Exercise from last time: "subset sum":

for a collection of integers V, and target t,

$\exists\ S \subseteq V$ s.t. $\sum_{x \in S} x = t$?

What if $|S| = 2$?

E.g. if $V = \{-3, 7, 19, 4, 5\}$

and $t = 16$, then the answer is "yes":

$-3 + 19 = t$.

However if $t = 100$, the answer would be "no".

Hint: just use "brute force"...

(Semi-) new topic: std::string.

Behind the scenes, string ≈ vector<char>.

However strings have a slightly different interface (e.g. s.length() gives the size, rather than V.size(), as you would for a vector.)

Also provides useful functions like "find":

string s = "abcdef";

s.find("def") would give 3, as the

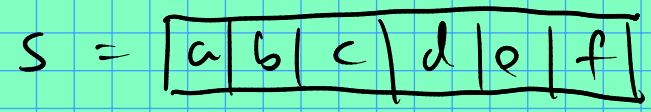
First match starts @ index 3:

```
abcdef
012345
```

s.find("deff") would give -1 to indicate "deff" was not found.

How to write our own find function?

```
size_t find(const string& s, const string& t);
// return index of match if t is a substring
//of s;  -1 other wise.
```



Strategy: "brute force" — look for t at all possible offsets in s.

Possible offsets: 0, 1, ... s.length()-t.length()

Adding some detail:

```
for (i=0; i<= s.length()-t.length(); i++) {
    //check if t matches @ offset i
    // i.e., t[0] ==? s[i], t[1] ==? s[i+1]...
```