

index

Symbols

- ! (negation operator) 39
- \$ (dollar) (AWK)
 - only used for field variables 126
- \$ (dollar) (Shell)
 - differences from Perl in use
 - with variable names 23, 25
- \$ (dollar) metacharacter 63
 - altered meaning with /m modifier 69
- \$ (dollar), as identifying mark for scalar variable 23
- #! (OS error-number) variable 341, 428
 - setting to define die's exit value 253, 292
- \$\ (output record separator) variable 42, 127
 - used for multi-character record separators 42
 - See also* output record separator
- \$" (string formatting) variable 43, 300–301, 435
- \$# (argument count) variable (Shell) 334
- \$#ARGV (maximum index) variable for @ARGV array 346
- \$\$ (process-ID) variable 112
- \$& (match) variable 63, 127
 - used in replacement field of substitution operator 95
 - used to display the match only 64
- \$' (post-match) variable 127
- \$() (Shell). *See* command substitution
- \$, (print's formatting) variable 43, 435
 - used with grep 228–229
 - used with map 233
- \$. (record number) variable 20
 - contains number of last record in END block 264
 - used with substitution operator for record-specific substitutions 97
- \$/ (input record separator) variable 42, 127, 307
 - need to reset in dual input-mode scripts 308
 - used for multicharacter record separators 42
 - See also* input record separator
- \$? (command error) variable 257, 428
- \$@ (eval error) variable 283–284
- \$^I (in-place edit) variable 429
 - used in clobber-proofing editing scripts 112
- \$_ (data) variable 126, 427
 - as default argument for file-test operators 181
 - introduction to 24
 - may get masked in nested loops 373
- \$` (pre-match) variable 127
- \$0 (record) variable (AWK) 126
- \$0 (script name) variable
 - used to show source of diagnostic message 36
 - used with warn and die 36
- \$1, etc. (field) variables (AWK) 126
- \$1, etc. (numbered) variables 71, 359
 - used in sed-like commands 99
- \$1, etc. (positional parameter) variable (Shell) 334
 - compared to \$ARGV[0] in Perl 334
 - See also* positional parameters
- \$a (sort item) variable 429
 - used in sort's coding rules 225
- \$ARGV (filename) variable 42, 67, 127, 146, 191
 - definition of 42
- \$b (sort item) variable 429
 - used in sort's coding rules 225
- \$F[n] (field) value 126

- \$SECONDS (Shell) variable, used in clobber-proofing editing commands 112
- % (modulus) operator 159, 405
- %ENV (environment) hash 313–314
 - as inter-process communication mechanism 313
 - using in place of switches 315–316
- && (AWK) as similar to Perl's logical and 140
- && (Shell), as similar to Perl's logical and 35
- () (parentheses) as metacharacters 71, 359
- * (multiplication) operator 159
- * (star) metacharacter 73
 - vs. meaning of * in FNG 237
- * (star) metacharacter (FNG) 236, 259
- + (addition) operator 159
- + (plus) metacharacter 73, 109
- ++ (increment) operator 159, 346
- (range in character class) metacharacter (FNG) 236
- (range in character class) metacharacter 63
- (subtraction) operator 159
- (decrement) operator 159
- . (dot) metacharacter 63
 - becomes literal character in character class 303
- . (string concatenation) operator 265–268
 - syntax and examples 265
 - tips on using 268–269
 - used in enhancement to `most_recent_file` 267
- .. (range) operator, AWK-like version 152–153
 - used in array indexing 301
- ... (range) operator, sed-like version 152–153
 - used for logfile analysis 170
- .= (string concatenation) compound assignment operator 265–267
 - common errors with 269
 - used to incrementally build HTML document 419
- / (division) operator 159
- / (slash) 438
 - See also* slash; backslash
- // (matching) operator 60
 - capabilities of 58
 - introduction to 60–62
 - modifiers for 68–70
 - when to use `split` instead 213
- /e (eval) substitution modifier 95
 - for computed replacements 114–118
 - for function-generated replacements 116
- /g (global) match modifier 69, 360
- /g (global) substitution modifier 95
- /i (ignore case) match modifier 69, 95, 187
- /m (multi-line mode) match modifier 69, 95, 359
- /s (single-line mode) match modifier 69, 95
- /x (expanded format) match modifier 69, 95, 110
- : (null) command (Shell)
 - used in bottom-tested `while/until` loops 338–339, 352
- < > (globbing) operator 234–239
- < > (input) operator
 - used in implicit loop 337
- <=> (numeric comparison) operator 157
 - used with `sort` 224
- <> (globbing) operator
 - syntax and examples 235
 - tips on using 237–238
- <ENTER> key
 - as needed but generally unshown in terminal sessions xxviii
 - definition 434
- <SPACE> key, definition 438
- <TAB> key, definition 439
- ~ (match-binding) operator 60, 69, 96
- ? (question mark)
 - metacharacter 99, 238
 - for stingy matching 124
 - for stingy matching, advantages over AWK 125
 - in Perl regex, for stingy matching 73
- ? (question mark) metacharacter (FNG) 236
- @_ (sub argument) array 364
- @ARGV (argument) array 42, 127
- @EXPORT (default exports) module array 391
 - when to use vs. `@EXPORT_OK` 391
- @EXPORT_OK (on-request exports) module array 391
 - when to use vs. `@EXPORT` 391
- @F (field) array 127, 131–132, 141, 145
 - when to access using indexing vs. assignment to variable list 145
 - See also* \$F[n]
- @INC (include) array, relation to `PERL5LIB` variable 397
- [] (character class) metacharacter (FNG) 63, 236
- [!] (complemented character class) metacharacter (FNG) 236
- [^] (complemented character class) metacharacter 63

\ (backslash) (Shell)
 as line-continuation character xxx
 differences from Perl in effects of 31
 See also backslash; slash
 \ (backslash) definition 432
 \ (backslash) metacharacter 63
 See also backslash; slash
 \& (function-address) operator 411
 \ (\) backslash (backslashed parentheses) metacharacters 71
 \{ \} (backslashed braces) metacharacters 73
 \040 (space) string escape 105
 \047 (single quote) string escape used as nested single quote in commands 62
 \1, etc. (backslashed-number) metacharacter 71
 \b (word-boundary) metacharacter 63
 \d (digit) metacharacter 67, 115
 \D (non-digit) metacharacter 67
 \E (end-modification) string modifier 113, 289
 cases where it can be omitted 227
 See also \U; \L
 \E (end-quoting) metacharacter 63
 See also \Q
 \L, \l (lowercase) string modifiers used to achieve case insensitivity 113, 227, 322
 \Q (start-quoting) metacharacter 63
 See also \E
 \S (non-whitespace) metacharacter 67
 \s (whitespace) metacharacter 67
 \U, \u (uppercase) string modifiers 113, 146, 348
 See also \E

\W (non-word) metacharacter 67, 324
 \w (word) metacharacter 67
 ^ (caret) metacharacter 63
 altered meaning with /m modifier 69
 _ (underscore)
 as reference to previously tested file 219
 __PACKAGE__ (package name) keyword 395
 ` ` (Perl). *See* command interpolation
 ` ` (Shell). *See* command substitution
 { } (braces) metacharacter 73
 | (vertical bar) metacharacter 71

Numeric Options

-00 (paragraph-mode) option 45
 special setting of \$/ as equivalent 428
 used in field processing 146
 -0777 (file-mode) option 45
 special setting of \$/ as equivalent 428
 -0*digits* (input record separator) option 17, 22, 428
 only used for single-character record separators 42
 used to print records by number 101
 See also \$/ (input record separator) variable

