Index

#include
   and using, 108
   vs. forward declaration, 40
#include guards, 27, 33
   internal vs. external, 43
#undef
   as soon as possible, 33
   &&
      preferable to nested ifs, 38
?:, 36
[]. See operators, []
++C, 50

A
Abelson, Harold, 13
Abrahams, Dave, xv
abstraction, 20
   and dependency management, 11
   and get/set, 20, 72, 73
   and interfaces, 62
abstractions
   build higher-level from lower-level, 12
   depending upon instead of details, 41
   vs. details, 128
accumulate, 125
Acyclic Visitor, 41
ADL, 104, 105, 106, 107, 122
   and template customization, 122
   disabling unwanted, 124
aggregates, 20
Albaugh, Tyrrell, xv
algorithmic complexity, 14
   and STL, 14
   exponential, 15
   linear-looking that is really quadratic, 15, 156
algorithms
   and design patterns, 162
   are loops, 159
   binary_search, 165
   count, 165
   count_if, 165
   equal_range, 165
   find, 165
   find_if, 165
   lower_bound, 165
   nth_element, 166
   partial_sort, 166
   partial_sort_copy, 166
   partition, 166
   searching, 165
   sort, 166
   sorting, 166
   stable_partition, 166
   stable_sort, 166
   upper_bound, 165
   vs. loops, 38, 162
alignment, 176
Allison, Chuck, xv
allocation, 111
   never allocate more than once per statement, 25
allocator
element use of, 5
ambiguities, 77
ambiguities,
   avoiding declaration, 13
amortized constant time, 155
append, 135
arithmetic operators. See
   operators, arithmetic
arrays
   fixed-size, 15
   inferior to containers, 152
assert, 33, 130, 135
   example of, 5, 98, 175
   macro needed for, 33
   only for internal programming errors, 132, 134
   prefer instead of logic_error, 131
assertions. See assert
assignment
   copy. See copy assignment
   self, 99, 138
assignment operators. See
   operators, assignment
asymptotic complexity. See
   algorithmic complexity
at
   vs. [], 136
atomic operations, 21
auto_ptr, 94, 154
B

Bajaj, Samir, xv
BankAccount, 72
Barbour, Marc, xv
base classes. See classes, base
base two, 176
basic_string, 12, See also
containers
append, 135
find_first_of, 136
insert, 135
monolithic, 79
behavior
undefined. See undefined
behavior
Bell, Gordon, 13
Bentley, Jon, 13, 16
BetweenValues, 164
Big Four, 55, 85, 94, See also
default constructor; copy
construction; copy
assignment; destructor
Big-Oh. See algorithmic
complexity
binary compatibility, 116, 120
binary_function, 172
binary_search, 165
bind2nd, 162, 163
example use of, 163, 164
Bird, 67
bloat, 112
Boedigheimer, Kim, xv
Boost, 3, 147, See also
shared_ptr
discriminated unions library,
121
format library, 184
Lambda library, 4, 162, 163,
164
Lambda library, example use
of, 163
preprocessor library, 33
bounds checking, 29, 152
brace placement, 2
braces. See brace placement
matching, 38
branch prediction, 16
Bridge, 162
buffer overruns. See security
bugs. See insects
build
breaking, 8
unit tests, 8
build system
automated, 7
build times, 76
C
C, 36, See also C, obsolete uses
of
C, obsolete uses of, xi
arrays, 37, 152, 186
casts, 180, 181
global namespace, 108
Hungarian notation, 3
implicit cast from const
char[] to (non-const) char*
hole in the type system,
179
macros, 32, 33
manual memory
management, 24, 152
manual resource
management, 24, 152
memcpy/memcmp (except
for PODs), 182
null-terminated character
array strings, 37, 152
pointer arithmetic, 152
printf, 184
realloc, 12
sprintf, 184
switching on a type flag, 174,
175
unions to reinterpret
representation, 183
unsafe functions
(strcmp/strncmp, strcmp,
sprintf, gets, etc.), 185
varargs, 46, 184
variable definition at
beginning of scope, 35, 36
C++
vs. ++C, 50
caching, 16
caffeine
lack of, 96
callback functions, 133
and exceptions, 114
instead of locking, 23
Carlson, Richard
reference to, 2, 144, 155
casts, 180
and not const, 179
explicit preferred, 6
catch
difficulties, 81, 93, 114, 115, 133, 140
Catch-22, 127
cerr, 19, 113
char_traits, 125
check in. See version control
system
check out. See version control
system
checked STL implementation,
160
checked_cast, 178
cin, 19, 113
clarity
prime importance of, 13
class templates. See also
templates
specialization, 127
classes
and namespces, 104
and nonmember functions,
104
and portability, 116
base, 56, 69, 90, 91, 96, 101
composition vs. inheritance,
58, 61
canonical, 60, 91
data members, 72
derived. See polymorphism
and substitutability
exception, 56
ekinds of, 56
minimal vs. monolithic, 57
mixin, 65
policy, 56, 65, 91
traits, 56
unions, 183
value, 56, 101, 154

