

# Section 13: Strings

**Introduction:** The `String` data type is used to store sequences of characters, which we commonly think of as “text.” Strings share some similarities to arrays of `char`s but with special properties and restrictions.

---

**String Literals:** Arbitrary strings created by placing text between double-quotes. We’ve already been using these in the course! Notice that for a known string that you are only going to use once (e.g. as an argument to `println()` or `text()`), you can use a string literal without creating a variable!

```
Examples: String s = "stored text!"; // string literal stored in variable s
          text(s,10,50); // draws string on canvas
          text("stored text!",10,50); // same result as the statement above
```

---

**Using Strings:** You are provided with a number of useful ways to use and manipulate strings:

- `s.length()` – Returns the length of the string (i.e. the number of characters).
- `s.charAt(i)` – Returns the `char` at the index `i`, where the indices start from the left with index 0.
- `s1 + s2` – The plus (+) operator concatenates the strings `s1` and `s2` together. That is, it returns a new `String` that is the combination of `s1` and `s2`, with the characters in `s1` coming *before* (to the left of) the characters in `s2`.
- `s1.equals(s2)` – Returns `true` if the *contents* of the two strings match (i.e. they are the same length and have matching characters in every index) and `false` otherwise. `s2.equals(s1)` is equivalent.

```
Examples: String s1 = "ate", s2 = "con", s3 = "nation"; // create 3 strings
          int i = s3.length(); // stores 6
          char c = s2.charAt(0); // stores 'c'
          boolean b1 = s1.equals(s2); // stores false
          boolean b2 = s2.equals("con"); // stores true
          println( s2 + s2.charAt(0) + s1 + s3 ); // prints "concatenation"
```

---

**String and Array Comparison:** These are conceptually similar so it is easy to confuse them. We hope this comparison will help you recognize the differences in syntaxes between the two.

	String	(Character) Array
Declaration:	<code>String s;</code>	<code>char[] ar;</code>
Assignment:	<code>s = "call";</code>	<code>ar = {'c','a','l','l'};</code>
Get length (4):	<code>s.length()</code>	<code>ar.length</code>
Get character ('c'):	<code>s.charAt(0)</code>	<code>ar[0]</code>
Change character:	Not allowed	<code>ar[0] = 'w';</code>
Concatenation:	<code>s = s + " me, maybe?";</code>	No easy way to do this

## Exercises:

- 1) What do the following lines of code print to the console?

```
String word = "ice";
println( word.length() + " bl" + word.charAt(0) + "nd m" + word );
```

- 2) Fill in the blanks in the Processing code for the function `frequency()`, which returns the number of times that a particular `char c` appears in a `String s`. For example, `frequency("missus", 's')` returns 3.

```
int frequency(String s, char c) {
    int count = 0;
    int i = ____;
    while ( i < _____; ) {
        if ( _____ ) {
            count = count + 1;
        }
        i = ____;
    }
    return count;
}
```

- 3) Write Processing code below to create the string "1, 2, 3, 4, 5, 6, 7, 8, 9" using a while-loop and store it in the variable `result`. Pay special attention to the spaces and commas!

```
String result = "";
```

- 4) After the following code is executed, what string is stored in `msg`?

```
char[] alphabet = {'a','b','c',..., 'x','y','z'}; // assume all 26 written out
int[] nums = {7,0,15,15,24};
String msg = "";
int i = 0;
while (i < nums.length) {
    msg = msg + alphabet[ nums[i] ];
    i = i + 1
}
```

- 5) Go to the course website and start working on the lab titled "Word Guessing." [*partners*]