A

a2p command
 for translating AWK to Perl 175
 tips on using 175
 address of function. *See* \& (function-address) operator
 Alexis, featured character for PFD project 388

aliases for Perl commands
 perl_o, perl_io, etc. 46
 amatch function
 for approximate (“fuzzy”) matching 85
 See also String::Approx
 analyzing log files 81
 Andy “yDNA” Sweger.
 See Sweger, Andy “yDNA”
 ARGV (argument count) variable (AWK) 126
 argument generation 259
 generating all readable, regular files 259
 argument pre-processing 256–259
 filtering arguments 257–258
 removing names of non-text files 258
 removing non-filename arguments in BEGIN block 256–257
 sorting arguments 257
 argument processing
 dealing with multi-word filenames 196–197
 filtering out binary files 189–191
 reporting names of non-text files 258
 using Perl to filter-out undesirable ones 188–192
 argument, definition 432
 ARGV (argument vector) variable (AWK) 42, 126
 ARGV filehandle 42
 definition of 43
 arithmetic operators
 comparing AWK’s and Perl’s 158–159
 array indexing
 syntax 145
 arrays 296–308
 illustration of storage in memory 297

- arrays (*continued*)
 - indexing techniques 300–304
 - indexing with random numbers 304–308
 - initialization methods 299–300
 - initialization using push 299
 - piecemeal initialization 299
 - Shell vs. Perl syntax comparison 298
 - slices: better name would be “index groups” 301
 - syntax for indexing 300
 - syntax for slice indexing 301
 - tips on using 308
- ASCII 103
- Ashanti, featured character for line-specific substitution command 98
- associative array (AWK)
 - is like a Perl hash 296
 - See also* hashes
- automatic line-end processing. *See* -l (in-place editing) option
- automatic looping. *See* -n (automatic input-reading) option; -p (automatic input-reading, with printing)
- automatic printing. *See* -p (automatic input-reading, with printing) option
- AWK
 - advantages vs. Perl 130
 - books on 123
 - definition 432
 - effects of delayed documentation on popularity 123, 162
 - features compared to Perl 123–130
 - flavors of 123
 - functions and Perl equivalents 161
 - functions, list of 160
 - has a variable Perl doesn’t have 123
 - history of 122–123
 - introduction to 121
 - special variables 126–128
 - summary of differences with Perl 129–130
 - See also* awk command; GAWK; NAWK; POSIX AWK
- awk
 - definition 432
 - GNU version, defined 433
- AWK commands. *See* commands (AWK)
- AWK scripts. *See* scripts
- AWKish, definition 432
- B**
- B (binary) file test operator 182, 187
- b (block) file test operator 182
- B. B. King, on KISS principle in music 8
- backslash, definition 432
 - See also* \ (backslash); slash
- Bali, web-scraping for travel tips about 81
- BEGIN block 338
 - as place to validate a script’s arguments 40
 - equivalent coding for use when implicit loop not used 338
 - introduction to 39
 - used to validate a script’s arguments 109
- Bell Labs rookie, featured character for `nexpr` 163
- Bell Labs veteran, featured character for `nexpr` xxvi, 163
- Bell System 432
- Benchmark module 315
- Boulder, Fox, featured character for rock-star biodata system 133–138
- break (loop control) directive (Shell) 349
 - compared to Perl’s last 349
- Brian “Ingy” Ingerson. *See* Ingerson, Brian “Ingy”
- bugs, Column of Ones 234
- Business::UPS module 398–405
- C**
- c (character) file test operator 182
- c (check-syntax) option 395
- C language
 - approaches to Perl programming 13
 - breeds mistrust in its programmers 7, 349
- C language refugees
 - are understandably phobic about uninitialized variables 317
- C shell
 - excluded from coverage xxvii
 - fundamental differences from Shell xxiv
- Camel book, definition 433
 - See also* glossary definition
- `carp` function
 - advantages over `warn` in modules 394–395
 - is provided by `Carp` module 391
- `Carp` module 391
 - See also* `carp` function; `croak` function
- case conversions 113–114
- `cat` command
 - emulating with Perl command 12, 24
 - emulating with Perl script 29
- `cd` command, Perl counterpart is `chdir` 239
- cell processing. *See* field processing, extracting data from tables

- Center module 393–395
 - CGI module 231, 414–422
 - :standard tag 416
 - cheatsheet for essential functions 422
 - cheatsheet for form-related functions 423
 - Dump function of 418
 - provides functions named after HTML tags 416
 - tips on using 421
 - See also* CGI programming
 - CGI programming
 - dual-mode programs 418–419
 - introduction to 414
 - testing programs without a browser 419
 - w option not used 416
 - CGI::Carp module
 - provides a CGI version of the carp function 416
 - character sets 104
 - chdir function, Unix counterpart is cd 239
 - check_links script 405
 - check_symlinks script 411, 414
 - chgrp command, related to Perl’s chown 240
 - chmod function 240
 - Choi, Dora 347
 - chomp function 219–221
 - example of need for 220, 339, 357
 - requires parentheses around multiple arguments 221
 - syntax and examples 220
 - tips on using 221
 - Unix relatives 210
 - used on find’s output 274
 - vs. using -l option 146
 - chown function 240
 - circle forehead marking. *See* forehead markings of Perlstanis
 - classic AWK 123, 126–127, 164
 - classic grep 54, 91
 - classic sed 91, 93
 - classic UNIX utilities
 - definition 433
 - clobberation of variables. *See* variable clobberations
 - clobberation, definition 433
 - clobbering variables. *See* variable clobberations
 - close function 68
 - cmp (string comparison)
 - operator 157
 - used with sort 224
 - code block
 - \$_ as final statement in, with map 234
 - used with grep 228
 - used with map 233
 - code blocks
 - coding rules for sort 225
 - used with sort 224
 - code snippets
 - allowing zero tip for waiter, using defined function 250
 - emulating env command using %ENV hash 314
 - for avoiding variable masking in nested loops 373
 - for conditionally appending one of two strings to another 266
 - for ensuring confirmation from interactive user 352
 - for exiting script if no arguments 249
 - for invoking command on different files using for loop 336
 - for obtaining confirmation from interactive user 339–340
 - for printing environment variables in sorted order 342
 - for removing non-text filenames from arguments 258
 - for responding to environment variables 315
 - for retrieving a phone number from a hash 309
 - for selecting a filename from a menu 355
 - for sorting arguments 257
 - for summing numeric arguments 334
 - for using names of all readable, regular files as arguments 259
 - make_money_fast, demonstrates repetition and compound-assignment concatenation operators 267
 - providing default argument for script 260
 - reporting names of non-text files 258
 - requiring non-zero tip for waiter 250
 - using local declaration to make temporary change to “\$,” variable 383
- coding conventions used in this book xxxi
 - col command 331
 - doesn’t take filename arguments 331
 - Colin “Shroomy” Meyer. *See* Meyer, Colin “Shroomy”
 - command interpolation 189, 201, 269–275
 - compared to command substitution 270
 - doesn’t work within quotes 274
 - may require multi-level quoting 357
 - returns output—not exit code, as Shell does 275
 - tips on using 274–275
 - See also* command substitution

