C++ Glossary

A glossary is a list of terms in a subject or field, with accompanying **definitions**. It explains or defines difficult or unusual words and expressions used in the field.

Below follows a glossary of C++ terms extracted from:
https://sites.google.com/site/simplestjava/c-glossary

You can also consult Bjarne Stroustrup's (who designed and implemented C++) C++ glossary at http://www.stroustrup.com/glossary.html

Term	English
abstract class	a class that can only be used as a base class for some other class. A class is abstract if it has at least one pure virtual function.
access control	a C++ mechanism for prohibiting or granting access to individual members of a class
access declaration	a way of controlling access to a specified member of a base class when it is used in a derived class
access specifier	a way of labeling members of a class to specify what access is permitted
allocation	the process of giving memory space to an object
ANSI	acronym for American National Standards Institute, a standards body currently standardizing C++
argument	when calling a function, refers to the actual values passed to the function
argument matching	the process of determining which of a set of functions of a specified name matches given arguments in a function call
array	an ordered and index-able sequence of values. C++ supports arrays of a single dimension (a vector) or of multiple dimensions
assignment	the process of giving a value to a pre-existing object
assignment operator	an operator for doing assignment, the '=' operator
base class	a class that serves as a base for a derived class to inherit members from
bool	C++ keyword used to declare a Boolean data type

break	C++ keyword used to specify a statement that is used to break out of a for or while loop or out of a switch statement
browser	a software development tool used for viewing class declarations and the class hierarchy
built-in type	see fundamental type
C	a programming language in widespread use. C++ is based on C.
C-string	a partially filled array of characters with the `'\0'` character used as sentinel to mark the end of the c-string.
call by reference	passing a pointer to an argument to a function. The function can then change the argument value.
call by value	passing a copy of an argument to a function. The function cannot then change the argument value. C and C++ use call by value argument passing.
calling conventions	refers to the system-specific details of just how the arguments to a function are passed. For example, the order in which they are passed on the stack or placed in machine registers.
case	a C++ keyword used to denote an individual element of a switch statement
cast	a way of doing explicit type conversion via a cast operator
catch	a C++ keyword used to declare an exception handler
cerr	in C++ stream I/O, the standard error stream
char	a C++ keyword used to declare an object of character type. Often considered the same as a byte, though it is possible to have multi-byte characters.
cin	in C++ stream I/O, the standard input stream
class	a C++ keyword used to declare the fundamental building block of C++ programs. A class has a tag (name), members, access control mechanisms, and so on.
class hierarchy	see base class, derived class
class layout	the way in which data class members are arranged in a class object

class library	a set of related classes declared in header files and defined in object files
class member	a constituent member of a class, such as a data declaration, a function, or a nested class
class template	a template used for generating class types
comments	C++ has C-style comments delimited with /* and */, and new C++-style line-oriented comments starting with //
compilation unit	see translation unit
compiler	a software tool that converts a language such as C++ into a different form, typically assembly language
const	a C++ keyword used to declare an object as constant or used to declare a constant parameter
constant	a literal or variable declared as const
constant expression	a C++ expression that can be evaluated by the compiler. Used to declare bounds for an array among other things.
constructor	a function called when a class object comes into scope. The constructor is used to initialize the object.
const_cast	a C++ keyword used as a style of cast for explicitly casting away const
container class	a type of class or template that is used to hold objects of other types. Lists and stacks would be examples of container classes.
continue	C++ keyword used with for and while statements to continue the iteration at the top of the loop
conversion	to convert from one data type to another
copy constructor	a special type of constructor that is called when an object is copied
cout	in C++ stream I/O, the standard output stream
data abstraction	the idea of defining a data representation (for example, to represent a calendar date), and a set of operations to manipulate that representation, with no public access to the representation except via the operations.
deallocation	the processing of freeing memory space previously used by an object

