

INDEX

Symbols

:: (double colon) operator, 205
! (text escape) operator, 6–9
& (text expansion) operator, 5–6
% (value expansion) operator, 8

A

ABI (application binary interface). *See*
 Microsoft (Windows) ABI
abstract methods, 413
accessing nonlocal variables, 216
acos (arc cosine) function, 147, 157
acot (arc cotangent) function, 147, 160
acsc (arc cosecant) function, 147, 163–164
activation record lifetimes and thunks, 597
address binding, 206
address space, 298
aligning the stack, 614
allocating new class objects, 427
.allocstack directive, 506–507
ancestor procedures, 214
aoaMacros.inc header file, 39
aoa namespace entries, 39
application binary interface (ABI). *See*
 Microsoft (Windows) ABI
asec (arc secant) function, 147, 167
asin (arc sine) function, 148, 171–172
atan (arc tangent) function, 148, 164–168,
 174–177
atomic instructions, 380
automatically resetting an event, 316

B

bare return instruction, 284
base-10 logarithm, 148, 186–191
biased frame pointer, 289
BMP (Basic Multilingual Plane,
 Unicode), 77
bool_t macro, 266, 275

_bswap32 macro, 53
btr instruction, 383–385
bts instruction, 383–385
byte swap macro, 53
byte_t macro, 267–269, 275
byte types, 610

C

callee- and caller-preservation
 (registers), 613–614
calling conventions
 fastcall calling convention, 611
 Microsoft (Windows) ABI, 610–618
 nonvolatile registers, 612
 preserving registers, 257
 returning function results,
 647–649
 shadow storage, 611
 stack alignment, 614
 volatile registers, 612
call tree, 246
CancelWaitableTimer function, 342
canonical equivalence (Unicode), 79
CAS (compare and swap), 387
catstr directive, 19
@CatStr macro, 20
change code page in Windows
 console, 89
character names (Unicode), 78
char_t macro, 275
char type, 610
child procedures, 213
classes, 398
 constructors, 426–431
 declarations in C++, 399
 destructors, 431
 functions, 425
 instantiating, 398
 macros for creating, 419–425
 new operations, 427–431

- CloseHandle function, 317, 325, 342
- closures, 581, 599–601
- CLU language, 651
- cmpxchg8b instruction, 389
- cmpxchg16b instruction, 389
- coarse-grained parallelism, 298
- cocalls, 676
- code snippets, xxi
- code stream parameters, 618, 624
- combining characters (Unicode), 78–83
- command line, xxiii
- compare and swap (CAS)
 - instruction, 387
- compute-bound tasks, 356
- concatenation, 19–20
- concrete classes, 696–697
- concurrency
 - context-switching, 357
 - cooperative multitasking, 298
 - critical sections, 306–314
 - deadlock, 314, 326–339
 - deadly embrace, 314, 332
 - events, 306, 314–323
 - fibers, 298, 675
 - lock-free operations, 389
 - memory fences, 393
 - memory ordering, 393
 - multiprocessing, 297
 - multitasking, 296
 - mutexes, 306, 318, 323–324
 - race conditions, 380
 - semaphores, 306, 324–326
 - starvation, 356, 358
 - threads, 296, 298
 - synchronization, 303
- constructors, 426–431
 - multiple inheritance, 418
 - passing parameters to, 429
- context-free grammars, 58–59
- context-free macros, 54, 72
- context-switching, 357
- continuable exceptions, 526
- ConvertFiberToThread function, 680
- ConvertThreadToFiber
 - function, 678
- cooperative multitasking, 298
- copyTh macro, 595