- command substitution 189, 192, 269–275
 - alternative \$(command) syntax 269
 - compared to command interpolation 270
 - returns exit code—not output, as Perl does 275
 - used to provide validated arguments to grep 189*See also* command interpolation
- commands (AWK)
 - for printing rock-star birthdays 130
 - Perl equivalents for simple tasks 141
 - simple ones compared to Perl equivalents 128
- commands (Perl)
 - a one-line grepper 61
 - a one-line sed command 93
 - differences from scripts 11
 - emulating date command 214
 - emulating Shell’s `-nt` file test operator 217
 - `fix_newsletter` 104
 - for calculating space allowance for each file in collection 335
 - for data validation 66
 - for editing files 105–107
 - for extracting “File doesn’t exist” errors from logfile 152
 - for extracting POD documentation from scripts 156
 - for filtering find’s output 183
 - for finding JPEG-oriented scripts 187
 - for generating a random number 221–222
 - for line-specific substitutions 96–97
 - for matching lines between specified days 152
 - for numbering paragraphs 150
 - for printing “Hello, world!”, down-under version 49
 - for printing first 80 characters of each line 161
 - for printing rock-star biodata 134
 - for printing rock-star birthdays 133
 - for printing square roots 161
 - for processing multi-word filenames 197
 - for record-specific substitutions 97–98
 - for web-scraping slashdot.org 86
 - like `scan4oops` script 174
 - like `scan4oops` script, Felix’s scathing review 174
 - `make_meeting_page` 102
 - matching a range of dates 154
 - of AWKish Pattern/Action type 139
 - one-line 11, 14, 18
 - printing lines by number 100
 - printing records by number 101
 - using field processing and in-place editing 134
 - with nesting of single quotes 62*See also* `-e` (code) option
- commands, definition 433
- commas
 - permitted after last aggregate initializer for list variable 311
- comments xxix
- common mistakes 111
- comparison operator
 - two forms of 158
- comparison operators 157
 - See also* `cmp` operator; `<=>` operator
- compartmentalize
 - as preferred term to modularize when discussing subroutines 363
- compound assignment operators 140, 158–159, 348
- construct, definition 434
- Consultix xxi
- context sensitivity
 - advantages of 206
- continue (loop control) directive (Shell) 349
 - compared to Perl’s `next` 349
- continue blocks 350, 353–355
 - give Perl loops an advantage over Shell loops 355
- control characters 104
 - string escapes for 56
- convert command (ImageMagick) 335
- Conway, Damian xx, xxii, 347
 - as author of *Perl Best Practices* book xxxii
 - as winner of Larry Wall award 28
 - his foreword to this book xvii
 - See also* *Object Oriented Perl* book; *Perl Best Practices* book
- copy function, related to Unix `cp` 240
- `cp` command, related to Perl’s `copy` 240
- CPAN
 - is the envy of non-Perl programmers 425
 - link to advice on searching for modules 425
 - searching to find modules 398
- CPAN module 401
 - using the shell function 401
- CPANPLUS module
 - relation to CPAN module 401

- creating modules
 - checking for warnings and syntax errors 395
 - defining subroutines 392
 - specifying exports 391–392
 - testing with a command 397
 - testing with a script 396
 - using “1;” as last line 392
 - with the Simple Module Template 390–392
 - croak function
 - provided by Carp module 394
 - vs. use of die in modules 394
 - Crocodile Hunter 49
 - cut command, Perl relatives 210
- D**
- d (directory) file test operator 182
 - Dan Sugalski. *See* Sugalski, Dan
 - date command, Perl counterpart is localtime 210
 - debugging strategies, adding/subtracting another backslash 376
 - default loop variables, are automatically declared with local 373, 383
 - defined function 249–252
 - advantages over testing values for True/False 249
 - needed for testing values before using them 251
 - tips on using 252
 - used for identifying empty arguments 249
 - used for validating input from keyboard 250–252
 - used to test for <^D> from keyboard 251, 286
 - delete function
 - used on array elements 298, 307
 - used on hash elements 310–311
 - df command 248
 - diamond operator. *See* input operator
 - die function
 - introduction to 35
 - preferred to warn and exit, outside BEGIN blocks 253
 - Diggity Dog
 - featured character for 4letter_word xxv, 372
 - featured character for ups_shipping_price 404
 - directive, definition 434
 - directory management functions 239
 - do until loop. *See* do while/until loop
 - do while/until loop 338–340
 - arranging for working loop control directives in bottom-tested loops 351–352
 - compared to Shell equivalent 338
 - doesn't respond to loop-control directives 340
 - tips on using 339
 - Don, featured character searching for epistle2dippy.txt 185
 - Dora Choi. *See* Choi, Dora
 - double quotes
 - for list to scalar conversion 209
 - link to article on Shell vs. Perl differences 298
 - permit command substitution but not command interpolation 274
 - used to suppress secondary substitutions in Shell 325
 - double-quoted string
 - used as output-template for print function 25
 - downloading the source code of this book's examples xxxii
 - Dump function
 - of CGI module 418
 - DWIMity 191, 206
- E**
- e (code) option 11, 17, 45
 - unused on Perl shebang line 30
 - E0 (end of range) marker 153
 - used in scan4oops script 171
 - egrep command
 - improving on with Perl 70–72
 - Perl relatives 223
 - egrep, GNU version, defined 433
 - elif keyword (Shell)
 - Perl counterpart is elsif 261
 - elsif keyword
 - Shell counterpart is elif 261
 - See also* if/else
 - END block 338
 - equivalent coding for use when implicit loop not used 338
 - introduction to 39
 - used for calculating average of input data 264
 - used to print final report 324
 - English module
 - for using AWK variables in Perl 126
 - entertainment value of this book xxv–xxvi
 - env command
 - emulating with help from %ENV 314
 - relation to %ENV hash 314
 - eof function 307, 322
 - eq (string equality) operator 157, 340
 - errata page for this book xxxii
 - error messages
 - BEGIN not safe after errors—compilation aborted 369

- error messages (*continued*)
 - Can't locate
 - Some_Module.pm 397, 401
 - Can't open (some_file): No such file or directory 256
 - Execution of some_script aborted due to compilation errors 374
 - See also* warning messages
- eval function 283–292
 - as used in preg 290
 - examples and Shell vs. Perl comparison 283
 - for handling user-supplied array indices 304
 - need for 283
 - security concerns for evaluating user-submitted text 304
 - why it's necessary 304
- evaluation context
 - effects of 207–208
 - how programmer controls 207–208
 - making use of 208–210
 - understanding and managing 206–210
- execution-trace mode (Shell) 282
 - using to debug commands submitted by Perl 282
- exists function 322
 - used on hash elements 310
- exit function 253–254
 - examples and Shell vs. Perl comparison 253
 - meanings of numerical exit values 253
- expand function (Text::Tabs) 366
- explicit list, definition 434
- export command
 - used to convey switch-like variables to scripts 315
 - used with setting of PERL5LIB variable 397
- Exporter module
 - as basis for creating a new module 389
- expr command
 - shortcomings of 162
 - See also* nexpr; expr_p expression, definition 434
- F**
 - F (custom field separator) option 136–138
 - f (regular) file test operator 182
 - False value
 - of Shell, converting to Perl's False 275
 - Shell and Perl definitions of 32
 - False, definition 434
 - fatalToBrowser
 - requests that error messages be displayed by web browser 417
 - See also* CGI::Carp
 - featured character
 - Alexis, for PFD project 388
 - Ashanti, for line-specific substitution command 98
 - Bell Labs rookie, for nexpr 163
 - Bell Labs veteran, for nexpr xxvi, 163
 - Diggity Dog, for 4letter_word xxv, 372
 - Diggity Dog, for ups_shipping_price 404
 - Don, searching for misplaced file 185
 - Felix, for scan4oops xxv, 168–175
 - Fox Boulder, for rock-star bio-data system 133–138
 - Gabriella, for expand_acronyms 343
 - Guillermo, for city rainfall comparison project 143–148
 - Ivan, for compress_image* scripts xxvi, 335–336, 344
 - Martina, for Apache logfile analysis 151–154
 - Murray, for scan4oops project 175
 - Oscar, for scan4oops-like command 174–175
 - Patrick, for city rainfall comparison project xxvi, 143–151
 - PerlDude, for find2perl examples 200
 - Ramon, for check_length 116
 - Steffi, having “lingering thumb” 186
 - TVM guy, rock-star biodata system 137–138
 - Vitas, for city rainfall comparison project 143–149
 - WinDude, for find2perl examples 200
 - Yoko, for fuzzy_match xxvi
 - Felix, featured character for scan4oops xxv, 168–175
 - fgrep command
 - improving on with Perl 64
 - Perl relatives 223
 - fgrep, GNU version, defined 433
 - field processing 130–138
 - accessing fields with Perl 131–132
 - extracting data from tables 143–151
 - introduction to 130–132
 - printing fields with Perl 132–133
 - using undef in 131
 - with AWK 90
 - See also* table processing
 - field separators
 - customizing 136–138
 - See also* -F (custom field separator) option

