 Warm Up

- Linked list is sorted
- Linked list has $-\infty$ and $\infty$
- Want largest key whose value is at most $k$.

```plaintext
find(k)
    t ← front // t.key ≤ k always
    while (t.next.key ≤ k)
        t = t.next
    return t
```
Improved running time

- Add a layer. Does it help?
- Which get added to higher layer?
- How about another layer?

New Find Function

old-find(k) // for reference
  t ← L.first
  while t.next.key ≤ k do
    t ← t.next
  return t.key

How does find change for layered list?

  t ← t[0]-left // invariant: t.key ≤ k
  while (t.next.key ≤ k)
    if t.down ≠ null then
      t ← t.down
    while (t.next.key ≤ k)
      t ← t.next

  return t
Inserting into a Skip List

Insert\((k, v)\)

// assume we have insertAbove\((p, q, k, v)\)
// and no new levels needed

\[ p = \text{find}(k) \]
\[ q = \text{iAA}(p, \text{nullptr}, k, v) \]
While \( \text{coinflip}(\cdot) \) is heads.
\[ p = p \rightarrow \text{up} \]
\[ q = \text{iAA}(p, q, k, v) \]

Running time for find

► Suppose skip list has height \( h \)
► How long does a find take?

► We added key. Probability stored \( i \) levels up?

\[ \left( \frac{1}{2} \right)^i \]
7. **Height of a skip list**

- Insert $n$ keys to initially empty list
- $P_i$ = prob level $i$ has at least one item?

$$P_i \leq \frac{n}{2^i}$$

- Probability height at least $3 \log n$?

$$P \leq \frac{n}{2^{3\log n}} = \frac{n}{(2^{\log n})^3} = \frac{1}{n^3}$$

8. **Time for find?**

```
find(k)
    t ← topmost left node of list
    while t.below ≠ nullptr do
        t ← t.below
        while t.after.key ≤ k do
            t ← t.after
        return t
```

- How many drops?
  - $\text{height}$
    - 50% each
- How many scan forward?
Size of a skip list

- $E[\# \text{ items}]$ at level $i$ is $n/2^i$