

&, 15, 140, 167, 168, 191, 214, 302, 342, 358  
 &&, 62, 63, 65-66  
 <>, 378-379  
 <, 42, 62, 63  
 <=, 62, 63  
 <<, 363-364, 369  
 >, 41, 62, 63  
 >=, 62, 63  
 >>, 363-364  
 \*, 17, 167-168, 189  
 \n, 59, 60  
 ^, 358  
 :, 100, 462  
 , (comma), 4, 12, 13, 34, 154, 155, 370-371  
 { }, 3, 6, 7, 26, 29, 46, 84, 154, 155  
 . (decimal), 12  
 . (dot operator), 302, 314, 315, 330  
 ... (ellipsis), 200  
 =, 12, 367-369  
 ==, 42, 62, 63  
 !=, 62, 63  
 ->, 314-315, 330  
 -- (decrement operator), 54-56, 172, 173, 182  
 ( ), 4, 18, 19, 173, 175, 281, 380  
 %, 13, 17, 18  
 . (period), 243  
 ++ (increment operator), 54-56, 173, 175  
 #, 393-58  
 ##, 393-395  
 ?, 365-367  
 \*\*, 4  
 \*\*, 12  
 ; (semicolon), 2, 11, 12, 154  
 /\* \*/, 20  
 //, 21  
 [ ], 139, 151, 216, 248  
 ~, 358, 359, 360-361  
 |, 358  
 ||, 62, 63, 65-66

## A

abort() function, 444-445  
 abs() function, 445

Access modifiers, 349-352  
 acos() function, 425  
 Addressing, 32-bit vs. 16-bit, 475  
 Allocation, dynamic, 402-407, 440-444  
 AND bitwise operator, 358-359, 359-360, 361  
 AND logical operator, 62  
 ANSI C Standard  
   and function arguments, 32  
   keywords, 35  
   library functions, 3  
 API (Application Program Interface), 474-475  
 APIENTRY calling convention, 476  
 argc parameter, 216-217, 219  
 Argument(s)  
   command-line, 215-220  
   definition of, 4, 33  
   function, 4, 32-34  
 Argument list, 34  
   variable-length, 200  
 argv parameter, 216-217, 267-268  
 Arithmetic  
   expressions, 17-20  
   operators, 17  
   pointer, 172-175  
 Array(s)  
   bounds checking and, 140-141, 295  
   definition of, 138, 139  
   function pointer, 397-400  
   indexing, 139  
   initializing, 154-158  
   multidimensional, 151-154  
   one-dimensional, 139-144  
   of pointers, 186-187  
   with pointers, accessing, 176-178, 179-181  
   string as character, 145-150,  
   of strings, 159-162  
   of structures, 303, 305-310  
   unsized, 155-156  
 Arrow operator, 314-315, 330  
 ASCII character  
   codes, values for, 72-73, 88  
   set, 60, 383  
 asctime() function, 435  
 asin() function, 425-426  
 Assembly code, C as replacement for, 100, 129

Assignment(s)  
 and arrays, 141, 142  
 multiple-variable, 367-368  
 shorthand, 368, 369  
 statement, 12  
 type conversion in, 129-131

atan() function, 426

atan2() function, 427

atof() function, 217-218, 445-446

atoi() function, 150, 217-218, 446-447

atol() function, 217-218, 447

ATOM data type, 485

auto storage class specifier, 113, 339, 458-459

**B**

Background color of window, creating, 484-485

Binary stream, 259

Bit-fields, 324-328

Bit-shift operators, 363-364

Bitmaps, 472

Bitwise operators, 358-362

Block of code, 46-48, 52-53

BOOL data type, 478

Borland C++ compiler, 8, 9

break statement, 459  
 in loop, 89-92  
 in switch statement, 95, 98-99, 466

Brush, 484-485

bsearch() function, 402, 448-449

Bubble sort, 143-144

BYTE data type, 478

**C**

.C extension, 8

C. *The Complete Reference* (Schildt), 258

Call by reference, 211-212

Call by value, 211-212

Callback function, 477

CALLBACK calling convention, 477

calloc() function, 440-441

Case sensitivity and C, 3, 12, 35

case statement, 95-96, 98, 99, 459, 466

Cast, type, 132-133

ceil() function, 427

char data type, 10, 459  
 signed and unsigned modifiers with, 108, 109  
 promotion to int, 126  
 variable in place of int, using, 111

