
Index

- aand 191
- abbrev 214
- abbrevs 214
- abbreviations 213
- Abelson, Harold 18
- Abelson, Julie 18
- ablock 193
- Abrahams, Paul W. 391
- :accessor 365
- accumulators 23, 47, 394
- acond 191
- acond2 198, 239
- Adams, Norman I. 396
- after 50
- aif 191
- aif2 198
- alambda 193
- Algol 8
- allf 169
- :allocation 367
- always 227
- alrec 205
- anaphora—see macros, anaphoric
- ANSI Common Lisp ix
- antecedent 322
- append
 - Prolog implementation 331
- append1 45
- apply 13
 - with macros 110
 - on *&rest* parameters 137
- =apply 267
- arch
 - Lisp as 8
 - bottom-up program as 4
- architects 284
- Armstrong, Louis vii
- artificial intelligence 1
- asetf 223
- assignment
 - macros for 170
 - order of 177
 - parallel 96
 - in Prolog 343
 - and referential transparency 198
 - see also: generalized variables
- assoc 196
- ATNs 305
 - arc types 311
 - correctness of 312
 - destructive operations in 313
 - like functional programs 316
 - for natural language 305
 - nondeterminism in 308
 - operations on register stack 398
 - order of arcs 308
 - recursion in 306
 - registers of 306, 312
 - represented as functions 309
 - tracing 309
- atrec 210
- augmented transition networks—see ATNs

- Autocad 1, 5
- automata theory 292
- avg 182
- awhen 191
- awhen2 198
- awhile 191
- awhile2 198

- backtraces 111
- backtracking 292
- backquote (‘) 84
 - in ATNs 307
 - nested 214, 217, 395
- bad-reverse 29
- barbarians 283
- Basic 30, 33
- battlefield 8
- before 50
- Benson, Eric 137
- best 52
- Bezier curves 185
- =bind 267
- binding 239
- binding lists 239
- bindings, altering 107
- blackboards 281
- block 154
 - implicit 131, 155
- block-names 131
- body (of expressions) 87, 91, 87
- body (of a rule) 322
- &body 87
- bookshops 41
- bottom-up design v, 3, 321
 - and functional arguments 42
 - and incremental testing 38
 - and shape of programs 4
 - multilayer 321
- bound—see variables, bound
- break-loop 56
- brevity viii, 43
- bricks, furniture made of 117
- Brooks, Frederick P. 5

- C 388
- C++ 398

- call-next-method 200, 375
 - sketch of 358
- call-with-current-continuation (call/cc) 260
 - at toplevel 292
- capital expenditures 43
- capture 118
 - avoiding with gensyms 128
 - avoiding with packages 130
 - avoiding by prior evaluation 125
 - of block names 131
 - detecting potential 121
 - free symbol capture 119
 - avoiding 125
 - of function names 131, 392
 - intentional 190, 267, 313
 - macro argument capture 118
 - of tags 131
- case 15
- >case 152
- case-sensitivity 331
- chains of closures 76, 269
- Chocoblobs 298
- choose 287
 - extent of 291
- choose
 - Common Lisp version 295
 - Scheme version 293
- choose-bind 295
- chronological backtracking 292
- classes
 - defining 364
 - see also: superclasses
- Clinger, William 395
- CLOS 364
 - as an embedded language 349, 377
 - see also: classes, generic functions, methods, slots
- closed world assumption 249
- closures 17, 62, 76
- CLTL—see *Common Lisp: the Language*
- code-walkers 237, 273
- Common Lisp: the Language* ix
- Common Lisp
 - case-sensitivity of 331
 - definition of ix

