# Today’s announcements:

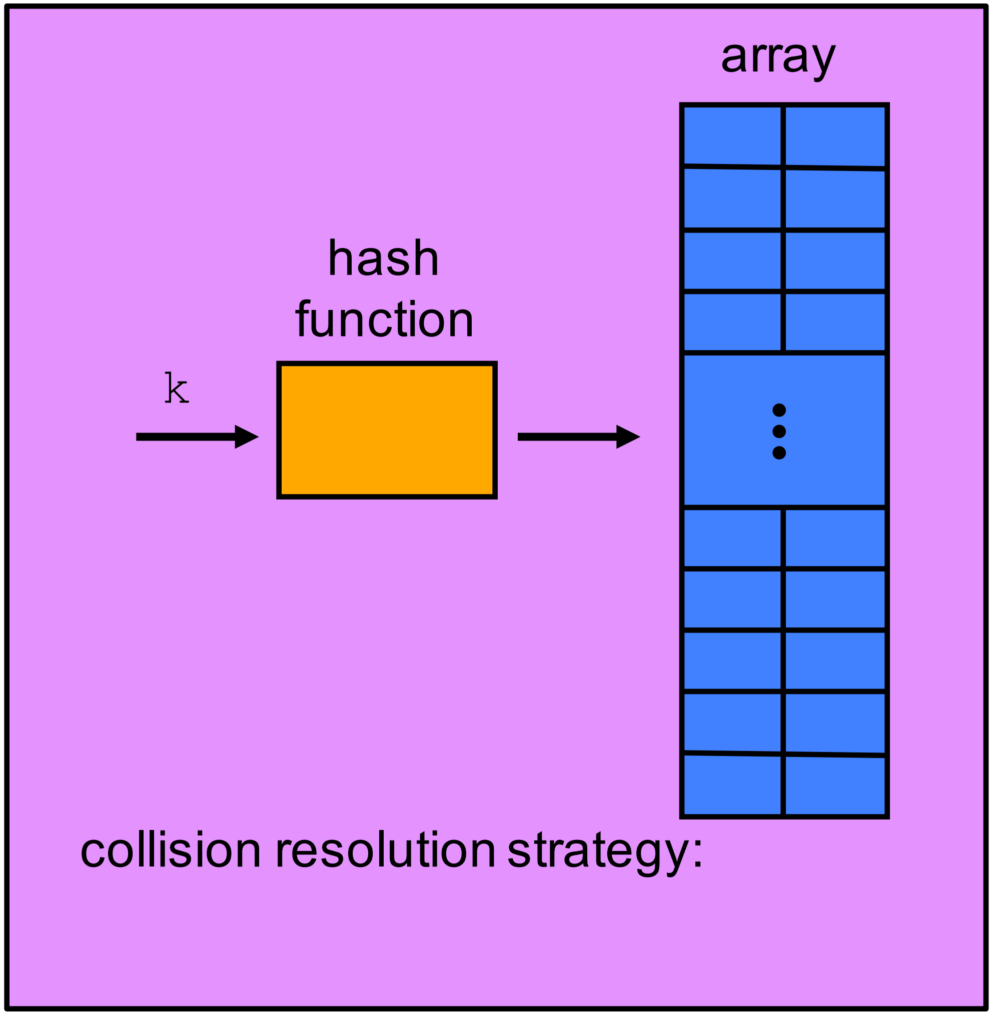
MP6 available, due 11/17, 11:59p.  EC due 11/10, 11:59p