Character(s)  
 arrays, strings as, 145

ASCII. *See* ASCII character constants, 12, 120

input, interactive, 233, 235-237

input, line-buffered, 69-74, 233, 234

output with printf(), 12-13, 234

output with putchar(), 233-235

reading and writing in file I/O, 262-268

Class, window, 477

clock() function, 343-344, 436

CLOCKS\_PER\_SEC macro, 435, 436

clock\_t type, 344, 435

Code block, 46-48, 52-53, 230

Comma operator, 370-371

Comments, 20-22

Compilation, conditional, 381-388

Compiler(s)  
 command line, 7  
 compiling C programs with C++, 8  
 error messages and, 8-9  
 header files, 4  
 preprocessor directives, 4-5

Concatenation, 147

Conditional statements, 41

CONIO.H header file, 72, 236

const access modifier, 349-351, 459

Constants, 12, 119-122  
 backslash-character, 58-61  
 character, 12, 120  
 floating-point, 119, 120-121  
 integer, 119, 121  
 numeric, 120-121  
 octal and hexadecimal, 121  
 string, 121

continue statement, 92-94, 460

cos() function, 428

cosh() function, 428-429

cprintf() function, 236, 237, 238

CreateWindow() API function, 485-487

cscanf(), 72, 236

ctime() function, 436-437

CTYPE.H header file, 179, 412

Current location (position)  
 definition of, 259  
 determining, 286  
 to start of file, positioning, 290, 291-292

Cursor, mouse, 484

CW\_USEDEFAULT macro, 486

**D**

Data type(s)  
 basic, in C, 10-11  
 modifiers, C, 107-111

table of all C, 109  
 Windows, 478  
 \_\_DATE\_\_ macro, 392-393  
 Debugging  
   #error directive and, 389, 390-391  
   example programs for, 384-387  
   #line directive for, 389, 390-391  
 Declaration vs. definition, 201  
 default statement, 95, 98, 460, 466  
 #define directive, 229-232, 377-378  
 defined compile time operator, 383, 387-388  
 Definition files, 490  
 DefWindowProc() API function, 489  
 Desktop model in Windows, 471-472  
 Dialog boxes, 473  
 difftime() function, 437  
 Directives, preprocessor, 4-5, 229-232, 388-391  
 DispatchMessage() API function, 489  
 do loop, 84-86, 460  
 Domain error, 425  
 Dot operator, 302, 314, 315, 330  
 double data type, 10-11, 461  
 DWORD data type, 478  
 Dynamic allocation, 402-407, 440-444

## E

#elif directive, 381, 382, 387  
 #else directive, 381, 382, 383  
 else statement, 44-45, 461  
   and code blocks, 46, 48  
   with nested if statements, 75-78  
   target statements and, 48, 51, 52  
 #endif directive, 381, 382  
 enum type specifier, 461  
 Enumerations, 352-355  
 EOF macro, 233, 234, 239, 247, 262, 264, 266-267, 269-270  
 #error directive, 388-389, 389-390  
 Errors  
   and function prototypes, 197, 201-202  
   syntax, 8  
   warning messages and, 8-9  
 exit() function, 220, 449-450  
 EXIT\_FAILURE macro, 449  
 EXIT\_SUCCESS macro, 449  
 exp() function, 429  
 Expressions  
   arithmetic, 17-20  
   definition of, 17  
   type conversions in, 126-128  
 extern storage class specifier, 339-341, 347, 461

## F

fabs() function, 429  
 False and true in C, 41  
 fclose() function, 262, 294  
 feof() function, 269-270, 279, 281  
 ferror() function, 269, 270, 279, 281  
 fflush(), function, 291, 292  
 fgetc() function, 262-267  
 fgets() function, 274-276, 295-296  
 File(s)  
   access modes for, 260-261  
   closing, 262  
   current location in. *See* Current location  
   definition of, 259-260  
   erasing, 290, 291  
   errors in, checking for, 269-274  
   executable, 8  
   extension when naming, 8  
   flushing disk buffer of, 262, 291, 292  
   header, 4  
   linking, 339  
   object, 8  
   opening, 260-261  
   random access to, 285-289  
   reading and writing any type of data in, 279-285  
   reading and writing text, 274-277  
   reading and writing bytes from or to, 262-268  
   renaming, 290  
   source, 8  
   streams and, 259-260  
 FILE data type, 260  
 \_\_FILE\_\_ macro, 392-393  
 float data type, 10-11, 462  
 Floating-point values, 10, 12  
 floor() function, 429-430  
 fopen() function, 260-261, 294  
 for loop, 49-53, 462  
   infinite, 81  
   nested, 87-88  
   variations, 79-81  
 Format specifiers  
   for printf(), 13, 110, 241-243  
   for scanf(), 15, 16, 110, 246-253  
 Forward declaration/reference, 198-199  
 fprintf() function, 275, 276  
   data conversion in, 278-279  
   printing output to screen with, 293  
 fputc() function, 262-263  
 fputs() function, 274, 275-276  
 fread() function, 279-285, 405  
 free() function, 403, 404-407, 441-442  
 fscanf() function, 275, 276  
   data conversion in, 278-279

