

INDEX

A

ACM. See Association for Computing Machinery
 actual parameters, 205–208
 addition example, while loop, 94–97
 age example, compound statement/block of
 statements, 63
 age example, if statement, 72–73
 ampersand, 46
 “and” operator, 58–61
 arithmetic expressions, 23–31
 array, character frequencies, 173–175
 array, definition of, 158–159
 array errors, 163
 array, in struct, 275–279
 array initialization short-cut, 162
 array, maximum value, 166–167
 array, mode of, 167–173
 array, necessity of, 175–177
 array, value, counting appearance of, 165–166
 array, value in, 163–165
 array, variable assessment in, 159
 arrays, loops and, 159–161
 arrays, of strings, 257–261
 assignment operator, 42
 assignment statement, 10–12
 Association for Computing Machinery (ACM), 140

B

bank account examples, pass by reference, 228–233
 bank account examples, trace through, 233–234
 bank example, menu driven program, 120–121
 block of statements, 61–64
 Boolean expressions, 59, 134
 Boolean operators, 59
 braces, 63–64, 133
 break statement, 113–114

C

calculator example, switch statement, 82–84
 calling functions, 196–197
 casting, 26
 character constant, 46
 character counting, strings and, parameter to function, 250
 character frequency, array example, 173–175
 character frequency, arrays of strings, 260–261
 character literals, 42–43
 character printing, multiple times, void function, 209
 comma operator, 112–113
 comments, 4, 43, 152
 comp_grade function, 203–205
 compilation, 44–45
 compiler, 2
 compile-time error, 38
 compound boolean expressions, 58–62
 compound expressions, mistakes, 60–61
 compound statement, 61–64
 conditional expressions, 50–52
 constants, use of, 17–19
 continue statement, 116
 cost, tax and, sample program, 53–55

D

data types, 8
 decrement operators, 40–42, 106–107
 #define, 18
 distance sample program, 37–38
 double loop, 121
 doubles, 22
 do-while loop, 102–104
 dynamic memory allocation, 286–289

E
 ease of use, software quality, 150
 efficiency, software quality, 150
 else clause, 70–71
 empty statement, 64–66
 equal signs, double, 52
 equal signs, single, 52, 55–56
 errors, programming, 45–47
 escape sequences, 22–23
 expressions, printing the values, 12–14
 extendibility, software quality, 150
 external quality factors, 150–151

F
 feet to mile conversion sample program, 14–17
 file, 138
 file, closing, 140
 file input, reading from, 139
 file, writing to, 146–148
 first C program, 3–4
 first string, array of, 259–260
 flag-controlled loop, 100–101
 floats, 22
 Flow Chart, 53
 for loop, 104–113
 formal parameters, 205–208
 fprintf, 138–148
 free function, 288–289
 frequencies, array example, 173–175
 fscanf, 138–148
 function main, 5
 function printf, 5
 function specifications, 31–32
 function, writing, 197–199
 functions, mistakes, 203–205
 functions, program layout and, 199–203

G
 guessing game example, do-while loop, 103–104
 guessing game example, flag-controlled loop, 100–101
 guessing game example, while loop, 99–100

I
 identifiers, 9–10
 IDEs. See Integrated Development Environments
 if statement example, age, 72–73
 if statement example, box dimensions, 78–81
 if statement example, leap year, 76–78
 if statement example, Price is Right guessing game, 73–75
 if-else statement, 56–58
 implicit nesting, multiple options, 68–70
 increment operators, 40–42, 106–107
 increment statement, 11–12
 indentation, 153–155
 infinite loops, 132–133
 information hiding, software quality, 150
 initialization of variables, 15–17
 inner loop, 124–125
 insert, front, linked list, 295–298
 insert, in order, linked list, 298–300
 integer division, 25–28, 46
 Integrated Development Environments (IDEs), 2–3
 internal quality factors, 150–151

K
 Kakuro Konundrum, 144–146

L
 leap year, if statement example, 76–78
 linked list, definition, 289–290
 linked list, functions, 291–294
 linked list, insertion, 295–300
 linked list, value in, 293
 list, numerical order in, 293–294
 logical error, 39–40
 loop control elements, 113–117
 loop index, 135–136
 loop, menu printing, 136
 loops, arrays and, 159–161

M
 Magic Squares, two-dimensional array example, 187–193
 malloc function, 287
 matching else problem, 70–71
 math library, 34–38
 Max3, actual vs. formal parameters, 205–208
 maximum value, list of numbers, 110–112
 memory, statically allocated vs. dynamically allocated, 286–289
 menu driven programs, 118–121
 minimum value, list of numbers, 110–112
 mod operator, 28–31
 modularity, software quality, 150
 multiplication table, two-dimensional array example, 182–184
 mypow function, 199–200

