Index

Note: Page numbers in **boldface** indicate where key terms are defined.

A

accelerate() method, 186–188, 190, 192
accumulator, 87
actual parameters, **143**
add() function, 168
addition operator (+), 19, 21
ADDRESS_LINE1 constant, 144
ADDRESS_LINE2 constant, 144
ADDRESS_LINE3 constant, 144
address operator (&), 163, 165
age variable, 163
algorithms, **126**
aliases, **161**
All apps button, 7
All Programs, Accessories, Notepad command, 7
All Programs, Accessories, Visual Studio 2012 Express, Developer Command Prompt for VS2012 command, 7
amount formal parameter, 151
ampersand operator (&), 161
AND logic, 63
AND operator (&&), 45–46, 47, 63
AnswerSheet.cpp file, 85
Answer Sheet program, 85
answer variable, 20, 90
arguments, **143**, 159–167
  changing value of, 162
  pass by address, 162–167
  pass by reference, 161–162
arithmetic operators, **18**–19
arrays, **94**
  accessing elements, 97
  assigning values to elements, 96–97
  bubble sort, 128–134
  constants, 128
  counting elements in, 132
  data types, 94
  declaring, **94**–96
  double data type, 95
  elements, 94
  as formal parameter, 158
  garbage values, **96**, 98
  initializing, 96–97
  int data type, 94
  memory allocation, 94–95
  multidimensional, 135–138
  named constants, 98
  naming, 94
  one-dimensional, **135**
  parallel, 103–106
  passing array and array element to, 156–159
  passing by value, 157
  passing to functions, 156–159
  primitive data types, 95
  printing out values, 158
  relationship between elements, 103–106
  searching for exact match, 100–102
  single-dimensional, **135**
  sorting records, 126
  square brackets ([ ]), 94
  staying within bounds, 98
  storing data, 132
  string objects, 94, 95
  subscripts, **94**, 97–98, 133
  two-dimensional, **135**–138
ArrayTest.cpp file, 97
assignment operator (/=), 20–21
assignment operator (%=), 20–21
assignment operator (*=), 20–21
assignment operator (+=), 20–21
assignment operator (-=), 20–21
assignment operator (=), 20–21, 153–154
assignment operators, **19**–20
assignment statements, **19**–20, 25, 70, 153–154
  assigning values to array elements, 96
associativity, 20–**21**
asterisk symbol (*), 163
AT&T Bell Labs, 2
attributes, **3**
  programmer-defined classes, 178
  setting values, 179
Automobile class, 184, 188–192
Automobile.cpp file, 191
automobileOne object, 191–192
average variable, 21

B

balance variable, 143, 150–151
bal variable, 170
base class, **184**
BEDROOMS constant, 136
behaviors, **3**
block statements, 51, 72–73, 80
bool data type, 13, 101–102
Boolean comparison, **72**
Boolean operators, **44**
  logical operators, **45**–46
  relational operators, **44**–45
  true or false values, 44

branching statements, 50
  See also decision statements
break statement, 60–61
bubble sort, 128–134
built-in functions, 172–173

C

calculatedAnswer variable, 31–32
calculateWeeklyPay() method, 179
calling methods, **3**
camel case, **13**
case keyword, 60
case sensitivity, 12, **49**
cd command, 7
C++ development cycle
  compiling C++ programs, 7–8
  executing C++ program, 8–9
  writing source code, 6–7
Celsius temperature: string, 25
celsius variable, 24–25
characters, 13–14
char data type, 13
cin object, 25
cin statement, 88
cities array, 94, 97
  memory allocation, 95–96
cityPopulations array, 94, 97
  memory allocation, 95
classes, **3**–4
  associating methods with, 179
  base class, **184**
  custom, **176**
  derived classes, **184**
  inheritance, **184**, 188
  instances, **3**
  programmer-defined, **176**–182
  reusing, 184–192
class keyword, 178, 189
classRank variable, 33
cl compiler, **7**–9, 25
Client By State program, 117–121
  City field, 117
  comments, 119
  main() function, 119–120
  Name field, 117
  State field, 117
client.dat file, 120
close() method, 113