fsck( ) function, 285-288

ftell( ) function, 286-288

Function(s)

arguments, 4, 32-34, 200-201

callback, 477

calling, 4, 24

creating, 23-26

declaration vs. definition, 201

definition of, 2

formal parameters of, 32-34

forward declaration/reference of, 198-199

general form of, 3, 197

library. *See* Library functions

parameterized, 33-34

passing arguments to, 211-214

pointers, 395-401

prototypes, 24, 26, 196-206

returning pointers from, 204-205

returning values with, 27-31

structures passed to, 304, 313, 315

structures returned by, 304, 312

window, 476-477, 478, 489

fwrite( ) function, 279-285, 405-407

## G

GDI (Graphics Device Interface), 474

getc( ) function, 262-263

getch( ) function, 235-237

getchar( ) function, 71-74, 203-204, 233, 234

getche( ) function, 72-74, 233, 235-236, 294

\_getche( ) function, 236

GetMessage( ) API function, 488-489

gets( ) function, 145-146, 176, 190, 238-240, 295

scanf( ) vs., 150, 248

GetStockObject( ) API function, 484-485

gmtime( ) function, 318, 435, 438

goto statement, 100-101, 462-463

Graphical User Interface (GUI), 474

Graphics Device Interface, 474

## H

.H extension, 4

Handle, 478

HANDLE data type, 478

Header files, 4, 5, 203

Heap (memory region), 403, 440

Hexadecimal

constants, specifying, 121

number system, 60, 121

HGDIOBJ data type, 485

Hoare, C.A.R., 452

HUGE\_VAL macro, 425

HWND data type, 478

HWND\_DESKTOP macro, 486

## I

Icons, 472, 483

IDI\_APPLICATION macro, 484

IDI\_WINLOGO macro, 484

#if directive, 381-382, 386-387

if statement, 41-43, 463

code blocks with, 46-48

else statement with, 44-45, 76-78

nested, 75-78

relational operators in, 41-42

if-else-if ladder, 76-77, 382

#ifdef directive, 381, 382-383, 384-386

#ifndef directive, 381, 383

#include directive, 4-5, 378-379, 380

Indirection, 168, 170

multiple, 188-190

In-line code vs. function calls, 378

int data type, 10, 473

as default function return value, 29

Integer(s)

size in 16-bit vs. 32-bit environments, 10, 107, 108, 475

values for signed and unsigned, 108, 109

variables, 10

Integral promotions, automatic, 126

Interface, command-based, 149-150

I/O

console, 229-253

file, 258-296

redirection, 293-294

*See also* Streams

isalnum( ) function, 413

isalpha( ) function, 413-414

isctrl( ) function, 414

isdigit( ) function, 415

isgraph( ) function, 415-416

islower( ) function, 416

isprint( ) function, 416-417

ispunct( ) function, 417-418

isspace( ) function, 418

isupper( ) function, 418-419

isxdigit( ) function, 419

## J

jmp\_buf type, 451

**K**

- kbhit() function, 236, 237-238
- Keyboard
  - inputting numbers from, 15-16
  - reading characters from, 69-74, 233-240
  - reading strings from, 145-146
- Keywords, C, 35-36, 458-468
  - for basic data types, table of, 10

**L**

- Label, 100, 462
- labs() function, 450
- Library functions, 3-4, 412
  - dynamic allocation, 440-444
  - mathematics, 424-434
  - miscellaneous, 444-455
  - and prototypes in header files, 203
  - string and character, 412-424
  - time and date, 434-440
- #line directive, 389, 390-391, 393
- \_LINE\_ macro, 392-393
- LoadCursor() API function, 484
- LoadIcon() API function, 483-484
- localtime() function, 316-317, 435, 438-439
- log() function, 430
- log10() function, 430-431
- LONG data type, 478
- long type modifier, 107-110, 464
- longjmp() function, 450-452, 454
- Loops
  - exiting, 89-92
  - forcing next iteration of, 92-94
  - infinite, 81
  - message, 477, 478
  - nested, 87
- LPARAM data type, 488
- LPCSTR data type, 478
- LPSTR data type, 478
- LPVOID data type, 486
- LRESULT data type, 477