- differences between versions
 - compilation of closures 25
 - complement 62
 - defpackage 384
 - destructuring-bind 93
 - dynamic-extent 150
 - environment of expanders 96, 393
 - no expansion in compiled code 136
 - function-lambda-expression 390
 - *gensym-counter* 129
 - if-not deprecated 62
 - ignore-errors 147
 - inversions from defun 179
 - Lisp package 384
 - name of user package 381
 - redefining built-in operators 131, 199
 - &rest parameters not fresh 137
 - symbol-macros 205
 - with-slots 236
 - see also: CLOS, series
- evaluation rule 392
- long names in 393
- vs. Scheme 259
- Common Lisp Object System—see CLOS
- common-lisp 384
- common-lisp-user 381
- compall 388
- compilation 24
 - bounds-checking during 186
 - computation during 109, 181, 197, 254, 335
 - of embedded languages 116, 254
 - errors emerging during 139
 - inline 26, 109, 110
 - testing 388
 - of local functions 23, 25, 81, 346
 - of macro calls 83, 101, 136
 - of networks 79
 - restrictions on 25
 - senses of 346
 - of queries 254
 - see also: tail-recursion optimization
- compile 24, 116, 388
- compile-file 25
- compiled-function-p 24
- complement 62
- compose 66
- composition—see functions, composition of
- conc1 45
- conc1f 170, 174
- concf 170
- concnew 170
- conditionals 108, 150
- condlet 146
- congruent parameter lists 372
- consequent 322
- consing
 - avoiding 31, 150, 197, 363
- constitutional amendments 391
- constraints 332
- *cont* 266
- context
 - and referential transparency 199
 - see also: environments; macros, context-creating
- continuations 258
 - destructive operations in 261, 313
 - cost of 284
 - see also: call-with-current-continuation
- continuation-passing macros 266
 - use in multiprocessing 283
 - use in nondeterministic choice 296
 - restrictions on 270
 - and tail-recursion optimization 298
- continuation-passing style (CPS) 273
- cookies 184
- copy-list 71, 206
- copy-tree 71, 210
- courtiers 375
- cut 337
 - with *fail* 342
 - green 339
 - red 339
 - in Lisp 298
- cut 301
- databases
 - caching updates to 179
 - locks on 148

- natural language interfaces to 306
 - queries on 246
 - representation of 247
 - with Prolog 398
- dbind 232
- def! 64
- defanaph 223
- defclass 364
- defdelim 228
- defgeneric 371
- define-modify-macro 168
- defmacro 82, 95
- defpackage 384
- defprop 354
- defun 10, 113
 - defining inversions with 179
- =defun 267
- defsetf 178
- delay 211
- delete-if 64
- density of source code 59, 389
- destruc 232
- destructive operations 31, 64
- destructuring
 - on arrays 234
 - on instances 236
 - on lists 230
 - in macros 93
 - and reference 236
 - on sequences 231
 - on structures 235
- destructuring-bind 93, 213, 230
- differences 207
- disassemble 388
- dispatching 370, 371
- do 98
 - implicit block in 131
 - multiple-valued version 162
 - order of evaluation in 392
- do-file 199
- do-symbols 388, 393
- do-tuples/c 156
- do-tuples/o 156
- do* 97
 - multiple-valued version 159
- dolist 94
 - generalization of 156
- Dolphin Seafood 219
- dotted lists 70, 390
- duplicate 50
- dynamic extent 127, 150
- dynamic languages 398
- dynamic scope 16

- Edwards, Daniel J. 391
- elt 244
- Emacs—see Gnu Emacs
- embedded languages 7, 188, 246
 - ATNs as 309
 - benefits of 110, 116, 246, 377
 - borderline of 246
 - compilation of 116
 - not quite compilers 346
 - implementation of 116
 - for multiprocessing 275
 - Prolog as 321
 - query languages as 246
 - see also: CLOS
- end-of-file (eof) 197, 225
- English 306
- environment
 - argument 95
 - interactive 8
 - of macro expanders 96, 393
 - of macro expansions 108
 - null 96, 278, 394
- error 148
- error-checking 45
- eval
 - explicit 34, 163, 197, 278
 - on macroexpansions 92
 - sketch of 391
- evaluation
 - avoiding 151, 181
 - lazy 211
 - order of
 - in Common Lisp 135
 - in Scheme 259
 - sketch of 391
- evaluation rule 392
- evenp 14
- evolution