clean compiles. See compiler

warnings

clear

better than cute, 13

cliff, 85

Cline, Marshall, xv

clog, 113

Clone, 96, 97

vs. copy construction, 97

Cobol, 36

code reviews, 9

this book’s table of contents

as checklist, 9

coding style

vs. design style, 11

cohesion, 12, 38

COM, 7, 63, 91, 115, 133

Command, 41, 121

comments, 2

CompareThings, 171

compatibility

source vs. binary, 73

compile

cleanly. See compiler

warnings

compile time

and errors, 28

compiler firewall. See Pimpl

compiler warnings, 4

compiler-generated functions,
85, See copy construction;
copy assignment; destructor

compile-time

conditions, 29

errors, 27

polymorphism, 29

complex

simple better than, 13

complexity

algorithmic. See algorithmic

complexity

asymptotic. See algorithmic

complexity

compose, 163

compose2

example use of, 164

composition

vs. inheritance, 58, 61

concurrency, 19, 21, See also

locking

vast majority of objects not

shared, 22

conditional compilation, 33

conditions

compile-time, 29

const, 27, 30

and pointers, 30

avoid on pass-by-value

parameters, 31

instead of magic numbers,
34

not deep, 30

simplifies code, 30

viral, 30

const_cast, 179

const-correctness, 31, 128, 179

construction

copy. See copy construction

construction order

of member variables, 86

ConstructionWasOK

not recommended, 141

constructor parameters

prefer named variables

instead of temporaries, 13

constructors

and virtual functions, 88

copy. See copy construction

default. See default

constructor

initialization list, 87

initialization list ordering

not significant, definition

order significant, 86

post-constructors, 88

prefer initializer list instead

of assignment, 18

reporting errors from, 141,

142

virtual constructors, 88

containers

and copy

construction/assignment,

95

and smart pointers, 95

and thread safety, 21

choosing, 150

default, 150

hash-based, 15, 150, 181

heterogeneous, 154

index, 154

map, and optional values,

154

of non-value types, 154

range vs. single-element

functions, 155, 156

shrink-to-fit, 157

store values, 154

string, 152

vector, 150, 152, 153

vector vs. list, 151

vector, advantages of, 150

conversion sequences, 70

conversions

implicit, 70. See implicit type

conversions

named functions, 70

copy, 107

copy assignment, 25, 55, 85, 87,

99

and containers, 95

and copy construction, 94, 95

and destructor, 94

and swap, 101

not virtual, 99

copy construction, 25, 55, 85

copy constructors

and containers, 95

and copy assignment, 94, 95

and destructor, 94

vs. Clone, 97

copy-on-write, 23

CORBA, 7, 63, 91, 115, 133

correct

better than fast, 13

correctness

prime importance of, 13

corruption, 21

count, 165

count_if, 165

coupling, 19

cout, 19, 113

covariance, 69
COW. See copy-on-write
CPU-bound, 17
Create, 89
curly braces. See brace
placement
CustomAllocator, 80
customization
and C++ standard library,
125
of templates, 122
CustomString, 117
cute
clear better than, 13
cvs, 8
cyclic dependencies, 40
cycling, 41