- coroutines, 675
 - reentrancy and recursion, 688
 - task switching, 676
- cos (cosine) function, 148, 177–181
- cot (cotangent) function, 148, 196–200
- CPU
 - affinity, 367
 - pinning, 367
 - releasing to another task, 358
- CreateEventA function, 316
- CreateEvent function, 316
- CreateEventW function, 316
- CreateFiber function, 678
- CreateMutexA function, 323
- CreateMutex function, 323
- CreateMutexW function, 323
- CreateProcessA function, 360
- CreateProcess function, 360
- CreateProcessW function, 360
- CreateSemaphoreA function, 324–325
- CreateSemaphore function, 324–325
- CreateSemaphoreW function, 324–325
- CreateThread function, 299–300
- CreateWaitableTimerA function, 340
- CreateWaitableTimer function, 340
- CreateWaitableTimerW function, 340
- CreateWindowExA, 85
- creating new threads, 299
- critical sections, 306–314
 - data structure, 307
 - vs. events, 314–315
 - multiple entries by the same thread, 314
- csc (cosecant) function, 148, 192–196

D

- data declaration directives, 39
- data types
 - creating with MASM, 264–266
 - Microsoft (Windows) ABI, 610
 - opaque, 307
- deadlock, 314, 326–339
- deadly embrace, 314, 332
- deep (recursive) text expansion, 11
- default stack space (for a thread), 300
- deferred execution, 586
- DeleteCriticalSection function, 308
- DeleteFiber function, 679

- destructors, 431
- diamond problem, multiple
 - inheritance, 416
- directive expansion from
 - macros, 13
- direct memory access (DMA), 383
- displays, 204, 218–221
- DLL (dynamically-linked library), 298–299
- domain-specific language (DSL), 2
- double-byte character sets, 75
- double colon (::) operator, 205
- double type, 611
- dword_t macro, 275
- dword type, 610
- dynamic link, 210, 657

E

- echo directive, 8
- emitVMT macro, 421, 470, 473
- endc macro, 464
- _endif macro, 56, 68
- endpr (protocol) macro, 447
- .endprolog directive, 506–507
- endTh (thunk) macro, 594
- endtry macro, 541, 556
- endxp macro, 541, 558, 566
- enter instruction, 222, 247
- environment strings, 362
- epilog code, 284
- epiloguedef option, 284
- epilogue:macroName option, 284
- epilogue:none option, 284
- escape character (macros), 7
 - sequences in, 27
- events, 306, 314–323
 - automatically resetting, 316
 - vs. critical sections, 314–315
 - handles, 315
 - manually resetting, 316
 - signaling an event, 316
- exception filter, 522
- exception flags (FPU), 142
- exception handling, 521
 - continuable exceptions, 526
 - handlers, 484
 - macros, 541, 569
 - structured, 487

- thunks, 605
 - unwinding the stack, 494
- exception macro, 541, 548
- ExceptionRecord_t structure, 528
- execution environment (thunks), 587
- execution queue, 296
- exiting a thunk, 603
- exitm (exit macro) directive, 11
- ExitProcess function, 363
- exp (natural exponent) function, 148, 181–186
- expansion of text equate symbols, 24
- exponential functions, 148, 181–186
- external namespace, 32
- extractIP macro (IP addresses), 40

F

- f2xm1 instruction, 142, 145, 182
- fastcall calling convention, 611
- fcos instruction, 142, 146
- fibers, 298, 675
 - context switches and registers, 683
 - fiber-local storage, 689
 - in Windows, 677
- finally macro, 541, 554–556
- finally section, 527
- first-class objects, 582
- fld1 instruction, 145
- fld12e instruction, 145
- fld12t instruction, 145
- fldlg2 instruction, 145
- fldln2 instruction, 145
- fldpi instruction, 145
- fldz instruction, 145
- float type, 611
- FLS (fiber-local storage), 689
- FlsAlloc function, 689
- FlsFree function, 689
- FlsGetValue function, 689
- FlsSetValue function, 689
- forcing stack alignment, 614
- fpatan instruction, 142, 146
- fprem and fprem1 instructions, 142, 144
- fptan instruction, 142, 146, 196–200
- FPU (floating-point unit), 139–141
 - busy bit, 142
 - condition code bits, 142
 - exception flags, 142

