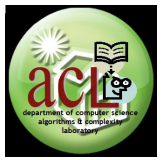
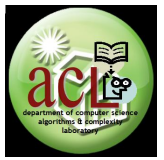
Hash Tables

Lesson 12



Outline

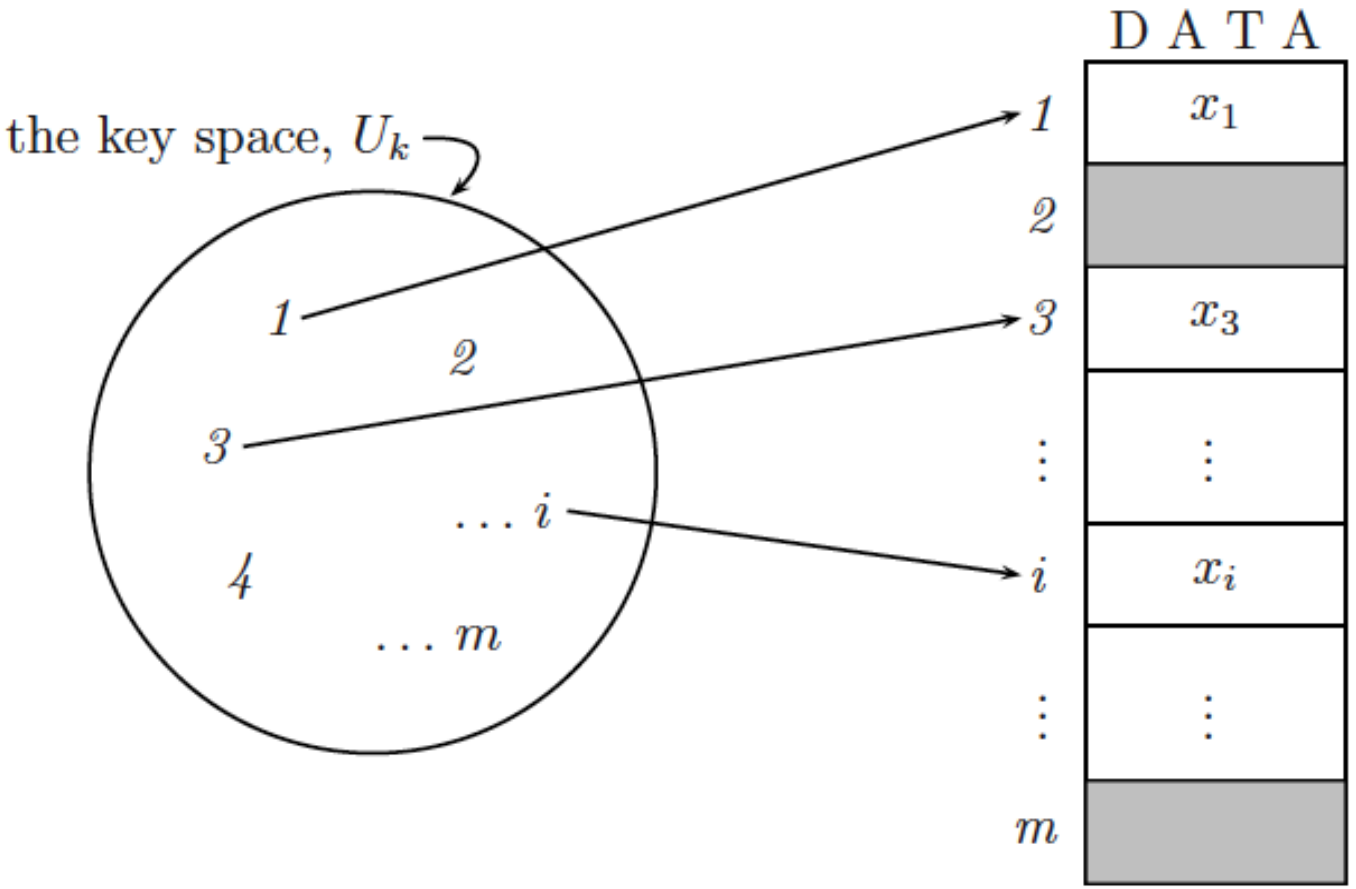
- Direct Address Tables and Hashing
- Hash Functions
  - Division Method
  - Multiplication Method
- Collision Problem and Resolution
  - Chaining
  - Open Addressing
    - Linear Probing
    - Double Hashing
  - Addressing Deletions



