

- as component, 420
 - basic operations, 57
 - importance, 55
 - order, 180
 - tail-recursive, 74
- procedure value (closure), 54, 66–67, 795
 - anonymous, 54
 - common limitation, 182, 558
 - distributed lexical scoping, 728
 - encoding as an object, 547
 - higher-order programming, 180
 - relation to inner class, 558
- process
 - concurrent program design, 372
 - CSP, 629
 - distributed system, 713
 - Erlang, 357, 397
 - large program design, 454
 - operating system, 259
 - producer and consumer, 730
 - runtime error, 97
 - small program design, 221
- processor, 241
 - cluster computing, 717
 - dataflow machine, 343, 473
 - parallel functional programming, 337
 - shared-memory multiprocessor, 716
- producer, 262
- profiling, 180, 455
- program design, *see* software development
- program point, 448
- programming, *xxx*, 3
 - component-based, 420
 - concurrent, 581
 - constraint, 45
 - data-centered, 584
 - declarative, 31, 415
 - descriptive, 117
 - need for algorithms, 119
 - programmable, 117
 - Erlang, 395
 - flow-based, 261
 - functional, 415
 - future developments, 464
 - good style, *xxxv*
 - Haskell, 327
 - higher-order, 116, 126, 180–197
 - introduction, 15
 - relation to object-oriented, 544
 - imperative, 31, 415
 - Java, 558, 625
 - kernel language approach, *xxx*
 - logic, 45, 102, 145, 415, 644
 - multi-agent, 420, 584
 - multiparadigm, *xxviii*, *xl*
 - event manager, 573
 - object-based, 21, 543
 - object-oriented, 21, 421
 - open, 107, 205
 - paradigm, *xxvii*, *xxx*, *xxxi*, 32, *see* computation model
 - Prolog, 676
 - real-time, 309
 - relational, 633
 - stateful, 31
 - stateless, 31
 - synchronous, 271
 - techniques not covered in this book, *xli*
- programming model, *xxvii*, 31
- Prolog, 673–684
- proof engineering, 119
- proof rule, 447
- propagate-and-search, 641
- property
 - liveness, 611
 - object, 502
 - safety, 611
- propositional logic, 644
- protected scope, 514
 - C++ sense, 515
 - Java sense, 574
- protection boundary, 205
- protector, 326
- protocol, 361
 - by-need, 287
 - communication, 721
 - consistency, 718
 - DHCP, 209
 - distributed binding, 739
 - distributed unification, 739
 - eager copy, 739
 - eager immediate copy, 740

- interaction (in GUI), 693
- invalidation, 739
- IP, 209
- lazy copy, 739
- meta-object, 522
- mobile state, 739
- negotiation, 384
- short-circuit, 567
- stationary state, 738
- TCP, 718, 745, 746
- timer, 376
- pseudorandom numbers, 477
- Psion Series 3 palmtop computer, 385
- public scope, 513
- pure object-oriented language, 549
- quadratic equation, 181
- quantifier, 445, 448, 645, 657
 - existential (in Prolog), 684
- quantum (in thread scheduling), 257
- query
 - database, 668
 - logic programming, 646
- queue, 149
 - amortized ephemeral, 150
 - amortized persistent, 303
 - breadth-first traversal, 159
 - concurrent, 387, 592
 - priority, 614, 623
 - worst-case ephemeral, 151
 - worst-case persistent, 304
- race condition, 22, 238
- raise** statement, 94, 803
- random number generation, 476
- Raymond, Eric, 466
- reachable memory, 75
- real-time computing
 - garbage collection, 78
 - hard, 177, 257, 259, 309
 - soft, 309
- reasoning
 - algebraic, 114, 118, 324
 - atomic action, 588
 - causal, 361, 583
 - lift control system, 381
 - logical, xxxiii, 114, 644
 - message-passing concurrent model, 360
 - shared-state concurrent model, 324
 - stateful model, 324, 444
- receive
 - asynchronous, 338
 - nonblocking, 338
 - synchronous, 338
- receive** expression (in Erlang), 399
- record, 20, 53, 826
 - adjoin, 828
 - basic operations, 56, 828
 - importance, 54
 - usage trade-offs, 442
- recurrence equation, 171
- recursion, 5, 116, 126
 - direct, 116
 - indirect, 116
 - mutual, 112
 - polymorphic, 327
 - programming with, 130
 - tail recursion optimization, 74
- red cut (in Prolog), 683
- Red Hat Corporation, xl, 204, 475
- reduction order, 336–337
 - applicative, 336
 - normal, 336
- reengineering, 527
- refactoring, 455
- reference, 720
- referential transparency, 116
- reflection, 522
- register
 - abstract machine, 57
 - distributed binding, 743
 - forwarding, 633
 - in finalization, 486
- relation, 667
- relative error, 122
- rendezvous, 629
- replicator, 326
- resolution
 - deadlock, 614
 - logic programming, 647, 652, 675
 - video display, 321
- resource, 735, 752
 - external, 79, 485

