219
graph 22
graph, directed 22
graphics xiii

H

half-adder 70
halting state 31
halting theorem 32
Hanoi, Tower of 169
hardware, computer 69
Harnadek, Anita xvii, 48
hash table 130

Haynes, Christopher T. 345
heap 131
heapsort 131
heuristic 294, 298
hierarchy 131, 146
hierarchy, syntactic 213
Hil®nger, Paul xvii
Hoare, C. A. R. 125
Hofstadter, Douglas R. 342

I

if (Pascal) 221
immediate 232
implication rule 60
incremental compiler 174
independent 72
index register 235
index variable 205
individual 49
induction, mathematical x
inductive de®nition 80, 91, 94, 112
inference system 47
inference, rules of 47
in®nite loop 32
in®nite set 89
insertion sort 119
instruction, computer assisted 294
integers, sum of the 119
integer (Pascal) 187
integrated circuit 230
intellectual property xv
intelligence, arti®cial 277
intelligent CAI 294
interactive language 169
intermediate language 173
internal state 30
interpreter 172
intractable 129
inverter 69
Iverson, Kenneth E. 224

J

Jensen, Kathleen 345
joke 112

K

keyword 219
Kleene, Stephen C. 16
knowledge engineering 278
knowledge representation 309
knowledge, declarative 16
knowledge, general 300
knowledge, procedural 16
Knuth, Donald E. 129

L

label 234
language, context-free 310
language, interactive 169
language, intermediate 173
language, machine 172, 209
language, non-interactive 169
last 110
leaf node 132
lexical analysis 215
lexical environment 184, 198, 241
lexical scope 183
limited context 278
linear 129, 286
linear search algorithm 143
Lisp 185, 196, 282
list, association 287
list, property 301
list, sorted 130
Liu, Chung Laung 343
load 147
local optimization 110
local procedure 176
lock, combination 74, 86
lock, Simplex 85, 114
logarithm 129
logic problem 45, 48
logic programming 168
logic, computer 71
logic, predicate 67
logic, propositional 46

logic, ternary 73
Logo 169, 294
Logo pattern matcher 283
Logo variable 199
lookahead 215
lookahead, one-character 216
loop, infinite 32

M

machine language 172, 209
machine, finite-state 3, 15, 213, 309
machine, nondeterministic 6, 22
machine, theoretical 1
mandatory substitution 280
matching, pattern 282, 308
mathematical induction x
mathematical model 1
mathematics, discrete 45
memoization 112, 312
memory 147, 230
memory, computer 28
memory, external 30
mergesort 126
Meteor 282
microworld 278, 297
Minsky, Marvin 342, 346
model, mathematical 1
modification, tree 139
monadic 224
multinomial coefficient 94
multinomial expansion 112
multiplication rule 72
mutator 141

N

nand 70
network, augmented transition 312
network, recursive transition 213
Newell, Allen 277
node, branch 132
node, leaf 132
node, root 132
non-interactive language 169
nondeterministic machine 6, 22
nor 70
null pointer 148
number, binary 25, 88
number, random 83
numerical analysis xiii
numof 281, 286

O

object-oriented programming 167
offset 235
one-character lookahead 216
one-to-one correspondence 89
operating systems xiii
operator precedence 223
operator, binary 223
operator, relational 224
operator, unary 223
optimization, global 111
optimization, local 110
optimizing compiler 109
optional substitution 304
ordered subset 74
ordering 146
ordering relation 68
ordinal type 188
overflow signal 71

P

P-code 173
packed array 188, 189
pair 147
Papert, Seymour x, 295
paradigm, programming 162
parameter, formal 176
parameter, value 197
parameter, variable 197
parentheses 227
parentheses, balancing 13, 28
Parry 31
parser 211, 212, 217
parser generator 217
partition sort 120, 204

Pascal 161, 310, 345
Pascal compiler 209
Pascal program 172
Pascal variable 199
Pascal's Triangle 82, 94
Pascal, Blaise 82
pattern matcher, Logo 283
pattern matching 282, 308
periodic 81
Perlis, Alan J. xi
permutation 74, 79
philosophy xiv
Piaget, Jean x
piracy, software xv
pointer 147, 190
pointer, frame 236
pointer, global 235
pointer, null 148
pointer, stack 236
portable 173
precedence 225, 285
precedence, operator 223
predicate function 67
predicate logic 67
predictive grammar 219
probability 73
problem, logic 48
procedural knowledge 16
procedural representation x
procedure call 178
procedure, effective 31
procedure, local 176
procedure, recursive 75, 80
procedure (Pascal) 176, 194
process ix
processor 147, 230
production rule 13, 212, 310
program verification 107
program, Pascal 172
programming languages, declarative 16
programming paradigm 162
programming, declarative 168
programming, dynamic 115
programming, functional 162
programming, logic 168
programming, object-oriented 167
programming, sequential 162
program (Pascal) 171, 172
Prolog 16, 68, 344
proof by contradiction 34
property list 55, 301
property, intellectual xv
proposition, compound 46
proposition, simple 46
propositional logic 46
psychology xiv, 277