Outline

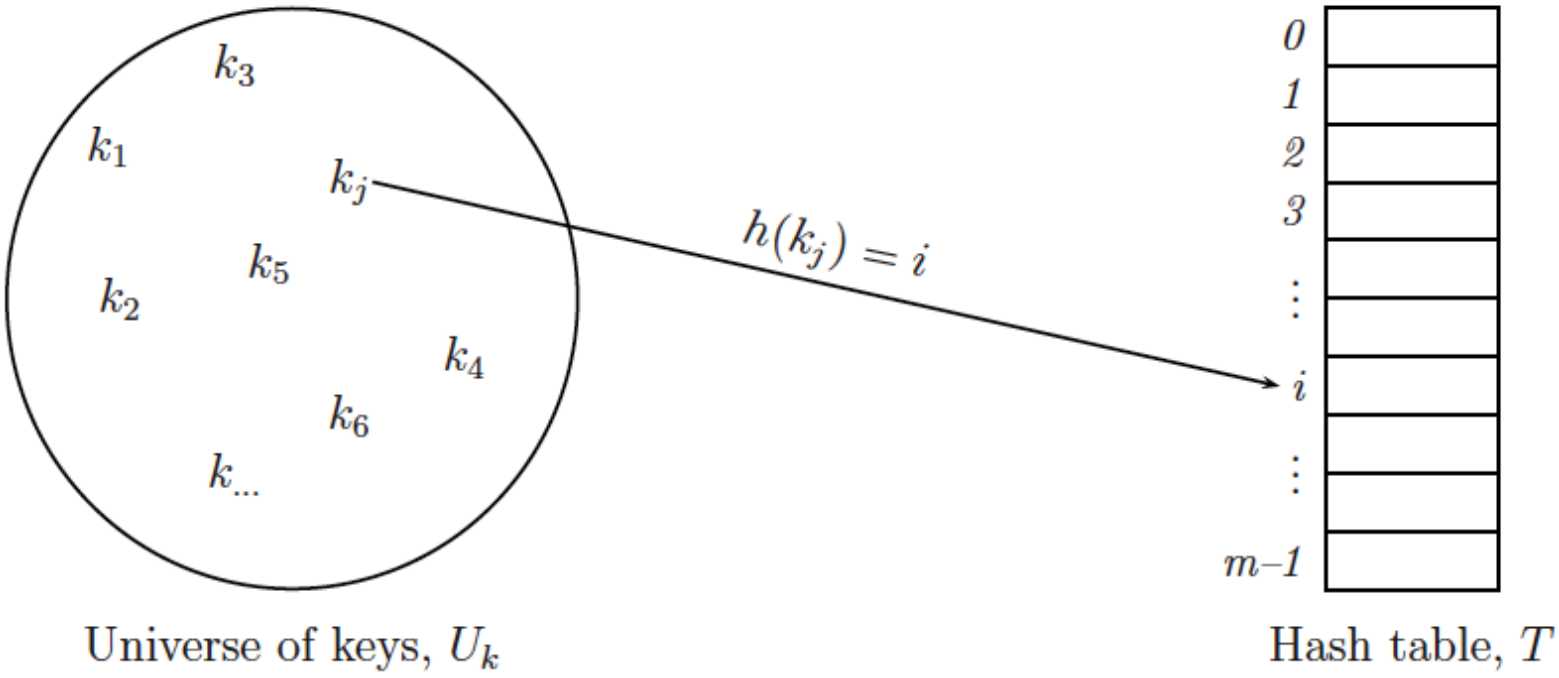
- **Direct Address Tables and Hashing**
- Hash Functions
  - Division Method
  - Multiplication Method
- Collision Problem and Resolution
  - Chaining
  - Open Addressing
    - Linear Probing
    - Double Hashing
  - Addressing Deletions