- file descriptor, 298
- localized, 715
- producer/consumer pipeline, 265
- use of laziness, 294
- resting point, 344
- restriction (environment), 64
- retract/1** operation (in Prolog), 675
- Reuter, Andreas, 590, 609
- right, *see* name
- RISC (Reduced Instruction Set Computer) microprocessor, 633
- RMI (Remote Method Invocation), 361, 715, 730, 732
- Round** operation, 824
- RPC (Remote Procedure Call), 361, 715
- rubber band, 255
- runic inscription, 783
- Runnable** interface, 625
- s-expression, 663
- Sacks, Oliver, 413
- safety, 611
- Saint-Exupéry, Antoine de, 113
- Santayana, George, 703
- Saraswat, Vijay, 343, 675, 810
- scalability
 - compilation, 461
 - multiprocessor, 717
 - program development, 107
 - weighted reference counting, 743
- scheduler
 - Delay** operation, 310
 - deterministic, 257
 - lift control system, 375
 - nondeterministic, 257
 - randomness, 477
 - round-robin, 257, 261
 - thread, 243, 256
 - transaction, 612
- Schulte, Christian, xxxvii, xl
- science, xxix, xxxii
- scientific method, xxxi
- scope, 58
 - attribute, 516
 - dynamic, 60
 - lexical, 58, 65, 514, 546
 - absence in Prolog, 674
 - distributed, 728
 - hiding, 224, 419, 431, 445, 488, 501, 555
 - substitution, 805
 - private, 513, 514
 - protected, 514
 - public, 513
 - static, *see* lexical
 - user-defined, 514
- search
 - aggregate, 683
 - all-solutions, 638
 - binary, 155
 - breadth-first, 656
 - constraint programming, 279
 - contribution of AKL, 811
 - danger, 652
 - database query, 670
 - depth-first, 634, 656
 - deterministic, 634
 - encapsulated, 637
 - generate-and-test, 641
 - iterative deepening, 656
 - linear, 200
 - logic programming, 674
 - n*-queens problem, 641
 - one-solution, 638
 - overuse, xxxv
 - propagate-and-search, 641
 - pruning, 675
 - relational computation model, 635
 - relational programming, 634
- search space, 634
- search tree, 637
- security
 - abstract data type, 204–213, 427
 - application, 750
 - atom vs. name, 514
 - capability, 211
 - distributed resources, 736
 - distributed system, 749
 - engineering, 750
 - hardware, 750
 - human society, 211
 - implementation, 750
 - kernel language concepts, 849
 - language, 211, 750

