Chapter 1

Summary

Elements of a C++ program

Basic elements

- `#include` directives
  - declares functions, objects and classes that are being used
  - ex. `iostrem` allows use of `cout`, provides further support of console input / output
- `using` statement
  - enables all names within a given namespace to be referenced directly
  - not strictly necessary buy a major convenience
  - without the `using` statement we would have to refer to `cout` with `std::cout`, the qualified name
- `main` function
  - the part of the program that actually does something

General Structure of a C++ program

1. Declarations, `include`.
2. `using` statement.
3. Type declarations, including classes.
4. Global variable decorations.
5. Function prototypes.
6. Function definitions including `main`.

Namespaces

Namespaces can be used more explicitly by referring to objects in the namespace directly. For example instead of `using namespace std;`, `using std::cout;` can be used to refer to the `cout` object.

The number of libraries available in C / C++ is quite large, which meant when many libraries are in use within a program, there is the potential for name conflicts.

The `using` statement grants access to all the symbols in a library only within the scope in which it is defined.

Namespaces can be defined with the `namespace` keyword