D
dangling pointers, 185
data
exposing, 20
global. See global variables
data validation, 29
data volumes
growth of, 14
database-bound, 17
Date, 72
deadlock, 21
deallocation, 111
deallocation functions
never fail, 92
Dechev, Damian, xv
declaration
vs. definition, 40
declaration ambiguities
avoiding, 13
default, 175
default arguments
and virtual functions, 66
default constructor, 55, 85, 87,
156
default container
vector, 150
definition
of member variables, 86
vs. declaration, 40
delete. See also operators, delete
and polymorphism, 91
with new, 80
dependencies, 103
and templates, 42
compile-time, 58
cyclic. See cyclic
dependencies
managing, 20
upon abstractions instead of
details, 41
dependency cycles
across modules, 41
Dependency Inversion
Principle, 41, 62
dependency management, 74,
See also encapsulation and
information hiding
broad importance of, 11
member vs. nonmember
functions, 79
dependent names, 125
development
ease of, 57
design patterns
and algorithms, 162
design style
design vs. coding style, 11
destructor, 55, 85
and copy assignment, 94
and copy construction, 94
nonvirtual, 61, See also
slicing
public and virtual, 63
destructors, 68, See also RAI
and exceptions, 115
and virtual functions, 88
in base classes, 90
never fail, 92
details
vs. abstractions, 128
Dewhurst, Steve, xv
Diamond, Norman, 85
Dimov, Peter, xv
dint
gratuitous use of odd word,
162
disabling warnings. See
warnings
disk capacity
growth of, 14
disk-bound, 17
distance, 107, 156, 165
divide and conquer. See
minimal vs. monolithic
DLLs, 103
DoClone, 98
downcasts, 29
Draw, 175
dusty corners, 13
dynamic_cast, 69, 178
downcasting with, 29
dynamically checked errors. See
d errors, dynamic checking

E
EBO. See empty base class
optimization
ECO. See empty base class
optimization
efficiency. See performance
empty base class optimization,
59, 63
empty()
vs. size() == 0, 128
equal_range, 165
ER units
comparison with, xiii
erro, 140, See also error codes
error code
overuse, 142
error codes
translating to/from
exceptions, 115
vs. exceptions, 140
error handling policy. See
errors, policy for handling
error messages
and macros, 33
error safety, 57, 59, 77
Index

and RAII, 24
errors
and modules, 133
and operators, 141
assert, 130
categorizing, 133
compile-time, 28
constructors, 141
detection, 133
dynamic checking, 28
dynamic checking, 28
exponent, 28
handling, 133, 145
identifying, 132
ignoring, dangers of, 140
internal assumptions, 130
invariants to test for. See invariants
link-time, 28
policy for handling, 132
prefer compile- and link-time to run-time, 27, 28
propagating, 140
propagation, 133
reporting, 133, 145
run-time, 132
severity, 133
static checking, 28
translating, 144, 145
vs. non-errors, 134
error-safety, 150
basic guarantee, 137
copy construction, 99
no-fail guarantee, 137
not penalizing code that doesn’t need stronger guarantees, 137
strong guarantee, 137
evil
root of all, 11
exception
what, 147
exception classes. See classes, exception
exception handling. See also errors; error-safety
catch by reference, 144
overuse, 142
throw by value, 144
warning against disabling, 143
exception safety. See error safety
exception specifications, 93, 146
avoid, 146
static vs. dynamic checking, 147
exceptions
and callback functions, 114
and destructors, 115
and main, 114
and modules, 114
and slicing, 144
and threads, 114
not across module boundaries, 114
translating to/from error codes, 115
vs. error codes, 140
explicit, 70, 97
explicit loops
fewer in STL-using programs, 162
explicit qualification, 77, 110
expression templates, 50, 53
external locking, 22

F
facets
mistakes of, 121
factory
example use of, 89
Factory, 162
factory functions, 19
fast
correct better than, 13
File, 72, 136
find, 18, 165
find_first_of, 136, 142
find_if, 165, 169
FlagNth, 169
Fly, 67
fools, 11
for_each, 15, 162
example use of, 161
formatting, 2
Fortran, 36
forward declaration
vs. #include, 40
French
graffiti use of, 51
friend, 55
fudgeFactor, 112
full build, 7, See also build system
Fuller, John, xv
gfunction
to avoid uninitialized variables, 37
unit of work, 134
function arguments
order of evaluation, 54
function objects, 162, See also predicates
gexample use of, 164
gvs. functions, 170
gwriting correctly, 172
gfunction parameters, 45
gand binders, 162
gand compile-time dependencies, 76
gand const, 31, 46
gand conversions, 48
gand copying, 46
gand null, 46
gand preconditions, 134
gand primitive types, 46
gand smart pointers, 46
gand unary_function/binary_function, 170
gand user-defined types, 46
gand varargs, 46
in constructors, 89
ginput, 46
goutput, 46
gpass by value vs. pass by reference, 46
gunary and binary operators, 48
function templates, 113
gand not specialization, 126
gand overload resolution, 126
gfunctions
compiler-generated, 85
deallocation, 92
length, 38
member vs. nonmember, 48, 79
nesting, 38
vs. function objects, 170
functions, compiler-generated.
See default constructor; copy construction; copy assignment; destructor