- linguistic abstraction, 41
- mechanism, 211
- open distribution, 717
- policy, 211
- right, 795, 849
- static typing, 107
- threat model, 750
- self**
 - clone, 523
 - delegation, 517
 - dynamic binding, 511
 - forwarding, 517
 - Java, 559
 - this** notation, 558
- self** (in Erlang), 398
- semantic stack, 63
- semantics, 33
 - abstract machine, 57–80, 93–95, 243–246, 286–288, 355–357, 424–425
 - axiomatic, 40, 444–453, 644
 - by-need trigger, 286
 - cell, 424
 - common abstractions, 809
 - denotational, 40
 - exceptions, 93
 - interleaving, 784
 - kernel language, *see* abstract machine
 - kernel language approach, 39
 - logical, 40, 644–653
 - operational, 40, 61, 647, 783–814
 - port, 355, 391
 - secure types, 205
 - semantic statement, 62
 - SOS (Structural Operational Semantics), 783
 - thread, 243
- Send** operation, 356
- send
 - asynchronous, 338
 - latency, 268
 - nonblocking, 338
 - synchronous, 338
- separation of concerns, 574
- serializability, 609
- serialization, 715
- serializer, 325
- setof/3** operation (in Prolog), 638, 679, 683
- Shakespeare, William, 817
- shared-state concurrency, *see* atomic action, *see* lock, *see* monitor, *see* transaction
- sharing, 426
 - Browser tool, 104, 830
 - distributed state, 726
 - distributed value, 722
 - thread, 385
- short-circuit
 - concurrent composition, 282
 - Flavius Josephus problem, 567
- side effect, 420
 - declarative, 293
- signal** operation (in monitor), 600
- signature (for procedure), 131
- simulation
 - components, 421
 - digital logic, 271–277
 - inadequacy of declarative model, 176
 - Internet, 421
 - multi-agent, 420
 - Small World, 491
 - Word of Mouth, 481
- single-assignment store, 44–50, 62, 785
 - importance, 45
- sink (consumer), 263
- skip** statement, 64, 789
- SLDNF resolution, 675
- Small World
 - graph, 465
 - simulation, 491
- Smolka, Gert, xl
- snapshot, 441, 724
- software development, 221, 453
 - bottom-up, 455
 - compositional, 456
 - distributed programming, 751
 - evolutionary, 455
 - extreme programming, 456
 - importance of names, 514
 - in the large, 453
 - in the small, 221
 - interactive interface, 88
 - middle-out, 455

- stepwise refinement, 469, 613
- thin-to-thick, 455
- top-down, 10, 455
- software engineering, 454
 - component as unit of deployment, 224
 - concurrency, 237
 - distributed lexical scoping, 728
 - further reading, 466
 - informatics curriculum, xxxvi
 - lexical scoping, 60
- software rot, 463
- Solve** operation, 638, 778
- Sort** operation, 831
- SOS (Structural Operational Semantics), 783
- source (producer), 263
- source code, 225
 - inexistent, 496
 - interactive, 818
 - million line, xxx, 38, 395, 461
 - preprocessor input, 319
 - reengineering, 527
 - set of functors, 290
 - textual scope, 65
 - variable name, 46
- specification, 418
 - component, 465
- specification language, 119
- speculative execution
 - non-strict language, 337
- stack
 - declarative object, 431
 - depth-first traversal, 159
 - open declarative, 198, 429
 - proving it correct, 445
 - secure declarative bundled, 431
 - secure declarative unbundled, 208, 430
 - secure stateful bundled, 431
 - secure stateful unbundled, 433
 - semantic, 62
 - stateful concurrent, 585
- standalone application, 225
 - declare** not allowed, 88
 - Java, 561
 - uncaught exception, 95
- starvation, 280
 - wait set implementation, 605
- state
 - cell (mutable variable), 422
 - declarative, 416
 - explicit, 18, 417
 - implicit, 416
 - lazy execution, 486
 - lazy language, 337
 - memory management, 79
 - non-strict language, 337
 - port (communication channel), 354
 - reasoning with, 40, 444
 - revocable capability, 433
 - threading, 142
 - transformation, 136
- state transition diagram, 361, 376
 - component design, 373
 - floor component, 377
 - lift component, 379
 - lift controller component, 377
 - transaction, 616
- stateless (declarative programming), 113
- statement
 - case**, 68, 793
 - catch** (clause in **try**), 95
 - choice**, 636, 778
 - conc**, 283
 - declare**, 4, 88
 - fail**, 636
 - finally** (clause in **try**), 95
 - for**, 190
 - fun**, 85
 - functor**, 227
 - gate**, 277
 - if**, 68, 793
 - local**, 58, 64, 790
 - lock**, 23, 591
 - proc**, 66, 795
 - raise**, 94, 803
 - skip**, 64, 789
 - thread**, 245, 789
 - try**, 94, 802
 - break, 491
 - declarative kernel language, 51
 - interactive, 88
 - procedure application, 68

