Glossary

forwarding reference **–** a function template *parameter* whose type is an unqualified rvalue reference to a template parameter (`T&&`). The deduced template argument corresponding to the template parameter will have the same value category as the function argument; reference collapsing will cause the function parameter of the instantiated function to be an lvalue reference or rvalue reference as appropriate. Forwarding References (377), Range **for** (680), *Rvalue* References (732), Variadic Templates (918), *Generic* Lambdas (971), **auto** Return (1184)

fragmentation **–** the process by which initially densely packed blocks of allocated memory, due to repeated allocation and deallocation, spread out across the address space over time — e.g., even when the total memory utilization does not increase. As a consequence, an allocation request for a sizable block of (contiguous) memory, which might have succeeded if requested earlier in the process lifetime, cannot be honored, despite there being an ample quantity of (noncontiguous) free memory available; see also memory diffusion. **alignas** (183)

free function **–** one that is *not* a member function; see also hidden friend idiom. Deleted Functions (58), Generalized PODs '11 (442), `initializer_list` (558)

free operator **–** a built-in or user-defined operator that is not implemented as a member of a class; see also member operator. User-Defined Literals (839)

freestanding **–** implies a subset of the C++ Standard Library and core language (due to missing pieces of the library) intended to be run on platforms (such as embedded systems) that might not have a fully functioning underlying operating system. `initializer_list` (570)

full expression **–** one that is not a subexpression of another expression; see outermost expression. Range **for** (693)

full specialization **–** a (colloquial) synonym for explicit specialization.

fully associative **–** a form of cache associativity where any cache line can reference any region in main memory, obviating evictions until every cache line has been populated. **alignas** (182)

fully constructed **–** implies, for a given object, that its initialization, if any, has been completed. In particular, if initialization requires invoking a nondelegating constructor, fully constructed implies that the constructor has finished running, whereas if initialization requires invoking a delegating constructor, fully constructed implies that the target constructor has finished running. Delegating Ctors (47)

function designator **–** a term in the C Standard used to describe an expression having function type. *Rvalue* References (815)

function object **–** a callable object (a.k.a. functor); see also invocable object. **constexpr** Functions (292), Default Member Init (328), Lambdas (574), Lambda Captures (990)

function parameter list **–** the parameter declarations that specify the arguments with which a function may be invoked and, along with the function's name, contribute to its signature.

function parameter pack **–** a pack expansion within a function parameter list that defines a function parameter for each element of a named template parameter pack. Variadic Templates (879)

function prototype **–** (the specification of) a nondefining declaration of a function, which includes its signature, return type, and any other features, such as a **noexcept** specification (see Section 3.1."**noexcept** Specifier" on page 1085) that would distinguish it from other like-named functions. *Rvalue* References (733)

function scope **–** used (colloquially within this book) to mean block scope. Note that the C++ Standard as published (through C++20) has an entirely different definition of this term,