G
Gaffney, Bernard, xv
generic programming. See templates
genius, 11
get/set, 73
and abstraction, 20, 72, 73
GetBuffer, 75
GetBuilding, 66
GetLastError, 140
getstr, 53
global data. See global variables
global state. See global variables
global variables, 19, 39
and dependency management, 11
initialization of, 19
limit parallelism, 19
Gordon, Peter, xv
greater
example use of, 164
grep, 181
Griffiths, Alan, xv
guarantees
for error safety. See error-safety

H
handles
to internal data, 74
hash-based containers. See containers, hash-based
Haskell, 28
header files
self-sufficient, 42
wrapping third-party
headers, 4
header guards. See #include guards
headers
and linkage, 112
and not unnamed
namespaces, 113
and static, 113
precompiled, 42
Henney, Kevlin, xv
Henning, Michi, xv
heterogeneous containers, 154
hide information. See information hiding
hiding
names, 66, 82
hijacking
and macros, 32
Hinnant, Howard, xv
Hoare, C.A., 16
Hungarian notation, 3
hygiene
and not macros, 32
Hyslop, Jim, xv
implicit conversions, 70
benefits of, 71
dangers of, 71
implicit interface, 122
and customization, 122
implicit type conversions
avoided by overloading, 51
import this, xv
incremental build, 7, See also build system
indentation, 2
index containers, 154
indexing
vs. iterators, 128
information hiding, 72
and dependency management, 11
inheritance
and dependency management, 11
and reuse, 64
misuse of, 64
not from concrete base classes, 60
public, 64
vs. composition, 58, 61
initialization
and constructors, 87
default, 87
of global variables, 19
of member variables, 86
of variables, 35, 36
static vs. dynamic, 39
variables. See variable, not initialized
zero, 39
initialization dependencies, 39
inline, 17, 113
and profiler, 17
in- "new" \t "See also operators, new" XE "delete"
\t "See also operators, delete"
place new. See new
insects, 9, 12, 28, 30, 35, 36, 39, 52, 81, 137
insert, 135, 139, 156
at a specific location, 150
inserter
example use of, 163
interface
implicit. See implicit interface
Interface Principle, 104
interfaces
abstract, 62
intermittent crashes, 36
internal locking, 22
internals
exposing, 20
invalid iterators, 185
invariants, 18, 20, 28, 64, 72, 73, 74, 130, 131, 132, 134, 135, 136, 137, 138, 140, 141, 142
iostreams, 113
is_in_klingon, 61
Index 215

is-a. See substitutability, See substitutability
IsHeavy, 170
iterator ranges, 161
iterator_traits, 125
iterators, 151
  comparing with != instead of <, 128
invalid, 161, 185
ranges, 161
vs. indexing, 128

J
Java, 28, 147
Johnson, Curt, xv
Josuttis, Nicolai, xv
juggling, 152

K
K&R style. See brace placement
Kalb, Jon, xv
Kanze, James, xv
Kernighan, Brian, 173
Khesin, Max, xv
KISS, 13
Knuth, Donald, 11, 16
Koenig lookup. See ADL

L
Lafferty, Debbie, xv
Lambda library. See Boost, Lambda library
land mines, 27
Last Word
  not this book, xii
Latin
  gratuitous use of, 59, 141
LaunchSatellite, 139
Law of Second Chances, 63
leak
  memory, 81
leaks, 137
Leary-Coutu, Chanda, xv
Leddy, Charles, xv
length
  of lines, 2
less
  example use of, 164
libraries
  shared, 103
lifetime. See object lifetime
line length, 2
link time
  and errors, 27, 28
linkage
  and headers, 112
  external, 19
Lippman, Stan, xv
Liskov Substitution Principle. See substitutability
Lisp, 28
list. See also containers
  vs. vector, 151
literals
  and magic numbers. See magic numbers
livelock, 21
locality of reference, 151
localized_string, 61
locking
  external, 22
  in increasing address order, 23
  internal, 22
lock-free designs, 23
not needed for immutable objects, 23
using callback functions instead of, 23
logic_error
  example of, 5
  prefer assertions instead of, 131
lookup
  two-phase, 125
loops
  fewer explicit loops in STL-using programs, 162
  vs. algorithms, 162
lower_bound, 165

