Strings (more practice w/ vectors)

String is really just vector<char> with a slightly different interface.

vectors	strings
v.size()	s.length()
v.push_back(x)	s += x ← could be char or a string
v[i]	s[i]

warm up: for a character c, & a string s, count the # of times c occurs in s. E.g., s = "hello", c = 'l'

⇒ answer = 2.

```
size_t countChars(const string& s, char c)
{
    size_t count = 0;
    for (size_t i = 0; i < s.length(); i++) {
        if (s[i] == c) count++;
    }
    return count;
}
```

low == s

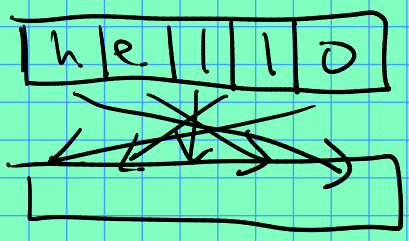
s[0] s[1] ... s[4]

s[0 ... s.length()-1]

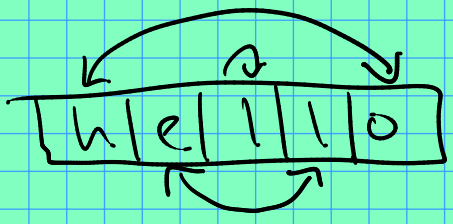
Exercise 2: reverse a string.

write a function to

void reverse(string& s);



requires a lot of extra space!



say length = n = 5

$0 \longleftrightarrow 4 = n-1$

$1 \longleftrightarrow 3 = n-2$

$2 \longleftrightarrow 2 = n-3$

in general: $i \longleftrightarrow n-i-1$

```
void reverse(string& s) {
    for (i=0; i< s.length()/2; i++) {
        // swap s[i] <--> s[s.length()-i-1]
        char temp = s[i];
        s[i] = s[s.length()-i-1];
        s[s.length()-i-1] = temp;
    }
}
```

s1 = "abc"

s2 = "lololabclol"

one idea: check all possible offsets in s2.

say length s1 = n1

length s2 = n2

possible offsets: $[0, 1, \ldots n_2 - n_1]$