- file management functions
 - 239–241
 - tips on using 241
- file mode, enabling by assignment to \$/ 428
- file test operators
 - \$_ as default argument 181
 - comparing find and Perl 180–184
 - syntax 181
- file tests
 - comparison of find vs. Perl 181
 - Perl better than find for permissions 183
- File::Copy module 240
- File::Find module 411–414
 - relation to find command 180, 201
- filehandles, using with close function 68
- file-management functions
 - tips on using 241
- FILENAME (filename)
 - variable (AWK) 42, 126
- filename generation 235
 - See also* FNG
- filename generation operator.
 - See* < > operator
- filter programs 12
 - basics of implementing 12–14
 - cascading filters, using grep/egrep 72
 - using one Perl command vs. multiple grep/egrep commands 75
- financial calculations
 - compound interest 165–167
 - compound interest2 166–167
 - Rule of 72 for estimating investment growth through compounding 165
- find | xargs, problems with multi-word filenames 196
- find command
 - as aid for finding misplaced files 179
 - as argument pre-processor for Perl 197–198
 - as emulated by find2perl 198–200
 - augmenting with Perl 183–184
 - enhancing by adding Perl command 179
 - exec option 192
 - exec's deficiencies vs. xargs alternative 193
 - filtering output of with Perl 183
 - for finding recently modified scripts 179
 - use of -follow option 274
 - used with xargs 192–197
- find function (from File::Find) 411
 - requires special handling 412
- find, GNU version, defined 433
- find2perl command
 - as OS-portable alternative to find 198–200
 - supports -exec rm 200
- finding files
 - by name matching 184–187
 - by pathname matching 187–188
 - having multi-word names 186–187
- fmt command 28, 280
 - emulating with Perl command 28
 - improving on with Text::Autoformat 28
 - See also* Text::Autoformat
- FNG (filename generation) notation
 - corresponding metacharacters from regex notation 238
- FNR (file-specific record-number) variable (AWK) 126
- footnotes, authors philosophy of
 - using xxv
- for loop 345–349
 - good for index-oriented array processing 345
- for loop (Perl)
 - syntax of 345
- for loop (Shell) 331
 - compared to Perl's foreach 340
- foreach loop 340–344
 - compared to Shell's for loop 340
 - preferred to for loop for list processing 346
 - preferred to while/until loop for list processing 334
 - similarities to Perl's select loop 356
 - used in emulating env command 314
 - used with unlink to get file-specific error messages 341
- forehead markings of Perlstanis 6–7
- format_mode function,
 - for converting permissions strings 218
- formatting variables
 - introduction to 43–44
 - See also* '\$'" variable; '\$,' variable
- Fox Boulder, featured character
 - for rock-star biodata system 133–138
- Fox Boulder, featured character
 - for rock-star programs alien conspiracy theories of 137
- Free Software Foundation 435
- FS (input field-separator) variable (AWK) 126
- function calls, compared with method calls in CGI
 - example 423

functions
 data flows backwards vs. Unix pipelines 223
 definition 434
 for directory management 239
 for file 241
 for file management 239–241
 for list processing 223–234
 having multi-valued return codes 240–241
 in series are processed from right to left 208
 Shell vs. Perl functions 364
 using built-in ones 159–164
 vs. methods 400
 functions of AWK, list of 160
 functions of Perl, list of 160

G

-g (set-GID) file test operator 182
 Gabriella, featured character for `expand_acronyms` 343
 gawk 432
 GAWK, capabilities of 125
See also gawk command; AWK; NAWK; POSIX AWK
 ge (string greater-than or equal-to) operator 157
 getgrgid function 218
 getpwuid function 218
 global variables 367
 accessing by simple vs. explicit package names 374–375
 globbing operator. *See* < > operator
 GNU AWK 123–124
See also GAWK
 GNU find 180, 182
 GNU grep 273
 capabilities 58
 GNU sed 92–93, 107
 capabilities 91

GNU utilities versions used in this book, defined 433
 GNU, definition 435
 grep command
 capabilities compared to Perl's 58
 diversity of regex dialects for 55
 emulating -i option with Perl 70
 emulating -l option with Perl 67–68
 emulating -v option with Perl 65
 for validating data 66
 history of 53
 origin of name 54
 Perl equivalents for common cases 87
 Perl relatives 223
 relationship to ed command 54
 screen corruption resulting from binary matches 188, 190
 shortcomings of 54–60
 grep function 227–229
 as "gating" mechanism 227
 compared to map 232
 differences from grep command 227
 syntax and examples 228
 Unix relatives 223
 grep, GNU version, defined 433
 grepper
 capabilities compared to Perl's 57
 definition 435
 grepping, definition 435
 gt (string greater-than) operator 157
 guidelines, for parenthesizing code 430–431
 Guillermo, featured character for city rainfall comparison project 143–148

H

hashes 296
 advantages of printing with `foreach over while/each` 343
 advantages of printing with `while/each over foreach` 337
 aggregate initialization 311
 as basis for simple database systems 319–322
 automatic initialization to zero in numeric context 317
 illustration of storage in memory 309
 indexing syntax 312
 indexing techniques 312–313
 initialization methods 311–312
 introduction to 308–311
 piecemeal initialization 311
 preferred to arrays for associating strings with values 310
 printing techniques 314–315
 printing with `foreach` 342–343
 printing with `while/each` 336–337
 slice-indexing a hash using an array 313
 slices used to impose ordering on retrieved values 312
 summary of indexing methods 313
 syntax and examples 310
 tips on initializing 311
 used as unique-ifiers 316–319
 used for counting word frequencies 323–325
 used to reduce list to unique elements 316
 head command
 as poorly suited for printing specific lines 100

head function (from
LWP::Simple) 405–406
hickory ruler
as motivator for grammatical
correctness 166
HTML
generating with CGI
module 419–422
listing of output generated by
survey.cgi 420

I

-i (in-place editing) option 45
backing up original file
111–113
editing pantaloony file 106
for clobber-proofing editing
commands 112
for clobber-proofing editing
scripts 112
for mass editing of HTML
files 107
pitfalls of using .bak
extension 111
used in AWKish
commands 134
using .bak extension 109
if/else 259–265
advantages over logical and/
or 260–261
comparison of Shell and Perl
syntaxes 260
curly braces can't be
omitted 265
elsif keyword 261
mixing with and/or 261–264
nesting of 261
tips on using 264–265
IFS (internal field separators)
variable (Shell) 325, 327,
341
compared to split
function 211
Perl relatives 210
implicit loop 12
compared to equivalent explic-
it loop 338
how it works 337–338
LINE is label for 338
See also -n (automatic input-
reading) option; -p (auto-
matic input-reading, with
printing)
Ingerson, Brian “Ingy” 102, 347
inode (Unix), accessing with
stat 215
in-place editing 109–113
input operator 13
definition 435
input record separator
automatic stripping of with -l
option 20
definition of 20
example of 42
how to change 22
newline as default 20
See also \$/ (input record sepa-
rator) variable
input/output variables (\$/, \$\\)
introduction to 42–43
installing modules
configuring the CPAN
module 401–402
determining which ones you
have 400
using non-root
privileges 402–403
using root privileges 401–402
invocation options
advice on ordering 44, 47
effects of 17
for automatic input process-
ing. *See* -n (automatic input-
reading) option; -p (auto-
matic input-reading, with
printing) option
for automatic line-end process-
ing. *See* -l (line-end process-
ing) option

for automatic switch handling.
See -s (switch) option
for loading modules. *See* -M
(module-loading) option
for one-line commands. *See* -e
(code) option
for warning messages. *See* -w
(warnings) option
introduction to 17–23
option clusters 17
standard option clusters 17

Ivan

featured character for
compress_image*
scripts 335–336, 344
featured character for com-
pression of stamp
collection xxvi

J

jalebi (confection) 66
JAPH
as resident of Perlstan 5
definition 435
JAPHly
definition 435
join function 229–232
equivalent to using \$" with
double quotes 210
for list to scalar
conversion 209
syntax and examples 229
tips on using 230
Unix relatives 223
used in constructing HTML
code 232
used in constructing passwd
file entry 230

K

-k (sticky) file test operator 182
keys function, retrieves all keys
from a hash 310

