Strings

It's time to learn how to declare strings and how C handles them.

We'll cover the following

I'll be honest, C is slightly awkward for dealing with character strings, especially compared to languages like Python.

String is one of the most popular data types in programming.

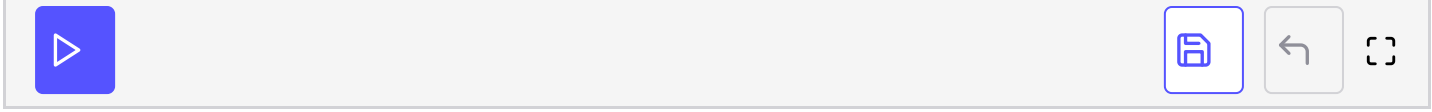
> A collection of characters grouped together forms a **string**.

Constant character strings are written inside double-quotation marks (line 5 below).

Strings are actually, under the hood, arrays of characters. Single character variables are declared using single-quotation marks (line 7 below).

String variables can be declared as indicated on line 8 below. The double square brackets signify an array. Just like arrays, we can access string elements through indexing.

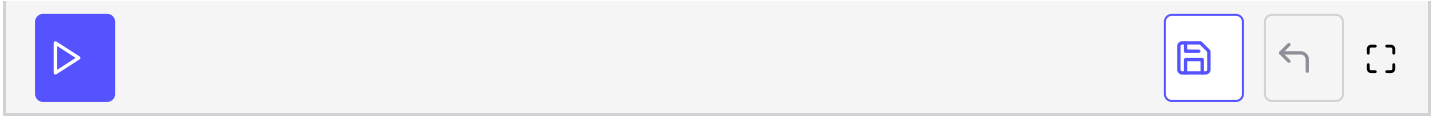
```
#include <stdio.h>

int main(void) {

  printf("Hello world\n");

  char c = 'p';
  char s[] = "paul";

  printf("c=%c and s=%s\n", c, s);

  return 0;
}
```



null-termination

An important thing to remember about strings is that they are always **null-terminated**. This means that the last element of the character array is a “null” character, abbreviated `\0`. When you declare a string as in line 8 above, you don’t have to put the null termination character in yourself, the compiler does it for you. We can use the `sizeof()` function to inspect our character string above to see how long it actually is:

```
#include <stdio.h>

int main(void) {

  char s[] = "paul";

  printf("s is %ld elements long\n", sizeof(s));

  return 0;
}
```



Modifying Strings

In C, it is a bit tricky to modify a declared string.

Importantly, once a string is declared to be a given length, you cannot just make it longer or shorter by reassigning a new constant to the variable.

Well, you can sort of make it shorter, by writing a new shorter string in the old string array, and terminating the new (shorter) string with a null character. Then essentially you will have a short string sitting in a long array, but that’s ok, since we know where the end is (the null termination character).

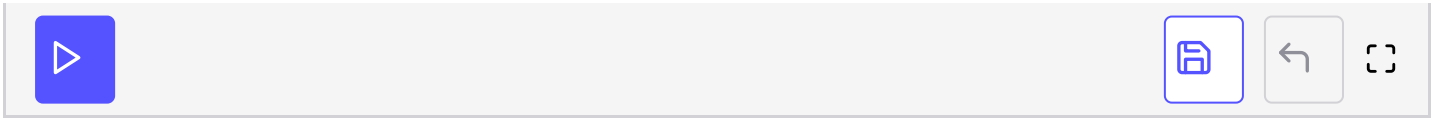
We can also edit the contents of a string, since it is simply an array of characters.

```
char s[] = "String";
s[3] = 'o';
```

`s` would now be `Strong` instead of `String`.

```
#include <stdio.h>
```

```c
int main(void) {

  char s[] = "This is a string";
  printf("Original String: %s\n", s);

  //Change the contents of the string
  s[2] = 'a';
  s[3] = 't';
  printf("Modified String: %s\n", s);

  //Shorten the string
  s[7] = '\0';
  printf("Shortened String: %s\n", s);

  return 0;
}
```



So, to summarize, in C, strings are simply **null-terminated arrays of characters**.

Next, we'll look at the various functions we can perform on strings.