C++ strings vs C-Strings

C-strings are not first-class types like the C++ `string` type. They do not work like the built-in types. Look at this example, which tries to **assign**, **compare** and **concatenate** two strings:

```
string str1 = "Hello", str2 = "World";
char cstr1[] = "Hello", cstr2[] = "World";

str1 = "Goodbye";            // assignment OK
cstr1 = "Goodbye";           // ILLEGAL
if (str1 < str2) ...         // comparison OK
if (cstr1 < cstr2) ...       // INCORRECT
str1 += ", ";                 // OK
cstr1 += ", ";                // ILLEGAL
```

For the C++ `string` class, assignment, comparison and concatenation work in the same manner as the built-in types. Use the **assignment operator**, the **relational operators**, and the `+=`. **Not so** for C-strings, where you must use functions from the `<cstring>` header to perform the same functionality.

- `strcpy(dest, src)` is used instead of assignment
- `strcat(dest, src)` is used instead of `+=`
- `strcmp(cstr1, cstr2)` is used instead of the relational operators

In addition, in place of the member function `size()`, you use the `strlen(cstr)` function which counts the number of characters before the `'\0'`.