client code

declares an object
    of ADT dictionary
    `dict<ktype, vtype> d;`

ex: insert is `d[k] = v;`

class `dict`



array

hash function

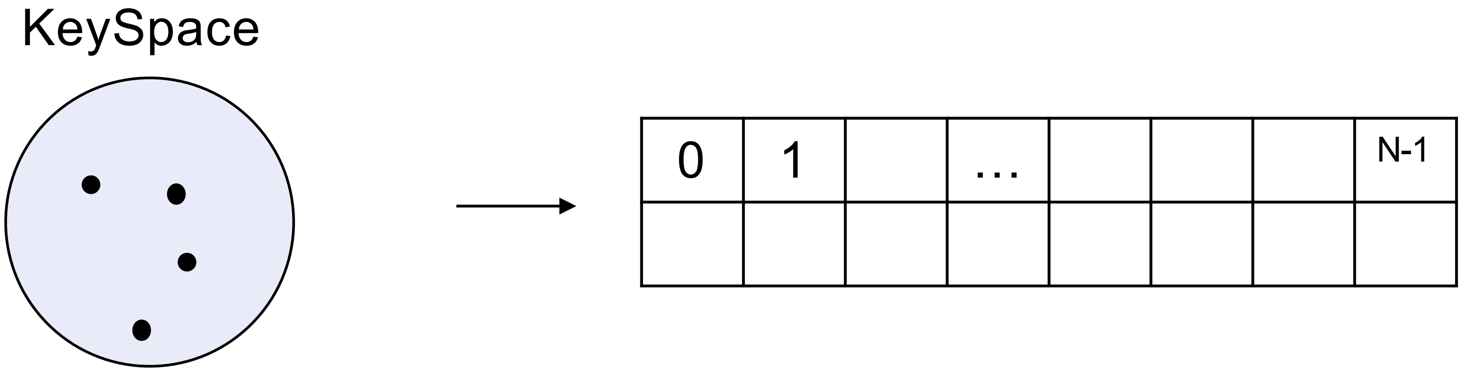
k

collision resolution strategy:

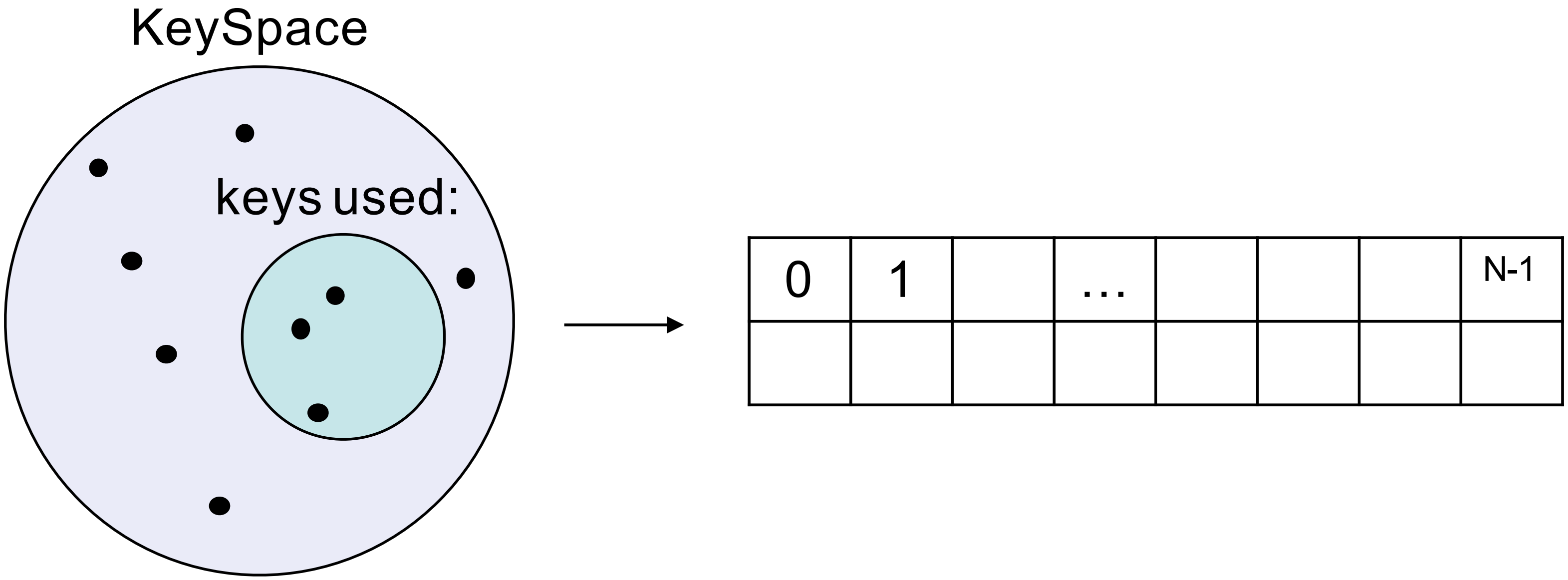
# Hash Functions:

- Consist of 2 parts:
  - A Hash: Function mapping a key to an integer `i`
  - A compression: function mapping `i` into the array cells 0 to N-1.
- Choosing a hash function is tricky...
  - Don’t create your own (yet)
  - Smart people can produce poor hash functions (what’s a bad hash function?)
    - Knuth’s multiplicative hash in “the Art of Computer Programming”
- Characteristics:
  - Computed in ________ time.
  - Deterministic.
  - Satisfy the SUHA.