M
macros, 27, 32
  and conditional compilation, 33
  interfering with template instantiations, 33
  to enable/disable threading support, 23
magic numbers, 34
main
  and exceptions, 114
make, 7. See also build system
malloc, 131
mem_fun, 170
mem_fun_ref, 170
member variables
  public vs. private, 72
member vs. nonmember functions, 79
memcmp, 182
memcpy, 182
memory leaks, 81
memory management
  and containers, 152
memory-bound, 17
MemoryPool, 82
Meyers, Scott, xv
Ming vases, 152
minimal vs. monolithic, 55, 57
missing return. See return, missing
mixin classes. See classes, mixin
ML, 28
modules
  allocating and deallocating memory in same, 111
  and error handling, 133
  and exceptions, 114
and not exceptions, 114
defined, 103
interdependence between, 40
interfaces use only
sufficiently portable types, 116
monolithic classes, 79
monolithic vs. minimal, 55, 57
Moore’s Law, 14
Mullane, Heather, xv
mutable, 30

N
name hiding, 66, 82
name lookup, 77
two-phase, 125
named variables
prefer as constructor
parameters, 13
names
dependent, 125
symbolic vs. magic numbers, 34
namespaces, 103
and using, 108
pollution of, 13, 108, 109, 110
type and its nonmember
functions in same, 104
type and unrelated functions
in separate, 106
unnamed. See unnamed
namespace
classifying.

O
object lifetime
minimizing, 35
objects
temporary. See temporary
objects
Observer, 162
obsolete practices, 2, See C,
obsolete uses of
external #include guards, 43
Hungarian notation, 3
SESE. See single entry single
exit
Occam, William of, 51
ODR. See one definition rule
offsetof, 176
ointment
fly in the, 81
one definition rule, 110
operator delete
ever fails, 92
operator overloading
adjuncts, 13
preserve natural semantics,
47
operators, 45
&&, 52
(), 168
"52
[][, 135, 136
[[vs. iterators, 128
| |, 52
++, 17, 18, 50
and ADL, 105
and namespaces, 104, 105
arithmetic, 48
assignment, 48, 78, 93
binary, 48
const char* (on strings), 71
copy assignment. See copy
assignment
decrement, 50
delete, 80, 82, 93, 111
increment, 50
member vs. nonmember, 48
new, 80, 82, 111, 141
overloaded, 47
preserve natural semantics,
47, 48, 50
reporting errors from, 141
optimization. See also
temporary objects, See also
temporary objects, See also
temporary objects, See also
temporary objects and exception specifications,
146
and inline, 17
and libraries, 17
by using STL, 18
compile-time evaluation, 121
copy-on-write outdated, 157
empty base class, 63
enabling compiler’s, 49, 99
capsulate where possible,
17
in STL implementations, 94
indexing vs. single-element
definitions, 128
must be based on
measurement, 16
prefer improving
algorithmic complexity
over micro-optimizations,
17
premature, 13, 14, 15, 16, 17,
18, 50, 51, 59, 87, 171
range vs. single-element
definitions, 156
self-assignment check, 138
static binding, 121
optional values
and map, 154
order dependencies, 19, 23, 25,
39, 52, 53, 54, 69, 86, 109, 110,
124, 169, 176
Ostrich, 67
out_of_range, 136
overload resolution, 77
overloading
and conversions, 70
and function templates, 126
of operators, 13
to avoid implicit type
conversions, 51
overriding, 66

