

Bold typeface indicates most important reference

!, See logical operator, not
!=, See not equal operator
%, See remainder operator
%=, See compound assignment operators
&&, See logical operator, and
(), See parentheses
*, See multiplication operator
*+=, See compound assignment operators
--, See decrement operator
-, See subtraction operator
., See class, access using dot operator
/* and */, See comment delimiters
/, See division operator
//, See comment delimiters
/+=, See compound assignment operators
:: 390
;, See semicolon
?:, See conditional operator
[and], See array, brackets
\, See escape character
'\n', See newline character
'\t', See tab character

for finding the largest element (without using
an array) 343
for linear search 458-59, 482
 analysis of 460-61
for linear sort 445-46, 448, 480-81
for finding smallest element in an array 445-
46
for summing a series of terms 152-53, 173
worst case behavior 484
alignment 18
 in **if-else** 108
 left 102, 129
 output 102, 129
 right 102, 129
analysis of
 accelerated rejection algorithm for searching
 487-89
 binary search algorithm 482, 479-80, 488-89
 bubble sort algorithm 481, 483-84
 linear search algorithm 480, 482, 488
 linear sort algorithm 481, 483-84
angle brackets <> 5
argument, See function, argument to
arithmetic expression 8
arithmetic operations 27-28, 35
arithmetic precedence 52-53, 65
arithmetic, mixed mode 66

array 315-82 (Chapter 7), **323-32**, 364-66
 vs. class 497-98, 502-03, 534
 advantages of using 348-49, 365
 as a parameter 335, 365
 as a reference parameter 336-37
 bounds 326, 331-32, 365
 brackets [and] 324, 336
 column subscript 350
 concept of 324
 declaration 326, 365
 for two-dimensional 351-52
 disadvantages of using 348, 365
 elements of 324-25
 expression as subscript 329
 for loop to process 332, 365
 inside class definition 511
 multidimensional 356-58, 366
 of objects 510-12, 535-36
 of type **string** 417-19, 425
 restrictions on 497-98, 534
 row subscript 350
 sending an element as a parameter 331
 sending as a parameter 339
 size 326, 331-32, 365
 subscript 324, 326, 330, 364
 two-dimensional 349-57, 365-66
 as a parameter 353-54
 processing a column 356
 type **char** 328
 type **double** 326-27
 type **int** 326

- array, continued:
 - use in a function 335
 - using a constant as size 332
 - with **for** loops 325
- ASCII code 424,557
- ASCII values 68, 481
 - sorting 470
- assignment operator = 8
- assignment statement 8, 35
- associativity 53, 74
- average, finding 320, 364

- behavior-less class, *See* class, without behavior
- binary search 474-80
- binary search algorithm, analysis of 482, 488-89
- binary search, function for 477-78
- Bloodshed Dev C++ compiler 529, 558-71
 - basics of use 557-58
 - compile and run 565
 - compiling a program 563
 - creating a new file 560-61
 - downloading and installing 568-69
 - editing, saving and retrieving files 569-70
 - entering 560
 - infinite loop 565
 - printing your output 565-66
 - printing your program 565
 - quitting 567
 - running a program 564-65
 - saving a file 560-63
- body of a loop 17
- body of the program 5
- bool** data type 293, 299
- boolean expression 290-93, 295, 299
- boolean operator 291, 298
- boolean variable 293-95
 - using as a condition 294
- bottom-up approach 444, 448, 464-65, 480
- braces { and } 5, 90
- branch 49
- break** statement 285-87, 298
- bubble sort 451-58, 481
 - function for 454-58
- bubble sort algorithm, analysis of 483-84
- buffer 412