- King, B. B., on KISS principle in music 8
- ksymoops command (Linux) 171
- L**
- l (line-end processing)
 - option 11, 17, 20
 - doesn't affect input read from <STDIN> 339
 - programs that omit it 226
 - using instead of "\n" 21
- l (symlink) file test operator 182
- Larry, definition. *See* Wall, Larry
- last (loop control) directive 349
 - compared to Shell's break 349
- Laziness
 - as a virtue 9
 - in programming practice 12, 15
- le (string less-than or equal-to) operator 157
- Leaning Toothpick Syndrome 60, 92
- length function 116
- Library for Web Programming. *See* LWP::Simple
- LINE, is label for implicit loop 338
- Lingua::EN::Inflect module
 - used to conditionally pluralize nouns 167
- Lingua::EN::Namegame 372
- list context
 - effects of 207
 - ways to request 207
- list generators
 - Shell vs. Perl techniques 325–327
 - Shell's command substitution vs. Perl's command interpolation 327
 - Shell's FNG vs. Perl's globbing 326
 - Shell's variable substitution vs. Perl's variable interpolation 327
- list to scalar conversion
 - method can be specified by programmer 208
 - needed for matching against arrays 230
 - tools for 209–210
- list variables
 - introduction to 295–296
 - See also* arrays; hashes
- List::Util module 329
 - for shuffling list elements 227
- listfile script 217
- local declaration
 - arranges for the previous value of a variable to be restored later 374
 - introduction to 375
 - is automatically used for default loop variables 373, 375
 - syntax and examples 374
 - used for temporary changes to "\$," and "\$"" 383
 - used with "\$," and "\$"" 44
 - See also* variable declarations
- localtime function 214–215
 - syntax and examples 214
 - tips on using 215
 - Unix relatives 210
- logfile analysis 81, 168–175
 - of Linux /var/log/messages file 168
- logical operators
 - introduction to and/or 37
 - Perl's compared to Shell counterparts 37
- longest-anything pattern 78, 109, 114, 117, 359
- longest-something pattern 78
- loop control directives
 - Perl syntax example 351
 - Perl's better than Shell's for nested loops 351
- Shell syntax example 350
- Shell vs. Perl comparison 349
- looping
 - comparison of Shell vs. Perl resources 331, 333
 - introduction to 330–331
 - Shell vs. Perl differences in concepts, terms, and basic syntax 332
- loops
 - basic types of 332
 - nesting within others 350–351
 - using loop control directives 349–355
- ls command
 - emulating in Perl using stat 217–218
 - Perl relatives 210
 - See also* listfile script
- lt (string less-than) operator 157
- LWP::Simple module 405–408
- lwp-request command 80–81, 405–406
 - used for web-scraping 323
- M**
- M (module-loading) option 27, 45
 - used only with commands, not scripts 41
 - used to enable strict mode 369
 - using to test availability of module 400
- m// (matching) operator 60
 - using custom delimiters with 109, 188, 213
 - See also* // (matching) operator
- Main (program segment)
 - definition of 379, 435
 - See also* scoping
- managing modules 398–403
- map function 232–234
 - compared to grep 232

- map function (*continued*)
 - syntax and examples 233
 - Unix relatives 223
 - used in emulating env
 - command 314
 - Martina, featured character for
 - Apache logfile
 - analysis 151–154
 - masking of variables. *See* variable masking
 - matching
 - across lines, step-by-step guide 79
 - against arrays 230
 - against files 77
 - against paragraphs 75
 - disqualifying undesirable matches 83
 - fuzzy matching with
 - String::Approx 85–86
 - in context, Perl’s superiority to greppers for 75–77
 - range of records 151–157
 - record separators 125
 - metacharacter, definition 436
 - method calls, compared with
 - function calls in CGI
 - example 423
 - methods, vs. functions 400
 - Meyer, Colin “Shroomy” 347
 - Minimal Perl
 - as a simplified subset of Perl 8
 - isn’t a version of Perl 8
 - minimizes problems with operator precedence 252
 - motivation for creating xix
 - practical and eclectic nature of 9
 - public debut xx
 - mkdir function 239
 - modularize
 - means convert code to a Perl module 363
 - vs. the term compartmentalize when discussing subroutines 363
 - module
 - definition of 27
 - meaning of double colons in name 28
 - modules
 - advantages over subroutines 389
 - and explicit package names for variables 393
 - Benchmark 315
 - Business::UPS 398–405
 - Carp 391
 - Center 393–395
 - CGI 231, 414–422
 - CGI::Carp 416
 - comparison of procedural and Object-Oriented 422–424
 - determining which ones you have 400
 - Exporter 389
 - File::Copy 240
 - File::Find 201, 411–414
 - introduction to 388–389
 - Lingua::EN::Inflect 167
 - Lingua::EN::Namegame 372
 - List::Util 227, 329
 - LWP::Simple 405–408
 - provide aliases to their exports for user convenience 393
 - Regexp::Common 372
 - Shell::POSIX::Select 356
 - Stat::IsMode 217
 - String::Approx 85–86
 - Term::ANSIColor 294
 - Term::Cap 271
 - Text::Autoformat 28, 116, 278
 - Text::Tabs 117, 366
 - Text::Wrap 395
 - See also* using modules; creating modules; installing modules; managing modules; Simple Module Template
 - move function, related to Unix mv 240
 - MULTICS OS 269
 - Murray, featured character for
 - scan4oops project 175
 - mv command
 - related to Perl’s move 240
 - related to Perl’s rename 240
 - my declaration
 - definition 436
 - for defining a variable’s scope 378
 - introduction to 374
 - is preferred declaration for user-defined variables 374
 - syntax and examples 374
 - with aggregate assignment to list of variables 405
 - See also* our declaration
- ## N
- n (automatic input-reading)
 - option 14, 17, 19
 - cases where it’s omitted 229
 - using non-filename arguments with 256
 - See also* implicit loop
 - name clashes
 - are more likely in larger programs 385
 - are more likely with file-scoped variables 380
 - as consideration in choosing exports from modules 391
 - avoided by using package declaration in modules 393
 - other techniques for avoiding 367, 386, 393
 - natural language processing, conditionally pluralizing nouns 166–167
 - See also* Lingua::EN::Inflect
 - nawk command 123, 432
 - NAWK. *See* nawk command; AWK; GAWK; POSIX AWK

ne (string inequality)
operator 157
new AWK 123
See also NAWK
newline xxvi
definition 436
See also glossary
nexpr 163
not a complete replacement for
expr 163
the legend of 162–164
See also scripts, nexpr_p
next (loop control) directive 349
compared to Shell's
continue 349
NF (number of fields) variable
(AWK) 127
NR (record-number) variable
(AWK) 126
-nt (newer-than) file test operator
(Shell), emulating in Perl using
stat 216–217

O

-o (owned by effective ID) file
test operator 182
-O (owned by real ID) file test
operator 182
Object Oriented Perl book xx,
347
Object-Oriented modules
tips on using 422–424
Object-Oriented programming
vs. procedural
programming 405
octal numbers, left-padding with
zeroes 105
OFS (output field-separator)
variable (AWK) 126
open function 361
operand, definition 436
operating system, definition 436
operator precedence
use of parentheses to
override 38, 184, 252, 351

operator, definition 436
operators, vs. functions 436
ORS (output record-separator)
variable (AWK) 42, 126
OS, definition 436
Oscar, featured character for
scan4oops-like
command xxv, 174
our declaration
allows use of simple name for
global variable 374
for defining a variable's
scope 381
for marking switch variable as
optional 35
introduction to 374–375
is used for all switch variables
in strict mode 375, 381
is used for global variables 375
is used for variables exported
by modules 375, 381
omitted for mandatory
switches 108
requires parentheses around
multiple arguments 33
syntax and examples 374
used for optional switches 117
See also my declaration
output record separator
automatic printing of 20
example of 42
newline as default 20