code
  polymorphic, **169**
  reusing, 4
COLUMN_HEADING constant, 36–37
Command Prompt window, 7–8
comments, **23**
  multiline (/* */), 24
  single-line (//), 24
comparing strings, 48–50
comparisons variable, 132–133
compiling C++ programs, 7–8
computeTax() function, 149–151
Compute Tax program, 149–151
constants, **17**
  array size, 128
  global, 35
  local, 35, 37–38
  named, **17**
  naming, 17, 37, 98
  numeric, **17**
  string, **17**
  unnamed, **17**
const keyword, 17
constructors, **181**
controlBreak() function, 120
control break logic, 116–121
convertibleStatus data member, 190–191
convertible variable, 191–192
counters, 87
  controlling loops, 74–75
counter variable, 15, 83
count variable, 75, 80–81, 120–121
cout statement, 5, 25, 88, 154, 164, 187–189, 192
.cpp extension, 7
C programming language, 2
C++ programming language, 2
  case sensitivity, 12, **49**
  data types, **13**–14
C++ programs
  See also programs
  compiling, 7–8
  executing, 8–9
  structure, 4–6
currentSpeed attribute, 186, 190
custom classes, **176**
customerAge variable, 51
Customer Bill program, 142–144

D

data
  encapsulated, 3, **176**
  fields, **110**
  hierarchy, 110
  records, **110**
  user input, 24–25, 31
data_in object, 110–112
data_out object, 112–113
data types, **13**
  arrays, 94
  pointer variables, 163
  variables, 13–14
decimal numbers, number of places after decimal point, 172–173
decision statements, 50
  dual-alternative, 54–55
  dual-path, 54–55
  if else statements, 54–55
  if statements, 50–52
  AND logic, 63
  multipath, 57–58, 60–61
  multiple comparisons, 63–64
  nesting if statements, 57–58
  OR logic, 64
  single-path, **50**–52
  switch statements, 60–61
declaring
  arrays, **94**–96
  functions, 143
  pointer variables, 163
  variables, 12, **14**–15
decrement operator (--)
  postfix form, **70**–71, 73
  prefix form, **70**–71, 73
deduct variable, 36, 78
default constructor, **181**
default keyword, 60
definite loops, **80**
dentalPlan variable, 63–64
dentPlan object, 52
deptName variable, 61
deptNum variable, 61
dereferencing pointers, **165**
derived classes, **184**
  declaring methods, 190
  defining, 185–191
  methods overriding inherited methods, **190**
  in programs, 191–192

`detailLoop()` module, 38–39
detail loop tasks, **35**
Developer Command Prompt for VS2012 command, 7
`didSwap` variable, 132–133
`dir` command, **8**, 9
directory listing, 8
`displayArray()` function, 133–134
division, floating-point, 25
division operator (/), 19, 21
documenting source code, 23–24
`done` variable, 120
`double amount` formal parameter, 150
`double amt` parameter, 162
`double` data type, 13–15, 17, 25, 97, 154, 172
  arrays, 95
`double` keyword, 154
`double& new_sal` parameter, 162
`do until` loops, 82
`do while` loops, 82–83
dual-alternative decision statements, 54–55
dual-path decision statements, 54–55
dynamic memory allocation, **95**–96

E

elements
  accessing, 97
  arrays, 94
  counting, 132
  garbage values, **96**
  passing to functions, 156–159
`else` keyword, 54–55, 57
`empDept` variable, 57–58
`Employee` class
  adding attributes, 178
  creation, 176–178
  get methods, **180**
  methods, 179–182
  private section, 179
  public section, **179**
  set methods, **179**
`Employee.cpp` file, 181
empty string (“”), 96
encapsulated, **176**
encapsulating data and tasks, **3**
`END_LINE` constant, 36, 39, 78
`endl` (newline), 5–**6**, 25
`endOfJob()` module, 39–40
end-of-job tasks, **35**
EOF (end of file), 120
  Ctrl+D (UNIX/Linux) keyboard shortcut, 112
  Ctrl+Z (Windows) keyboard shortcut, 112
eof (end of file) condition, 32
`eof()` method, 112
equal to operator (==), 44–45, 47–49, 101
`evenOrOdd()` function, 146–147
Even or Odd program, 146–147
executable code, **7**–8
executing
  programs, 8–9
  statements, **2**
`.exe` extension, 8
expressions, **19**
  comparing with integer constants, 60–61
extraction operator (>>), **25**, 111

