

Index

• *Symbols and Numerics* •

/ (division), 104
\$ (dollar sign), 111–112, 166
|| (double vertical line symbol), 127
= (equal sign), 94, 98
^ (exponentiation), 104
> (greater than), 140
>= (greater than or equal to), 140
\ (integer division), 104
< (less than), 140
<= (less than or equal to), 140
* (multiplication), 104
<> (not equal), 140
+ (plus sign), 104, 312
– (subtraction), 104
2600, 347

• *A* •

addition (+), 104
Ajax, 33
alert dialog boxes, 313
algorithms
 binary search, 235–238
 bubble sort, 220–223
 defined, 215, 232
 efficiency, measuring, 216
 insertion sort, 216–219
 quicksort, 227–231
 right, selecting, 251
 search, selection, 238
 searching, 232–238
 sequential search, 233–235
 shell sort, 223–226
 sort, built-in, 231–232
 sort, selection, 231
 wrong, choosing, 232
alpha testing, 45
anchors, 297
AND operator. *See also* Boolean operators
 defined, 124
 IF-THEN statement, 252

 truth table, 125
 values, 124
Animation Magazine, 343
AntiOnline, 347
Apple Open Source, 348
applets. *See* Java applets
arrays
 associative, 181
 in C++, 179–180
 creating, 175–182
 defined, 175
 dynamic, 182–189
 element definitions, 176
 initializing, 177, 180
 in Liberty BASIC, 176–178
 multi-dimensional, 189–193
 one-based, 176
 in REALbasic, 178–179
 Revolution, 181–182
 static, 187
 storing data in, 177
 string storage/retrieval in, 179
 two-dimensional, 189–193
 zero-based, 176, 179
ASCII files, 50–51
assembly language
 advantages/disadvantages, 21
 commands, 20
 defined, 20
 program example, 20–21
 source code, 20
associative arrays, 181
Axon Idea Processor, 11–12

• *B* •

BASIC
 advantages, 25
 compilers, 66, 67
 dialects, 2, 25
 feedback, 67
 learning, 66–69
 Liberty BASIC, 2, 66–67

- BASIC (*continued*)
 - philosophy, 67
 - popularity, 25
 - power, 66
 - programs, writing, 25
 - REALbasic, 2, 27, 67–69
 - beta copies, 242
 - beta testing, 45
 - Big-O notation, 216
 - binary search. *See also* searching
 - algorithms
 - C++, 237–238
 - defined, 235
 - illustrated, 236
 - Liberty BASIC, 236–237
 - list sort requirement, 237
 - steps, 235
 - Boolean expressions
 - defined, 120, 144
 - in endless loop creation, 154–156
 - evaluating, 120
 - storing as variables, 121–122
 - using, 119–128
 - value comparison, 120
 - variables in, 122–123
 - Boolean operators
 - AND, 124–126
 - defined, 124
 - OR, 126–128
 - use, 123–124
 - bootstrapping, 53
 - branching instructions. *See also* instructions
 - alternative instructions, 81
 - code, 81
 - defined, 80
 - illustrated, 81
 - using in program, 84, 85
 - branching statements
 - common structures, 352–354
 - defined, 119
 - function, 119
 - IF-THEN, 120–121, 128–129
 - IF-THEN-ELSE, 129–132
 - SELECT CASE, 132–142
 - switch, 135–138
 - breakpoints, 57, 246
 - browsers, 281
 - bubble sort. *See also* algorithms; sort
 - in C++, 221–223
 - defined, 220
 - functioning, 220
 - illustrated, 220
 - in Liberty BASIC, 220–221
 - using, 251
 - buffer overflow, 247
 - bugs
 - anatomy, 240
 - defined, 57
 - logic errors, 240, 243–246
 - run-time errors, 240, 242–243
 - syntax errors, 240–242
 - tracking down, 57–58
 - verifying, 46
 - building blocks, programming, 83–85
 - built-in commands, 256–257
 - built-in mathematical functions, 108–111
 - built-in sorting algorithms, 231–232
 - buttons, 299
- C •
- C programming language
 - advantages/disadvantages, 24
 - compilers, 23
 - defined, 22
 - goals, 23
 - programs, 24
 - C#, 330, 331
 - C++
 - AND operator, 125–126
 - arrays, 179–180
 - arrays, initializing, 180
 - binary searching, 237–238
 - Boolean expressions in variables, 122
 - bubble-sort algorithm, 221–223
 - class creation, 198
 - comments, 101
 - defined, 69
 - defining constants in, 100
 - design, 70
 - Dev-C++, 69
 - difficulty in understanding, 255–256
 - DO-WHILE loop, 153–154
 - dynamic arrays, 187–188
 - FOR-NEXT loop, 146
 - FOR-NEXT loop backward count, 150