Direct Address Tables - Review



Converting Keys to Indices

$h : U_k \rightarrow \{0, 1, 2, \ldots, m-1\}$



Implementing Hash Tables

- Convert non-numeric keys to numbers

$$\text{KEYS} = 11052519$$

$$\text{KEYS} = 11 \times 26^3 + 05 \times 26^2 + 25 \times 26^1 + 19 \times 26^0 = 197385$$

$$\text{KEYS} = 01001011\,01000101\,01011001\,01010011_2$$

$$= 75 \times 256^3 + 69 \times 256^2 + 89 \times 256^1 + 83 \times 256^0 = 1262836051_{10}$$

- Choose a good ***hash function***
- Choose a ***collision resolution*** policy

Example

FOLK	1179601995	FALL	1178684492	EVIL	1163282764	FARM	1178686029
TEST	1413829460	OPEN	1330660686	BELL	1111837772	TAXI	1413568585
BACK	1111573323	BALI	1111575625	AVIV	1096173910	HOLY	1213156441
ROAD	1380925764	GRAB	1196572994	RING	1380535879	BANK	1111576139
HOME	1213156677	ZOOM	1515147085	BOMB	1112493378	BIRD	1112101444
CREW	1129465175	MESS	1296388947	BABY	1111573081	JAVA	1245795905
FAUX	1178686808	QUIZ	1364543834	HIGH	1212761928	MATH	1296127048
OPUS	1330664787	WARM	1463898701	ROOF	1380929350	BODY	1112491097
SODA	1397703745	HALF	1212238918	ALSO	1095521103	DROP	1146244944
DEEP	1145390416	ERGO	1163020111	UNIX	1431193944	PILI	1346980937
MENU	1296387669	THOU	1414025045	IRAQ	1230127441	OVER	1331053906
BEAR	1111834962	FIAT	1179205972	TOOL	1414483788	PULP	1347767376
SORT	1397707348	TEAR	1413824850	EPIC	1162889539	SINE	1397313093
IRON	1230131022	DEAD	1145389380	JAZZ	1245796954	ZINC	1514753603
DATA	1145132097	LONG	1280265799	SHOW	1397247831	LAUS	1279350099

Outline

- Direct Address Tables and Hashing
- **Hash Functions**
  - **Division Method**
  - **Multiplication Method**
- Collision Problem and Resolution
  - Chaining
  - Open Addressing
    - Linear Probing
    - Double Hashing
  - Addressing Deletions

Criteria for a Good Hash Function

- ***Can be calculated very fast***
  - Hash tables require lesser comparisons for search than most structures but computation of the hash function h(k) usually requires some computation time
- ***Minimizes collisions***
  - ***Collisions happen when two distinct keys get hashed to the same index***
  - ***There are some cases where perfect, collisionless scenarios exist, but mostly, this is not the case***

Division Method

- $h(k) = k \bmod m$
  - m is the size of the table
- Criteria for choosing m
  - m should not be even
  - m should not be a power of the radix of the computer
  - m should not be of the form $2^i \pm j$, where j is some small integer, if the keys are character strings interpreted in radix 2^r

Example – Division Method

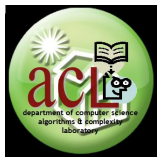
HASH ADDRESS	COLLIDING KEYS		
0	BELL		
1	MESS	EVIL	BIRD
2			
3	SORT	TEAR	MATH
4	ZOOM		
5			
6			
7			
8			
9	FOLK		
10			
11	TEST	THOU	EPIC
12	GRAB	RING	ROOF
13			
14			
15	FALL		
16	ERGO		
17	ALSO		
18	FARM	BODY	
19	DEAD	LONG	
20	IRON	HALF	
21	MENU		
22	BEAR	PILI	
23	WARM		
24	DROP		
25	ROAD		
26			
27	OPUS		
28	UNIX		
29	TOOL		

HASH ADDRESS	COLLIDING KEYS		
30	FAUX		
31			
32	CREW	BANK	
33			
34	HOME	TAXI	HOLY
35	DATA	QUIZ	
36	JAZZ		
37	SHOW		
38	IRAQ		
39			
40			
41	OVER		
42	BABY		
43			
44			
45	SINE		
46			
47			
48	BACK	AVIV	
49	BALI	JAVA	
50	PULP		
51			
52	DEEP	ZINC	
53	FIAT		
54	HIGH		
55			
56	OPEN		
57	BOMB		
58	SODA	LAUS	