P

-p (automatic input-reading,
with printing) option
17, 19
used with sed-like
command 93
using non-filename arguments
with 256
See also implicit loop
-p (named-pipe) file test
operator 182
See also implicit loop

package declaration, why it's used
in modules 393
paragraph mode, enabling by
assignment to \$/ 428
parentheses
guidelines for using on
code 430–431
mandatory for our declaration
with multiple
arguments 33
optional use of 161
required for chomp with mul-
tiple arguments 221
required for our declaration
with multiple
arguments 374
used in explicit list 434
used to enclose all function
arguments 243
used with function
arguments 242–243
using to control allocation of
arguments to
functions 242–243
See also appendix B
Patrick, featured character for
city rainfall comparison
project xxvi, 143–151
pattern ranges
matching a range of
dates 153–154
matching multiple
ranges 155–157
syntax 153
pattern, definition 436
Pattern/Action programming
AWK vs. Perl syntax 139
combined with field
processing 143
introduction to 138–142
patterns
longest-anything 78, 109,
114, 117
longest-something 78
shortest-anything 78
shortest-something 78, 124

- patterns (AWK)
 - matching type 139
 - relational operator type 140
- Perl
 - advantages over AWK 129
 - as argument pre-processor for other commands 192
 - as derived from UNIX utilities xvii
 - as the “Swiss Army chainsaw” xvii
 - commands hybridized with Unix commands 180
 - common problems of
 - beginners xxv
 - complexity of xix, 3, 51
 - cryptic programming styles of 14
 - definition 437
 - dialects of 4, 9
 - efficiency advantages over Shell 206
 - functions, list of 160
 - idiomatic programming style of 6
 - link to article on OS-portable programming techniques 199, 201, 245
 - Object-Oriented programming approach 8
 - redundancies of xix, 7
 - sensitivity to context 243
 - similarities to Yiddish 6
 - source-code beautifiers xx, 265
 - summary of advantages over find command 201
 - summary of advantages vs. AWK 129
 - summary of facilities superior to Unix counterparts 206
 - syntax 10
 - See also* perl
 - Perl Best Practices* book xxxii
 - Perl commands. *See* commands (Perl)
 - Perl functions. *See* functions
 - Perl Mongers xxii, 368
 - Perl scripts. *See* scripts
 - Perl shell. *See* scripts, psh
 - perl, definition 437
 - See also* Perl
 - PERL5LIB (library search) variable (Shell)
 - preserving setting in Shell startup file 397–403
 - setting to help Perl find modules 397–403
 - perldoc
 - advantages over man command 51
 - q option, for searching FAQ 52
 - PerlDude, featured character for find2perl examples 200
 - Perlstan xxvii, 3–5
 - definition 437
 - need for citizens to wear dialect markings 6
 - PFD, as Precursive Frobination of Defragulations 388
 - POD documentation, example of script containing 155
 - positional parameters 198
 - POSIX AWK 124
 - POSIX find 186–187
 - capabilities 180, 186–187
 - POSIX grep, capabilities 58
 - POSIX sed 92
 - capabilities 91
 - POSIX, definition 437
 - Primary Option Cluster for Input Processing 93
 - print function 10
 - differences from AWK’s print 134–136
 - treats commas differently than AWK’s print 134
 - with arithmetic expression as argument 11
 - printf command, Perl relatives 223
 - printf function 25, 407
 - % as special character with 22
 - for formatted printing 218
 - for printing without automatic newlines 21
 - used for printing floating point numbers 324
 - used for printing with fields of fixed widths 324
 - used for prompts 21
 - printing, Shell’s echo vs. Perl’s print 10
 - private variables, are declared with my 374
 - procedural programming vs. object-oriented programming 405
 - processing input, a line at a time using Shell vs. Perl 341–342
 - program segments, list of 381
 - Programming Perl* book 433
 - programs, step-by-step construction technique 47–51
 - push function 305, 349
 - for initializing arrays 299
 - PWD (present working directory) variable (Shell) 397
- Q**
- quoting
 - clever use in nexpr* scripts 164
 - differences between commands and scripts 30
 - differences between Shell and Perl 30, 341
 - introduction to 30
 - link to guidelines article (Shell) 49, 136
 - nesting of single quotes using backslash 62
 - of one-line commands 11
 - Shell-friendly Perl techniques 136
- qw (quote words) operator 391

R

-r (readable by effective ID) file test operator 182
unnecessary to use before -T test 274

-R (readable by real ID) file test operator 182

Ramon, featured character for `check_length` 116

rand function 221–222
syntax and examples 222
Unix relatives 210
used to select random element from array 305

RANDOM variable (Shell)
Perl relatives 210

Ravi, featured character for data validation commands, and jalebi consumption 66

RE (regular expression) notation
corresponding metacharacters from FNG notation 238
using for matching filenames 238

recursive grepping 191–192

redirection requests (Shell)
can be attached to constructs, unlike case in Perl 361

reference value of this book xxiv

references. *See* `\&` operator

regex, definition 437

Regexp::Common module
using to identify profane words 372

regular expressions
as superior to those of Unix utilities 56
classic dialect (UNIX) 91
comparison of dialects 57
essential syntax 63–64
line-spanning 77–79
metacharacters for grouping, and match capturing/referencing 71

quantifier metacharacters 73
shortcut metacharacters 67

relational operators
comparing AWK's and Perl's 157–158
used to define a range of lines 97

rename function, related to Unix `mv` 240

report generation
conditionally pluralizing nouns 166–167
using system 277–280

return function 364
returned value automatically converted for caller's context 365

reverse function
Unix relatives 223
used with `sort` 225

reverse video terminal mode. *See* `tput` command

rm command, related to Perl's `unlink` 240

rmdir function 239

rock-star biodata system (AWK) 133–134

rock-star biodata system (Perl) 134

RS (input record-separator) variable (AWK) 42, 126

Rule of 72
for estimating investment growth through compounding 165

S

-s (non-empty) file test operator 182, 197
easier to use than `stat` 219

-S (socket) file test operator 182

-s (switch) option 45, 278
used heavily in `preg` 288

s/// (substitution) operator
capabilities of 91

converting special characters with 103–105

differences from `sed` 234

introduction to computed replacements 91

modifiers for 95

syntax of 95

used to insert indentation 19

using backreferences and numbered variables with 99

using computed replacements 114–118

using context addresses 96

using function-generated replacements 116

s2p command, translates `sed` to Perl 118

scalar (data type)
introduction to 23

scalar context
effects of 208
ways to request 208

scalar function, for overriding list context 208

scalar to list conversion
tools for 209–210

scope, of type file, defined 434

scoping, definition 437

script, definition 437

scriptification, definition 437

scriptified, definition 437

scripts
`4letter_word` 372, 375, 383
a scripted grepper 84
advantages over commands 247
`award_cruises` 32
`award_cruises2` 32–33
`award_cruises3` 38
`beatles` 313
`c2f: Celsius to Fahrenheit` 233
`cd_report` 261, 263
`center` 365
`center2` 366
`center2.strict` 369, 396