- FOR-NEXT loop incremental count, 148–149
- function, creating, 173–174
- functions, 163–164
- gcc compiler, 69
- IF-THEN-ELSE statement, 129, 130–131
- inheriting classes with, 204–206
- insertion sort, 218–219
- learning, 69–71
- libraries, 158
- mathematical functions, 109–110
- object creation from class, 200–202
- OR operator, 127–128
- passing parameters, 169
- polymorphism, 209–210
- popularity, 2
- postfix operator, 255
- prefix operator, 255
- programs, 70–71
- quicksort algorithm, 229–231
- relational operator `SELECT CASE` statement, 142
- run speed, 255
- sequential search, 234–235
- shell-sort algorithm, 225–226
- string definition, 116
- subprograms, creating, 163–164
- switch statement, 135–137
- two-dimensional arrays, 192
- unary operators, 105
- value range checking, 139–140
- variable declaration as string, 112
- variables in, 98
- WHILE loop, 152
- C++ compilers
 - installing, 365
 - optimization settings, 258
 - standard libraries, 158
- C++ Robots game, 339
- CD compilers
 - CD problems, 367
 - Dev-C++, 365
 - Liberty BASIC, 365
 - REALbasic, 365
 - Runtime Revolution, 365
 - system requirements, 365–366
 - using, 366–367
- check boxes. *See also* controls; user interfaces
 - creating, with HTML code, 304–305
 - defined, 268, 299
 - event, 304
 - form, 303–305
 - HTML, 303–305
 - with REALbasic, 268–269
 - with Revolution, 269–270
- Clarion, 333–334
- classes
 - creating, 198–199
 - creating objects from, 200–203
 - defined, 198
 - inheriting, using C++, 204–206
 - inheriting, using REALbasic, 206–208
- COBOL, 22, 24–25
- CodeGuru, 336
- CodeWarrior, 53
- collections, 187
- colors
 - background, 289–290
 - hyperlink, 290–291
 - text, 289–290
- combo boxes. *See also* controls; user interfaces
 - contents, editing, 271
 - defined, 270
 - illustrated, 271
 - with REALbasic, 270–273
 - with Revolution, 275
- command buttons. *See also* controls; user interfaces
 - defined, 266
 - event, 302
 - form, 302–303
 - HTML, 302–303
 - illustrated, 268
 - REALbasic program, 266–267
 - Revolution, 267
- command-line interfaces. *See* user interfaces
- comments
 - C++, 101
 - creating, 100–103
 - HTML, 284–285
 - for ignoring lines, 102
 - Liberty BASIC, 101

- comments (*continued*)
 - REALbasic, 101
 - Revolution, 102
 - uses, 100–101
 - compilers
 - BASIC, 66, 67
 - bootstrapping, 53
 - C, 23
 - C++, 158, 258, 365
 - CD, installing, 365–367
 - CodeWarrior, 53
 - defined, 23, 49, 52
 - gcc, 69
 - high-level programming language, 26
 - interpreters versus, 49
 - Linux, 332
 - machine language, 52
 - Macintosh, 331
 - optimizing, 258
 - syntax error detection, 241
 - use determination, 56
 - using, 52–53
 - Visual Express, 330
 - Windows, 331
 - computer animation programming, 343
 - computer game programming, 341–343
 - Computer Graphics World Online, 343
 - computer programs. *See* programs
 - concatenating strings, 112
 - confirmation dialog boxes. *See also* dialog boxes
 - creating, 313–314
 - defined, 313
 - illustrated, 314
 - constants
 - defined, 99
 - defining, 100
 - using, 99–100
 - controls. *See also* user interfaces
 - check boxes, 268–270
 - combo boxes, 270–275
 - command button, 266–268
 - common, 264–265
 - list boxes, 270–275
 - radio buttons, 268–270
 - text boxes, 276–278
 - Core Wars, 337–338
 - counting, FOR–NEXT loops. *See also* FOR–NEXT loops
 - backward, 149–150
 - with different numbers, 147
 - in increments, 147–149
 - Cprogramming.com, 336
 - C-Robots and P-Robots game, 339
 - cross-platform programs
 - defined, 32
 - examples, 39
 - issues in program design, 39
 - Crystal Space game engine, 342
 - cube function, 172
 - CypherNet, 344
- D •
- data
 - bubble sort, 220–223
 - insertion sort, 216–219
 - protected, 203
 - public, 203
 - quicksort, 227–231
 - searching for, 232–238
 - shell sort, 223–226
 - sorting, 215
 - structure, 249–250
 - variables, 93
 - data types
 - common, 95
 - correct, using, 256
 - decimal numbers, 95
 - text, 95
 - understanding, 96
 - variable, 94–97
 - whole numbers, 95
 - database programming languages
 - advantages, 29–30
 - Clarion, 333–334
 - defined, 29
 - FileMaker, 334
 - limitations, 30
 - PowerBuilder, 334
 - SQL, 334
 - using, 333–334
 - databases, 29