# Hash Functions



Easy, if
|KeySpace| ~ N



# Hashing Strings (an example)

Given: 8 character strings are easy to hash

The idea: Select 8 random positions from long strings and hash that substring.

A bunch of strings:

```
Lookyhere, Huck, being rich ain't going
No! Oh, good-licks; are you in real dead
Just as dead earnest as I'm sitting here
nto the gang if you ain't respectable, y
Can't let me in, Tom? Didn't you let me
Yes, but that's different. A robber is m
irate is -- as a general thing. In most
Now, Tom, hain't you always ben friendly
ut, would you, Tom? You wouldn't do that
Huck, I wouldn't want to, and I DON'T wa
ay? Why, they'd say, 'Mph! Tom Sawyer's
t!' They'd mean you, Huck. You wouldn't
uck was silent for some time, engaged in
Well, I'll go back to the widder for a m
can come to stand it, if you'll let me
All right, Huck, it's a whiz! Come along
Will you, Tom -- now will you? That's go
he roughest things, I'll smoke private a
hrough or bust. When you going to start
Oh, right off. We'll get the boys togeth
```

# Hashing Strings (an example)

Given: 8 character strings are easy to hash

The idea: Select 8 random positions from long strings and hash that substring.

A bunch of strings:

```
http://en.wikipedia.org/wiki/Le%C5%9Bna_Grobla
http://en.wikipedia.org/wiki/Blow_the_Man_Down
http://en.wikipedia.org/wiki/Swen_K%C3%B6nig
http://en.wikipedia.org/wiki/2/7th_Cavalry_Commando_Regiment_(Australia)
http://en.wikipedia.org/wiki/Salman_Ebrahim_Mohamed_Ali_Al_Khalifa
http://en.wikipedia.org/wiki/Alice_High_School
http://en.wikipedia.org/wiki/Beautiful,_Dirty,_Rich
http://en.wikipedia.org/wiki/RFA_Sir_Bedivere_(L3004)
http://en.wikipedia.org/wiki/Birthright_(band)
http://en.wikipedia.org/wiki/Jacky_Vimond
http://en.wikipedia.org/wiki/Vachon
http://en.wikipedia.org/wiki/McCarthy_%26_Stone
http://en.wikipedia.org/wiki/Salisbury,_New_Hampshire
http://en.wikipedia.org/wiki/A_Line_of_Deathless_Kings
http://en.wikipedia.org/wiki/Newfoundland_Irish
http://en.wikipedia.org/wiki/Beatrice_Politi
http://en.wikipedia.org/wiki/Bona_Sijabat
http://en.wikipedia.org/wiki/Sour_sanding
http://en.wikipedia.org/wiki/Dr_Manmohan_Singh_Scholarship
http://en.wikipedia.org/wiki/Religion_in_Jordan
```

# Collision handling - Separate Chaining: (an example of open hashing)

$S = \{16, 8, 4, 13, 29, 11, 22\}$ $|S| = n$ $h(k) = k\%7$

| | |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |

| | **Worst case** | **Under SUHA** |
|---|---|---|
| Insert | | |
| Remove/find | | |

# Collision Handling - Probe based hashing: (example of closed hashing)

S = {16, 8, 4, 13, 29, 11, 22}   |S| = n   h(k) = k%7

| | |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |

Try h(k) = (k + 0) % 7. If full...
try h(k) = (k + 1) % 7. If full...
try h(k) = (k + 2) % 7. If full...
try...

# Probe based hashing – 2 problems…

## Removals:

| | |
|---|---|
| 0 | 22 |
| 1 | 8 |
| 2 | 16 |
| 3 | 29 |
| 4 | 4 |
| 5 | 11 |
| 6 | 13 |

## Clustering:

http://groups.engin.umd.umich.edu/CIS/course.des/cis350/hashing/WEB/HashApplet.htm

# Probe based hashing: (double hashing)

$S = \{16, 8, 4, 13, 29, 11, 22\}$ $|S| = n$ $H(k,i) = h_1(k) + ih_2(k)$

| | |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |

Try $h(k) = (k + 0^*h_2(k))$ % 7. If full...
try $h(k) = (k + 1^*h_2(k))$ % 7. If full...
try $h(k) = (k + 2^*h_2(k))$ % 7. If full...
try...