- FPU (*continued*)
 - registers, 620
 - top of stack pointer, 142
- framed procedures, 520
- frame option (proc directive), 507, 536
- frame pointer biasing, 289
- fscale instruction, 142–143, 182
- fsincos instruction, 142, 146
- fsin instruction, 142, 146, 192
- fsqrt instruction, 142
- fstsw instruction, 141
- functional syntax for macros, 11
- function instance, 206
- function results
 - returning, 647–649
 - thunks, 595
- fxtract instruction, 144
- fyl2x instruction, 142, 147
- fyl2xp1 instruction, 142, 147

G

- generators, 675, 690
 - restart operation on, 694
 - return addresses in, 704
- generic\$Class\$Constructor
 - function, 428
- genID macro, 54
- getByte macro, 41
- GetCommandLine function, 363
- GetCurrentProcess function, 369
- GetCurrentThread function, 369
- GetEnvironmentStrings function, 363
- GetExitCodeProcess function, 363
- GetModuleHandle function, 126
- getPort macro, 40, 43
- GetProcessAffinityMask
 - function, 368
- getRem (get remainder) macro, 41–43
- global memory locations as
 - parameters, 618
- global namespace, 32
- global variables, 233

H

- handling exceptions, 521
 - in thunks, 605
- hot patching, 292
- hyperthreading, 368

I

- Icon language, 651
- _if macro, 56, 68
- implementing inheritance in MASM, 413
- indexing local variables off RSP, 251
- indirect recursion, 207
- infinity, 149
- inheritance, 409
 - in MASM structs, 413
- InitializeCriticalSectionAndSpinCount
 - function, 313
- InitializeCriticalSection function, 307
- initializing a class object, 426
- initializing thunks, 593
- instances, routine, 206
- instantiating a class, 398
- instr directive, 17–18
- @InStr macro, 20
- instructions that support lock prefix, 384
- int8_t macro, 274
- int16_t macro, 274
- int32_t macro, 274
- int64_t macro, 274
- __int64 type, 611
- interfaces, 444
- intermediate variables, 233
- internationalization (strings in an application), 124
- int type, 610
- inverse cosine, 157
- inv macro, 422, 477
- I/O-bound tasks, 356–357
- IP4 address macro, 39
- ip macro, 40
- iterators, 651–652
 - implementation with resume frames, 657
 - object-oriented design pattern, 695

L

- lazy-evaluation parameters, 637
- leaf functions, 489
- leaf procedures, 246, 519
- leave instruction, 222–232, 259, 511, 616, 624

- lexical nesting, 204
- lex programming language, 2
- lfence instruction, 394
- listings, xxi
- ln (natural logarithm) function, 148, 186–191
- LoadString function, 126
- LoadStringW function, 124, 126
- local directive (macros), 9
- localizing strings in an application, 124
- local labels, 205
- locals macro, 240–242
- local variables, 233
 - RSP access, 251
- lockable instructions
 - lock prefix, 383
 - performance, 385
 - syntax, 384
- lock-free operations, 389
- log (base-10 logarithm) function, 148, 186–191
- logarithmic functions, 139
- long int type, 611
- long long unsigned type, 611
- long type, 611
- long unsigned type, 611
- lpProcessInformation data
 - structure, 363

M

- macros, 1
 - arguments passed from another macro, 6
 - concatenation, 19–20
 - context-free, 54, 72
 - for creating classes, 419–425
 - data declaration directives, 39
 - directive expansion, 13
 - exception handling, 569
 - functional syntax, 11
 - invocation, 3
 - local symbols, 8
 - passing arguments to another macro, 6
 - performance, 73
 - simulating directives with, 13
 - substrings, 18
 - text expansion operator (&), 5