P
pair, 56
parameters
pass by value vs. pass by
reference, 18
unused. See unused
parameters
partial specialization. See
specialization, partial
partial_sort, 166
example use of, 167
partial_sort_copy, 166
partition, 162, 166
dexample use of, 166
Pascal, 36
Peil, Jeff, xv
pejorative language
and macros, 32
performance, 28, 141
Perlis, Alan, xi, xv, 11, 27, 45, 60,
103, 129, 173
personal taste
matters of, 2
pessimization, 18
Pimpl, 30, 58, 69, 72, 76, 78, 101,
172, See also encapsulation
and dependency
management
and shared_ptr, 78
pipelining, 16
Pirkelbauer, Peter, xv
placement
of braces. See brace
placement
plain old data. See POD
platform-dependent operations
wrapping, 21
Plauger, P.J., 173
plus, 162, 163
dexample use of, 163
POD, 176, 183
point_to_unary_function, 170
pointers
and const, 30
and not static_cast, 178
dangling, 185
points of customization. See
customization
class. See classes,
class
policy
policy-based design, 63
pollution (of names and
namespaces), 19, 35, 108, 109,
110
polymorphism, 66
ad-hoc, 120
del and delete, 91
destruction, 90
and not arrays, 186
and slicing, 96
compile-time vs. run-time,
29
decomposed, 64, 120
inclusion, 120
static, 63, 120
static vs. dynamic, 120, 175
static vs. slicing, 65
vs. slicing, 144
vs. switch on type tag, 38
descriptor on type, 174
Port, 24
portable types
and module interfaces, 116
postconditions, 66, 69, 124, 130,
131, 134, 135, 136, 138, 140,
142
and virtual functions, 66
post-constructors, 88
PostInitialize, 89
pragmatists, 11
Prasertsith, Chuti, xv
precompiled headers, 42
preconditions, 66, 69, 132, 134,
135, 136, 142
and virtual functions, 66
predicates. See also function
doctors
pure functions, 168
premature optimization. See
optimization, premature
pressure
schedule pressure, xiii
priority_queue, 166
processes
multiple, 21
profiler
and inline, 17
using. See optimization
proverbs
Chinese, 8
German, 177
Latin, 16, 156
level of indirection, 126
Romanian, 177
Prus, Vladimir, xv
ptr_fun, 170
public data, 20
push_back, 15, 155
Python, 28
Q
qualification
explicit, 77
qualification, explicit, 110
qualified
vs. unqualified, 123
R
race conditions, 21
RAII, 5, 24, 38, 56, 94, 95
  and copy assignment, 25
  and copy construction, 25
range checking, 135
ranges
  of iterators, 161
realloc, 12
Rectangle, 64
recursive search
  not reporting result using
  exception, 142
reference counting, 135
registry
  factory and, 19
reinterpret_cast, 177, 180, 181,
  183, 184, 185
release
  unit of. See module
reliability, 27
remove_copy_if, 169
remove_if, 169
replace_if, 162
resource acquisition is
  initialization. See RAIIResource management, 94, See
  also RAIIRun time
  and errors, 27, 28
Ruby, 28
S
  safety, 27
Saks, Dan, xv
scalability
  coding for, 14
schedule pressure, xiii
Schwarz counters, 113
Schwarz, Jerry, 113
Second Chances
  Law of, 63
security, 15
  and checked STL
    implementation, 160
  and exception handling
    performance, 142
  arrays and, 15
  buffers, 152
  pointers, 152
  printf, 184
  ssh, 8
  strcpy, 185
security, 72
self-assignment, 99, 138
self-sufficient header files, 42
serialization
  of access to shared objects,
  21
SESE. See single entry single
  exit
shallow const, 30
Shape, 175
shared libraries, 103
shared state
  and dependency
  management, 11
shared_ptr, 111, 121, 149
  and arrays, 186
  and containers, 154
  and modules, 111
  and optional values in maps,
  154
  and overuse, 25
  and Pimpl, 78
  example use of, 24, 25, 76, 78,
  89, 182
  throwing, 144
shared_ptr, 149
shared_ptr, 172
sheep’s clothing, 39
shrink-to-fit, 157
signed/unsigned mismatch, 6
simple
  better than complex, 13
simplicity
  prime importance of, 13
single entry single exit, 3
Singleton, 39
skins, 139
slicing, 61, 96
  and polymorphism, 96
  of exceptions, 144
Smalltalk, 28
smart pointers, 98
  and containers, 95
  and function parameters, 46
  and overuse, 25
  for resource management, 24
Socket, 74
sort, 18, 125, 166
spaces
  vs. tabs, 3
spaghetti, 17
special member functions. See
default constructor; copy
  construction; copy
  assignment; destructor
specialization
  and not function templates,
  126
  of class templates, not
  function templates, 127
  partial, 126