F

`Fahrenheit temperature:` string, 25
`fahrenheit` variable, 24–25
fields, **110**
`fileIOTest.cpp` file, 113
files
  input, 110
  naming, 7
  object code, 7
  opening for reading, 110–111
  opening for writing, 112–113
  output, 110
  reading data, 111–112
  saving, 7
  sequential, **116**–121
  writing data to, 113–114
`fillArray()` function, 132
`finishUp()` function, 121
`firstName` variable, 13, 111, 113, 143
`flag` variable, 128
`float` data type, 13
floating-point division, 25
floating point numbers, number of places after decimal point, 172–173
floating-point values, 13
`FLOORS` constant, 136
flowcharts, 30–34
flow of control, **50**
`fn` variable, 13
`for` loops, 80–81, 88
formal parameters, **144**
  arrays as, 158
`foundIt` variable, 101–102
`freshmanStudentFirstName` variable, 13
`fstream` class, 110
`fuelCapacity` attribute, 186, 190
function declaration, **143**
function prototype, **143**
functions, **5**
  actual parameters, **143**
  arguments, **143**, 159–167
  braces ({}) and, 5
  built-in, 172–173
  declaring, 143
  encapsulated, **176**
  formal parameters, **144**
  headers, **5**, **143**–144
  local constants, 35
  local variables, **35**
  manipulators, **172**–173
  multiple-parameter, 149–151
  multiple versions with same name, 168–171
  overloading, **168**–171
  pass by address, **161**
  pass by reference, **161**
  passing argument by value, **147**
  polymorphic, 169
  return type, **152**
  signature, **169**
  single-parameter, 145–147
  value-returning, 5, 152–154
  without parameters, 142–144

G

garbage values, 87, **96**, 98
`getConvertibleStatus()` method, 190–192
`getFuelCapacity()` method, 186, 190
`getHourlyWage()` method, 180

`getHoursWorked()` function, 152–154
`getLastName()` method, 180
`getMaxSpeed()` method, 186, 190
get methods, **180**, 186
`getReady()` function, 120
`getSpeed()` method, 186, 188, 190–192
`getWeeklyPay()` method, 180–181
global constants, 35
global variables, **35**, 37
`GoKart` class, 184
`grades` array, 103–104
greater than operator (>), 44–45
greater than or equal to operator (>=), 44–45
`grossPay` variable, 52, 55
`gross` variable, 36, 78, 153–154

H

header files, 4, 181
headers, **5**, **143**–144
HelloWorld command, 8
`HelloWorld.cpp` file, 7–9
`HelloWorld.exe` file, 8–9
`HelloWorld.obj` file, 8–9
Hello World program, 4–6, 8
`HIGH` constant, 181–182
high-level programming languages, **2**
`hourlyWage` attribute, 178–182
`HOURS_IN_WEEK` constant, 54–55
`hours` variable, 153–154
`hoursWorked` variable, 55
`housekeeping()` function, 37–38
`housekeeping()` method, 77
`housekeeping()` module, 37–38
housekeeping tasks, **35**

I

`if else` statements, 54–55
`if` keyword, 50, 54
`if` statements, 50–52, 101, 120, 147, 186, 190
  less than operator (<), 51
  nesting, 57–58
  parentheses () and, 50–51
  syntax, **50**
  testing string objects for equality, 52
`ifstream` class, 110–111, 111
`#include` preprocessor directive, 4, 33, 37, 143, 178, 181, 187, 189
`increase()` function, 161–162
`IncreaseSalary.cpp` file, 161–162
increment operator (++)
  postfix form, **70**–71, 73
  prefix form, **70**–71
indirection operator (*), **163**–166
infinite loops, **72**, 75
inheritance, **184**, 188
  `Automobile` class, 190
initialization operator (=), 20
initializing
  arrays, 96–97
  variables, **15**
input, validating with loops, 89–90
input classes, 110
`inputFile.dat` file, 111
input files, 110–111
input operator (>>), **25**
insertion operator (<<), **5**, 113
instances, **3**, 4
instantiation, **3**, 4
`int` data type, 13–15, 17, 19
  arrays, 94
integers, 5–6, 13
interactive input statements, **24**
`int` keyword, 5, 6
invoking methods, **179**
`iostream` header file, **4**, 181

