Chapter 2: Hello, World!

2.1 Programs

C++ is a programming language designed for a wide variety of programming tasks.

2.2 The classic first program

String literals are delimited by double quotes ("). The name `cout` refers to the standard (character) output stream. Whatever characters are "put into `cout`" will be output to the screen.

Anything written after the token `//` on a line is a comment. Comments are ignored by the compiler.

Comments are written to describe what the program is intended to do and in general to provide information useful for humans that can't be directly expressed in code. The person most likely to benefit from the comments in your code is you (when you come back to your code next week, next month or next year)!

A program is written for two audiences: (1) we write code for computers to execute, (2) programmers are another audience for programs. Writing code is also a form of human-to-human communication. Code is for reading!

The first line of the program is typically a comment that tells the (human) reader what the program is *supposed* to do. The code itself says what the program does, not what we meant it to do (the two should be, but are not necessarily, synonymous).

An `#include` directive instructs the computer to make available facilities from a file. A **header file** (or just **header**) contains definitions of terms that we use in our programs.

Every C++ program must have a function called `main` to tell it where to start executing. A function is a named sequence of instructions. A function (e.g. `int add1(int i) { return i + 1; }`) has four parts:

- a *return type* which specifies what kind of result, if any, the function will return to whoever asked for it to be executed
- a *name*
- a *parameter list* enclosed in parenthesis
- a *function body* enclosed in a set of "curly braces", `{ }`, which lists the actions that the function is to perform

A zero (`0`) returned by `main()` indicates that the program terminated successfully.

A part of a C++ program that specifies an action (and isn't an `#include` or other preprocessor directive) is called a **statement**.

2.3 Compilation

A **compiler** translates human-readable **source code** (or **program text**) into an **executable** (also called **object code** or **machine code**) that the machine can "understand" and run. The compiler looks at your source code to see if your program is grammatically correct, if every word has a defined meaning, and if there is anything else obviously wrong that can be detected without actually executing the program.

There is no really short, fully correct and non-technical way of summarizing where semicolons are needed.

The compiler is your friend; possibly, the compiler is the best friend you have when you program.

2.4 Linking

A program usually consists of several parts. These separate parts (sometimes called *translation units*) must be compiled and the resulting code files must be linked together to form an executable program. The program that combines (*links*) such parts together is called a **linker**. Object code and executables are not portable between operating systems (e.g. code compiled for a Windows machine will not run on a Linux machine.)

A **library** is simply some source code that we access using declarations found in an `#included` file. A **declaration** is a program statement specifying how a piece of code can be used.

Errors found by the compiler are called **compile-time errors**, errors found by the linker are called **link-time errors**, and errors not found until the program is executed are called **run-time errors** or **logic errors**. Generally, compile-time errors are easier to understand and fix than link-time errors and link-time errors are often easier to find and fix than run-time errors or logic errors.

2.5 Programming environments

An **IDE** (for *interactive* [or *integrated*] *development environment*) combines into one program all the most useful tools for programming (editor, compiler, linker, etc.).

Terms

`//`
`<<`
C++
comment
compiler
compile-time error
`cout`
executable
function
header
IDE
`#include`
library
linker
`main()`
object code
output
program
source code
statement