- C vs. C++ 493
- C++ 2
 - compiler 558
 - program 1-2, 4, 33-35
 - running 22
- call by reference 223
- call by value 223
- call to a function, *See* function, call to
- called function 262, 296
- calling function 262, 296
- cascading **if** 277
- case**, *See* **switch** statement, **case**
- casting 321, 364
- cctype, functions from,
 - isalnum() 414, 425
 - isalpha() 414, 425
 - isdigit() 414, 425
 - islower() 414, 425
 - ispunct() 414, 425
 - isspace() 414, 425
 - isupper() 414, 425
 - tolower() 415-16, 425
 - toupper() 415-16, 425
- ceil() function 68
- cerr 121-25, 130
- char** data type 69, 73, 387, 410, 414, 424
- character-oriented I/O 410, 424
- cin 86-87, 121, 128, 130
 - reading until input failure (end-of-file) 360
 - with prompts 99
- class 385, 534, **497-517, 483-555 (Chapter 10)**
 - access using dot operator (.) 499, 517, 535
 - accessing members of a nested 506
 - data member 534
 - definition 498-99, 517, 534
 - general form 499
 - location of 512, 537
 - order of 509
 - reusing 506
 - fstream 498
 - general form of class definition 499
 - members of 498, 536
 - nested definition 504-12, 506, 535
 - picture of 506, 508
 - object 385
 - overview 517
 - picture of 498, 500, 501, 503, 504, 506, 507, 508, 511, 517, 519, 534
 - programmer-defined 498
 - string**, *See* **string** class
 - without behavior 497-98
 - without member functions, *See* class, without behavior
- classes **483-555** (Chapter 10)
 - and objects **385, 497-517, 499, 501, 534**
- class **string**, *See* **string** class
- clear() string member function 391, 422
- clear() stream member function 408
- close() stream member function 116, *See also* closing a file
- closing a file 116
- cmath 67, 69, 74, 232
 - header file 385
- cmath.h 67
- column headings 58
- comment 4, 27, 33, 168-69, 174, 224
 - delimiters (symbols) // 4, 33
 - delimiters /* and */ 27, 33
 - mentioning functional dependencies 521
- compilation error 70-71, 74
- compiler 24
 - directive 5, 34
 - warning 71
- compiling a program 24, 37

- compound assignment operators 64, 74
 - precedence of 64
- compound statement 18, 90
- concatenation, *See* string, concatenation operation
- condition 49-50
- conditional 49, 73
- conditional expression 112-13, 129
 - general form 113
- conditional operator ?: 112-13, 129
- const** definition 156,158,173
 - general form 158
- constant, *See* **const** definition
- continue** statement 285-87, 298
- control structures, *See* **do-while** loop, **for** loop, **if** statement, **if-else**, nested **if-else**, **switch** statement, **while** loop
- control variable of a **for** loop 17, *See also* **for** loop, control variable
- cos() function 68, 74
- counter 96
- counting, starting at 0 318, 320
- cout 10-11, 36, 73, 121, 130
 - redirection of 122-24
 - splitting lines 103
- cout.precision() 100-01, 128-29
- cout.setf() 100-02, 128-29
- cout.width() 102, 129
- cursor 11, 99

- data 84, 127-28
 - encapsulation 156
 - file 127
 - creating 571
- data member of a class 499, 534, 536
- data member of an object 536
- data set 127
- data structure 323, 365, 497
 - choosing the best 351
- data type 6, 47-48, 73
 - bool** 293
 - char** 69, 387
 - double** 47, 65
 - float** 66, 73
 - int** 6
 - long int** 65
 - string** 383
- database 493, 495, 539
 - searching 527
- debug printout 170-71, 522
- debugging 41, 24-25, 70-72, 74
- declaration 6, 35
 - file 115
 - in **for** loop header 156-57, 173
 - multiple lines 93
 - with initialization 97
- decrement 30
- decrement operator -- 32-33, 37, 63
 - precedence 52
- default 11

- default** clause, *See* **switch** statement, **default** clause
- defensive programming 261
- defining a constant 156, 158, 173
- deMorgan's Laws 308
- detecting the end of a set of data 258, 357
 - See also* end-of-file method, parameter (header) method, trailer (sentinel) method, user-response method
- end-of-file method 358-61
 - header value method 316-18, 358
 - reading until input failure method 358-61
 - trailer method 358
 - user-response method 358
- division operator / 27, 35
- documentation 172, 174
- dot operator (.), *See* class, access using dot operator
- double** data type 73
 - printing 48
 - range of 65
- double spacing 58
- do-while** loop 260-61, 296
 - general form 260
- driver program 270

- empty string 391, 422
- end of a program 12
- endl 11, 36, 73
 - use in double spacing 58
- end-of-data signal 413
 - end-of-file method 358-61, 366-67, 399, 401, 406
- end-of-file
 - signalling 360-61, 401, 411
- eof() function 361
- error checking 261-62, 319-20, 347
- error trapping 284
- escape character \ 61, 74
- escape sequence 61-62, 74
- execution 25
- execution error 71, 74
- exit() function 287-88, 298, 320
- extraction operator >> 86-87, 411, 424