- debuggers
 - breakpoints, 57
 - defined, 57
 - Dev-C++, 69
 - illustrated, 57
 - methods, 57
 - stepping, 57
 - watching, 58
 - debugging
 - defined, 239
 - logic errors, 240, 243–246
 - run-time errors, 240, 242–243
 - syntax errors, 240–242
 - decimal numbers, 95
 - decompiling programs, 55
 - definition lists
 - creating, 293–295
 - illustrated, 295
 - tags, 294
 - Delphi, 330, 331
 - DelphiSource, 336
 - Dev-C++, 69, 365
 - development cycle, 45
 - dialog boxes
 - alert, 313
 - confirmation, 313–314
 - creating, 313–315
 - prompt, 314–315
 - types, 313
 - DigiPen, 343
 - division (/), 104
 - DLLs. *See* dynamic link libraries
 - dollar sign (\$), 111–112, 166
 - dynamic arrays. *See also* arrays
 - advantage, 182
 - C++, 187–188
 - data types, 187
 - defined, 182
 - Liberty BASIC, 182–184
 - REALbasic, 184–187
 - resizing, 182
 - Revolution, 188–189
 - dynamic link libraries (DLLs), 43
- **E** ●
- editors
 - ASCII files, 50–51
 - defined, 50
 - Visual Basic, 51
 - writing programs in, 50–51
 - encapsulation. *See also* object-oriented programming
 - data isolation, 196, 197
 - defined, 196
 - instruction isolation, 196, 197
 - encryption
 - careers, 344–345
 - defined, 344
 - END statement, 171
 - endless loops. *See also* loops
 - avoiding, 154–156
 - Boolean expression initialization failure, 155–156
 - Boolean expression modification failure, 154–155
 - equal sign (=), 94, 98
 - errors
 - logic, 240, 243–246
 - run-time, 240, 242–243
 - syntax, 240–242
 - events
 - defined, 300
 - handling, 300–301
 - HTML codes, 300–301
 - EXE files, 49
 - exponentiation (^), 104
 - external hyperlinks. *See also* hyperlinks
 - address, 296
 - creating, 295–296
 - defined, 295
- **F** ●
- file compression, 61
 - FileMaker, 334
 - Flash, 310
 - The Flexible Learning Company, 336
 - forms
 - buttons, 299
 - check boxes, 299, 303–305
 - command buttons, 302–303
 - creating user interface on, 299–308
 - defined, 299
 - elements, 299–300
 - events, handling, 300–301
 - radio buttons, 299, 306–308
 - text boxes, 299, 301–302

FOR-NEXT loops. *See also* loops

- C++, 146
- count, 147
- counting backwards, 149
- counting in increments, 147–149
- defined, 144
- elements, 145
- Liberty BASIC, 145
- needless, avoiding, 254
- Revolution, 146–147

FORTTRAN, 22, 24–25

Free Software Foundation, 348

F-Secure, 346

functions. *See also* C++; subprograms

- arguments, 316
- C++, creating, 173–174
- creating, 315–317
- data, 170
- defined, 163, 170, 315
- elements, 316
- example, 163–164
- illustrated, 316
- instructions, 171, 316
- JavaScript, 315–317
- keyword, 316
- Liberty BASIC, creating, 171–172
- mathematical, 171
- name, 316
- parameter list, 169
- REALbasic, creating, 172–173
- Revolution, creating, 174
- using, 170–174

• G •

Game Developer, 342

Game Programmer, 342

GameJobs, 343

gcc compiler, 69

GeekFinder, 345

GIF format, 297

golden handcuffs, 41

graphics

- as background picture, 299
- formats, 297
- text alignment, 298
- on Web pages, 297, 298

greater than (>), 140

greater than or equal to (>=), 140

GUI operating systems, 260

• H •

hacking careers, 346–347

handlers

- defined, 164
- example, 164
- starting, 165

headings

- creating, 285–286
- defined, 285
- illustrated, 286

Hello World! Web site, 35

Help files

- creation programs, 60
- defined, 59
- writing, 59–60

high-level programming languages. *See also* programming languages

- advantages, 26
- BASIC, 25
- COBOL, 24–25
- defined, 24
- drawbacks, 26
- FORTTRAN, 24–25
- PASCAL, 25
- RAD, 26–29

H.M.S. Sheffield, 243

HTML. *See* HyperText Markup Language (HTML)

hybrid object-oriented language, 211

HyperCard, 72

hyperlinks

- coloring, 290–291
- creating, 295–297
- defined, 281
- external, 295–296
- internal, 295, 296
- to specific spots, 296–297

