**Identifiers.**

A valid identifier is a sequence of one or more letters, digits, or underscore characters (\_). Spaces, punctuation marks, and symbols cannot be part of an identifier. In addition, identifiers shall always begin with a letter. They can also begin with an underline character (\_), but such identifiers are -on most cases- considered reserved for compiler-specific keywords or external identifiers, as well as identifiers containing two successive underscore characters anywhere. In no case can they begin with a digit.

C++ uses a number of keywords to identify operations and data descriptions; therefore, identifiers created by a programmer cannot match these keywords. The standard reserved keywords that cannot be used for programmer created identifiers are:

*alignas, alignof, and, and\_eq, asm, auto, bitand, bitor, bool, break, case, catch, char, char16\_t, char32\_t, class, compl, const, constexpr, const\_cast, continue, decltype, default, delete, do, double, dynamic\_cast, else, enum, explicit, export, extern, false, float, for, friend, goto, if, inline, int, long, mutable, namespace, new, noexcept, not, not\_eq, nullptr, operator, or, or\_eq, private, protected, public, register, reinterpret\_cast, return, short, signed, sizeof, static, static\_assert, static\_cast, struct, switch, template, this, thread\_local, throw, true, try, typedef, typeid, typename, union, unsigned, using, virtual, void, volatile, wchar\_t, while, xor, xor\_eq*

Specific compilers may also have additional specific reserved keywords.

**Very important:** The C++ language is a "case sensitive" language. That means that an identifier written in capital letters is not equivalent to another one with the same name but written in small letters. Thus, for example, the RESULT variable is not the same as the result variable or the Result variable. These are three different identifiers identifying three different variables.