- text processing, 16
 - value expansion operator (%), 8
- manually resetting events, 316
- maskmovdqu instruction, 393
- maskmovq instruction, 393
- MASM (Microsoft Macro Assembler), xx
 - implementing inheritance, 413
 - nesting proc declarations, 235, 260
 - searching for substrings in textual constants, 17
 - Unicode strings, 91
- masm32.com* website, 299
- math_acos function, 147, 157
- math_acot function, 147, 160
- math_acsc function, 147, 163
- math_asec function, 147, 167
- math_asin function, 171
- math_atan function, 174
- math_cos function, 177
- math_cot function, 196
- math_csc function, 192
- math_exp function, 181
- math(inf) constant, 149
- math_ln function, 186
- math_log function, 186
- math namespace, 148
- math(nan) constant, 149
- math_sec function, 177
- math_sin function, 192
- math_tan function, 196
- math_tenTox function, 148, 181–182
- math_twoTox function, 148, 181–182
- math_yTox function, 148, 181–182
- memory
 - address space, 298
 - direct memory access (DMA), 383
 - fences, 393
 - non-temporal instructions, 393
 - ordering, 393
- Metaware Professional Pascal
 - Compiler, 656
- method0fs macro, 422, 474
- methods
 - calling class methods, 423
 - calling parent methods, 424
 - method dispatch, 417
 - static member functions, 403

- methods (*continued*)
 - super functions, 425
 - super keyword in class method invocation, 424
 - virtual member functions, 425
 - virtual method tables, 400
- mfence instruction, 394
- Microsoft (Windows) ABI, 610–618
 - data types, 610
 - nonvolatile registers, 612
 - shadow storage, 611
 - stack alignment, 614
 - volatile registers, 612
- Microsoft Visual C++ (MSVC), 744
- minimal constructor code, 426
- MMX instructions, 141
- MMX register set, 620
- modules (source code), xxi
- movntdq instruction, 393
- movnti instruction, 393
- movntpd instruction, 393
- movntps instruction, 393
- movntq instruction, 393
- MSVC (Microsoft Visual C++), 744
- multiple inheritance, 415, 445
 - issues in, 418
 - method dispatch, 417
- multiprocessing, 297
- multiprogramming, 298
- multitasking, 296
- mutexes, 306, 318, 323–324

N

- namespaces, 32–39
 - pollution, 33, 263
- nan (not-a-number), 149
- natural exponent (exp) function, 148, 181–186
- natural logarithm (ln) function, 148, 186–191
- nested procedures, 204
- nesting MASM proc declarations, 235, 260
- new operations for class objects, 427–431
- nontemporal instructions, 393
- nonvolatile registers, 612
- normalization (Unicode), 80

- noscoped option, 205
- not-a-number (nan), 149

O

- object-oriented programming (OOP), 398
 - classes, 398
 - constructors, 426–431
 - design pattern in iterators, 695
 - destructors, 431
 - inheritance, 409
 - interfaces, 444
 - method dispatch, 417
 - multiple inheritance, 415, 445
 - objects, 398
 - overloading, 412
 - polymorphism, 414
 - protocols, 444
 - static member functions, 403
 - super functions, 425
 - virtual member functions, 425
 - virtual method tables, 400
 - VMT, 399
- objects, 398
- one-shot timer, 339
- opaque data type, 307
- opcodes
 - UWOP_ALLOC_LARGE, 496
 - UWOP_ALLOC_SMALL, 496
 - UWOP_PUSH_MACHFRAME, 499
 - UWOP_SAVE_NONVOL, 498
 - UWOP_SAVE_NONVOL_FAR, 498
 - UWOP_SAVE_XMM128, 498
 - UWOP_SAVE_XMM128_FAR, 499
 - UWOP_SET_FPREG, 497
- OpenSemaphore function, 325
- OpenWaitableTimerA function, 340
- OpenWaitableTimerW function, 340
- option directive, 205, 284
- out-of-order execution, 393
- overloading, 405, 412
- override macro, 463

P

- parameters, 610
 - blocks, 626
 - callee- and caller-preservation (registers), 613–614
 - cleanup, 615