- design by 1
 - of Lisp 158
 - of programming languages 8
- expander code 99
- expansion code 99
- explode 58
- exploratory programming 1, 284
- export 383
- :export 384
- expt 32
- extensibility 5
 - of object-oriented programs 16, 379
- extent, dynamic 127, 150
-
- _f* 173, 222
- factions 167
- factorials 343, 387
- fail* 287
- fail
 - Common Lisp version 295
 - Scheme version 293
- failure 195
- fboundp 388
- fif 67
- filter 47
 - simpler version 389
- find2 50
 - evolution of 41
- find-if 41, 195
 - sketch of 206
 - version for trees 73
- finished programs 285
- fint 67
- flatten 47, 72, 210
 - simpler version 389
- Floyd, Robert W. 293
- fmakunbound 373
- fn 202, 229
- Foderaro, John K. v
- for 154
- force 211
- Fortran 8
- free—see variables, free
- fullbind 324
- fun x
- fun 67
-
- funcall 13, 259
- =funcall 267
- function calls, avoiding
 - by inline compilation 26
 - with macros 109
 - by tail recursion 23
- functional interfaces 35
- functional programs 28
 - almost 35
 - and bottom-up programming 37
 - from imperative ones 33
 - shape of 30
- functions
 - as arguments 13, 42, 177
 - constant 226
 - closures of 17, 62, 76
 - use in nondeterministic choice 296
 - stack allocation of 150
 - combined with macros 141, 149, 266
 - compiled 24
 - composition of 66, 201, 228
 - as a data type 9
 - defining 10
 - filleting 115
 - generating recursive 68, 204
 - generic—see generic functions
 - internal 172
 - interpreted 24
 - as lists 27
 - literal 11
 - recursive 21, 193
 - local 21
 - vs. macros 109
 - names of 11, 213
 - as properties 15
 - redefining built-in 131, 174
 - as return values 17, 61, 76, 201
 - set operations on 67, 201
 - with state 18, 65
 - tail-recursive 23
 - transforming into macros 102
 - undefining 373
 - see also: compilation; defgeneric; defun; labels
- function-lambda-expression 390

- Gabriel, Richard P. 23
- garbage
 - avoiding—see consing, avoiding
 - collection 8, 81
- generalized variables 107, 165
 - meaning of 179
 - see also: inversions
- generic functions 371
 - defining 371
 - removing 373
 - see also: methods
- gensym 128
 - to indicate failure 197
 - as unbound 244, 330
- gensym? 243
- *gensym-counter* 129
- gentemp 392
- Gelernter, David H. 198
- get 63
- gethash 196
 - recursive version 350
- get-setf-method 171
- gift-shops, airport 278
- GNU Emacs 1, 5
- go 100, 155
- gods 8
- gold 386
- good-reverse 30
- group 47
 - simpler version 389

- Hart, Timothy P. 391
- hash tables 65, 247, 350
- head 322
- hiding implementation details 216, 382
- hygienic macros 392

- ice-cream 370
- ice-skating 33
- if3 150
- if-match 242
- ignore-errors 147
- Igor 289
- imperative programming 33
- import 383
- in 152
- incf 171
 - generalization of 173
- incremental testing 37
- indexing 249
- inheritance
 - single 196
 - of slots 366
 - multiple 366
 - sketch of 351
- in-if 152
- :initarg 365
- :initform 365
- in-package 382
- inq 152
- instances 365
- intellectuals 374
- interactive development 37, 316
- interactive environment 8
- intercourse, lexical 108
- Interleaf 1, 5
- intern 128, 136, 266
- interning 128, 136, 381
- intersection 207
- intersections 207
- inversions
 - asymmetric 179
 - defining 178
 - see also: generalized variables
- iteration
 - macros for 108, 154
 - vs. nondeterministic choice 291, 325
 - without loops 264, 325

- Jagannathan, Suresh 198
- jazz vii
- joiner 62
- joke, practical—see Nitzberg, Mark

- keywords 386

- labels 21
- lambda 11
- =lambda 267
- lambda-expressions 11, 21
- last 45
- last1 45

- Latin 306
- lawyers 298
- let 144, 199
- let* 172
- lengths of programs 387
- Levin, Michael I. 391
- lexical scope 16
- life, meaning of 197
- lions 37
- Lisp
 - 1.5 95
 - defining features of 1, 8, 349, 398
 - integration with user programs 110
 - slowness of 285
 - speed of 388
 - see also Common Lisp, Scheme, T
- lists
 - accumulating 47
 - as binary trees 70
 - as code 116
 - decreased role of 44
 - disambiguating return values with 196
 - dotted 390
 - as facts 247
 - flat—see flatten
 - interleaving 160
 - operating on end of 170
 - quoted 37
 - recursers on 68, 204
 - as trees 262
 - uses for 70
- list processing 44, 398
- locality 36
- logic programs 334
- longer 47
 - simpler version 389
- loop 154
- loops
 - interrupting 154
 - see also: iteration
- lrec 69

