Sets + Maps

Sets: models mathematical sets, e.g.

$$S = \{1, 7, 13\}.$$

Desired features:

- Efficient membership test (is $x \in S$?)
- Efficient insertion / removal

How to represent in C/C++?

Can we use vectors?

If yes, how?

With no ordering, insert is efficient, but removal + search are not.

What if we sort the vector?

Then search is efficient, but insert + remove are not...

However, what if the elements come from a <u>small universe</u> (e.g. if they are of type char).

Remember: sizeof(char) = 1 (byte).

So # different char values is $2^8 = 256$.

In this case, vectors actually can be used to <u>efficiently</u> represent a set.

Hint: recall the "characteristic function" of a set $S \subseteq U$:

$$\chi_S : U \longrightarrow \{0,1\}$$

$$\chi_S(x) = \begin{cases} 0 & \text{if } x \notin S \\ 1 & \text{if } x \in S \end{cases}$$

$(S_0,\ S = \chi_S^{-1}(\{1\}).)$

If $|U|$ is "small", and furthermore, consists of consecutive integers, we can use a vector to store χ_S:

```
vector<bool> S;  // S[x] == true
                 //      <=> x ∈ S.
for (i=0; i<256; i++)
    S.push_back(false);  // set S = {}.

// Add 'a' to S:
S['a'] = true;

// Remove 'b', if present:
S['b'] = false;

:
```

Very efficient to add/remove and search!

What if $|U|$ not small, or values are not consecutive integers?

We can #include <set> ...

See exercises for more...

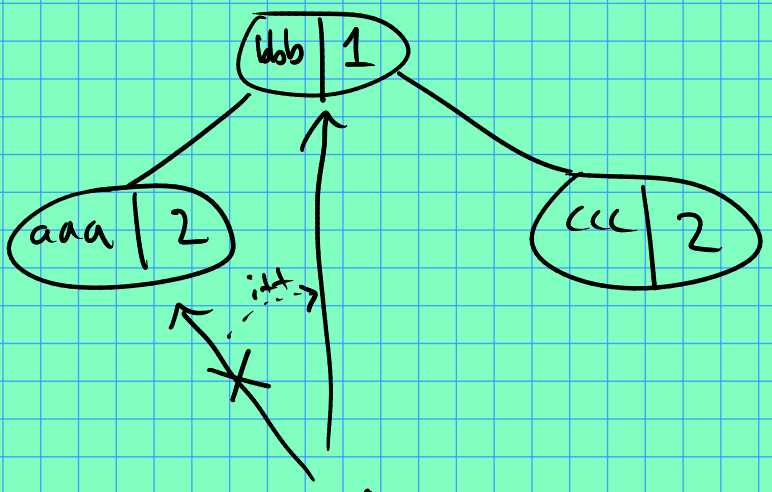
Maps: similar to a set, but you can associate a value to each element.

Example: make a frequency table of strings read from stdin.

```
echo aaa bbb aaa ccc ccc | ./freq
```

```
aaa: 2
bbb: 1
ccc: 2
```

Behind the scenes:




```
i = F.begin()
(*i).first  = key
(*i).second = value
```