- sequential composition, 64, 789
- suspendable, 67
- value creation, 65
- variable-variable binding, 65
- static
 - binding, 512
 - linking, 225
 - scope, *see* scope, lexical
 - typing, xl, 52, 106–108
- stdin** (standard input), 232, 560
- stdout** (standard output), 560
- Stirling's formula for factorial, 626
- storage manager, 325
- store, 785
 - equivalence, 788
 - mutable (for cells), 424
 - mutable (for ports), 356
 - predicate, 785
 - read-only, 209, 800
 - single-assignment, 44–50, 62, 101, 239, 785
 - trigger, 287, 798
 - value, 44
- stream, 798
 - deterministic, 261
 - Java, 560
 - merger, 403
 - producer/consumer, 261
 - usage trade-offs, 442
- strict ..., *see* eager ...
- strict two-phase locking, 613
- strictness analysis, 295, 329, 349
- string, 54, 832
- StringToAtom** operation, 826
- structure
 - compiler, 164
 - compositional, 465
 - difference, 144
 - distribution, 260
 - effect of concurrency, 256
 - grammar, 34
 - hierarchical, 456
 - interpreter, 665
 - noncompositional, 465
 - program, 222, 223
- structure equality, 105, 426, 729
- substitution, 129, 805
- substitution property, 525, 527
- subtype
 - basic types, 52
 - class hierarchy, 525
- Sun Microsystems, 466
- superclass, 508, 520, 563
- supercomputer, 178
- supply-driven execution, *see* eager execution
- suspension
 - Delay** operation, 310
 - due to program error, 50, 91
 - thread, 243, 280
- Sussman, Gerald Jay, xxxiii, 43
- Sussman, Julie, xxxiii, 43
- Symbian Ltd., 385
- symbolic link, 463
- synchronization, 339–343
 - dataflow, 793
- synchronized** keyword, 601, 625
- synchronized clocks, 313
- synchronous failure detection, 745
- syntactic sugar, 41, 80–85
 - local** statement, 41
 - state transition diagram, 377
- syntax, 33
 - convention for examples, xliii
 - language, 33
 - Oz language, 835
 - Oz lexical, 843
- synthesized argument, 163
- system exception, 97
- Szyperski, Clemens, 466
- tail recursion optimization, 74
- task, 784
- tautology, 645
- TCP (Transmission Control Protocol), 718, 745
- technology, xxix
 - dangers of concurrency, 23
 - magic, 314
 - software component, 466
 - synchronous digital, 271
 - transition to 64-bit, 80
- Tel, Gerard, 361
- tell operation, 786, 791

- temporal logic, 611
- temporary failure, 745
- term (in Prolog), 676
- termination
 - detection, 281, 390
 - ping-pong example, 311
 - failure in declarative program, 249
 - partial, 247, 344, 806
 - proof, 452
 - total, 806
- testing, 113, 222, 416
- text file, 213
- Thalys high-speed train, 390
- theorem prover, 119, 646, 675
- Therac-25 scandal, 23
- thin-to-thick software development, 455
- thinking machine, 633
- third-party independence, 341
- this**, *see* **self**
- Thompson, D'Arcy Wentworth, 413
- thread, 848
 - declarative model, 237
 - interactive interface, 90
 - introduction, 17
 - Java, 625
 - monotonicity property, 243, 785, 786
 - priority, 257
 - ready, 243
 - runnable, 243
 - suspended, 243
 - synchronization, 339
- thread** statement, 245, 789
- Thread** class, 625
- throughput, 268
- thunk, 437
- ticket, 485, 721
- ticking, 311
- time complexity, 13
- time out, 745
 - Erlang, 399–402
 - system design, 464
- time slice, 257–259
 - duration, 258
- time-lease mechanism, 485, 740, 744
- timer protocol, 376
- timestamp, 209, 611
- token equality, 426, 720, 729
- token passing, 587, 596, 599
- token syntax (of Oz), 835
- tokenizer, 34, 164
- top-down software development, 10, 455
- total termination, 806
- trade-off
 - asynchronous communication vs. fault confinement, 751
 - compilation time vs. execution speed, 461
 - composition vs. inheritance, 466
 - compositional vs. noncompositional design, 465
 - dynamic vs. static scoping, 60
 - functional vs. type decomposition, 547
 - helper procedure placement, 123
 - indexed collections, 439
 - kernel language design, 846
 - lazy vs. eager execution, 335
 - manipulability vs. expressiveness, 691
 - names vs. atoms, 516
 - non-strictness vs. state, 337, 351
 - optimistic vs. pessimistic scheduling, 612
 - planning vs. refactoring, 456
 - semantics vs. implementation, 791
 - single vs. multiple inheritance, 537
 - specification language, 119
 - type view vs. structure view, 526
- transaction, 588, 608–625
 - bank example, 205
 - database, 668
 - distributed, 625
 - incarnation, 616
 - logging example, 511
 - nested, 625
 - restart, 614
 - tell operation, 792, 813
- transaction manager, 619
- transaction processing monitor, 619
- transducer, 263
- transition, 376
- transitive closure, 467–475
 - declarative algorithm, 469, 473
 - Floyd-Warshall algorithm, 473
 - stateful algorithm, 471