Q

quadratic 129
quadratic formula 108
quantifier 283
quicksort 125

R

random access 148
random number 83
range 188
real (Pascal) 187
record 190
recursive definition 12
recursive descent 220
recursive procedure 75, 80
recursive transition network 213
reference, bound 184
reference, call by 197, 198
reference, free 184, 185
Reggini, Horacio xvii
register 230
register, index 235
regular expression 11, 15, 211
reject state 4
relation 67
relation, equivalence 68
relation, ordering 68
relational operator 224
repeat (Pascal) 178
repetition rule 11, 21
reserved word 183

retrieval time 131
robust 109, 146, 193
root node 132
round (Pascal) 190
RTN 213, 310
rule, production 13, 212
rules of inference 47

S

scalar type 187
science, cognitive 278
scope 239, 241
scope, dynamic 184, 185, 197, 198, 242
scope, lexical 183
search algorithm, binary 143
search algorithm, linear 143
searching 142
selection sort 117
selector 136
self-reference 32
semantics 180
sentence, simple 281
sentinel 205
sequential programming 162
set theory 89
set, infinite 89
sharable 150
sigma 91
Simon, Herbert A. 277
simple proposition 46
simple sentence 281
simple statement 177, 178
Simplex lock 85, 114
simulation 83
sine function 81
Smullyan, Raymond 343
sociology xiv
software engineering x, 174
software piracy xv
Somos, Michael xvii
sort, insertion 119
sort, partition 120, 204
sort, selection 117
sorted list 130
sorting 115
source file 171
space, time and 130
Spock, Mr. 72
spreadsheet 16
stack frame 235
stack pointer 236
start state 5
state 3
state, accepting 4
state, halting 31
state, internal 30
statement part 175
statement, assignment 178, 180, 195
statement, compound 177
statement, conditional 177
statement, simple 177, 178
statement, structured 178
storage time 130
store 147
string, constant 189
structure, block 177
structured statement 178
Student 278
subrange type 191
subset, ordered 74
substitution technique 285
substitution, mandatory 280
substitution, optional 304
sum of several terms 91
sum of the integers 119
Sussman, Gerald Jayix, 342
Sussman, Julie 342
symbol, generated 141, 239, 293, 304
symmetric 68
syntactic hierarchy 213
syntax 179
system, expert 278
system, inference 47

T

table of values 81
table, hash 130
ternary logic 73

text editor 9, 169
theoretical machine 1
thinking, formal ix
time and space 130
time, retrieval 131
time, storage 130
timesharing systems xvi
token 214
tokenization 211, 214
top-down 174
Tower of Hanoi 169
tractable 129
tradeoff 130
transition network, augmented 312
transition network, recursive 213
transitive 68
transitive rules 51
tree 131
tree modification 139
tree, balanced 146
tree, binary 132, 144
trunc (Pascal) 190
truth-valued function 46, 67
Turing machine 30, 311
Turing machine, universal 33
Turing's thesis 31
Turing, Alan M. 30
Turkle, Sherry xiv, xvii, 346
two-stackalgorithm 225
type, aggregate 187
type, data 187
type, enumerated 191
type, ordinal 188
type, scalar 187
type, subrange 191
type, user-defined 191
typed variable 187
type (Pascal) 191

U

unambiguous 217
unary operator 223
uniqueness rule 50, 67
unit 286
universal Turing machine 33
Unix xiii, xvi, 9
user-defined type 191

V

value parameter 197
value, call by 197, 198
variable parameter 197
variable, index 205
variable, Logo 199
variable, Pascal 199
variable, typed 187
var (Pascal) 176, 178, 197
verification, program 107

W

Wand, Mitchell 345
Weinberg, Gerald M. 347
Weizenbaum, Joseph 346
while (Pascal) 178
White, Dick xvii
Winograd, Terry xvii, 312, 345
Wirth, Niklaus 345
word 147, 189
word problems, algebra 278
word, reserved 183
workspace 171
workstations xvi
writeln (Pascal) 178
write (Pascal) 178

Y

YACC 219