debugger	a tool for stepping through the execution of a program, examining variables, setting breakpoints, and so on
declaration	a C++ entity that introduces one or more names into a program
declaration statement	a declaration in the form of a statement that may be used in C++ where statements would normally be used
declarator	a part of a declaration that actually declares an identifier name. A declarator appears after a sequence of type and storage class specifiers.
default argument	an optional argument to a function. A value specified in the function declaration is used if the argument is not given.
delete operator	C++ keyword and operator used to delete dynamic storage
delete[] operator	See delete operator. Used to delete array objects.
demotion	converting a fundamental type to another fundamental type, with possible loss of precision. For example, a demotion would occur in converting a long to a char.
deprecate	to make obsolete (a language feature)
derived class	a class that inherits members from a base class. See inheritance.
destructor	a function called when a class object goes out of scope. It cleans up the object, freeing resources like dynamic storage. See constructor and deallocation.
do	see while
dominance	refers to the case where one name is used in preference to another. See multiple inheritance.
double	C++ keyword used to declare a floating point type
dynamic storage	refers to memory allocated and deallocated during program execution using the new operator and delete operator
dynamic_cast	a C++ keyword that specifies a style of cast used with run-time type information. Using dynamic_cast one can obtain a pointer to an object of a derived class given a pointer of a base class type. If the object pointed to is not of

	the specified derived class, dynamic_cast will return 0.
else	C++ keyword, part of the if statement
encapsulation	a term meaning to wrap up or contain within. Used in relation to the members of a class. See access control.
enum	C++ keyword used to declare an enumeration
enumeration	a set of discrete named integral values. See enum.
enumerator	a member of an enumeration
exception	a value of some type that is thrown. See exception handling.
exception handler	a piece of code that catches an exception. See catch and try block.
exception handling	the process of signalling that an exceptional condition (such as divide by zero) has occurred. An exception is thrown and then caught by an exception handler, after stack unwinding has occurred.
explicit	a C++ keyword used in the declaration of constructors to indicate that conversion of an initializer should not take place
expression	a combination of constants, variables, and operators used to produce a value of some type
expression statement	a statement that is an expression, such as a function call or assignment
extern	a C++ keyword used to declare an external name
external name	a name available to other translation units in a program. See linker and global variable
false	C++ keyword used to specify a value for the bool type
finalization	to declare that an object or resource is no longer needed, and initiate cleanup of that object. See initialization.
float	a C++ keyword used to declare a floating point type
floating point	non-integral arithmetic. A floating-point number is typically represented as a base-two fraction part and an exponent.

for	a C++ keyword used to specify an iteration or looping statement
forward class	a class for which only the tag has been declared. Such a class can be used where the size of the class is not needed, for example in pointer declarations.
free store	see dynamic storage
friend	a type of declaration used within a class to grant other classes or functions access to that class. See access control.
front end	often refers to the early stages of C++ compilation, such as parsing and semantic analysis
function	a C++ entity that is a sequence of statements. It has its own scope, accepts a set of argument values, and returns a value on completion.
function template	a template used for generating function types
fundamental type	a type built in to the C++ language. Examples would be integral types like `int` and pointer types such as `void*`.
garbage collection	a way of automatically managing dynamic storage such that explicit cleanup of storage is not required. C++ does not have garbage collection. See new operator and delete operator.
generic programming	see template
global name	a name declared at global scope
global namespace	the implicit namespace where global variables reside
global scope	see global namespace
global variable	a variable that is accessible throughout the whole program, whose lifetime is that of the program
grammar	a way of expressing the syntax of a programming language, to describe exactly what usage is valid and invalid
header	see header file
heap storage	see dynamic storage
helper class	a class defined as part of implementing the details of another class
hiding	see encapsulation

if	C++ keyword used in conditional statements
implementation-dependent behaviour	not every aspect of a programming language like C++ is specified in a language standard. This term refers to behaviour that may vary from implementation to implementation.
implicit conversion	a conversion done as part of another operation, for example converting a pointer type to `bool` in an `if` statement
inheritance	the process whereby a derived class inherits members from a base class. A derived class will also add its own members to those of the base class.
initialization	to give an initial value to an object. See constructor and assignment.
initialize	the process of initialization
initializer	a value or expression used to initialize an object during initialization
inline	C++ keyword used to declare an inline function
inline function	a function that can be expanded by a compiler at the point of call, thereby saving the overhead time required to call the function
instantiation	see template instantiation
int	a C++ keyword and fundamental type, used to declare an integral type
integral conversion	the process by which an integer is converted to signed or unsigned
integral promotion	the process by which a `bool`, `char`, `short`, enumerator, or bit field are converted to `int` for use in expressions, argument passing, and so on
keyword	a reserved identifier in C++, used to denote data types, statements of the language, and so on
library	a set of object files grouped together. A linker will search them repeatedly and use whatever object files are needed. See class library.
lifetime	refers to the duration of the existence of an object. Some objects last for the whole execution of a program, while other objects have a shorter lifetime.
linkage	refers to whether a name is visible only inside or also outside its translation unit