- translator, *see* compiler
- tree, *153*
 - balanced, *154*
 - depth, *155*
 - drawing, *161*
 - Haskell expression, *328*
 - node depth, *159*
 - ordered binary, *154*
 - parse, *34, 164, 655*
 - search, *637*
 - stream merge, *406*
 - syntax, *164*
 - ternary, *154*
 - traversal, *158*
- trigger, *196, 286–290, 848*
 - by-need, *583*
 - internal, *286*
 - programmed, *286*
- true**, *826*
- try** statement, *94, 802*
- tuple, *53, 827*
 - usage trade-offs, *441*
- tuple space, *460, 594–598*
- Turing machine, *43, 117, 691, 848*
- two-phase locking, *612*
- type, *51, 197*
 - abstract, *197*
 - ADT, *197*
 - basic, *53*
 - bundled, *429*
 - class, *331*
 - concurrent ADT, *585*
 - declarative, *428*
 - descriptive, *132*
 - dynamic, *52, 106–108, 715*
 - hierarchy, *52*
 - inference, *99, 140, 327*
 - open, *427*
 - polymorphic, *330*
 - secure, *204–213, 427*
 - signature, *131, 327*
 - stateful, *428*
 - stateless, *428*
 - static, *xl, 52, 106–108*
 - strong, *106, 327*
 - unbundled, *428*
 - weak, *106*
- type decomposition, *528, 547*
- UML (Unified Modeling Language), *493, 534*
- undecidable problem, *212*
- Unicode, *462, 822*
 - Java, *560*
- unification, *100–106, 848*
 - algorithm, *102*
 - distributed, *739*
 - equivalence set of variables, *103*
 - examples, *101*
- unification grammar, *662*
- unit**, *826*
- Unix operating system, *xxxviii, 258, 311, 463, 560, 654*
 - pipe, *xxx, 261*
- URL (Uniform Resource Locator), *214, 721*
 - limited lifetime, *463*
- value, *45*
 - complete, *47*
 - failed, *334, 804, 849*
 - partial, *47*
- value store, *44*
- var** operation (in Prolog), *675*
- variable, *4*
 - binding, *45, 47, 791*
 - dataflow, *44, 50, 848*
 - declarative, *44*
 - determined, *67, 103, 209, 288, 338, 673, 794*
 - equivalence set, *103*
 - escaped, *506, 515*
 - final (in Java), *558*
 - free, *59, 645*
 - fresh, *663, 790*
 - global, *89*
 - identifier, *4, 46*
 - instance, *502*
 - interactive, *89*
 - local, *182*
 - logic, *102*
 - mutable, *see* cell
 - needed, *288, 799*
 - read-only, *209, 212, 355, 388, 800, 849*

- single-assignment, 342
- unification, 100
- Viriding, Robert, 590
- virtual machine, *xxxi*, 43, 685
- virtual string, 214, 833
- Visual Basic, 465
- Vlissides, John, 540
- VLSI-BAM microprocessor, 633
- VM (Virtual Machine) operating system, 43
- Vossen, Gottfried, 609
- Wait** operation (in dataflow), 794
- wait** operation (in monitor), 600
- wait point, 588
- wait-for graph, 614
- WaitQuiet** operation, 800
- WaitTwo** operation, 320, 321, 401, 404, 407, 653
- WAN (wide-area network), 717, 745
- Warren, David H.D., 633
- watcher, 745, 747
- Weikum, Gerhard, 609
- wheel, *xxxii*
- while** loop, 451
- widget (in GUI), 216, 692, 693
- Width** operation, 828
- Wikström, Claes, 590
- Wilf, Herbert S., 173
- Williams, Mike, 590
- window (in GUI), 693
- Windows operating system, *xxxviii*, 258
- wire, 371
 - many-shot, 371
 - one-shot, 371, 376
- Wood, David, 385
- word
 - frequency, 201
 - memory, 76
 - specification, 222
- wrapping and unwrapping, 207
- WWW (World-Wide Web), 214
- XML (Extensible Markup Language), 117