K

keyboard, 25
  EOF character, 112
keywords, **5**, 12

L

`lastName` attribute, 178–181
`lastName` variable, 111, 113, 143
`LENGTH` constant, 157
less than operator (<), 44–45, 98
  `if` statements, 51
less than or equal to operator (<=), 44–45, 98
local constants, 35, 37–38
local variables, **35**, 37–38, 142
logical operators, **45**–46
  precedence and associativity, 46–48
logic error, **51**–52, 72
loop body, **72**–73
  accumulator, 87
  counter, 87
loop control variable, **72**, 78, 85
`loopIndex` variable, 97–98
loops
  accessing array elements, 98
  accumulating totals, 87–88
  assigning values to array elements, 96–97
  braces ({}), 32, 33, 81
  counter-controlled, 74–75, 81
  decrement operator (--), 70
  definite, **80**
  `do until` loops, 82
  `do while` loops, 82–83
  Increment operator (++), 70
  infinite, **72**, 75
  loop body, **72**–73
  loop control variable, **72**, 85
  `for` loops, 80–81
  nesting, 84–85
  priming read, 31, **77**
  reading data, 112
  sentinel values, **72**, 76–78
  sequential statements, 32
  validating input, 89–90
  `while` loops, 72–74
`LOW` constant, 181
low-level programming languages, **2**
lvalue, **70**–71

M

`MailOrder2.cpp` file, 104
Mail Order program, 100–102
Mail Order 2 program, 104–106
`main()` function, 5–6, 31, 35, 37–38, 119–120, 131–132, 143, 147, 153, 157–159, 162, 166, 181, 187
`main()` module, 35
manipulators, **172**–173
`maxSpeed` attribute, 186, 190
`MAX_STUDENTS` constant, 17
`medicalPlan` variable, 63–64
memory allocation
  arrays, 94–95
  `cities` array, 95–96
  `cityPopulations` array, 95

  - dynamic, **95**–96
  - `string` objects, 95
- methods, **3**
  - associating with class, 179
  - calling, **3**
  - constructors, **181**
  - derived classes, 190
  - `Employee` class, 179–182
  - encapsulated, **176**
  - invoking, **179**
  - nonstatic, **179**
  - overriding inherited methods, **190**
  - programmer-defined classes, 179–182
  - public, 180
  - set methods, **179**
- modular programs, 35–40
- modulus operator (%), 19, 21, 147
- `Motorcycle` class, 184
- multidimensional arrays, 135–138
- multiline block (/* */) comments, 24
- multipath decision statements, 57–58, 60–61
- multiple comparisons, 63–64
- multiple-parameter functions, 149–151
- multiplication operator (*), 19, 21, 163
- `myAge` variable, 13
- `MyAutomobileClassProgram.cpp` file, 191–192
- MyEmployeeClassProgram, 180–182
- `MyEmployeeClassProgram.cpp` file, 182
- `myGardener` object, 181
- `My Program` folder, 7
- `myRent` variable, 136
- `MyVehicleClassProgram.cpp` file, 187–188

N

- nameAndAddress`() function, 143-144, 152`
- `name` argument, 179
- named constants, **17**
  - arrays, 98
- namespaces, 4–5
- `name` variable, 36, 38, 78
- naming conventions, 13
- nesting
  - `if` statements, 57–58
  - loops, 84–85
- `net` variable, 36, 78
- newline, 5–**6**
- `newNum` variable, 159
- `new_salary` argument, 162
- `newSalary` variable, 113
- `new_salary` variable, 162
- nonstatic methods, **179**
- Notepad, 7
- not equal to operator (!=), 44–45
- NOT operator (!), 45–46, 112
- null statement (;), **51**–52, **72**
- `NumberDouble.cpp` file, 31
- Number-Doubling program, 31–34
  - body of loop, 32–33
  - flowchart and pseudocode, 30
  - input and output, 33
  - `main()` function, 31
  - starting, 31
- `numberOfEls` variable, 132
- `number` parameter, 147
- `numbers` array, 98–99
- `number` variable, 70, 80–81, 146–147
- `number1` variable, 44–47
- `number2` variable, 44–47
- `num` data type, 14, 31
- numeric constants, **17**
- `num` loop control variable, 73–74
- `NUM_LOOPS` constant, 81
- `numStudents` variable, 88
- `num` variable, 163–164
- `num1` variable, 19–20, 166
- `num2` variable, 19–20, 166