linker	a program that combines object files and library code to produce an executable program
literal	a constant like 1234
local	typically refers to the scope and lifetime of names used in a function
local class	a class declared local to a function
local variable	a variable declared local to a function
long	C++ keyword used to declare a long integer data type.
long double	a floating point type in C++
lvalue	an expression referring to an object. See rvalue.
macro	a preprocessor feature that supports parameter substitution and expansion of commonly-used code sequences. See inline function.
member	see class member and namespace member
member function	a function that is an element of a class and that operates on objects of that class via the this pointer to the object.
memberwise copy	to copy an object a member at a time, taking into account a copy constructor for the member.
name method	see member function
mixed-mode arithmetic	mixing of integral and floating point arithmetic
module	see translation unit
multiple inheritance	a derived class with multiple base classes. See inheritance.
mutable	C++ keyword declaring a member non-constant even if it is a member of a const object
name	an identifier that denotes an object, function, a set of overloaded functions, a type, an enumerator, a member, a template, a namespace, or a label
name lookup	refers to taking a name and determining what it refers to, or its value, based on the scope and other rules of C++
name space	a grouping of names
namespace	a C++ keyword used to declare a namespace, which is a collection of names such as function declarations, classes, and so on

namespace alias	an alias for a namespace, that can be used to refer to the namespace
namespace member	an element of a namespace, such as a function, typedef, or class declaration.
nested class	a class declaration nested within another class
new operator	C++ keyword and operator used to allocate dynamic storage.
new[] operator	see `new` operator. Used to allocate dynamic storage for array objects
NULL	a special constant value that represents a null pointer
null pointer	a pointer value that evaluates to zero
object	has several meanings. In C++, often refers to an instance of a class. Also more loosely refers to any named declaration of a variable or other entity that involves storage.
object file	in C or C++, typically the output of a compiler. An object file consists of machine language plus an external name list that is resolved by a linker.
object layout -	refers to the ordering of data members within a class
object-oriented -	this term has various definitions, usually including the notions of derived classes and virtual functions. See data abstraction.
old-style cast	a cast written in C style, with the type in parentheses before the value being casted.
OOA / OOD	acronym for object-oriented analysis and object-oriented design, processes of analyzing and designing object-oriented software.
OOP	acronym for object-oriented programming
operator	a built-in operation of the C++ language, like addition, or an overloaded operator corresponding to a member function of a class. See function and operator overloading.
operator overloading	to treat a C++ operator like << as a function and overload it for particular parameter types
overload	to specify more than one function of the same name, but with varying numbers and types of parameters. See argument matching.
overload resolution -	see argument matching

parameter	refers to the variables passed into a function. See also argument
parameterized type	see template
parser	see parsing
parsing	the process by which a program written in some programming language is broken down into its syntactic elements
pointer	an address of an object
pointer to data member	a pointer that points at a data member of a class
pointer to function	an address of a function or a member function
pointer to member	see pointer to data member, pointer to function
programming environment - a set of integrated tools used in developing software, including a compiler, linker, debugger, and browser.	the ability to call a variety of member functions for a given class object using an identical interface in each case. See virtual function.
polymorphism	the ability to call a variety of member functions for a given class object using an identical interface in each case. See virtual function.
postfix	refers to operators that appear after their operand. See prefix.
prefix	refers to operators that appear before their operand. See postfix.
preprocessing	a stage of compilation processing that occurs before the compiler proper is invoked. Preprocessing handles macro expansion among other things. In C++ use of const and inline functions makes preprocessing less important.
preprocessor	see preprocessing
private	a C++ keyword used to specify that a class member can only be accessed from member functions and friends of the class. See access control, protected, and public.
programming environment	a set of integrated tools used in developing software, including a compiler, linker, debugger, and browser
promotion	see integral promotion
protected	a C++ keyword used to specify that a class member can only be accessed by member

	functions and friends of its own class and by member functions and friends of classes derived from this class. See private, public, and access control.
public	a C++ keyword used to specify that class members are accessible from any (non-member) function. See access control, protected, and private
pure virtual function	a virtual function with a "= 0" initializer. See abstract class
qualification	to prefix a name with the name of a class or namespace
reference	another name for an object. Access to an object via a reference is like manipulating the object itself. References are typically implemented as pointers in the underlying generated code.
repository	a location where an instantiated template class can be stored. See template instantiation.
resolution	see overload resolution
resumption	a style of exception handling where program execution continues from the point where an exception is thrown. C++ uses the termination style.
return	C++ keyword used for returning values from a function.
return value	the value returned from a function
run-time	refers to actions that occur during program execution
rvalue	a value that may appear on the right-hand side of an assignment
scope	the region of a program where a name has visibility
semantic analysis	a stage that a compiler goes through after parsing. In this stage the meaning of the program is analyzed.
semantics	the meaning of a program, as opposed to its syntax
separate compilation	refers to the process by which each translation unit of a program is compiled separately to produce an object file. The object files are then combined by a linker.