- scripts (*continued*)
 - center3 396–397
 - change_file 107, 198
 - check_length 116
 - check_length2 117
 - check_links 405–406
 - check_symlinks 411–412
 - compared to Shell scripts 29
 - compound_interest 165
 - compound_interest2 167
 - compress_image 336
 - compress_image2 345
 - confirmation 354
 - count_words 323–324
 - definition 29
 - double_space 40
 - expand_acronyms 343
 - expand_daynames 94
 - extract_cell 127, 147
 - extract_cell2 149
 - fcookie 305
 - fcookie2 306
 - fields2lists, converts input fields into HTML bullet items 231
 - for editing files 107–110
 - fuzzy_match 85
 - greperl 248, 286
 - highlight_trailing_ws 272
 - how nexpr* scripts work 164
 - incomplete 141
 - insert_contact_info 108
 - insert_contact_info2 110
 - intra_line_sort 226
 - introduction to 29–41, 247–248
 - listfile 217
 - m2k (miles to kilometers) 115
 - massage_data 254
 - mean_annual_precip 145
 - menu_ls 409
 - most_recent_file 195
 - most_recent_file2 267
 - mytime 212
 - mytime2 215
 - mytime3 386
 - news_flash 277
 - news_flash2 278, 281
 - nexpr (Shell/AWK) 164
 - nexpr_p 164, 283
 - of dual input-mode variety 308, 320
 - perl_cat 30
 - perlgrep 248, 257, 286
 - perlman 358–360
 - phone_home 371, 377, 380
 - phone_home2 384
 - phone_home3 385
 - preg 286–287, 291–292
 - preg, as replacement for grep-erl, text_grep, perlgrep, and rgrep 286
 - psh 284–286
 - raffle, demonstrates nested for loops 347–348
 - resignation_letter 251
 - rgrep 192, 273–274, 286
 - rm_files 341
 - running Shell commands from Perl scripts 248
 - scan4oops 170
 - scan4oops2 173
 - shell_types 299
 - show_fields2 302–303
 - show_fields2_1 155
 - show_files 34
 - show_pvars 337
 - show_user 357–358
 - survey.cgi 414, 417, 420–421
 - text_grep 190, 286
 - textfile_args 228
 - textfiles 184, 190, 198, 228
 - unique_args 316
 - unique_inputs 318–319
 - ups_shipping_price 404
 - user_lookup 321
 - See also* commands (Perl)
- Seattle Perl Users Group xx, xxii, 347
 - link to interview with founder Tim Maher xxii
 - subscribers to mailing list required “use strict” on submitted source code 368
- Seattle.pm. *See* Seattle Perl Users Group
- sed command
 - as inferior to Perl for file editing 110
 - capabilities compared to Perl’s 91
 - differences from substitution operator 234
 - eclipsed by AWK 90
 - emulating -f option with Perl 94
 - history of 89–91
 - matching ranges of records 152
 - n option compared to Perl’s -n 137
 - pattern-matching capabilities compared to Perl’s 124–126
 - performing line-specific substitutions with 96
 - Perl equivalents for common uses 119
 - Perl relatives 223
 - printing lines by number 100
 - relation to ed 90
 - shortcomings of 91–93
 - used to insert indentation 19
 - using “l” command, for special listing format 137
 - was eclipsed by AWK 90
- sed, GNU version, defined 433
- select function 361
 - is unrelated to Perl’s select loop 361
- select loop (Perl) 355–360, 408–410
 - author’s motivation for developing 356
 - enhancements over Shell version 357–358, 361, 409

- select loop (Perl) (*continued*)
 - facilitates development of terminal-based menu-oriented scripts 355
 - syntax 356
 - See also* Shell::POSIX::Select
- select loop (Shell) 331
 - syntax 356
- semicolon
 - as terminator of statement 438
 - customarily omitted for one-line code blocks 224
 - not needed at end of most constructs 265, 333
 - required at end of do while/until loop 339
 - use of 19
 - used in condensed if/else format (Shell) 260
 - used in condensed looping format (Shell) 333
 - vs. Shell's <ENTER> as statement terminator 10
- Senator Quimby, featured character for “matching in context” commands 76
- shebang line
 - as used in Perl scripts 11
 - definition 438
 - is not used in module files 395
- shebang, definition 438
- Shell xxvii
 - bash version, defined 433
 - meaning vs. shell xxiii
 - See also* glossary
- Shell commands, running from Perl scripts. *See* command interpolation; system
- Shell functions, vs. Perl functions 364
- Shell processing of command line is difficult to understand 325
- Shell programmer
 - as reader of this book xxiii
- skills required for readers of part 2 xxiv
- Shell prompt
 - in command-with-output vs. code-with-output displays xxxi
 - primary prompt xxx
 - secondary prompt xxx
- Shell scripts. *See* scripts
- Shell::POSIX::Select
 - module 356
 - link to documentation 361, 425
 - provides the select loop for Perl 408–410
- Sherpa, Yeshe xxii
- shift function 254–256
 - comparison of effects in Shell and Perl 254
 - examples and Shell vs. Perl comparison 255
 - illustration of effect on arguments 254
 - See also* unshift function
- shortest-anything pattern 78
- shortest-something pattern 78, 124
- Simple Module Template 390
- slash, definition 438
 - See also* / (slash); backslash
- Solaris 433
- sort function 224–227
 - case-insensitive sorting 227
 - programmer defines sorting rules 225
 - random sorting 226
 - random sorting with List::Util 227
 - syntax and examples 224
 - Unix relatives 223
 - used in printing hashes 337, 343–344
 - uses \$a and \$b in comparing list items 225
- source-code filtering 358
 - as used in Shell::POSIX::Select 356
- space character, definition 438
- special characters, converting with substitution operator 103–105
- special variables 23
- split function 211–213
 - compared to cut command 211
 - compared to IFS variable 211
 - for scalar to list conversion 209
 - syntax 211
 - tips on using 213, 227, 234
 - Unix relatives 210
 - used to supply iteration list to foreach loop 372
 - when to use matching operator instead 213
- split function (AWK), Perl counterpart is split 210
- sprintf command, Perl relatives 223
- SPUG 347
 - links to web pages 101
 - needed meeting-announcement software 101
 - See also* Seattle Perl Users Group
- square forehead marking. *See* forehead markings of Perlstanis
- squared JAPHs
 - are understandably uneasy about uninitialized variables 317
- standard error, definition 438
- standard input, definition 438
- standard option clusters, introduction to 44–47
- standard output, definition 438
- stat function 181–182, 196, 201, 215–219
 - syntax and examples 216

- stat function (*continued*)
 - tips on using 218–219
 - Unix relatives 210
 - using to emulate ls
 - command 217–218
 - using to emulate Shell’s -nt operator 216–217
 - Stat::lsMode module, for converting permission strings 217
 - statement, definition 438
 - STDERR (standard error), definition 438
 - STDIN (standard input) definition 438
 - STDIN (standard input)
 - filehandle 19, 220, 228, 250
 - as default input source for script 289
 - avoiding as default input source when user omits argument to script 274
 - STDOUT (standard output), definition 438
 - Steffi, featured character having “lingering thumb” 186
 - strict mode 363, 439
 - enabling using -M’strict’ option 369
 - See also* use strict
 - strictified 439
 - stricture 439
 - string modifiers, for case conversion 113
 - String::Approx module
 - for fuzzy matching 85–86
 - strings, definition 439
 - sub declaration 364
 - sub. *See* subroutines
 - subroutines
 - basics of 363–365
 - benefits of using 362
 - caller’s context defined 365
 - defining and using 365–368
 - passing data through arguments 367, 383, 392
 - passing data through global variables 367, 395
 - return value can be specially crafted for caller’s context 365
 - reusing in other programs 386–387
 - syntax and examples 364
 - vs. Shell functions 364
 - subroutines and variable scoping, introduction to 362–363
 - subs
 - center_line 367, 369, 386, 395–396
 - center_line, modularized 393
 - check_link (from check_links script) 407
 - check_symlinks (from check_symlinks script) 412
 - commafy 413
 - conscious 382
 - dial_phone 371
 - get_home_address 371, 384–385
 - show_times 414
 - uniquify (from check_links script) 408
 - Sugalski, Dan 347
 - Sweger, Andy “yDNA” 347
 - switch variable, definition 439
 - switch, definition 439
 - switches, introduction to 33
 - system administration tools
 - change_file script 107, 198
 - check_links script 405
 - check_symlinks 411, 414
 - command emulating Shell’s -nt file test operator 217
 - command for extracting “File doesn’t exist” errors from logfile 152
 - command for filtering find’s output 183
 - command for processing multi-word filenames 197
 - command like scan4oops script 174
 - highlight_trailing_ws script 272
 - most_recent_file script 195
 - most_recent_file2 script 267
 - preg script 287
 - scan4oops script 170
 - scan4oops2 script 173
 - shell_types script 299
 - show_fields script 302
 - show_user script 357–358
 - unique_args script 316
 - unique_inputs script 318
 - user_lookup script 321
 - w command 357
 - system function 275–283, 410
 - converting Shell’s True/False values to Perl’s 276, 335
 - debugging Shell commands submitted by Perl 281–282
 - may require multi-level quoting 335
 - requires multi-level quoting 276
 - syntax and examples 276
 - tips on using 280–283
 - used in compress_image 336
 - using to generate reports 277–280
- ## T
- T (text) file test operator 182, 184–185
 - incorporates -r (readability) test 274
 - tab character, definition 439
 - table processing
 - using array indexing 145
 - with switch-driven scripts 147
 - See also* field processing
 - tac command (Linux), Perl relatives 223