N
 nested if statements, 66–70
 nested loops, 121–131
 nesting, 66–70
 “not” operator, 61–62
 null character, 242
 Num_times, 90–93

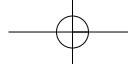
O
 off by one error, 134–135
 one-dimensional arrays, basic use, 158–163
 one-dimensional arrays, examples, 163–177
 operator precedence, 61
 “or” operator, 58–61
 order of execution, 15–17
 order of operations, 24
 outer loop, 124–125
 output formatting issues, 21–23
 overview, C program, 2–6

P
 Pascal’s Triangle, two-dimensional array example, 184–186
 pass by reference, Bank account examples, 228–234

pass by reference, motivation for, 222–226
 pass by reference parameters, 226–230
 pass by reference structs, 269–270
 pass by reference structs, alternate syntax, 270
 pass by value, 267–269
 pay calculation example, 57–58
 percent code, for types, 12, 46
 percent code, incorrect, 21–22, 46
 pointers, 223–225
 power function, 198–199
 preprocessor directives, 4
 pre-written C functions, 31–38
 Price Is Right guessing game, if statement, 73–75
 Price Is Right sample program, 36–37
 prime number testing example, 115–116, 128–131
 printf, 12–13, 138
 printing, in loop, 136
 printing value, multiple expressions, one printf, 13–14
 printing value, one variable, 12–13
 program layout, functions and, 199–203
 programming contests, 140–146
 programming errors, common, 45–47
 programming style, 44, 151–152, 156

R
 rand function, 32–34
 random number generator, 33
 readability, software quality, 150, 152
 reading in, valid test scores example, 117
 real number division, 26–28
 relational operators, 50–52
 reliability, software quality, 150–151
 return values, 5–6
 reusability, software quality, 150
 reverse function, 253–256
 Ritchie, Dennis, 2
 run-time error, 38

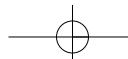
S
 scanf, 19–21, 138
 scrabble example, switch statement, 84–85
 security, software quality, 150
 semicolons, #define statement, 46
 semicolons, end of statement, 46



semicolons, unintended, 133
 sentinel controlled loop, 97–99
 separate loops, 121
 short-circuit evaluation, 71–72
 shorthand assignment, 107–108
 simple if statement, 52–56
 Skippy and His New Jetski, 141–144
 software quality, 150–151
 spacing, 152–153
 standard library (`string.h`) functions, 243–249
 stars example, backwards right triangle, 126–127
 stars example, left-justified right triangle, 127–128
 stars example, right triangle, 125–126
 statements, order of execution, 46
 statically allocated memory, 286–289
`stmt1`, 71
`stmt2`, 71
`stmt3`, 71
`streat` function, 247–249
`stremp` function, 245–247
`strep` function, 243–245
 string, first, finding, 259–260
 string literals, 42–43
 string variable, 242–243
`string.h` library, 243–249
 strings, arrays of, 257–261
 strings, passing as parameters to function, 249–256
 struct, array in, 275–279
 struct definition, 264–265, 290
 struct examples, pass by value and reference, 270–275
 struct, in struct, 279–283
 struct, passing into function, 267–270
 struct variable, 265–266
 structures, simple example program, 266–267
 style, programming, 44, 151–152, 156
 sum example, for loop, 108–109
 swap function, 226–228
 swap function, trace through of, 227–228
 switch statement, 81–85
 syntax, assignment statement, 11

T

tip chart example, for loop, 109–110
 trace through, 236–239



tracing code, pass by reference and, 234–239
 tracing examples, 123–124
 tracing examples, inner loop, outer loop, 124–125
 tracing questions, 234–235
 truth table, Boolean expressions, 59
 two-dimensional array example, Magic Square, 187–193
 two-dimensional array example, multiplication table, 182–184
 two-dimensional array example, Pascal’s Triangle, 184–186
 two-dimensional arrays, basic use, 180–181
`typedef`, for structs, 266

U

unterminated string, 46
 uppercase function, 250–252

V

valid identifiers, 9–10
 variable, in array, 159
 variable names, 152
 variables, 8–9, 46–47
 variables, initializing values, 46
 variables, printing values, 12–14
 variables, swapping, 222–223
 void function, 208–209
 void function, example, printing shapes, 209–220

W

while loop, 88–101
 white space, 44–45
 word search, 258–259
 words, set of, storage, 258
 writing functions, 197–199