Multiplication Method

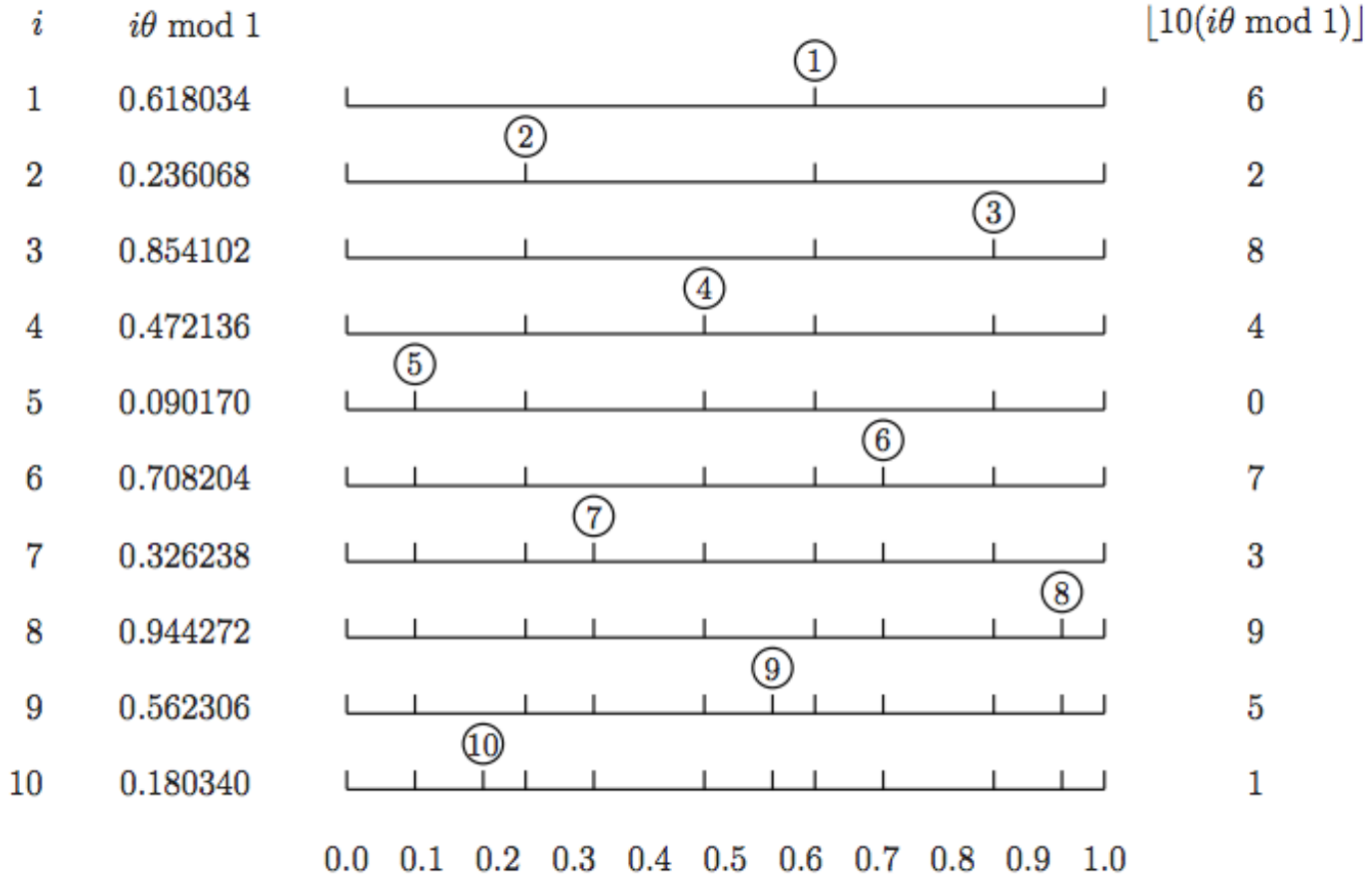
- $h(k) = \lfloor m(k\theta \bmod 1) \rfloor$
  - $\theta = (\sqrt{5} - 1)/2$ is called the ***conjugate of the golden ratio φ***
- ***Basis of the hash function***

THEOREM 15.1 (Vera Turán Sós). Let θ be any irrational number. When the points $\theta \bmod 1, 2\theta \bmod 1, \ldots, n\theta \bmod 1$ are placed in the line segment [0..1], the $n+1$ line segments formed have at most three different lengths. Moreover, the next point $(n+1)\theta \bmod 1$ will fall in one of the largest existing segments.