- fabs() function 68
- field width 100, 129
- file 357
 - declaration 115
 - I/O 114-21, 130
 - advantages 120-21
 - comparison with interactive I/O 398
 - stream 115, 358-59, 361-62, 367
 - as a parameter to a function 363, 367
 - member function is_open() 521
- find() string member function 391-93, 422
- finding the largest element in an array, algorithm for 341-42
- float** data type 66, 73
- floor() function 68

- flowchart 142-43
- for** loop 15-19, 21, 36
 - body 17
 - control variable 17
 - declaring index inside the header 156-57, 173
 - header 17, 31
 - general form 32
 - index 17
 - declaration of 156-57, 173
 - nested 161-68, 173-74, 355-56, 366, 449, 481
 - omission of semicolon after header 18-19
 - using a compound statement in 18
 - using decrement 30
 - variants 30
- formatting output 101
- fstream file stream 115
- function 5, **183-251 (Chapter 5)**
 - advantages 185, 232, 257
 - argument to 188
 - array of string as a parameter 418, 425
 - call to 67-68, 74, 184, 191-93, 230-31
 - efficiency 202-03, 230
 - calling another function 262, 296
 - declaration 195
 - definition 191, 195, 230
 - driver program 207
 - header 187, 231-32
 - for array parameter 335, 338, 365
 - input parameter 218
 - input-output parameter 218
 - interchanging two values 221
 - local variable 196-99, 231
 - location 204-05
 - multiple return statements 199-200
 - output parameter 218
 - parameter 185, 192-93, 202, 230, 232-33
 - actual (real) 188, 231-32
 - formal (dummy) 187-88, 232
 - multiple 200-03
 - passing by reference 223, 233
 - passing by value 223, 233
 - scope of 188
 - sent by value 189, 233
 - transmission by value 189, 233
 - type matching 188
 - parameter transmission by reference 223, 233-34
 - parameter transmission by value 223, 233
 - parameterless 210-13, 233
 - parameterless void 210-12, 233
 - parameterless, main program as 212
 - parameterless, returning a value 212, 233
 - programmer-defined 190-203, 230
 - prototype 191, 193-96, 232
 - for array parameter 335, 338, 365
 - location 205-06, 269
 - reference parameter 213-23, 233-34
 - when to use 223
 - return value 187, 193, 207-08, 212, 233
 - returning an object 516-17
 - returning more than one value 216
 - separate compilation 205-06
 - string as parameter to or return value 397, 423, 426
 - testing 207, 270
 - using a **for** loop 203
 - void** 207-12, 232-33
 - call to 209
 - prototype 209
- functional flowchart 348-49
- general form
 - class definition 499
 - conditional expression 113
 - const** definition 158
 - do-while** loop 260
 - for** loop header 32
 - get() function, call to 411
 - if** statement 50
 - if-else** statement 108
 - if-else** with compound statement 109
 - put() function, call to, 413
 - reading from cin 87
 - switch** statement 282
 - while** loop 89
 - while** loop with compound statement 90
- get() function 410-11, 424-25
 - call to, general form 411
- getline() 399, 406
- goto** statement 288
- hard copy 11
- header file 5, 34, 194, 384-85
 - cctype 410, 425
- header of a **for** loop 17
- header value 316-18, 364
- headings
 - column 58
 - output 57
- I/O redirection 125-27, 130
- identifier 28-29, 34
 - names 174
 - uppercase vs lowercase 29-30, 158, 160
- if** statement 49-50, 73
 - general form 50
- if-else** 106-108, 129
 - ambiguous case 280-81
 - general form 108
 - nested 276-81, 296-97
 - organizing 110
 - vs. conditional expression 113
 - with compound statement 108
 - general form 109
- ifstream file stream 115
- increment 13, 16-17
- increment operator ++ 32-33, 37, 63
- precedence 52

