

Strings

$\text{std::string} \approx \text{std::vector} \langle \text{char} \rangle$
(+ some fancy functions).

vector	string
$v.size()$	$s.length()$
$v[i]$	$s[i]$

Bonus features of String:

- concatenation: $S = S1 + S2$

(if $S1 = \text{"abc"}$, $S2 = \text{"def"}$, $S1 + S2 = \text{"abcdef"}$)

- searching for substrings:

$s1.find(s2)$ (like `strtlf`, or
/ `strstr`)

E.g., if $S1 = \text{"hello world"}$

$S2 = \text{"wo"}$

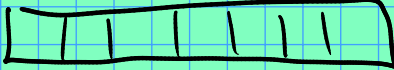
$S1.find(S2)$ would return 6

Exercise: let's write `find`.

We'll use this prototype:

```
size_t find(const string& s1, const string& s2);
```

```
// say  $n1 = s1.length()$ ,  $n2 = s2.length()$ 
```

S1 = 

S2 = 

Idea: look for match at all possible starting indexes in S1.

Possible indexes: $[0, 1, \dots, n1 - n2]$

make sure $n1, n2$ signal...



out line:

```
for (int i = 0; i <= n1 - n2; i++) {  
    // check for match starting @ i.  
    // Detail: compare S1[i] vs S2[0]  
                    i+1 vs 1  
                    :  
                    :
```