- McCarthy, John I. 391
- mac 92
- macroexpand 91
- macroexpand-1 91

- macro-characters—see read-macros
- macros 82
 - as abbreviations 213
 - access 167, 216
 - anaphoric 189
 - defining automatically 218
 - for distinguishing failure from falsity 195
 - for generating recursive functions 204
 - multiple-valued 198
 - and referential transparency 198
 - see also: call-next-method
 - and apply 110
 - applications of 111
 - arguments to 107
 - for building functions 201
 - calls invertible 166, 216
 - clarity 99, 233
 - and CLOS 378
 - for computation at compile-time 181
 - context-creating 143
 - combined with functions 141, 149, 266
 - compiled 83, 101, 136
 - complex 96
 - defining 82
 - efficiency 99
 - environment argument to 95
 - environment of expander 96, 393
 - environment of expansion 108
 - errors in
 - modifying arguments 137
 - modifying expansions 139
 - non-functional expanders 136
 - nonterminating expansion 139
 - number of evaluations 133, 167
 - order of evaluation 135
 - see also: capture
 - expansion of 83
 - in compiled code 136
 - multiple 136, 138
 - non-terminating 139
 - testing 92
 - time of 83
 - from functions 102

- vs. functions 109
- hygienic 392
- justification of 392
- macro-defining 213, 266
- parameter lists 93
- position in source code 102, 266
- as programs 96
- proportion in a program 117
- recursion in 139
- redefining 101, 138
 - built-in 199
- simple 88
- skeletons of 121
- style for 99
- testing 91
- unique powers of 106
- when to use 106
- see also: backquote, read-macros, symbol-macros
- mainframes 348
- make-dispatch-macro-character 226
- make-instance 365
- make-hash-table 65
- make-string 58
- map-> 54
- map0-n 54
- map1-n 54
- mapa-b 54, 228
- mapc 163
- mapcan 41, 46
 - nondestructive version 55
 - sketch of 55
- mapcar 13
 - version for multiple lists 55
 - version for trees 55
- mapcars 54
- mapcon 176, 218
- mappend 54
- mappend-mklist idiom 160
- mapping functions 53
- mark 301
- match 239
- matching—see pattern-matching
- maxmin 207
- Meehan, James R. 396
- member 88
 - misuse of 151
 - Prolog implementation 332
 - returns a cdr 50
- Miller, Molly M. 137
- member-if 196
- memq 88
- memoizing 65, 174
- message-passing 350
 - vs. Lisp syntax 353
- methods
 - adhere to one another 369
 - after- 374
 - sketch of 357
 - around- 375
 - sketch of 356
 - auxiliary 374
 - sketch of 356
 - before- 374
 - sketch of 357
 - of classes 368
 - without classes 371
 - as closures 378
 - redefining 372
 - removing 373
 - sketch of 359
 - isomorphic to slots 368
 - specialization of 369
 - on objects 371
 - on types 370
 - see also: generic functions
 - method combination
 - and
 - sketch of 363
 - operator 376
 - sketch of 362
 - or
 - sketch of 363
 - progn
 - sketch of 362
 - standard 376
 - sketch of 358
 - :method-combination 377
- Michelangelo 11
- mines 264
- mklist 45, 160
- mkstr 58

- modularity 167, 381, 382
- de Montaigne, Michel 2
- most 52
- most-of 182
- mostn 52
- moving parts 4
- multiple inheritance—see inheritance, multiple
- multiple values 32
 - to avoid side-effects 32
 - to distinguish failure from falsity 196, 239
 - in generalized variables 172
 - iteration with 158
 - receiving—see multiple-value-bind
 - returning—see values
- multiple-value-bind 32
 - leftover parameters nil 234
- multiprocessing 275
- mvdo 162
- mvdo* 159
- mvpsetq 161
- Mythical Man-Month, The* 5
- name-spaces 12, 205, 259, 273, 384, 392
- natural language—see ATNs
- nconc 31, 35, 137
- negation
 - of facts 249
 - in Prolog 325
 - in queries 252
- networks
 - representing 76, 79
- next-method-p 375
 - sketch of 358
- :nicknames 384
- nif 150
- nil
 - default block name 131
 - forbidden in case clauses 153
 - multiple roles of 51, 195
- nilf 169
- Nitzberg, Mark—see joke, practical
- nondeterministic choice 286
 - Common Lisp implementation 295
 - need for CPS macros 296
 - restrictions on 297
 - and tail-recursion optimization 298, 396
- Scheme implementation 293
- appearance of foresight 289
- breadth-first 303
- correct 302
- depth-first 292
 - in ATNs 308
 - nonterminating 293
 - in Prolog 334
- in functional programs 286
- vs. iteration 291, 325
- optimizing 298
- and parsing—see ATNs
- and search 290
- see also: *choose, fail*
- Norvig, Peter 199
- nreverse 31
 - sketch of 388
- nthmost 183
- object-oriented programming
 - dangers of 379
 - defining features of 350
 - like distributed systems 348
 - and extensibility 16, 379
 - name of 349
 - in plain Lisp 349
 - see also: C++; classes; CLOS; generic functions; inheritance; methods; message-passing; slots; Smalltalk
- on-cdrs 205
- on-trees 210
- open systems 6
- open-coding—see compilation, inline
- orthogonality 63
- *package** 125, 381
- packages 381
 - aberrations involving 384
 - avoiding capture with 130, 131
 - creating 382
 - current 381
 - using distinct 131, 382
 - inheriting symbols from 384