Multiplication Method

- VTS Theorem, illustrated



Example – Multiplicative Method

HASH ADDRESS	COLLIDING KEYS		
0	HOME	FALL	
1	ROAD	BELL	
2			
3	FOLK		
4	BODY		
5			
6			
7			
8	ERGO	DEAD	
9	IRON	HOLY	
10	ZINC		
11			
12	CREW	SHOW	PILI
13	EPIC		
14	WARM	PULP	
15			
16	BALI	GRAB	
17	BIRD	JAVA	
18	ROOF		
19			
20	SORT		
21	BEAR	MATH	SINE
22	LONG		
23	OPEN	FIAT	TEAR
24	FAUX		
25			
26	DEEP	ZOOM	
27			
28	TOOL		
29			
30	QUIZ		
31	DATA		

HASH ADDRESS	COLLIDING KEYS		
32	TEST		
33			
34	BACK	SODA	
35			
36			
37	JAZZ		
38			
39	EVIL		
40	LAUS		
41			
42			
43	THOU		
44	OVER		
45	HIGH		
46			
47	HALF		
48			
49	DROP		
50			
51	BOMB		
52	MESS	UNIX	
53	RING		
54			
55			
56	TAXI		
57	AVIV		
58			
59	OPUS	FARM	BANK
60			
61			
62	MENU	BABY	IRAQ
63	ALSO		

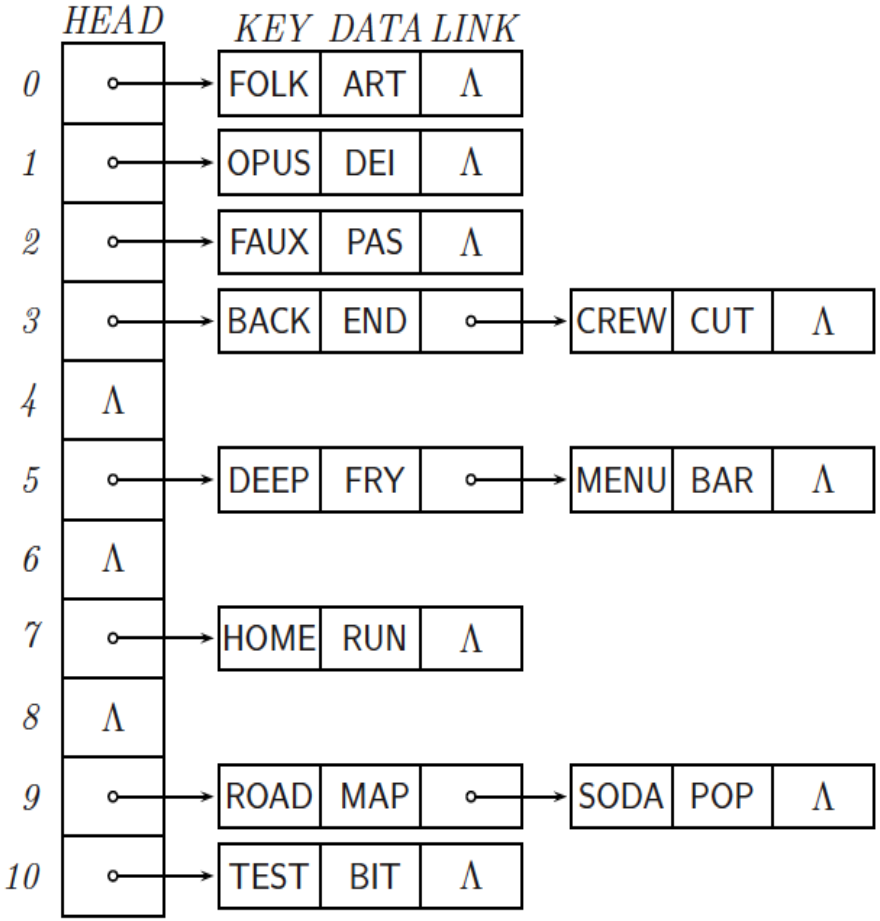
Outline

- Direct Address Tables and Hashing
- Hash Functions
  - Division Method
  - Multiplication Method
- **Collision Problem and Resolution**
  - **Chaining**
  - **Open Addressing**
    - **Linear Probing**
    - **Double Hashing**
  - **Addressing Deletions**