- tail command, as poorly suited for printing specific lines 100
 - taint-checking mode 304
 - template processing 101–103
 - advantages of Perl over sed 103
 - replacing placeholders 102
 - See also* Template Toolkit
 - Template Toolkit 103
 - Term::ANSIColor module 294
 - Term::Cap module, compared to tput 271
 - text processing
 - quieting spam 113–114
 - upper/lowercase conversions 113–114
 - text substitutions, comparisons of sed vs. Perl 93–99
 - Text::Autoformat module 116, 278
 - as replacement for fmt command 28
 - Text::Tabs module 117, 366
 - provides expand function 366
 - Text::Wrap module 395
 - The Perl Foundation 347
 - time function, returns current time as large integer number 414
 - Tinker toys 363
 - TMTOWTDI, definition 439
 - TPF. *See* The Perl Foundation
 - tput command 271–273
 - compared to Term::Cap module 271
 - determining terminal’s dimensions using lines and cols options 271, 277
 - for displaying text in reverse-video 285
 - for manipulating terminal display modes 271
 - for underlining text on screen 275
 - testing output for error 279
 - tr command 113, 331
 - doesn’t take filename arguments 331
 - training on Perl, minimal vs. maximal approaches 7
 - Travelers tale 3–5, 14
 - triangle forehead marking. *See* forehead markings of Perlstanis
 - True value
 - of Shell, converting to Perl’s True 275
 - Shell and Perl definitions of 32
 - True, definition 439
 - Truthiness and Falsity 32
 - TVM guy, featured antagonist of Fox Boulder 137–138
 - typographical conventions used in this book xxvii–xxxi
- U**
- u (set-UID) file test operator 182
 - umask function 240
 - undef function
 - definition 439
 - used in assignments to explicit lists 132
 - used to return undefined value from sub 368
 - undefined values
 - definition 439
 - detecting and replacing 367
 - only serious uses trigger warnings 368
 - returning from sub 368
 - Unix
 - definition 440
 - skills required of readers xxiv
 - See also* UNIX
 - Unix people, definition 440
 - Unix pipelines, data flows backwards vs. Perl functions 223
 - Unix utilities, POSIX versions, defined 437
 - UNIX, definition 440
 - See also* Unix
 - unlink function 341
 - is irreversible like rm 240
 - related to Unix rm 240
 - unset command 315
 - unshift function 255, 303–304
 - illustration of effects 302
 - used to achieve friendlier field numbers 302
 - See also* shift function
 - until loop. *See* while/until loop
 - use directive
 - loading modules with 41
 - used with strict 439
 - use strict
 - as quality standard 368
 - as used in modules 391
 - efficacy compared to that of myopic lavatory attendant 386
 - implements strictures, whose violations are fatal errors 368
 - is very limited as a quality-control tool 370–371, 373, 384–385, 387
 - poorly conceived declarations undermine its benefits 375
 - proper placement in script 377
 - purpose of 368–373
 - See also* strict mode
 - user interaction, obtaining confirmation 352–353
 - user-defined variables 25
 - using modules 403–424
 - augmenting Perl’s search path by setting PERL5LIB 397
 - must import all desired components, if not using defaults 396

using modules (*continued*)
preserving setting of
PERL5LIB in Shell startup
file 397, 403
using `defined_for_keyboard_in`
`put` 250
using `re_notation_for_filename`
`_filtering` 238

V

validating data, with `grep`
command 66
value, definition 440
values function, retrieves all values from a hash 310
variable assignment, aggregate assignment to list of variables in declaration 405
variable clobberations 371
are more likely with file-scoped variables 380
more easily avoided in small programs 385
tips on avoiding 373
variable declarations, syntax and examples 374–384
variable interpolation
Perl has, AWK lacks 128–129
with `print`, compared to AWKish approach 128
variable masking 372–373
`$_` may get masked in nested loops 373
tips on avoiding 373
variable scoping
benefits of curly braces around `Main` 379
block scope 377
effects of curly braces 377
employing user-defined loop variables 383
example of widely-scoped variable 413
file scope 377

graphical illustrations of 378–382
introduction to 373
loop scope 383
preventing variables from leaking into subs 379
using my declaration 378
using our declaration 381
See also Variable Scoping Guidelines
Variable Scoping
Guidelines 369
case study of applying:
`phone_home2` 384–385
for complex programs 376–386
for simple programs 376
introduction to 375
tips on using 385
variable declarations. *See* my declaration; our declaration; local declaration
variables
AWK variable names usable in Perl 23
cases requiring explicit initializations 142
common problems 370–373
data variable. *See* `$_` (data) variable
declarations for switch type 436
declared with `local`, defined 435
declared with `my`, defined 436–437
declared with `our`, defined 436
default initialization values 25
drawbacks of using spaces around “=” in assignment 27
introduction to 23
leakage of 363, 376, 381
list (data type), compared to scalar 23

masking of, defined 436
Perl compared to AWK 126
record number variable. *See* `$.` (record number) variable
scope, defined 437
See also special variables; user-defined variables
variables, scalar (data type)
introduction to 23–27
Vitas, featured character for city rainfall comparison project 143–149
void context 252

W

`-w` (warnings) option 11, 17–18, 395
not used in CGI scripts 416
`-w` (writable by effective ID) file test operator 182
`-W` (writable by real ID) file test operator 182
`w` command 357
Wall, Larry 347
as apparition in Fox Boulders dream 134
as creator of Perl xxii, xxvi, 6–7, 9, 14, 20, 51, 59, 93, 118, 123, 158, 175, 211, 436
as discreet critic of your programs 18
debugs using `print` statements, not Perl debugger 285
on relation of Perl to Unix xvii
pays homage to AWK 123
See also glossary
`wantarray` function 364
`warn` function
introduction to 35
plus `exit` preferred to `die`, in `BEGIN` blocks 253
vs. use of `carp` in modules 394–395

- warning messages
 - Applying substitution (*s///*) to
 - @array will act on scalar(@array) 230
 - Can't open –
 - switch_var_name: No such file or directory 34
 - Global symbol “\$whatever” requires explicit package name 369, 374
 - Name
 - “main::switch_var_name” used only once, possible typo 35
 - odd number of initializers 312
 - quantifier follows nothing in regex 237
 - readline() on unopened filehandle 237
 - reference found where even-sized list expected 312
 - Scalar value @some_name[0] better written as \$some_name[0] 308
 - use of uninitialized value in print 249, 368
 - use of uninitialized value in string ne 251
 - use of uninitialized value on line ... 383
 - used only once: possible typo 18
 - useless use of a variable in void context 221
 - useless use of defined operator in void context 252
 - See also* error messages
 - web scraping 86
 - while/until loop 333–338
 - comparison of Shell vs. Perl syntax 333
 - tips on using 334
 - using “infinite” version in replacement for do while/until 351–352
 - while/until loop (Shell), emulating advanced features of in Perl 353–354
 - whitespace, definition 440
 - Willy, featured character for insert_contact_info 108
 - WinDude, featured character for find2perl examples 200
 - wl (output generation) cluster 45
 - wnl (input/output) cluster 45
 - wnla (field processing) cluster 45, 233
 - wnlaF (custom field-processing) cluster 45, 138, 185
 - word splitting (Shell) 325
 - wpl (input/output with printing cluster), used in sed-like commands 97
- X**
- x (executable by effective ID) file test operator 182
 - X (executable by real ID) file test operator 182
 - x (string repetition) operator 265–267
 - syntax and examples 265
 - xargs command 195–198
 - advantages of Perl for sorting applications 193–196
 - advantages over find's -exec option 193
 - not an alternative to find's -exec on all OSs 200
 - relation to command substitution 189
 - used with find 192–197
 - XPG4 433
- Y**
- YAPC xx
 - definition 440
 - Yet Another Perl Conference. *See* YAPC
 - Yoko, featured character for fuzzy_match xxvi, 85
- Z**
- z (empty) file test operator 182