Maps: motivational example: frequency table.

Read strings from stdin, and print each one with the # of times it appeared (the frequency).

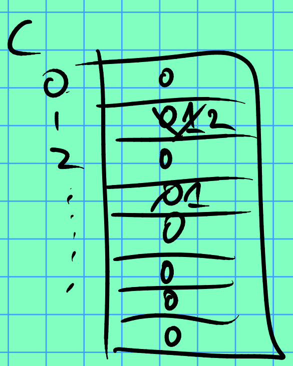
```
echo a a a b b c | ./freq
```

```
a  3
b  2
c  1
```

Warm up: what if we read small integers ($x \in \{0, \dots, 9\}$) instead of strings?

Use a vector<int> of size 10:

```
vector<int> c; // c[i] stores count for i
for (i=0; i<10; i++) c.push_back(0);
  // Set all 10 counts to 0.
  int n;
  while (cin >> n) {
      c[n]++;
  }
  // Now just print i, c[i]
```



This solution totally breaks for strings (or if the # of distinct integers is large...)

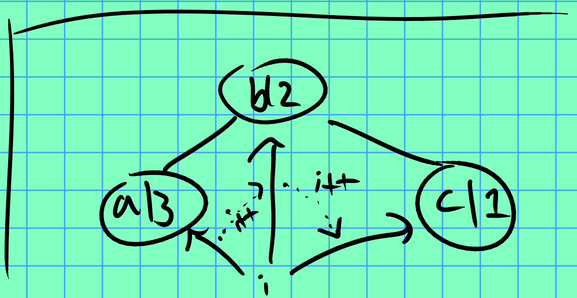
Back to the original. Use maps.

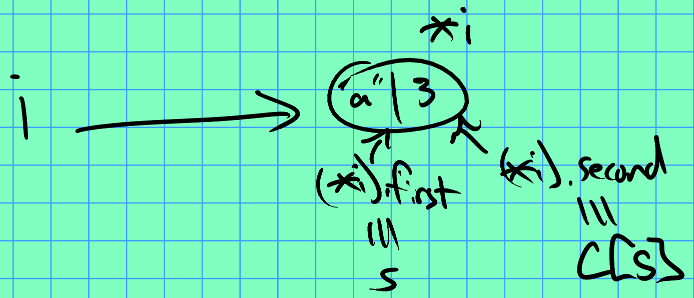
```
map<string,int> c; // c[s] stores count
```

// for string s.

```
string s;
while (cin>>s)
      C[s]++;

// Now just print!
for (map<string, int>::iterator i = C.begin(); i!=C.end();
                                                  i++) {
      cout << (*i).first << ":"
           << (*i).second << "\n";
}
```





*i
i
"a"|3
(*i).first
|||
s
(*i).second
|||
C[s]