Searching a String

To search for both characters and substrings, the `string` class contains a member function **`find()`**, which comes in several forms. The simplest form looks like this:

```
auto index = str.find(target)
```

The argument **`target`** is what you're looking for.

- **`target`** may be a **`string`**, a **`char`** or a C-string **literal**.
- The function searches through **`str`** looking for the **first occurrence** of **`target`**.
- If **`target`** **is** found, **`find()`** returns the index at which the match begins. Use **`auto`** or **`size_t`** to store this.

If you want to find the **last occurrence** of target, use **`rfind()`** instead.

Not Found

If **`target`** **is not found**, then **`find()`** returns the constant named **`string::npos`**. This constant is defined as part of the string class and therefore requires the **`string::`** qualifier. This is a good candidate for a **named constant** in your code:

```
const auto kNotFound = string::npos;
```

The **`find()`** member function takes an optional second argument to indicate the index at which to start the search. Both styles of the **`find()`** member function are illustrated here:

```
string str{"hello, world"};
auto a = str.find('o');         // char, 4
auto b = str.rfind("o");        // C-string, 8
auto c = str.find('l', 4);      // 10
auto d = str.find("waldo");     // string::npos
```

The **`find()`** member functions consider uppercase and lowercase characters to be different. Unlike Java, there is no built-in **`toUpperCase()`** or **`toLowerCase()`** member function in the **`string`** class.

Variations

In addition to **`find()`** and **`rfind()`**, you can find the position of the first (or last) occurrence of a character that **appears in a set** or that **doesn't** appear in a set. Here are some examples:

```
string s{"\"Hooray\", the crowd cheered!"};
auto a = s.find_first_of("aeiou");      // first lower-case vowel
auto b = s.find_last_of("\",.!:;");     // last punctuation
auto c = s.find_first_not_of(" \t\n");  // first non-whitespace
```