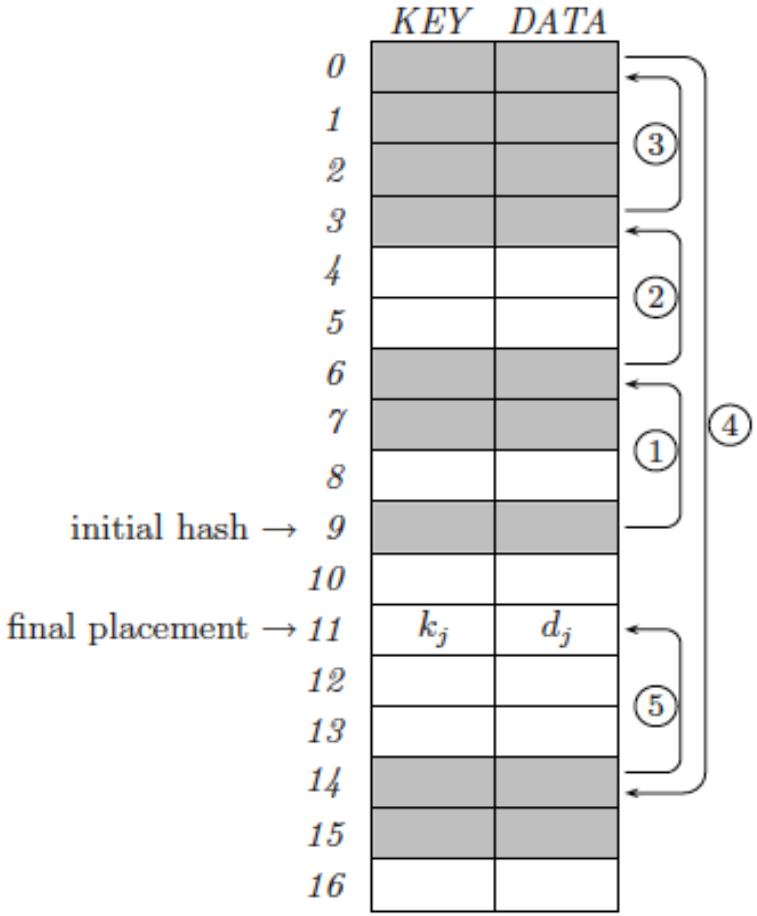
The Birthday Paradox

- How many persons you have to put in a room so you could have a good 50% chance that two of those persons have the same birthday (not necessarily the same year)?
- This is a basis for the collision problem, and is in fact common even in sparsely occupied hash tables (Standish)

Collision Resolution by Chaining



Collision Resolution by Open Addressing



Open Addressing: Linear Probing

$$h(K),\ h(K)-1,\ h(K)-2,\ \ldots,\ 1,\ 0,\ m-1,\ m-2,\ \ldots,\ h(K)+1$$

$$p(K) = [h(K) - i] \bmod m \qquad i = 0, 1, 2, \ldots, m-1$$

$(k\,,\,d)$	$h(k)$
FOLK ART	10
TEST BIT	16
BACK END	2
ROAD MAP	7
HOME RUN	8
CREW CUT	10
FAUX PAS	15
OPUS DEI	15
SODA POP	0
DEEP FRY	2
MENU BAR	18

Open Addressing: Double Hashing

$$p(K) = [h(K) - i \cdot h'(K)] \bmod m \qquad i = 0, 1, 2, \ldots, m-1$$

$$h'(k) = 1 + k \bmod (m-2)$$

$(k\,,\,d)$	$h(k)$	$h'(k)$
FOLK ART	10	12
TEST BIT	16	15
BACK END	2	2
ROAD MAP	7	6
HOME RUN	8	9
CREW CUT	10	17
FAUX PAS	15	3
OPUS DEI	15	5
SODA POP	0	7
DEEP FRY	2	15
MENU BAR	18	4