- nicknames for 384
 - switching 382
 - user 381
 - see also: `intern`; `interning`
 - parsers, nondeterministic—see `ATNs`
 - paths, traversing 155
 - `pat-match` 242
 - pattern-matching 186, 238
 - pattern variables 238
 - phrenology 30
 - planning 2
 - pointers 76
 - pools 313
 - `popn` 173
 - `pop-symbol` 220
 - position 49
 - `*print-array*` 245
 - `*print-circle*` 70
 - `print-names` 57, 129, 382
 - processes 275
 - instantiation of 278
 - scheduling of 279
 - state of 278
 - `proclaim` 23, 45
 - productivity 5
 - programming languages
 - battlefield of 8
 - embedded—see `embedded languages`
 - expressive power of vii
 - extensible 5
 - high-level 8
 - see also: `Algol`; `Basic`; `C`; `C++`; `Common Lisp`; `Fortran`; `Lisp`; `Prolog`; `Scheme`; `Smalltalk`; `T`
 - `Prolog` 321
 - assignment in 343
 - calling `Lisp` from 343
 - case-sensitivity of 331
 - conceptual ingredients 321
 - nondeterminism in 333
 - programming techniques 332
 - restrictions on variables 344
 - rules 329
 - bodyless 323, 330
 - implicit conjunction in `body` 328
 - left-recursive 334
 - order of 329
 - subverting 346
 - syntax of 331
 - promises 211
 - prompt 56
 - property lists 15, 63, 216
 - `propmacro` 216
 - alternative definition 393
 - `propmacros` 216
 - `prune` 47
 - simpler version 389
 - pruning search trees—see `cut`
 - `psetq` 96
 - multiple-valued version 161
 - `pull` 173, 223
 - `pull-if` 173
 - `push-nreverse` idiom 47
 - `pushnew` 174
- queries
- complex 249, 335
 - conditional 191
 - query languages 249
- quicksort 345
- quote 84, 391
- see also: `'`
- quoted lists, returning 37, 139
- rapid prototyping 1, 284
- of individual functions 24, 48
- `read` 56, 128, 197, 224
- `read-delimited-list` 227
- `:reader` 367
- `read-eval-print` loop 57
- `read-from-string` 58
- `read-line` 56
- `readlist` 56
- `read-macros` 224
- `recurser` 388
- recursion
- on `cdrs` 68, 204
 - in grammars 306
 - in macros 139, 192
 - without naming 388
 - on subtrees 70, 208
 - `tail-` 23, 140

- reduce 207, 363
- Rees, Jonathan A. 395, 396
- referential transparency 198
- remove-duplicates
 - sketch of 206
- remove-if 14
- remove-if-not 40
- rep_ 324
- reread 58
- &rest parameters 87
 - not guaranteed fresh 137
 - in utilities 174
- return 131, 155
- return-from 131, 154
- return values
 - functions as—see functions, as return values
 - multiple—see multiple values
- re-use of software 4
- reverse 30
- rfind-if 73, 210
 - alternate version 390
- rget 351
- rich countries 285
- rmapcar 54
- Rome 283
- rotatef 29
- rplaca 166
- rules
 - structure of 322
 - as virtual facts 323
 - see also: Prolog, rules in
- Scheme
 - vs. Common Lisp 259
 - cond 192
 - macros in 392
 - returning functions in 62
- scope 16, 62
- scoundrels, patriotic 352
- scrod 219
- search trees 265
- sequence operators 244
- series 55
- set 178
- set-difference 207
- setf 165
 - see also: generalized variables, inversions
- set-macro-character 224
- setq
 - destroys referential transparency 198
 - ok in expansions 100
 - now redundant 170
- Shapiro, Ehud 398
- sharp (#) 226
- shuffle 161
- side-effects 28
 - destroy locality 36
 - in macro expanders 136
 - mitigating 35
 - on &rest parameters 137
 - on quoted objects 37
- signum 86
- simple? 242
- single 45
- Sistine Chapel 11
- skeletons—see macros, skeletons of
- sketches 284
- sleep 65
- slots
 - accessor functions for 365
 - declaring 364
 - as global variables 379
 - initializing 365
 - isomorphic to methods 368
 - read-only 367
 - shared 367
- Smalltalk 350
- some
 - sketch of 206
- sort 14, 352
- sortf 176
- sorting
 - of arguments 176
 - partial 184
 - see also: stable-sort
- special 17
- special forms 9, 391
- specialization—see methods, specialization of
- speed 23