O

- object code, **7**–8
- object-oriented programming, 176
- object-oriented programming languages, **2**–3
- object-oriented programs, **3**
- object-oriented terminology, 2–4
- objects, **3**
- `.obj` extension, 7–8
- `ofstream` class, 110, 112–113
- `oldState` variable, 120
- one-dimensional arrays, **135**
- opening files for reading, 110–111
- `open()` method, 111–113
- operators, **18**
  - arithmetic operators, **18**–19
  - assignment operators, **19**–20
  - assignment statement, **19**–20
  - Boolean operators, **44**–48
  - logical operators, **45**–46
  - precedence and associativity, 20–21
  - relational operators, **44**–45
- `originalNumber` variable, 31–32
- OR logic, 64
- OR operator (||), 45–47, 64
- output classes, 110
- `outputFile.dat` file, 113
- output files, 110
  - writing data to, 112–114
- output operator (<<), **5**
- `Overloaded.cpp` file, 171
- overloading functions, **168**–171
  - signature, **169**
- `OVERTIME_RATE` OVE constant, 54–55
- `overtime` variable, 55

P

- parallel arrays, 103–106
  - subscripts, 104
- parentheses () precedence and associativity, 21
- `partCounter` variable, 85
- pass by address, **161**, **162**–167
- `PassByAddress.cpp` file, 167
- pass by reference, **161–162**
- Pass Entire Array program, 156
  - `main()` function, 157–159
  - `quadrupleTheValues()` function, 157–159
- passing argument by value, **147**
- passing by value, 157
- `PAY_RATE` constant, 153
- `PayrollReport.cpp` file, 37
- Payroll Report program
  - `detailLoop()` module, 38–39
  - endOfJob`()` module, 39–40
  - flowchart, 36–40
  - `housekeeping()` module, 37–38
  - `main()` function, 37
  - sentinel values to control while loop, 76–78

pcnt parameter, 162
PointerPractice.cpp file, 165
pointers, **162**–163
  address operator (&), 163
  dereferencing, **165**
pointer variables, 163–164
polymorphic code, **169**
polymorphic functions, 169
postfix form, **70**–71, 73
precedence, 20–**21**
Precision.cpp file, 173
prefix form, **70**–71, 73
preprocessor, **4**
preprocessor directives, 4–5
price variable, 104
priming loops, 31
priming read, 31, **77**, 120
primitive data types, **13**
  arrays, 95
printBill() function, 169–171
private keyword, 178–**179**
private section, **178**–179
procedural programming, 2–3, 176
procedural programs, **2**–3
  declaring variables and constants, 35
  detail loop tasks, **35**
  end-of-job tasks, **35**
  housekeeping tasks, **35**
  users, **3**
produceReport() function, 120
programmer-defined classes, **176**
  adding attributes, 178
  creation, 176–178
  get methods, **180**
  methods, 179–182
  private section, **178**
  public section, **179**
programming languages, 2–3
programs
  comments, **23**–24
  compiling, 7–8
  derived classes, 191–192
  executing, 8–9
  flowcharts, 30–34
  flow of control, **50**
  logic error, **51**–52
  main() module, 35
  modular, 35–40
  object-oriented, **3**
  procedural, **2**–3
  pseudocode, 30–34
  single-level control break, **116**
  structure, 4–6
prompt, **24**, 25
pseudocode, 30–34
ptr_age variable, 163
ptr_num variable, 164
public derivation, **189**–190
public keyword, 189
public methods, 180
public section, **179**

Q

quadrupleTheValues() function, 157–159
questionCounter variable, 85
QUIT constant, 36, 39, 78, 132

R

raise argument, 162
raise variable, 162
RATE constant, 36
rate formal parameter, 150–151
rate variable, 150–151
reading data
  from input files, 111
  with loops and EOF (end of file), 112
reading files, 110–111
records, **110**
  sorting, 126
relational operators, **44**–45
  comparing string objects, 48–50
  precedence and associativity, 46–48
rent array, 136–138
REPORT_HEADING constant, 36–37
return 0; statement, 33
return statement, 154
return type, **152**
reusing classes, 184–192
reusing code, 4
Ritchie, Dennis, 2

