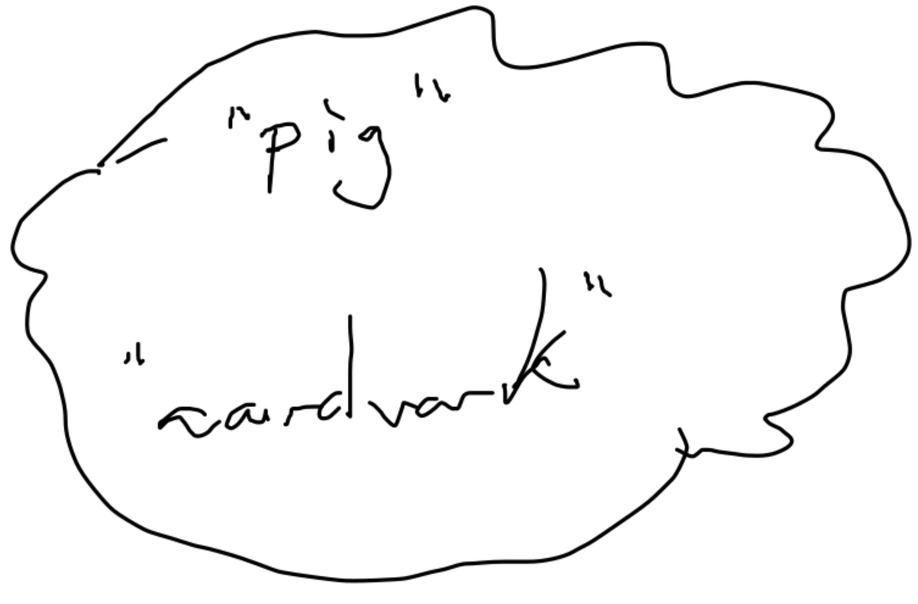
list:

set:

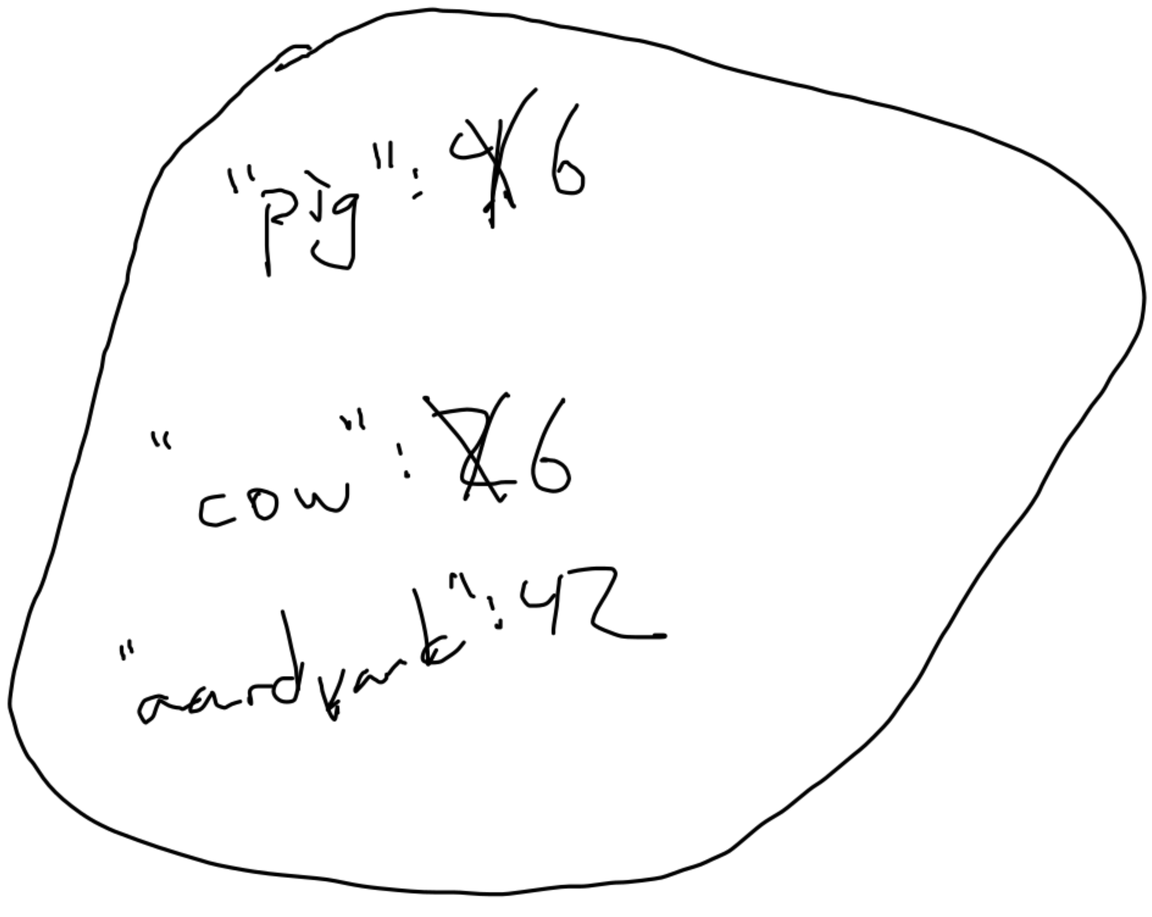


for add & remove,

true → change

false → no change

map:



Iterator

- a representation of moving through a collection
- methods:
  - hasNext
  - next

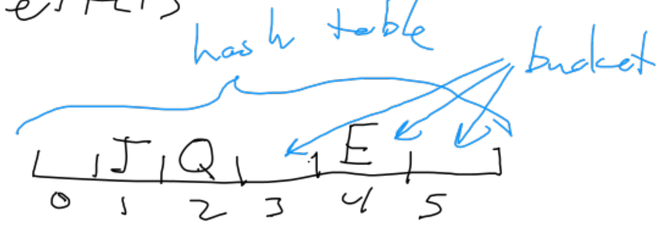
```
while(i.hasNext())
    thing = i.next()
```

Hash tables

- arrays, but better
- arrays are fast at random access
- arrays are slow to insert or remove
  - shifting is slow
- we want to avoid shifting

1. ~~always~~ gives an unused index
2. gives consistant answer

for a set of letters



adding

E → [?] → 4

Q → [?] → 2

J → [?] → 1

F → [?] → 4

queries

Q → [?] → 2

H → [?] → 5

collision

P → [?] → 2

hash function + compression function

table size = # of buckets

load factor = $\frac{\#\text{ of entries/values}}{\text{table size}} = \frac{3}{6} = 50\%$

Collision Resolution

- Find another bucket
  - Open addressing
    - linear probing
      - start at h → wrap around
      - increment until a blank spot
    - quadratic probing
      - try h
      - try $(h + 1^2)$ % size
      - try $(h + 2^2)$ % size

- Allow buckets to have multiple values
  - Closed addressing
    - chaining
      - each bucket is a linked list