C-Style Strings

C++ has two different "string" types; the `string` class from the standard library makes string manipulation easy, but is complex, since it uses dynamic memory. The original "string" type, inherited from the C language, is much simpler.

Though simpler, older C-strings are more **difficult to work with**. Sometimes more efficient, they are also **more error prone**, even **somewhat dangerous**. However, as C++ programmers, you can't ignore them.

Why should you dedicate **any** time to studying C-strings? There are several reasons:

- **Efficiency**. Library string objects use dynamic memory and the heap. C-strings are **built into the language**, so you don't need to link library code.
- **Legacy Code**. To **interoperate** with pure C code or older C++ code that predates the C++ `string` type.
- **Library Implementation**. You may want to **implement** your own `string` type. Knowing how to manipulate C-strings can greatly simplify this task.
- **Embedded Programming**. Programs written for embedded devices like those in your automobile or toaster, frequently use C-strings.
- **Platform O/S Programming**. For native Linux or Windows programming, you will need to use C-strings.

We will encounter many of these cases in the remainder of this course.