S

salary argument, 162
salary variable, 111, 113, 162
salePrice variable, 13
scope resolution operator (::), **179**
scores array, 131–134
Score Sorting program, 128–130
  displayArray() function, 132–134
  fillArray() function, 132
  main() function, 131–132
  sortArray() function, 132–133
score2 variable, 126
search algorithms, 126
searching arrays for exact match, 100–102
sentinel values, **72**, 76–78, 146
sequences, **23**
  interactive input statements, **24**
sequential files, **116**–121
sequential statements, **23**–24
  loops, 32
setConvertibleStatus() method, 190–192
setFuelCapacity() method, 186, 190
setHourlyWage() method, 179, 182
setLastName() method, 179
setMaxSpeed() method, 186–187, 190, 192
set methods, **179**, 186
setprecision() function, 172–173
setSpeed() method, 186–187, 190–192
signature, **169**, 190
simple statements, 51
single alternative statements, 50
single-dimensional arrays, **135**
single-level control break programs, **116**
single-line comments (//), 24
single-parameter functions, 145–147
single-path decision statements, **50**–52
single-sided statements, 50
SIZE constant, 131–132
someNums array, 157, 159
sortArray() function, 132–133
sorting
  bubble sort, 128–134
  records, 126

source array, 97
source code, **7**
  compiling, 7
  documenting, 23–24
  writing, 6–7
source code files, 7
  syntax errors, **8**
Standard C++ library, **5**
standard input device, 25
Start button, 7
Start screen, 7
state control break variable, 117, 120
statements
  block, 51
  ending with semicolon (;), 5
  executing, **2**
  interactive input statements, **24**
  sequential statements, **23**
  simple, 51
std namespace, 4–**5**
string constants, 5, **17**
  comparing to string objects, 48–49
string objects, 14, 37
  arrays, 94, 95
  ASCII values of characters in, 49
  comparing, 48–50
  comparing to string constants, 48–49
  dynamic memory allocation, **95**–96
  memory allocation, 95
  testing for equality, 52
strings, **14**
  comparing, 48–50
  empty (""), 96
string variables, 15, 17
Stroustrup, Bjarne, 2
stuCount variable, 88
stuID array, 103–104
subscripts, **94**, 97–98, 133
  parallel arrays, 104
subtraction operator (-), 19, 21
supervisorName variable, 57–58
swap() function, 133, 166
swapping values, 126–127
switch keyword, 60
switch statements, 60–61
syntax, **50**
syntax errors, **8**

T

target array, 97
tasks, objects encapsulating, 3
TaxiCab class, 184
Temperature.cpp file, 24–25
temp variable, 132, 166
TestAverage.cpp file, 88
Testing folder, 7
testScore variable, 88
testTotal accumulator, 88
testTotal variable, 87
test1 variable, 21
test2 variable, 21
text editors, 7
tripleTheNumber() function, 159
true or false variables, 13
two-dimensional arrays, **135**–138

U

unary minus operator (-), 19, 21
unary plus operator (+), 19, 21
unnamed constants, **17**
users, **3**
  data input, 24–25, 31
using statement, 178, 187, 189

V

VALID_ITEMS array, 104
VALID_PRICES array, 104
val1 parameter, 166
val2 parameter, 166
value-returning functions, 5, 152–154
values
  relationship between, 44–45
  swapping, 126–127
value variable, 15
variable declaration, **14**–15
variables, **12**–14
  aliases, **161**
  assigning initial value to, 15, 20
  camel case, **13**
  characters, 13, 14
  data types, **13**–14
  declaring, 12, **14**–15
  global, **35**, 37
  initializing, **15**
  invalid names, 12
  local, **35**, 37–38, 142
  memory address, 15
  naming, 12–13
  numeric values, 13–14
  pass by address, **161**, 162–167
  pass by reference, **161**
  swapping values, 126–127
  true or false, 13
  values stored in, 12
Vehicle class, 184–187, 189–190
  as base class, 188
Vehicle.cpp file, 189
vehicleOne object, 187–188
void keyword, 143–144, 147, 152, 179

W

wage argument, 179
weeklyPay attribute, 178, 180
while loops, 32, 38–39, 72–74, 90, 112, 120, 132–134, 146–147, 157–158
  counter-controlled, 74–75
  loop body, 32–33
  sentinel values, 76–78
whole numbers, 13
Windows 7
  opening Command Prompt window, 7
  starting Notepad, 7
Windows 8
  opening Command Prompt window, 7
  starting Notepad, 7
workHours variable, 154
WORK_WEEK_HOURS constant, 179
writing
  to files, 112–113
  source code, 6–7

X

x variable, 132–133, 157

Y

Y string constant, 52

Z

zero (0) indicating end of input, 32