HyperText Markup Language (HTML). *See also* tags, HTML

- anchors, 297
- basics, 282–285
- code, 32–33, 282, 283
- code, writing, 283
- colors, 289–290

- comments, 284–285
- defined, 32, 281
- event codes, 300–301
- events, handling, 300–301
- graphics display, 297–299
- headers, 283–284
- headings, 285–286
- hyperlink colors, 290–291
- paragraphs, 285, 286–287
- quotes, 285, 287
- radio buttons, 306–308
- text alignment, 289
- text boxes, 301–303
- text emphasis, 285, 288
- titles, 283–284
- Web pages, 33

• 1 •

IBIS, 11, 12

IF-THEN statements. *See also* branching statements

- AND operator, 252
- conditions, false first, 252
- conditions, true first, 252–253
- defined, 120, 128
- examples, 128–129
- use, 120–121

IF-THEN-ELSE statements. *See also* branching statements

- C++, 129, 130–131
- defined, 129
- example, 129
- Liberty BASIC, 130
- Revolution, 131
- using, 129–132

IF-THEN-ELSEIF statement, 252–253

inheritance. *See also* object-oriented programming

- class, using C++, 204–206
- class, using REALbasic, 206–208
- defined, 204
- protection, 204

input, 91

insertion sort. *See also* algorithms; sort

- in C++, 218–219
- defined, 217
- functioning, 217

- illustrated, 217
- in Liberty BASIC, 217–218

installation programs

- creating, 60–61
- defined, 60
- features, 61
- file compression, 61

instructions

- branching, 80–81
- function, 171, 316
- line of code, 76
- looping, 82–83
- pseudocode, 43
- repetitive, storing, 158
- sequential, 79–80
- as source code, 19, 76

integer division (\backslash), 104

internal hyperlinks. *See also* hyperlinks

- creating, 296
- defined, 295

International Animated Film Society, 343

International Game Developer's Association, 342

International PGP Home Page, 345

Internet programming, 345

interpreters

- compilers versus, 49
- defined, 49, 53
- use determination, 56
- using, 53–54
- for Web page programming languages, 54

• 1 •

Java

- defined, 33
- JavaScript versus, 322

Java applets

- adding to Web page, 323–326
- defined, 33, 321
- functioning, 321–322
- limiting power of, 323
- running, 322
- source code, 321

- space around, defining, 325–326
- Web sites offering, 322

- window location, aligning, 324–325
- window size, defining, 323–324

- writing, 321, 322

- Java Jobs, 345
 - JavaScript
 - basics, 310–315
 - code, storing, 311
 - dialog boxes, creating, 313–315
 - functions, 315–317
 - Java versus, 322
 - objects, 311
 - older browser support for, 310
 - programming code, 309
 - programs, inserting, 310
 - text display, 311–312
 - variables, creating, 312
 - window, closing, 319–320
 - window, opening, 318
 - window appearance, defining, 318–319
 - writing, programs, 309
 - The JavaScript Source, 336
 - JPEG format, 297
- **L** ●
- Lego Mindstorms NXT, 339–340
 - less than (<), 140
 - less than or equal to (<=), 140
 - Liberty BASIC. *See also* BASIC
 - AND operator, 125
 - array size, defining, 176
 - arrays, 176–178
 - arrays, storing data in, 177
 - binary searching, 236–237
 - bubble-sort algorithm, 220–221
 - comments, 101
 - compiler, installing, 365
 - Debugging window, 244
 - defined, 2
 - dynamic arrays, 182–184
 - FOR-NEXT loop, 145
 - FOR-NEXT loop backward count, 149
 - FOR-NEXT loop incremental count, 148
 - function, creating, 171–172
 - IF-THEN-ELSE statement, 130
 - insertion sort, 217–218
 - learning, 66–67
 - mathematical functions, 109
 - number conversion into string, 117
 - parameter list, 165
 - passing parameters, 165–168
 - precedence, 106–107
 - programs, stepping through, 244–245
 - pull-down menu, creating, 263–264
 - quicksort algorithm, 227–229
 - SELECT CASE statements, 134–135
 - sequential search, 233–234
 - shell-sort algorithm, 224–225
 - string conversion into numbers, 114–115
 - subprograms, creating, 158–160
 - subprograms, examples, 159, 160
 - subprograms, naming, 159
 - subprograms, 159
 - syntax errors, 240
 - two-dimensional arrays, 190
 - user interface creation, 259–260
 - value range checking, 138–139
 - variable declaration as string, 111–112
 - variables in, 97–98
 - WHILE loop, 151
 - libraries, subprogram, 87–88
 - life cycle. *See also* programs
 - development, 45
 - maintenance, 46
 - upgrade, 46–47
 - linkers, 43
 - Linux programs, 332, 348
 - LISP, 42
 - list boxes. *See also* controls; user interfaces
 - contents, editing, 271
 - defined, 270
 - illustrated, 271
 - multiple item selection, 272, 275
 - with REALbasic, 270–273
 - with Revolution, 273–275
 - lists
 - creating, 291–295
 - definition, 291, 293–295
 - ordered, 291, 292–293
 - types, 291
 - unordered, 291–292
 - local user group, 336
 - logic errors. *See also* debugging; errors
 - cause, 243
 - defined, 240, 243
 - detection difficulty, 244
 - occurrence, 243
 - stepping line by line, 244–245