- indentation 170, 174, 277
- indenting 18, 22
 - style 23
 - in **if-else** 108
- index of a **for** loop 17
- infinite loop 89, 91, 128, 456
- initial value 17
- initialization 17
 - in declaration 97
- input 84
- input data 84
- input file 118-20, 407
 - as a parameter 363
- input parameter, *See* function, input parameter
- input stream 86
- insert() string member function 395-96, 423
- insertion operator << 11, 36
- int** data type 6
 - range of 65
- integer division 28, 320-21
- interactive data entry 85-86, 98, 127
- interactive program 528, 539
- interchanging two values 219-21, 449-50, 472-73
- iostream 5, 194, 232
 - header file 385
- I-P-O comments 224-25, 227, 234
- is_open() file stream member function 362, 367, 521
- isalnum() string member function 414, 425
- isalpha() string member function 414, 425
- isdigit() string member function 414, 425
- islower() string member function 414, 425
- ispunct() string member function 414, 425
- isspace() string member function 414, 425
- istream 363, 367
- isupper() string member function 414, 425

keywords 29-30, 34

- left-alignment 102, 129
- length() string member function 390, 421
- library function 67, 186
- limit value, reading 159, 173
- linear search 458-61
 - algorithm, analysis 480, 482, 488
 - function for 459-60
- linear (selection) sort 444-51, 480-82
 - algorithm, analysis of 484
 - function for 448-50
 - problems 451
- linker error 70, 74
- literal 11
- literal string 11, 36, 384
- logic error 71-72, 75
- logical and relational operators 288-89, 298
 - precedence 289
- logical equals operator == 63
- logical operation 263
- logical operator 263, 289
 - and && 267-68, 298-99
 - exclusive-or 290, 308
 - inclusive-or 290
 - or || 263, 298-99
 - not ! 289, 298-99
 - truth tables 290
- long int** data type 65
- loop 15-16, 36, *See also do-while* loop, **for** loop, **while** loop
- loop invariant 141

- machine language 24
- main function 5
- main program 5
- main program header 5, 34
- maximum, *See* algorithm for finding the largest element in an array
- maximum, finding 341
- member function 389, 497-98
- menu 495-96, 524, 539
- mixed mode arithmetic 66
- mnemonic name 6, 169-70
- modular approach 444, 480
- modularization 184
- module 184, 206-07, 230, 273, 334, 348-49, 365
- modulus 28, 35, *See also* remainder
- multiplication operator * 9, 27, 35

- nested **if**, *See if-else*, nested
- newline character '\n' 62, 69, 73
- not equal operator != 63
- null **else** 281
- null statement 19

- null string 391, 422
- number conversions 320-21

- objects 385, 390, 420, **497-517**, 499, 534
 - array of 510-12, 535-36
 - as a parameter 515-16, 538
 - as a parameter to a function 536-37
 - as a reference parameter 537
 - assigning one to another 500, 537
 - changing in a function 514-15, 537-39
 - initialization in declaration 537
 - returned from a function 516-17, 538-39
 - use in a function 512, 536-37
- object-oriented programming 156, 493
- ofstream file stream 115
- open() stream member function 116, *See also* opening a file
- opening a file 116
- operator overloading 387
- order of operations 52
- ostream 363, 367
- output 2, 10, 19
 - alignment 129
 - appearance of 522-23
- output file 116-19
 - as a parameter 363

- output headings 57
- output parameter, *See* function, output parameter
- output stream 10-11, 36

- parallel arrays 330
- parameter, *See* function, parameter
- parameter to a function, *See* function, parameter
- parameter value 316-18, 364
 - reading 319
 - testing for invalid value 319-20
- parameterless function, *See* function, parameterless
- parentheses ()
 - empty 5
 - precedence of 53-54, 74
- phony data value 91-92, 127, 401
- piping 125-27, 130-31
- precedence of operations 52, 65, 74, 289, 291
 - parentheses 289
 - table 53, 65, 290
- printing
 - messages 56, 73
 - numbers 100, 128
 - real numbers 100
- program 2
 - header 4-5
 - listing 19
- programmer-defined identifiers 29-30, 35
- programming style 4
- prompts 128, 98-100
 - separating from output 122
- prototype, *See* function, prototype
- pseudocode 2-3, 7, 15, 34, 46
 - and stepwise refinement 154
 - indenting 47
- public** 499, 534
- put() function 410, 412-13, 424
 - call to, general form 413

