The strlen Function

In this lesson we're going to look at several implementations of the standard library functions beginning with **`strlen()`**. We'll finish by learning to write your own C-String functions.

To find the length of a string, you **count characters** until you reach the **`'\0'`**. Here is an implementation that uses array notation.

```
size_t strlen(const char str[])
{
    size_t len = 0;
    while (str[len] != '\0') len++;
    return len;
}
```

> *Note that the return type **must be** **`size_t`** (not **`int`**), because we can't have a negative length on a string. The array must be **`const`**, otherwise it **would be illegal** to call the function using a C-string literal.*

Another alternative is to advance the pointer until it reaches the end of the string, and then to **use pointer subtraction** (or **pointer difference**) to determine the number of characters. Here's a version that does that:

```
size_t strlen(const char *str)
{
    const char *cp = str;
    while (*cp != '\0') cp++;
    return cp - str;
}
```

We can actually write this in an even more cryptic style. I **don't encourage you to write code like this**, since it is quite a bit more error prone, but it **is a common C idiom** so you should recognize it when you see it.

```
size_t strlen(const char *str)
{
    const char *cp = str;
    while (*cp++) /* do nothing */;
    return cp - str - 1; // cp points to 1 past the NUL
}
```