tracing through programs, 245–246
watching variables, 246

looping instructions. *See also* instructions
defined, 82
illustrated, 82, 83
using in program, 84, 85

loops
cleaning, 254–256
common structures, 351–352
conditions, 144
creating, 143–144
defined, 143
endless, 154–156
fixed number, 144–150
FOR-NEXT, 144
trade-offs, 144
UNTIL, 152–154
WHILE, 150–152

• M •

McAfee, 346
machine language, 20, 52
Macintosh programs, 331
maintenance cycle, 46
malware, fighting, 346
mathematical formulas, 108
mathematical functions
C++, 109–110
defined, 108
Liberty BASIC, 109
list of, 108
REALbasic, 110
Revolution, 110–111
mathematical operators
list, 104
parentheses, 107–108
precedence, 105–107
memory addresses, 91
menus, pull-down, 263–264
MetaCard, 72
methods. *See also* REALbasic;
subprograms
defined, 160
defining as polymorphic, 210
examples, 160, 162
object, writing, 199–200
running, 161
modulo (mod), 104

Mozilla, 348
multi-dimensional arrays. *See also* arrays
C++, 192
as grids, 189
Liberty BASIC, 190
REALbasic, 191
Revolution, 193
two-dimensional, 189–193
types, 189
multiplication (*), 104

• N •

names, variable, 93
National Centre for Computer
Animation, 343
.NET framework, 43, 56
NetBeans program, 330, 331
newsgroups, 337
niche-market programming, 348–349
not equal to (<>), 140
NSBASIC, 332
numbers
converting into strings, 117
converting strings into, 114–116
FOR-NEXT loop count, 147

• O •

object files, 43
Objective-C, 10
object-oriented languages, 211
object-oriented programming
classes, 198–199
defined, 89, 195
encapsulation, 196–203
features, 90
inheritance, 204–208
polymorphism, 208–212
problems solved by, 196
objects
code-sharing, 89
creating, 198
creating, from class, 200–203
creation guidelines, 202
data, hiding/exposing, 203
data isolation, 197
defined, 89

objects (*continued*)