- rand() function 212, 242
- random number 212, 242
- readable output 56-59, 100-03, 114, 170, 174
- reading
 - a limit value 159, 173
 - data 84-85, 88, 127-28
 - from cin 85-88
 - general form 87
 - from input file 118-19, 130, *See also* input file
 - from keyboard 85, 128
 - groups of values 359-60
 - to input failure 358-61
 - matching data types 88
- real division 321
- real numbers 66
 - printing 100
- redirection of I/O 30, 125-27, 131
- reference parameter 336, 397, 423, *See also* function, reference parameter
 - string as 426
- relational operators 49, 62-63, 289, 299
- remainder operator % 28,35
- replace() string member function 393-95, 422-23
- return** statement 12, 187, 191, 208, 231
- right-alignment 102, 129
- running a program 25, 37
- run-time error 71

- saving a program 24, 37
- searching 444, **458-64**, 481-82, *See also* binary search, linear search
 - a sorted list 461
 - accelerated rejection 461
- selecting one from a series of alternatives 274-85, 296
- selection sort, *See* linear sort
- selector, *See* **switch** statement, selector
- semicolon ; 7, 34
 - omission after comment 7
 - omission after **if** clause 50
 - omission after **while** condition 90
 - omission after #include 7
- sentinel value 89, 91, 127
- separate compilation, *See* function, separate compilation
- sequential search, *See* linear search
- short-circuit evaluation 292, 299
- sin() function 68, 74
- size() string member function 390, 421
- software 25
- software development cycle 25-26, 37
- sorting **444-58, 480-82**, 444-91 (Chapter 9), *See also* bubble sort, linear sort algorithms, comparison of 454
 - array of strings 470-72
 - ASCII values 470
 - descending order 470, 481
 - noninteger values 470, 481
 - with repeated elements 470
- spacing of program 170, 174
- sqrt() function 67-68, 74, 185-89, 230
- standard function library 5
- standard I/O stream 121-22, 130
 - as a parameter to a function 363
- standard identifiers 29-30, 35
- standard library of functions 67, 74, 186, 230
- stepwise refinement 7, 152-54, 172, 185, 254-57, 347
- stream, member function clear() 408
- string **383-441 (Chapter 8)**
 - array of 417-19, 425-26
 - accessing one character 419
 - as parameter to a function 397, 418, 423
 - as return value from a function 397, 424
 - as reference parameter 423
 - assignment 385, 420
 - class 383, 385, 420, 498, 498
 - data type string::size_type 390, 393, 422
 - member functions 389-90
 - clear() 391, 422

string, continued:

- erase() 394-95
- find() 391-93, 422
- insert() 395-96, 423
- length() 390, 421
- replace() 393-94, 422-23
- size() 390, 421
- substr() 396-97, 423
- npos 391, 422
- objects 390
- string::npos 391, 422
- comparison 387-88, 421
- concatenation + 386-87, 420
- declaration 385, 420
- empty 391, 422
- file I/O 406
- header file 384
- index operation 388-89, 421
- length 390
- member functions, *See* string, class, member functions
- printing to cout 385, 420
- reading from a file 406
- reading from cin 385-86, 420
- reading using getline() 386, 420
- sorting, *See* sorting, array of strings

string data type 385

string header file 385

structured programming 168, 174

structured read loop 94-95, 128

style of programming 168-70, *See also* structured programming

style, indenting 24

subprogram 184

subscript, *See* array, subscript

subscripted variable 324, 364

substr() string member function 396-97, 423

subtraction operator - 27, 35

summation 152-53, 173

summing a series 152-53, 173

switch statement 281-85, 297-98, 525

- break** 282-83, 285, 298
- case** 282, 298
- default** clause 283-84, 298, 526
- general form 282
- restrictions 285
- selector 282, 285, 298

syntax error 70

`system("pause");` 529

tab character '\t' 59-60, 69, 73

tan() function 68, 74

test data 347-48

testing 347-48

text editor 22, 558

token 410

token-oriented I/O 410

tolower() function 415-16, 425

top-down approach 258

- comparison to bottom-up 465, 480
- top-down programming 152, 172-73, 185, 399, 494
- toupper() function 415-16, 425
- tracing 9-10, 17-18, 20-21, 51, 104-05
- trailer value 89, 91-92, 127
- truth tables 290, 299

type casting, *See* casting

type checking 187, 232, 193-94

unary minus 52, 74

unary plus 52

user-response method 258

using namespace std 5

variable 6, 35

void 207-08, *See also* function, **void**

void function, *See* function, **void**

while loop 89-91, 127-28

- compound statement 90
- compound statement, general form 90
- condition 89-90, 127-28
- general form 89

while loop vs. **do-while** loop 259, 274, 296

whitespace characters 386, 399, 411