

Chapter 6, Hash tables:

- ▶ General introduction; separate chaining (**Today**)
- ▶ Open addressing (next week Monday)
- ▶ Hash functions (next week Wednesday)
- ▶ Practice separate chaining (Thursday lab)
- ▶ Perfect hashing (week-after Monday)
- ▶ Hash table performance (week-after Wednesday)

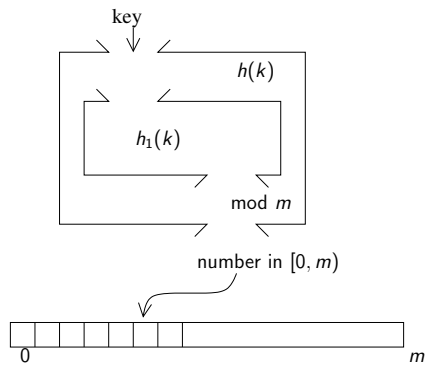
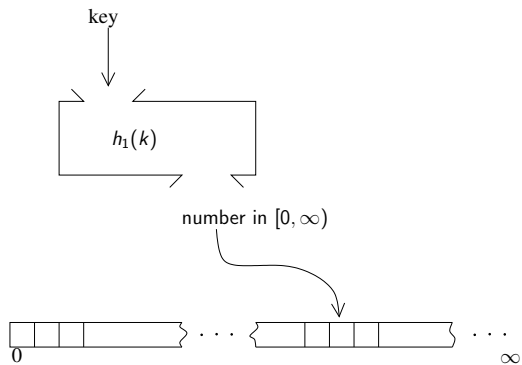
Today:

- ▶ Test problems
- ▶ The story of the Map ADT
- ▶ Goals and terminology of the unit
- ▶ Separate chaining implementation
- ▶ Variables and metrics of performance

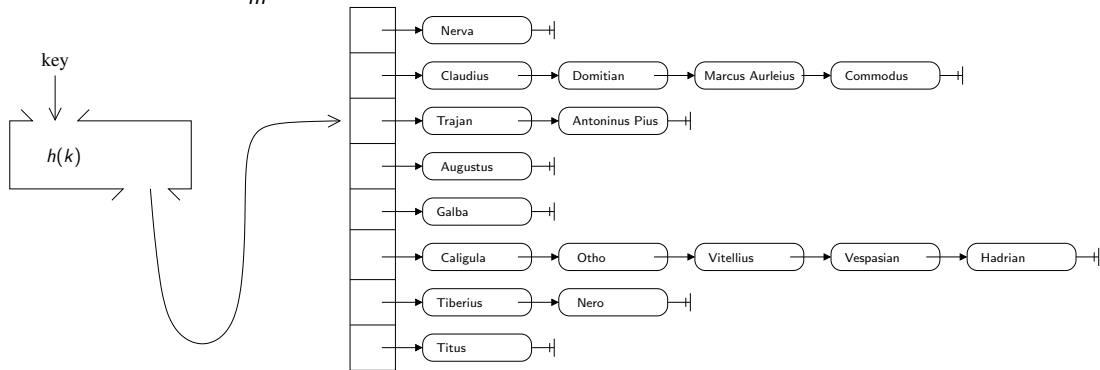
Find	Search the data structure for a given key
Insert	Add a new key to the data structure
Delete	Get rid of a key and fix up the data structure

<code>containsKey()</code>	Find
<code>get()</code>	Find
<code>put()</code>	Find + insert
<code>remove()</code>	Find + delete

	Find	Insert	Delete
Unsorted array	$\Theta(n)$	$\Theta(1)$ [$\Theta(n)$]	$\Theta(n)$
Sorted array	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n)$
Linked list	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$
Balanced BST	$\Theta(\lg n)$	$\Theta(1)$ [$\Theta(\lg n)$]	$\Theta(1)$ [$\Theta(\lg n)$]
What we want	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$



Separate chaining: $\frac{n}{m} < \alpha$ where $\alpha > 1$



Open addressing: $\frac{n}{m} < \alpha$ where $\alpha < 1$

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

A	D	E	G	F	H	B	C	J	I	
---	---	---	---	---	---	---	---	---	---	--

Unit agenda:

- ▶ Solution 1: Separate chaining (plus basic concepts and terminology). (**Today** and lab next week)
- ▶ Solution 2: Open addressing. (next week Monday)
- ▶ All about hash functions. (next week Wednesday)
- ▶ Solution 3: Perfect hashing. (week-after Monday)
- ▶ Looking carefully at performance. (week-after Wednesday)

Hash table terminology:

- ▶ Hash table: A *data structure*, not an ADT ...
- ▶ Bucket: A position in the (main) array, or, abstractly, an index in the range $[0, m)$.
- ▶ Hash function: A function from keys to buckets.
- ▶ Collision: When two keys are hashed to the same bucket.
- ▶ Chain: A sequence of keys that needs to be searched through to find a given key.
- ▶ Load factor (α): An upper bound on the ratio of keys to buckets.

Factors in best vs worst vs expected case:

- ▶ State of the table
- ▶ Length of the bucket
- ▶ Position of key in the bucket.

Parameters that can be adjusted for engineering a hash table:

- ▶ Load factor α
- ▶ Rehash strategy
- ▶ Hash function

$$\begin{array}{lcl}
 & O(1) & c_0 \\
 & O(1) & c_0 \\
 & O(1) & c_0 \\
 & \vdots & \\
 & O(1) & c_0 \\
 \text{rehash} \longrightarrow & O(n) & c_1 + c_2 n \\
 & O(1) & c_0 \\
 & \vdots & \\
 & O(1) & c_0
 \end{array} \left. \vphantom{\begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \end{array}} \right\} \begin{array}{l} T(n) \\ = \\ = \\ = \end{array} \begin{array}{l} (n-1)c_0 + c_1 + c_2 n \\ (c_0 + c_2)n + c_1 - c_0 \\ \Theta(n) \end{array}$$

Hash functions should distribute the keys *uniformly* and *independently*.

Uniformity:

$$P(h(k) = i) = \frac{1}{m}$$

Independence:

$$P(h(k_1) = i) = P(h(k_1) = i \mid h(k_2) = j)$$

Coming up:

*Do **Optimal BST** project (Due Mon, Apr 14)*

*Due **Fri, Apr 11** (end of day)*

Read Sections 7.(1 & 2)

Take quiz (actually due Mon, Apr 14)

*Due **Tues, Apr 15** (end of day)*

Read Section 7.3

Do Exercises 7.(4,5,7,8)

Take quiz

*Due **Mon, Apr 21** (end of day)*

Read Sections 7.(4 & 5) (No exercises or quiz)