speculative execution, 16
Spencer, Henry, 173, 177
Square, 64
ssh, 8
stable_partition, 166
stable_sort, 166
stack unwinding, 92
standards, xi
  advantages of, xii
what not to include, 2
Star Trek
   gratuitous reference to, 61
state
   global. See global variables
   static
   misuse of, 112
static type checking, 120
static_cast, 181
   and not pointers, 178
   downcasting with, 29
statically checked errors. See errors, static checking
STL
   algorithms. See algorithms
   checked implementation
   valuable, 160
   containers. See containers
   iterators. See iterators
   searching, 165
   sorting, 166
   use leads to fewer explicit
   loops, 162
   using, 18
STL containers
   and thread safety, 21
string. See basic_string, See basic_string
String, 75
Stroustrup, Bjarne, xv, 32, 55, 119, 129, 149, 159
strtok, 54
style
   design vs. coding, 11
substitutability, 59, 64, 66
subsumption, 120
SummarizeFile, 116
super_string, 60
surprises
   programmers hate, 53
Sussman, Gerald Jay, 13
swap, 93, 100, 125, 126, 127
   never fails, 92
swap trick, 157
switch
   default case, 5
T
   tabs
   vs. spaces, 3
taste
   matters of personal, 2
tautologies
   perfect for assertions, 131
template customization. See customization
   Template Method, 68, 90
templates
   and implicit interface. See implicit interface
   and source-level dependencies, 42
   function. See function
templates
   function templates not in same namespace as a
   type, 106
   macros interfering with, 33
   unintentionally nongeneric
   code, 128
temporaries
   avoid as constructor
   parameters, 13
temporary objects, 18, 51, 70, 98
tensor, 47
terminate, 146
testing, 20
tests
   unit tests, 8
TeX
   The Errors of TeX, 11
this
   import, xv
thread safely, 21
thread safety, 21
   “just enough”, 23
   and STL containers, 21
threads, 133
   and exceptions, 114
   multiple, 21
   vast majority of objects not
   shared across, 22
thrill sports, 152
time pressure, xiii
traits classes. See classes, traits
   transform, 162
   example use of, 163
Translate, 117
Transmogrify, 54, 96
Transmogrify2, 97
Transubstantiate, 96
Tree, 25
TreeNode, 73
try, 38
two-phase lookup, 125
two’s complement, 176
type checking
   static, 120
type safety, 28, 173, 176
type switching
   vs. polymorphism, 174
type system
   and not macros, 32
   and not memcpy/memcmp,
   182
   hole in, 179
   making use of, 28, 29, 30,
   131, 146, 173
type systems
   static vs. dynamic, 28
typename
   example use of, 122, 123, 125
types
   vs. representations, 176
U
   unary_function, 91, 170, 172
   Uncle Bob, xv
undefined behavior, 19, 25, 27,
   36, 39, 61, 71, 88, 90, 91, 93,
   173, 179, 181, 182, 183, 184,
   185
   unexpected_handler, 146
   uninitialized variables, 36
   unintentionally nongeneric
   code, 128
   unions, 183
   unit of work. See function
   unit tests, 8
   UnknownException, 146
   unnamed namespace
and not headers, 113
unqualified
vs. qualified, 123
unsigned
mismatch with signed. See
signed/unsigned
mismatch
unused parameters, 5
unwinding
stack, 92
upper_bound, 15, 165
using, 83
avoiding need for, 105
is good, 108
not before an #include, 108

V
validation
of input data, 29
value-like types. See classes,
value
Vandevoorde, Daveed, xv
varargs, 184
variable
defined but not used, 5
not initialized, 5
variable naming. See
Hungarian notation
variables
declaring, 35
global. See global variables
initialization of, 35
initializing, 36
uninitialized, 27
VCS. See version control system
vector. See also containers
by default, 150
insert, 139
vs. list, 151
version control system, 8
versioning, 103, 138
and get/set, 72
viral const, 30
virtual constructors, 88
virtual functions, 66
and constructors and
destructors, 88
destructors, 90
nonpublic preferred, 68
Visitor, 41, 121, 162, See also
Acyclic Visitor
volatile, 37

W
Wagner, Luke, xv
warnings
compiler. See compiler
warnings
disabling, 6
none on successful build, 7
spurious, dealing with, 6
Weinberg, Gerald, 1
what, 147
Wilson, Matthew, xv
works-like-a. See
substitutability. See
substitutability
wrapping
header files. See header files,
wrapping third-party
headers
platform-dependent
operations, 21
Wysong, Lara, xv

Z
zero initialization, 39
Zolman, Leor, xv