If we don't know what a functions' argument or result types will be until we use it, we call it a ***generic function***. The ability to use and create generic functions is a key feature of the C++ language. The language feature that implements generic functions is called ***template functions***. They let us write a single definition for a family of functions - or types - that behave similarly, except for differences that we can attribute the types of their ***template parameters***.

Type parameters operate much like function parameters. They define names that can be used within the scope of the function. However, type parameters refer to types, not to variables. For `template <class T>`, when T appears in the function, the implementation will assume that T is a type.

The library defines five ***iterator categories***, each one of which corresponds to a specific collection of iterator operations. These categories classify the kind of iterator that each of the library containers provide.