dividing programs into, 88–90

JavaScript, 311

methods, 199–200

one-based array, 176. *See also* arrays

Open Source Initiative, 348

open-source programming, 347–348

optimization

algorithm selection, 251

compiler, 258

data structure selection, 249–250

defined, 249

source code, 252–257

OR operator. *See also* Boolean operators

defined, 126

truth table, 127

values, 127

ordered lists. *See also* lists

creating, 292–293

nested, 293

tags, 293

• p •

paragraphs

defining, 285–287

illustrated, 287

text, reading, 287

parameter lists

C++ function, 169

defined, 165

Liberty BASIC, 165

REALbasic, 168

Revolution, 170

parentheses, in mathematical operators,

107–108

Pascal, 25

passing parameters. *See also*

subprograms

C++, 169

Liberty BASIC, 165–168

REALbasic, 168

Revolution, 170

p-code

defined, 54

disadvantages, 55

Java use, 55

programs, decompiling, 55

running programs compiled into, 55

use determination, 56

Pixar Animation Studios, 343

Planet Source Code, 336

platforms, 32

plus sign (+), 104, 312

PNG format, 297

Pocket C, 332

polymorphism. *See also* object-oriented

programming

allowing, 208

in C++, 209–210

defined, 90

in REALbasic, 211–212

rewriting code, 208

portability

assembly language lack of, 21

in program design, 39

RAD language programs, 28

PowerBuilder, 334

precedence, 105–107

program design. *See also* writing programs

building blocks, 83–85

critical elements, 37–38

cross-platform issues, 39

knowledge of users, 38

portability issues, 39

problem solved by, 38

process, 77–85

program parts, 79–83

programming skill in, 39–40

steps, 77

target computer, 39

top-down, 44

Programmer's Paradise, 335

programming

battling robot, 338–339

building blocks, 83–85

as career, 11

curiosity requirement, 17

desire requirement, 17

for fun, 10

games, 337–338

goals, 22

handheld computers, 332

imagination requirement, 17

as intellectual challenge, 11

knowledge requirements, 17

learning for first time, 10–17

- object-oriented, 89, 195–207
 - as problem-solving, 14–15
 - reasons for learning, 10–13
 - skill in program design, 39–40
 - spaghetti, 77–79
 - as time-consuming, 15–16
- programming careers
 - computer animation, 343
 - computer games, 341–343
 - encryption, 344–345
 - fighting malware, 346
 - hacking, 346–347
 - Internet, 345
 - niche-market, 348–349
 - open-source projects, 347–348
 - own software, 349–350
 - teaching, 349
- programming languages
 - “best,” 34–35
 - C, 22–24
 - C-derived, 24
 - choosing, 41
 - COBOL, 22, 24–25
 - database, 29–30, 333–334
 - defined, 19
 - FORTRAN, 22, 24
 - general-purpose, 42
 - high-level, 24–26
 - hybrid object-oriented, 211
 - HyperCard, 72
 - LISP, 42
 - MetaCard, 72
 - multiple, 43
 - object-oriented, 211
 - Pascal, 25
 - purpose, 20
 - REALbasic, 2, 27
 - “religious” wars, 35
 - scripting, 30–32
 - SNOBOL, 42
 - specialized, 42
 - speed, 257
 - type-checking, 95
 - type-safe, 95
 - VBA, 31
 - Visual Basic, 27
 - Web page, 32–34
- programming resources
 - battling robot programming, 338–339
 - Clarion, 333–334
 - Core Wars, 337–338
 - FileMaker, 334
 - Lego Mindstorms NXT, 339–340
 - Linux, 332
 - local user group, 336
 - Macintosh, 331
 - mail-order houses, 335
 - PowerBuilder, 334
 - source code, 335–336
 - SQL, 334
 - Usenet newsgroups, 337
 - Windows, 330–331
- programs
 - assembly language, 20–21
 - beta copies, 242
 - branching instructions, 80–81
 - buying, 13
 - C, 24
 - C++, 70–71
 - cross-platforms, 32
 - debugging, 239–247
 - decompiling, 55
 - defined, 19, 75
 - distributing, 60–61
 - dividing into objects, 88–90
 - functioning definition, 42–44
 - functioning of, 13–16
 - Help-file-creation, 60
 - input/output, 13–14
 - installation, 60–61
 - Java, 33
 - life cycles, 44–47
 - Linux, 332
 - looping instructions, 82–83
 - machine language, 20
 - Macintosh, 331
 - own, selling, 349–350
 - parts, 79–83
 - p-code, 55
 - prototypes, 40–41, 257
 - RAD language, 29
 - REALbasic, 68
 - reliability, 195–196
 - Revolution, 72–73

programs (*continued*)
 securing, 247
 sequential instructions, 79–80
 small, 76
 stepping through, 244–245
 structure, 75–90
 subprograms, 85–88, 157–174
 tracing through, 245–246
 Windows, 330–331
 writing, 13, 37–47

prompt dialog boxes. *See also* dialog boxes
 creating, 314–315
 defined, 314
 illustrated, 314
 user data, 315

protected data, 203

prototypes
 choices, 257
 defined, 40
 function, 40
 as guideline, 41
 languages for creating, 41

pseudocode
 defined, 42
 examples, 42, 43, 44
 instructions, 43
 using, 44
 writing, 44

pull-down menus
 creating, 263–264
 defined, 263

• Q •

quicksort. *See also* algorithms
 in C++, 229–231
 defined, 227
 functioning, 227
 illustrated, 227
 in Liberty BASIC, 227–229
 recursion, 227

quotes
 defined, 285
 highlighting, 287

• R •

RAD programming languages. *See* rapid application development programming languages

radio buttons. *See also* controls; user interfaces
 creating, with HTML code, 307–308
 defined, 268, 299
 event, 306–307
 form, 306–308
 HTML, 306–308
 with REALbasic, 268–269
 with Revolution, 269–270

rapid application development (RAD) programming languages
 benefits, 28
 defined, 27
 drawbacks, 28–29
 examples, 27
 programs, 29
 in prototype creation, 41

RB Garage, 336

REALbasic. *See also* BASIC
 Append command, 184
 array resizing commands, 184
 arrays, 178–179
 Boolean expressions in variables, 121
 check boxes, 268–269
 class creation, 198–199
 collections, 187
 combo boxes, 270–273
 comments, 101
 compiler, installing, 365
 control creation, 265, 266
 cross-platform capabilities, 330
 defined, 2, 67
 DO-LOOP, 153
 dynamic arrays, 184–187
 FOR-NEXT loop backward count, 149–150
 FOR-NEXT loop incremental count, 148
 function, creating, 172–173
 inheriting classes using, 206–208
 Insert command, 184
 interface, 27
 learning, 67–69