- splicing 86
- splines—see Bezier curves
- split-if 50
- sqrt 32
- squash 160
- stable-sort 352, 399
- stacks
 - allocation on 150
 - of ATN registers 312
 - in continuations 260, 261
 - use for iteration 264
 - overflow of 396
- Steele, Guy Lewis Jr. ix, 43, 213, 395, 396
- Sterling, Leon 398
- strings
 - building 57
 - matching 231, 244
 - as vectors 233
- Structure and Interpretation of Computer Programs* 18
- structured programming 100
- subseq 244
- superclasses
 - precedence of 369
 - sketch of 352
- Sussman, Gerald Jay 18, 395
- symb 58
- symbols
 - building 57
 - as data 385
 - exported 383
 - imported 383
 - interning—see intern
 - names of 57, 129, 382
 - see also: keywords
- symbol-function 12, 388
- symbolic computation 398
- symbol-macrolet 105, 205, 210
- symbol-name 58
- symbol-package 381
- symbol-value 12, 178
- symbol-macros 105, 205, 236, 237
- swapping values 29

- T 396

- tagbody 155
- tail-recursion optimization 22
 - needed with CPS macros 298
 - testing for 388, 396
- taxable operators 32
- testing
 - incremental 37
 - of macros—see macros, testing
- \TeX vi, 5
- tf 169
- Theodebert 236
- three-valued logic 151
- till 154
- time 65, 359
- times of evaluation 224, 229
- toggle 169
- top-down design 3
- trace 111, 266, 309
- transition networks 306
- transformation
 - embedded languages implemented by 116, 241
 - of macro arguments 107, 112
- trec 75
- trees 70, 262
 - cross-products of 265
 - leaves of 72
 - recursers on 70
- true-choose
 - breadth-first version 304
 - T implementation 396
 - depth-first version 396
- truncate 32
- ttrav 74
- Turing Machines vii
- twenty questions 77
- typecase 62
- type-of 371
- typep 243
- types
 - declaration of 23
 - specialization on 370
- typing 44, 112

- undefmethod 373
- unification 394

- union 206
 - unspecified order of result 207, 364
- unions 207
- unspecialized parameters 373
- unwind-protect 148
- :use 384
- user 381
- utilities 40
 - as an investment 43, 392
 - become languages 113
 - mistaken argument against 59

- var? 239
- variable capture—see capture
- variables
 - bound 16
 - free 16, 121
 - generalized—see generalized variables
 - global 36, 125, 268, 379
- varsym? 239
 - redefined 335
- vectors
 - for ATN registers 313
 - creating with backquote 86
 - matching 231, 244
- visual aspects of source code 30, 213, 231
- voussoirs 8
- values 32
 - inversion for 393
- =values 267

- wait 280
- Wand, Mitchell 395
- Weicker, Jacqueline J. x
- when-bind 145
- when-bind* 145
- while 154
- with-answer 251
 - redefined 255
- with-array 234
- with-gensyms 145
- with-inference 324
 - redefined 335, 340
- with-matrix 234
- with-open-file 147
- with-output-to-string 58
- with-places 237
- with-slots 236
- with-struct 235
- writer's cramp 44
- &whole 95
- Woods, William A. 305
- workstations 348
- world, ideal 109

- X Windows vi, 5

- zebra benchmark 396

- #' 10, 226
- #(233
- #. 75
- #: 128
- #! 226
- #[227
- #\ 233
- #{ 229
- ' 225
 - see also: quote
- , 84
- ,@ 86, 138
- : 383
- :: 382
- @ 294
- _ 240, 252, 328
- ` see backquote
- | 58