- code stream parameters, 618, 624
- FPU registers, 620
- global memory locations as
 - parameters, 618, 622–623
- lazy evaluation, 637
- locations, 611, 618
- name parameters, 632
- parameter blocks, 626
- pass by value/result, 628
- registers, 618
- result parameters, 631
- returning function results,
 - 647–649
- shadow storage, 611
- stack parameters, 623
- value-returned parameters, 628
- parms macro, 240, 242–245
- partial remainder, 144
- pause instruction, 386
- PDA (push-down automata), 58
- peer procedure, 213
- performance of macros, 73
- periodic (synchronization) timers, 340
- per-thread variables, 345
- physical cores, 370
- pointer type, 611
- pollution, namespaces, 33, 263
- polymorphism, 414
- _pop macro, 59, 64
- port (Ethernet socket port number), 39
- powers of 2 and 10, 148, 181
- preemption, 298
- preserving registers, 257
- printing UTF-8 characters, 88
- priority of threads, 358
- privileged instructions, 381
- procedures
 - activation record lifetimes, 597
 - ancestor procedures, 214
 - call tree, 246
 - child procedures, 213
 - epilog code, 284
 - framed procedures, 520
 - function instance, 206
 - leaf procedures, 246, 519
 - nested procedures, 204
 - peer procedure, 213
 - preserving registers, 257
 - proc options, 205
 - prolog code, 284
 - returning function results,
 - 647–649
 - state information, 237
 - static links, 209–219
 - subroutine instance, 206
- processes, 359–366
 - definition, 298
 - synchronization, 306
- processor affinity, 367
- Professional Pascal, 656
- programs, 298
 - listings in this book, xxi
 - stand-alone in assembly
 - language, xxiii
- prolog code, 284
- prologuedef option, 284
- prologue:macroName option, 284
- prologue:none option, 284
- protocol macro, 447, 463–467
- protocols, 444
- pseudo-dynamic arrays, 253
- ptr_t macro, 275
- pure protocol definition, 447
- push-down automata (PDA), 58
- .pushframe directive, 506
- _push macro, 59, 64
- .pushreg directive, 506, 508
- put macro, 276

Q

- quantum (time slices), 357
- queuing tasks, 370
- qword_t macro, 274
- qword type, 611

R

- race conditions, 380
- Raise macro, 505
- RBP biasing, 289
- rc.exe resource compiler, 124
- rdtsc instruction, 394
- real4_t macro, 274
- real4 type, 611
- real8_t macro, 274
- real8 type, 611
- real10_t macro, 274

- recursion
 - in coroutines, 688
 - in text expansion, 11
- registers
 - callee- and caller-preservation, 613–614
 - FPU registers, 620
 - as function results, 647
 - MMX register set, 620
 - nonvolatile registers, 612
 - as parameters, 618
 - values in coroutine calls, 683
 - volatile registers, 612
- regular expressions, 54
- regular language, 54
- ReleaseMutex function, 324
- ReleaseSemaphore function, 325
- req operator in macro arguments, 14
- ResetEvent function, 317
- resource files, 92, 124
- restarting a generator, 694
- resume frames, 657–665
- returning function results
 - in a register, 647
 - on the stack, 648
 - thunks, 595–597
- round-robin (thread priority), 358
- RSP access to local variables, 251
- RTTI (runtime type information), 477
- run (execution) queue, 296
- RUNTIME_FUNCTION structure, 521