- lines, 68–69
 - list boxes, 270–273
 - mathematical functions, 110
 - methods, 160–163
 - MsgBox command, 162
 - number conversion into string, 117
 - object creation from class, 202–203
 - parameter list, 168
 - passing parameters, 168
 - polymorphism in, 211–212
 - programs, 68
 - radio buttons, 268–269
 - Redim command, 184
 - relational operator `SELECT CASE` statement, 141
 - Remove command, 184
 - string conversion into numbers, 115
 - subprograms, creating, 160–163
 - text boxes with, 276–277
 - two-dimensional arrays, 191
 - variable declaration as string, 112
 - variables in, 97–98
 - window creation, 262–263
 - writing, 67–68
 - recursion, 227
 - relational operators. *See also specific operators*
 - checking, 140–142
 - defined, 120
 - reliability, software, 195–196
 - Revolution
 - advantages, 71
 - AND operator, 125
 - arrays, 181–182
 - Boolean expressions in variables, 122
 - check boxes, 269–270
 - combo box, 275
 - comments, 102
 - defined, 2
 - defining constants in, 100
 - design, 2
 - dynamic arrays, 188–189
 - FOR-NEXT loop, 146–147
 - FOR-NEXT loop backward count, 150
 - FOR-NEXT loop incremental count, 149
 - function, creating, 174
 - handlers, 164–165
 - IF-THEN-ELSE statement, 131
 - learning, 71–73
 - List Field, 273–275
 - mathematical functions, 110–111
 - Menu Builder, 264
 - number conversion into string, 117
 - parameter list, 170
 - passing parameters, 170
 - program illustration, 73
 - program lines, 73
 - programs, 72–73
 - radio buttons, 269–270
 - Runtime, 365
 - Script Editor, 265
 - string conversion into numbers, 116
 - subprograms, creating, 164–165
 - switch statement, 137–138
 - text boxes with, 277–278
 - trial version, 72
 - two-dimensional arrays, 193
 - UNTIL loop, 154
 - user interface, 73
 - variable declaration as string, 112
 - WHILE loop, 152
 - RSA, 345
 - run-time errors. *See also* debugging; errors
 - defined, 240, 242
 - detection, 243
 - Runtime Revolution, 365
- S ●
- scripting programming languages. *See also* Revolution
 - benefits, 71
 - defined, 30
 - problems, 31
 - uses, 31
 - searching algorithms. *See also* algorithms
 - binary, 235–238
 - defined, 232
 - selecting, 238
 - sequential, 233–235
 - wrong, choosing, 232
 - security, program, 247

- SELECT CASE statements. *See also*
 - branching statements
 - checking relational operators, 140–142
 - defined, 133
 - example, 133
 - Liberty BASIC, 134–135
 - REALbasic support, 253
 - value range checking, 138–140
- sequential instructions. *See also* instructions
 - defined, 79
 - illustrated, 80
 - using in program, 84
- sequential search. *See also* searching
 - algorithms
 - advantage, 234
 - C++, 234–235
 - defined, 233
 - Liberty BASIC, 233–234
 - for small lists, 233
 - start, 233
- shell sort. *See also* algorithms; sort
 - in C++, 225–226
 - defined, 223
 - illustrated, 223
 - in Liberty BASIC, 224–225
 - principle, 223
 - steps, 223–224
- SNOBOL, 42
- software. *See* programs
- software patches, releasing, 46
- Sophos, 346
- sort
 - algorithm selection, 231
 - bubble, 220–223
 - built-in algorithm, 231–232
 - insertion, 216–219
 - quicksort, 227–231
 - shell, 223–226
- SORT command, 231–232
- source code
 - assembly language, 20
 - commenting, 100–103
 - defined, 19, 76
 - fine-tuning, 252–257
 - Java applets, 321
 - resources, 335–336
 - reusing, 204–208
 - rewriting, 208–212
 - step-by-step instructions, 76
- spaghetti programming. *See also* programming
 - code, 78
 - defined, 77
 - illustrated, 78
 - logic, following, 79
- SQL, 334
- stepping, 57
- STR\$ function, 117
- strings
 - concatenating, 112–113
 - converting numbers into, 113–116, 117
 - declaring variables as, 111–112
 - defined, 111
 - identification, 111
 - manipulating, 111–113
 - storage/retrieval in array, 179
- subprograms. *See also* programs
 - advantages, 87
 - C++, creating, 163–164
 - calling, 167
 - creating, 158–165
 - creation illustration, 86
 - defined, 85, 157
 - event, 265
 - illustrated, 86
 - Liberty BASIC, creating, 158–160
 - libraries, 87–88
 - naming, 159
 - parameter list, 165
 - passing parameters, 165–170
 - REALbasic, creating, 160–163
 - Revolution, creating, 164–165
 - running, 165–170
 - storing, in separate files, 88
 - tasks, 170
 - uses, 158
 - virtual, 89, 90
- subtraction (–), 104
- switch statement
 - C++, 135–137
 - Revolution, 137–138
- Symantec, 346
- syntax errors. *See also* debugging; errors
 - catching, 241
 - defined, 240

