String manipulation:

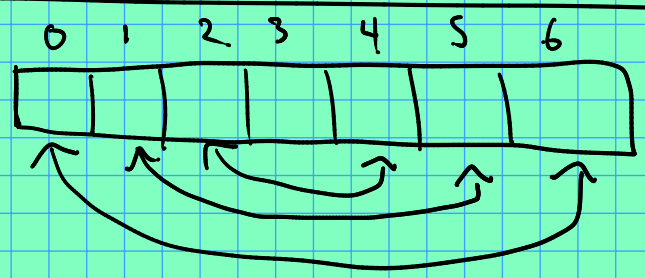
Note: very similar to the vector...

Vector	string
v.size()	s.length()
v.push_back(x)	s += c
v[i]	s[i]

Exercise: write a function that reverses a string. E.g.

```
string s = "abc";
reverse(s);
//Now s has "cba".
```

```
void reverse(string& s);
```



0 → 6
1 → 5
2 → 4
i → 6 − i
(generally, len − i − 1)

in words: repeatedly swap s[i] w/ s[length − i − 1]...

Boundary? for (i = 0; i < ?; i++)
? = len/2.

```
void reverse(string& s) {
  for (int i = 0; i < s.length()/2; i++) {
```

```
        char temp = S[i];
        S[i] = S[s.length() - i - 1]
        S[s.length() - i - 1] = temp;
    }
}
```

Exercise: "String matching"

Given 2 strings S1, S2, determine if S1 is a substring of S2.

(E.g. S1 = "bcd", S2 = "abcdef")

(Note: String class provides this.)

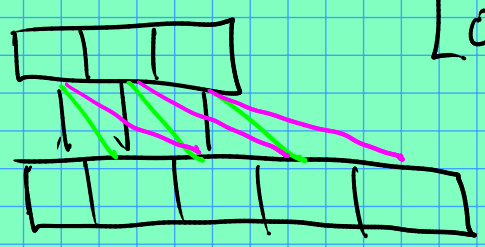
High-level: S2 = abcebcc bcf bcde

bcd

idea: test all possible "offsets" for a match.

What are the possible offsets?

[0 ... S2.len − S1.len]



```
int search(S1, S2)
    for (int i = 0; i <= S2.length - S1.length; i++) {
        // check for match starting @ i.
        bool found = true;
        for (int j = 0; j < S1.length; j++) {
            found &= (S1[j] == S2[j+i]);
```

```
        if (found) return i;
    }
    return -1;
```

TODO: re-write above inner loop to stop as soon as possible (stop at first non-match of s1[j] w/ s2[i+j])