**M**

- Macro(s)
  - built-in (C), 391-393
  - function-like, 377-378, 379-380, 393-394
  - substitution, 229-232

- main() function, 3, 6, 24
  - command-line arguments and, 215-220
  - and prototypes, 205-206
- malloc() function, 403-405, 441, 442-443
- MATH.H header file, 27, 203, 425
- Memory, dynamic allocation of, 402-407, 440
- Menus, 472-473
- Message(s)
  - loop, 477-478, 487-489
  - and Windows, 473, 477
- Microsoft Visual C++, 8, 9, 72
- Modulus operator, 17, 18
- Mouse and Windows, 472
- MSG structure, 478, 488, 489
- Multitasking and Windows, 474

**N**

- Naming conventions (Windows), 490
- NOT logical operator, 62
- NULL macro, 260
- Null
  - pointers, 169-170
  - string, 150
  - terminator, 145

**O**

- Octal
  - constants, specifying, 121
  - number system, 60, 121
  - 1's complement operator, 358, 359, 360-361
- Operator(s)
  - arithmetic, 17, 18
  - arrow, 314-315, 330
  - assignment, 12, 367-369
  - bit-shift, 363-364
  - bitwise, 358-362
  - comma, 370-371
  - decrement, 54-56
  - dot, 302, 314, 315, 330
  - increment, 54-56
  - logical, 61-66
  - modulus, 17, 18
  - precedence of, 372
  - relational, 41-42, 61-64
  - ternary, 365-367
  - unary, 17
- OR bitwise operator, 358, 361-362
- OR logical operator, 62

## P

- Parameters, 32-34, 211
  - declaration, classic vs. modern, 220-223
  - formal, as local variables, 114
  - to main(), 216-217
  - pointers as, 191-192, 211, 212-214
- Parity bit, 362
- POINT structure, 488
- Pointer(s)
  - accessing array with, 176-178, 179-181
  - arithmetic, 172-175, 179
  - arrays of, 186-187
  - base type of, 167, 168-169, 171
  - decrementing, 181-182
  - function, 395-401
  - incrementing, 173, 175, 181
  - indexing, 178-179
  - null, 169-170
  - operators, 63-168
  - as parameters, 191-192, 211, 212-214
  - to pointers, 188-190
  - returned from functions, 204-205
  - to string constants, 183-185
  - to structures, 314-317
  - void (generic), 279
- PostQuitMessage() API function, 489
- pow() function, 431
- #pragma directive, 389, 391
- Preprocessor, 4-5, 229, 388, 393. *See also*
  - Directives, preprocessor
- printf() format specifiers, 13, 110, 241-243
  - table of, 242
- printf() function, 4, 12-14, 241-246
  - backslash-character constants for, 58-61
  - performing disk file I/O with, 294
  - and pointers, 174
  - strings and, 121, 146
  - using putchar() instead of, 234
  - using puts() instead of, 239
- Programs
  - components of, 2-7
  - creating and compiling, 7-9
- Prototypes, 24, 26, 196-206
- putc() function, 262-263
- putchar() function, 233-235
- puts() function, 191, 238, 239, 240

## Q

- qsort() function, 400-401, 452-453
- Quicksort, 208, 400, 452

## R

- Range error, 425
- rand() function, 244, 453-454, 455
- RAND\_MAX macro, 453
- realloc() function, 443-444
- Recursion, 207-210
- register storage class specifier, 339, 341-342, 343-346, 418
- RegisterClassEx() API function, 485
- rename() function, 290
- remove() function, 290, 291
- return statement, 28-30, 418
- rewind() function, 290, 291-292

## S

- scanf() format specifiers, 16, 110, 246-253
  - table of, 247
- scanf() function, 15-16, 72, 246-253
  - and arrays, 140
  - and gets(), 150
  - pointers and, 191
  - and strings, 121, 247-248, 249-250, 250-251
- Scanset, 248, 250-251
- Scientific notation, 119-120, 242
- Scope rules, 112
- SEEK\_CUR macro, 286
- SEEK\_END macro, 286
- SEEK\_SET macro, 286
- setjmp() function, 450-451, 454-455
- SETJMP.H header file, 450, 451
- short type modifier, 107-111, 418
- ShowWindow() API function, 487
- Sign flag, 108
- signed type modifier, 107-111, 418
- sin() function, 431-432
- sinh() function, 432
- size\_t type, 279-280, 440, 448, 452
- sizeof operator, 281-282, 305, 330, 419
- Sorting with arrays, 143-144, 400-401, 452-453
- sqrt() function, 27-28, 132-133, 203, 433
- srand() function, 455
- Statement(s)
  - assignment, 12
  - conditional, 41
  - definition of, 2
  - null, 81
  - selection, 41
- static storage class specifier, 339, 342-343, 346-347, 419
- \_\_STDC\_\_ macro, 392
- stderr (standard error) stream, 293, 294