# Hash table performance: expected # of probes for Find(key) under SUHA

Linear probing -

successful: $\frac{1}{2}(1 + 1/(1-\alpha))$

unsuccessful: $\frac{1}{2}(1 + 1/(1-\alpha))^2$

Double hashing -

successful: $1/\alpha \ln 1/(1-\alpha)$

unsuccessful: $1/(1-\alpha)$

Separate chaining -

successful: $1 + \alpha/2$

unsuccessful: $1 + \alpha$

## Do not memorize these!

## Observe:

- As α increases, running times increase…
- If α is held constant then running times are constant…

## Which is better?

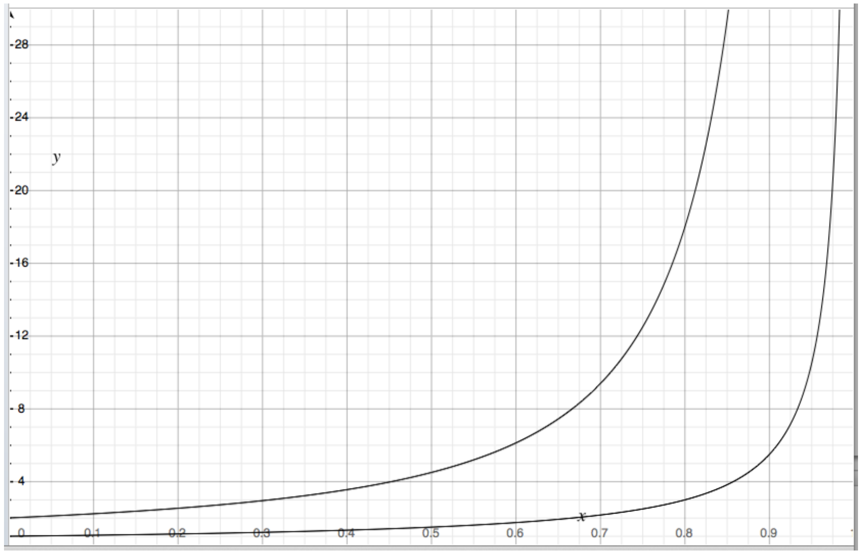
- Big records –
- Structure speed –

# Hash table performance: expected # of probes for Find(key) under SUHA

Linear probing -

successful: $\frac{1}{2}(1 + 1/(1-\alpha))$

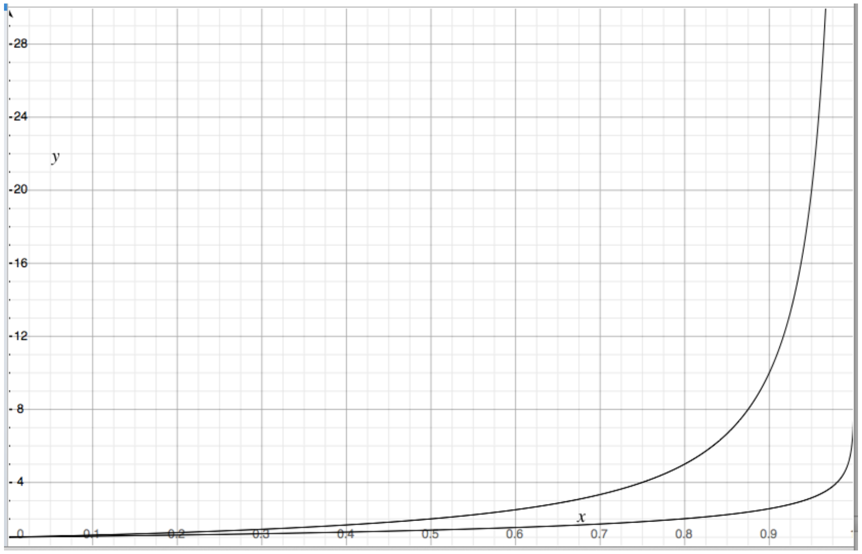
unsuccessful: $\frac{1}{2}(1 + 1/(1-\alpha))^2$



Double hashing -

successful: $1/\alpha \ln 1/(1-\alpha)$

unsuccessful: $1/(1-\alpha)$



# What’s left???

Running times of dictionary algorithms are a function of load factor, _____ ,but we hoped for _________ running times.

hmmmm….

What structures do hash tables replace for us?