Maps (Aka "dictionary" or an "associative array")

Idea: like an array, but you can use non-integral types as indexes. Even for integers, they need not be consecutive.

Example: if M is a <string, int> map, then stuff like

```
M["Easter"] = 27;
```

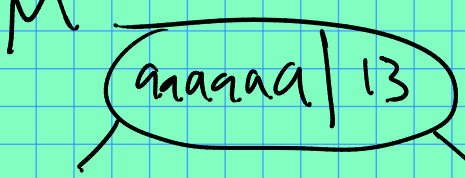
A <int, ___> map is like a normal array, kind of...

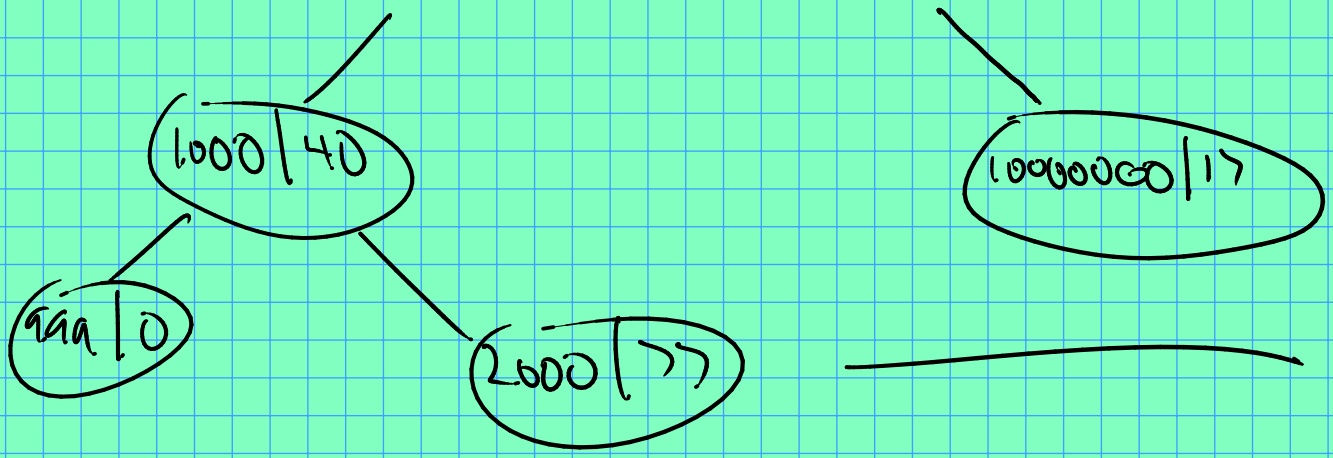
For a normal array, what happens if we do (or vector)

```
vector<int> V;
V[999999] = 13;   ?
// crash w/ seg fault.
```

For maps this is fine:

```
map<int, int> M;
cout << M[999999] << endl; // prints 0
M[999999] = 13; // works fine...
```





Accessing elements takes $\approx \log n$ steps.
(vector takes constant time.)

Example: histograms / freq. tables.

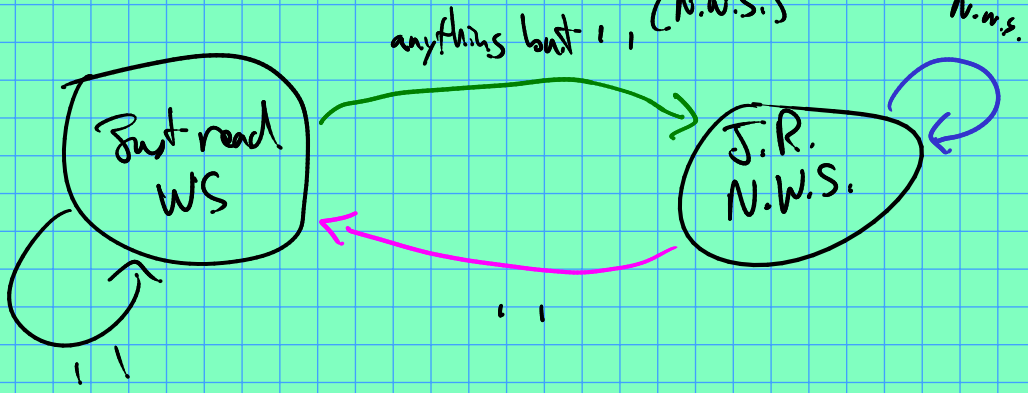
Read strings from stdin, print out a histogram of their frequencies.

Exercise: try to do this w/o maps.

With maps: very easy!

```
map<string, int> F;
while (cin >> s) F[s]++;
```

Back to word count.



hi hello how are you?

```
// wordcount++
// cur = c; state = N.W.S.
// cur += c
// add cur to set of words.
// state = W.S.
```