S

- .saverreg directive, 506, 508
- .savexmm128 directive, 506, 509
- sbyte type, 610
- scheduling
 - context-switching, 357
 - cooperative multitasking, 298
 - preemption, 298
 - priority of a thread, 358
 - quantum (time slices), 357
 - round-robin, 358
 - run (execution) queue, 296
 - time slices, 296
 - voluntarily giving up a time slice, 358
- scope, 204
- scoped option directive, 205
- scopeTable structure, 523
- sdword type, 610, 611
- searching for substrings in MASM
 - textual constants, 17
- sec (secant) function, 148, 177
- sehProlog macro, 562
- semaphores, 306, 324–326
- SetCriticalSectionSpinCount
 - function, 313
- SetErrorMode function, 363
- SetEvent function, 317
- .setframe directive, 506–507, 509
- SETL language, 651
- SetProcessAffinityMask function, 368
- SetThreadAffinityMask function, 368
- SetThreadIdealProcessor function, 368
- SetThreadPriority function, 359
- SetWaitableTimer function, 341
- sfence (store fence) instruction, 394
- shadow storage, 131, 361
 - for parameters, 611
- shared data in coroutine calls, 683–684
- short int type, 610
- short unsigned type, 610
- signaling an event, 316
- signed char type, 610
- signum function, 163
- simulating directives with macros, 13
- sin (sine) function, 148, 192–196
- single-instruction, multiple data (SIMD) instructions, 620
- sizestr directive, 16
- @SizeStr macro, 20
- Sleep function (Windows), 303
- small languages, 2
- source code in this book, xxi
- specific protocol definition, 447
- spin counters, 313
- spin loops, 385
- sqword type, 611
- stack
 - alignment, 614
 - default stack space, 300
 - frame pointer biasing, 289
 - parameters, 623
 - RSP access to local variables, 251
 - shadow storage, 611

- stack space for a thread, 300
- unwinding, 494
- stand-alone programs, xxiii
- starvation, 356, 358
- state information (procedures), 237
- static class functions, 425
- static links, 209–219
 - traversal, 215–219, 222
- static member functions, 403
- store fence (sfence) instruction, 394
- strchr function, 18
- strcspn function, 18
- stringable protocol, 448
- string concatenation
 - in macros, 19–20
 - in Unicode, 132
- string directives, 20
- string functions (Unicode), 131
- string length, 16
- string literals, 4
 - text expansion, 5
- string tables, 124
- string_t macro, 275
- strspn function, 18,
- strstr function, 17
- struct directive, 398
- structured exception handling, 487
- subroutine instance, 206
- substr directive, 18–19
- substrings
 - in macros, 18
 - searching in MASM textual constants, 17
- @SubStr macro, 20
- super functions, 425
- super keyword (class method invocation), 424
- supports macro (classes/protocols), 468, 470
- surrogate code points (Unicode), 77
- SwitchToFiber function, 679
- SwitchToThread function, 358
- sword type, 610
- synchronization (of threads), 303

T

- tan (tangent) function, 148, 196
- task switching (coroutines), 676

- tenTox (power of 10) function, 148, 181–182
- TerminateProcess function, 363
- test and set instructions, 385
- text equates, 22
- textequ directive, 22
- text escape operator (!), 6–9
- text expansion
 - inside a string literal, 5
 - levels, 10
 - operator (&), 5–6
- text literals, 4
- text processing in macros, 16
- this (pointer to an object), 407
- thread-local storage (TLS), 345
- threads, 296, 298
 - creating new threads, 299
 - priority, 358
 - processor affinity, 367
 - quantum (time slices), 357
 - stack space, 300
 - thread-local storage, 345
 - variables, 345
 - voluntarily giving up a time slice, 358
 - worker threads, 396
- thunks, 581, 587, 633
 - and activation record lifetimes, 597
 - exception handling, 605
 - execution environment, 587
 - exiting, 603
 - initializing, 593
 - returning as a function result, 595–597
 - unwinding the stack, 604
- time multiplexing, 296
- timers
 - one-shot timer, 339
 - periodic (synchronization) timers, 340
 - UTC time, 341
 - waitable timers, 339
- time slices, 296
 - quantum, 357
 - voluntarily giving up, 358
- TLS (thread-local storage), 345
- TlsAlloc function, 346
- TlsFree function, 346