- stdin (standard input) stream, 293, 294, 295-296  
 STDIO.H header file, 5, 145, 233, 238, 260, 279-286  
 STDLIB.H header file, 150, 244, 401, 403, 440, 452, 453  
 stdout (standard output) stream, 293-294  
 Storage class specifiers, 339-347  
 strcat() function, 147, 420  
 strchr() function, 420-421  
 strcmp() function, 147, 421-422  
 strcpy() function, 146-147, 150, 191, 192, 422  
 Streams, 259-260  
   standard, 293-296  
 String(s)  
   arrays of, 159-162  
   as character arrays, 145-150, 412  
   command-based interface and, 149-150  
   concatenating, 147  
   definition of, 4, 145  
   null, 150  
   printf() and, 121, 146  
   scanf() and, 121, 247-248, 250-251  
   table, 159-162, 183, 186-187  
 String constants  
   definition of, 121, 183  
   using pointers to, 183-185  
 STRING.H header file, 146, 412  
 strlen() function, 148, 191, 351, 422  
 strstr() function, 422-423  
 strtok() function, 423-424  
 struct keyword, 301, 419  
 Structure(s), 300-324  
   arrays of, 303, 305-310  
   definition of, 300  
   general form of, 301  
   members, accessing, 302, 304-305, 314-315  
   nested, 318-324  
   passed to functions, 304, 313  
   pointers to, 314-317  
   returned by functions, 304, 312  
   size of, determining, 305  
   variables, 301, 302-303  
 switch statement, 94-99, 420  
   nested, 96
- I**
- tan() function, 433  
 tanh() function, 433-434  
 Ternary operator, 365-367  
 Text stream, 259  
 Time  
   broken-down, 316-317, 435  
   calendar, 316-317, 434  
   time() function, 316, 317, 436, 438, 439-440  
   TIME.H header file, 316, 344, 434  
   \_\_TIME\_\_ macro, 392-393  
   time\_t type, 316, 434, 435  
   tm structure, 316, 434-435, 438  
   tolower() function, 179-181, 424  
   toupper() function, 179-181, 424  
   TranslateMessage() API function, 489  
   True and false in C, 41  
   Two's complement approach, 108-109  
   Type casts, 132-133  
   Type conversions  
     in assignments, 129-131  
     in expressions, 126-128  
   Type modifiers, 107-111  
   Type promotions and prototypes, automatic, 200  
   typedef statement, 356-358, 467
- U**
- UINT data type, 488  
 Unary operators, 17  
 #undef directive, 389, 390  
 union keyword, 467  
 Unions, 329-333  
 UNIX, 258  
 unsigned type modifier, 107-111, 467  
 UpdateWindow() API function, 487
- V**
- Values  
   assigning, to variables, 12  
   returning, from functions, 27-30  
 Variables  
   assigning values to, 12  
   automatic, 339  
   declaring, 10-12, 13-14, 112-114  
   initializing, 123-125  
   using register for fast access to, 341-342, 343-344  
 Variables, global, 11, 112, 114-118  
   extern and, 339-341, 347  
   initializing, 123  
   static, 343  
 Variables, local, 11, 112-114, 115-119  
   auto, 113, 339  
   initializing, 123, 124-125  
   static, 342-343, 346-347  
 void, 10, 23, 467  
   function prototypes and, 200, 201  
   pointers, 279  
   used to denote no return value, 201  
   volatile access modifier, 349, 351-352, 468

**W**

while loop, 82-84, 468

WIN32, 474-475

WINAPI calling convention, 476

Window

components of, 475-476

creating, 485-487

displaying, 487

style, macros for, 486

Window class, 477

Window function, 476-477, 478, 489

Windows

application basics, 476-478

application skeleton, 478-489

data types, common, 478

message-based interaction with programs,  
473-474

mouse and, 472

naming conventions for functions and variables,  
490, 491

programming philosophy, 470-471

WINDOWS.H header file, 478

WinMain( ), 476, 477, 479, 482

WM\_DESTROY message macro, 489

WM\_QUIT message macro, 489

WNDCLASSEX structure, 478, 482-483

WORD data type, 478

WPARAM data type, 488

WS\_OVERLAPPEDWINDOW macro, 486

**X**

XOR logical operation, 64-66

XOR bitwise operator, 358, 359