short	a C++ fundamental type used to declare small integers
signed	C++ keyword used to indicate a signed data type
sizeof	C++ keyword for taking the size of an object or type
stack frame	refers to a region of storage on the hardware stack, used to store information such as local variables for each invocation of a function
stack unwinding	see exception handling. When an exception is thrown, each active stack frame must be removed from the stack until an exception handler is found. This process involves calling a destructor as appropriate for each local object in the stack frame, and so on.
standard conversion	refers to standardized conversions between types, such as integral conversion
standard library	see library. The C++ standard library includes much of the C standard library along with new features such as strings and container class support.
statement	the parts of a program that actually do the work
static	see static member, static object, and static storage
static member	a class member that is part of a class for purposes of access control but does not operate on particular object instances of the class.
static object	an object that is local to a function or to a translation unit and whose lifetime is the life of the program
static storage	storage that persists throughout the life of the program. See static object and dynamic storage.
static type checking	refers to type checking that occurs during compilation of a program rather than at run-time.
static_cast	a C++ keyword specifying a style of cast meant to replace old-style C casts
storage class	see auto and static.
stream	an object used to represent an input or output channel. See stream I/O.
stream I/O	a C++ I/O library using overloaded operators<< and >>. It has more type safety than C-style I/O.

string	see C-string
struct	a kind of C++ class in which all the class members are by default public
switch	C++ keyword denoting a statement type, used to dispatch to one of several sequences of statements based on the value of an expression
symbol table	a compiler structure used to record type information about program names. The symbol table is used to generate compiler output.
syntax	the rules that govern how C++ expressions, statements, declarations, and programs are constructed. See grammar and semantics.
systems programming	refers to low-level programming, for example writing I/O drivers or operating systems. C and C++ are suitable languages for this type of programming.
tag	a name given to a class, struct, or union
template	a parameterized type. A template can accept type parameters that are used to customize the resulting type
template argument	an actual value or type given to a template to form a template class. See argument.
template class	a combination of a template with a template argument list via the process of template instantiation.
template declaration	a declaration of a template with its associated template parameter list
template definition	an actual definition of a template or one of its members
template instantiation	the process of combining template arguments with a template to form a template class
template parameter	a value or type declared to be passed in to a template. See parameter
temporary	an unnamed object used during the evaluation of an expression to store intermediate values
termination	a style of exception handling where control does not return to the point where a **true** C++ keyword used to specify a value for the bool type exception is thrown. C++ uses this style of exception handling.

this	C++ keyword used in a member function to point at the object currently being operated on
throw	C++ keyword used to throw (initiate) an exception. See exception handling.
translation unit	a source file presented to a compiler with an object file produced as a result.
true	C++ keyword used to specify a value for the bool type
try	C++ keyword used to delimit a try block.
try block	a statement that sets up a context for exception handling. A subsequent throw from a function called from within the try block will be caught by the exception handler associated with the try block or by a handler further out in the chain of handlers.
type	a property of a name that determines how it can be used. For example, an object of a class type cannot be assigned to an integer variable.
type checking	see type system
type conversion	converting a value from one type to another, for example via a constructor
type safety	see type system.
type system	a system of types and operations on objects of those types. Type checking is done to ensure that the operations for given types are appropriate, for example that a function is called with arguments of the appropriate types.
type-safe linkage	refers to the process of encoding parameter type information in external names so that the linker will reject mismatches between the use and definition of functions
typedef	a C++ keyword used to declare an alias for a type.
unsigned	a C++ keyword used to declare an integral unsigned fundamental type
user-defined type	a class or typedef
using declaration	a declaration making a class or namespace name available in another scope.
using directive	a way of making available to a program the members of a namespace.
using namespace	see using directive

variable	an object that can be assigned to
vector	a one-dimensional array
visibility	refers to the processing of doing name lookup without regard to whether a name is accessible. Once a name is found, then type checking and access control are applied
void	a C++ keyword used to declare no type. It has special uses in C++, for example to declare that a function has no parameter list
while	C++ keyword used to declare an iteration statement