- TlsGetValue function, 346
- TlsSetValue function, 346
- tostr macro, 5
- transcendental functions, 147
 - acos (arc cosine), 147, 157
 - acot (arc cotangent), 147, 160
 - acsc (arc cosecant), 147, 163–164
 - asec (arc secant), 147, 167
 - asin (arc sine), 148, 171–172
 - atan (arc tangent), 148, 164–168, 174–177
 - cos (cosine), 148, 177–181
 - csc (cosecant), 148, 192–196
 - cot (cotangent), 148, 196–200
 - exp (natural exponent), 148, 181–186
 - ln (natural logarithm), 148, 186–191
 - log (base-10 logarithm), 148, 186–191
 - sec (secant), 148, 177
 - sin (sine), 148, 192–196
 - tan (tangent), 148, 196
 - tenTox (power of 10), 148, 181–182
 - twoTox (power of 2), 148, 181–182
 - yTox (exponential) function, 148, 181–182
- traversing static links, 215–219, 222
- TryEnterCriticalSection function, 308
- try macro, 541, 545
- twoTox (power of 2) function, 148, 181–182
- types
 - byte, 610
 - char, 610
 - double, 611
 - dword, 610
 - float, 611
 - int, 610
 - __int64, 611
 - long, 611
 - long int, 611
 - long long unsigned, 611
 - long unsigned, 611
 - pointers, 611
 - qword, 611
 - real4, 611
 - real8, 611

- sbyte, 610
- sdword, 610, 611
- short int, 610
- short unsigned, 610
- signed char, 610
- sqword, 611
- sword, 610
- unsigned int, 610
- unsigned char, 610
- unsigned __int64, 611
- word, 610

U

Unicode

- in assembly language, 91
- BMP, 77
- canonical equivalence, 79
- characters
 - combining characters, 78–83
 - names, 78
 - surrogate code points, 77
- encodings, 80
 - UTF-8, 81
 - UTF-16, 81
 - UTF-32, 80
- normalization, 79
- printing UTF-8 characters, 88
- resource files, 92, 124
- strings, 75
 - character strings in MASM, 92
 - concatenation, 132
 - functions, 131
- Windows code page, 89
- unit activation, 206
- uns8_t macro, 274
- uns16_t macro, 274
- uns32_t macro, 274
- uns64_t macro, 274
- unsigned char type, 610
- unsigned int type, 610
- unwinding the stack, 494
 - thunks, 604
- UTC time, 341
- UTF-8, 81
- UTF-16, 81
 - character strings in MASM, 92
- utf16 macro, 92
- UTF-32, 80

- UWOP_ALLOC_LARGE opcode, 496
- UWOP_ALLOC_SMALL opcode, 496
- UWOP_PUSH_MACHFRAME opcode, 499
- UWOP_SAVE_NONVOL_FAR opcode, 498
- UWOP_SAVE_NONVOL opcode, 498
- UWOP_SAVE_XMM128_FAR opcode, 499
- UWOP_SAVE_XMM128 opcode, 498
- UWOP_SET_FPREG opcode, 497

V

- value expansion operator (%), 8
- value-returned parameters, 628
- vararg operator in macro arguments, 14
- variable lifetime, 206
- very high-level languages (VHLLs), 2
- virtual cores, 368
- virtual macro, 421, 462
- virtual member functions, 425
- VMT (virtual method tables), 399, 400–401
- vmtAdrs macro, 422
- volatile registers, 612
- voluntarily giving up a time slice, 358

W

- waitable timers, 339
- WaitForMultipleObjects function, 318
 - return values, 319

- WaitForSingleObject function, 317
 - return values, 318
- wcscat function, 132
- wcschr function, 132
- wcscmp function, 133
- wcscpy function, 133
- wcscspn function, 134
- wcslen function, 134
- wcsspn function, 134
- wcsstr function, 135
- weakly-ordered memory
 - operations, 393
- Windows code page, 89
- Windows fibers, 677
- word_t macro, 275
- word type, 610
- worker tasks, 370
- worker threads, 326
- wstring_t macro, 275

X

- xchg instruction, 382
- xproc macro, 541, 558, 561

Y

- yacc programming language, 2
- yTox (exponential) function, 148, 181–182