- detection, 241, 242
- Liberty BASIC, 240
- misspellings, 240, 241
- system requirements, CD, 365–366

• T •

- tags, HTML
 - attributes, 289–291
 - background picture, 299
 - blank Web page, 283
 - body, 284
 - check box, 303–305
 - color, 289–290
 - comments, 284–285
 - defined, 282
 - defining text with, 285–288
 - definition list, 294
 - ending, 282
 - external hyperlink, 295–296
 - header, 283–284
 - heading, 285–286
 - hyperlink color, 290–291
 - internal hyperlink, 296
 - mixed-up, 282
 - ordered list, 293
 - paragraph, 286–287
 - quote, 287
 - radio button, 306–308
 - text alignment, 289
 - text box, 301–302
 - text emphasis, 288
 - title, 284
 - unordered list, 291–292
 - Web page picture, 298
- teaching, 349
- text
 - aligning, 289, 298
 - colors, 289–290
 - data type, 95
 - displaying, in JavaScript, 311–312
 - emphasis, 285, 288
- text boxes. *See also* controls; user interfaces
 - creating, 301–302
 - defined, 276, 299
 - form, 301–302

- labels, 276, 278
 - with REALbasic, 276–277
 - with Revolution, 277–278
- Therac-25, 242
- top-down design, 44
- TRACE command, 245–246
- tracing, 245–246
- tracking, 245–246
- Trend Micro, 346
- truth tables
 - AND operator, 125
 - defined, 125
 - OR operator, 127
- two-dimensional arrays. *See also* arrays
 - C++, 192
 - Liberty BASIC, 190
 - REALbasic, 191
 - Revolution, 193
- type-checking, 95
- type-safe languages, 95

• U •

- unary operators, 105
- unordered lists, 291–292
- UNTIL loop. *See also* loops; WHILE loop
 - C++, 153–154
 - defined, 152–153
 - REALbasic, 153
 - Revolution, 154
- upgrade cycle, 46–47
- Usenet newsgroups, 337
- user interfaces
 - check boxes, 268–270
 - combo boxes, 270–275
 - command buttons, 266–268
 - controls, 264–278
 - creating, 259–278
 - creating, in Liberty BASIC, 259–260
 - defined, 259
 - on forms, 299–308
 - list boxes, 270–275
 - pull-down menus, 263–264
 - radio buttons, 268–270
 - text boxes, 276–278
 - windows, 261–263

• V •

VAL function, 114

variables

- in Boolean expressions, 122–123
- Boolean expressions in, 121–122
- in C++, 98
- creating, 92–93
- data, 93
- data type, assigning, 94–97
- declaring, 96–97, 98
- declaring, as strings, 111–112
- defined, 92
- inappropriate data, 95
- initial values, 97
- JavaScript, creating, 312
- in Liberty BASIC, 97–98
- naming, 92, 93–94
- parameter list, 165
- in REALbasic, 97–98
- uses, 92
- values, assigning, 94, 97–98
- watching, 246

Visual Basic, 27, 51

Visual Basic Code Source, 336

Visual Basic for Applications (VBA), 31

Visual Express, 330

• W •

watching

- defined, 58
- variables, 246

Web Jobs USA, 345

Web page programming languages. *See also* programming languages

- advantages/disadvantages, 34
- HTML, 32–33
- interpreters for, 54
- types of, 33

Web pages

- animating, with Flash, 310
- graphics, 297–299
- interactive, with JavaScript, 309–320

- Java applets, 321–326
 - linking to spot on, 296–297
- Webroot Software, 346
- WHILE loop. *See also* loops
 - C++, 152
 - defined, 151
 - Liberty BASIC, 151
 - Revolution, 152
 - stopping, 152
 - UNTIL variation, 152–154
- whole numbers, 95
- windows. *See also* user interface
 - appearance, 318–319
 - closing, 319–320
 - creating in REALbasic, 262–263
 - defined, 261
 - designing, 261–263
 - displaying, 262
 - opening, 318
 - pull-down menus, 263–264
 - purposes, 261
 - size, defining, 318–319
- Windows Mobile, 332
- Windows programs, 330–331
- writing programs
 - design before, 37–38
 - in editors, 50–51
 - Help files, 59–60
 - JavaScript, 309
 - large, 158
 - program function definition, 42–44
 - program language selection, 41
 - prototyping, 40–41
 - technical details, 40–44

• X •

Xtras.Net, 335

• Z •

zero-based arrays, 176, 179. *See also* arrays

Z-Write, 10–11