Addressing Deletions

- Chaining
  - Simply **delete record from table or chain**
- Open Addressing
  - Mark the previously occupied slot as "vacated" (**lazy deletion**)
    - Additional field and can **affect searching and insertion**
  - Rehash all records that might have been affected by deletion
    - **Extra work** for deletions
    - Feasible only for **linear probing**

Addressing Deletions

- Chaining
  - Simply **delete record from table or chain**
- Open Addressing
  - Mark the previously occupied slot as "vacated" (**lazy deletion**)
    - Additional field and can **affect searching and insertion**
  - Rehash all records that might have been affected by deletion
    - **Extra work** for deletions
    - Feasible only for **linear probing**

Addressing Deletions

(k, d)		$h(k)$
JAVA	PGM	3
MATH	LAB	8
BODY	FIT	3
DROP	BOX	0
PILI	NUT	3
OVER	PAR	0
PULP	BIT	7
SINE	DIE	7
ZINC	ORE	12
LAUS	DEO	1

(a)

	KEY	DATA
0	DROP	BOX
1	PILI	NUT
2	BODY	FIT
3	JAVA	PGM
4		
5		
6	SINE	DIE
7	PULP	BIT
8	MATH	LAB
9		
10	LAUS	DEO
11	ZINC	ORE
12	OVER	PAR

(b)

	KEY	DATA
0	DROP	BOX
1	PILI	NUT
2		
3	JAVA	PGM
4		
5		
6	SINE	DIE
7	PULP	BIT
8	MATH	LAB
9		
10	LAUS	DEO
11	ZINC	ORE
12	OVER	PAR

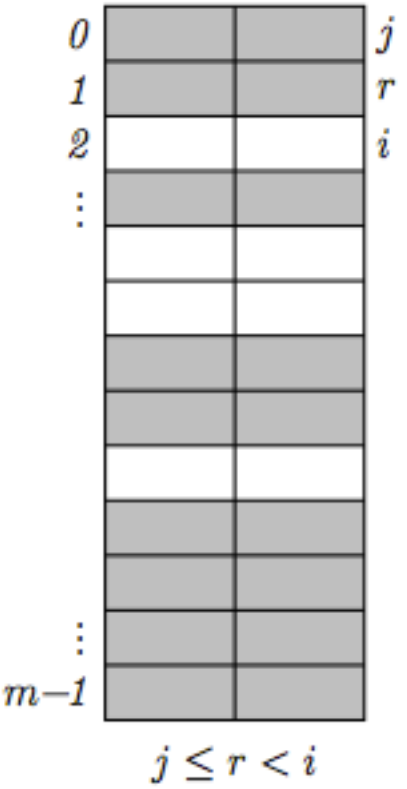
(c)

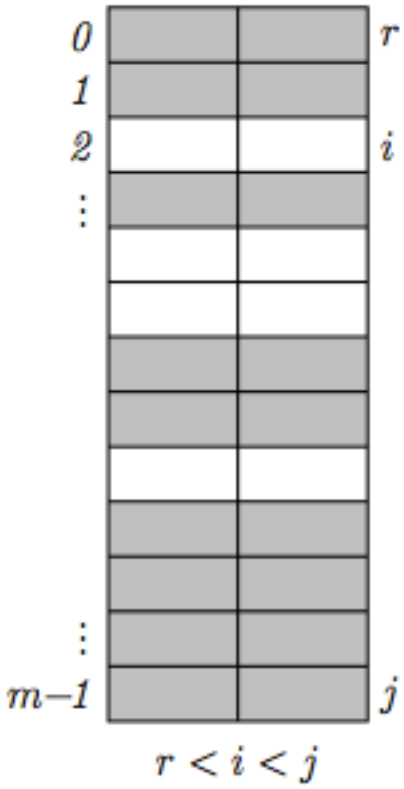
	KEY	DATA
0	DROP	BOX
1	LAUS	DEO
2	PILI	NUT
3	JAVA	PGM
4		
5		
6	SINE	DIE
7	PULP	BIT
8	MATH	LAB
9		
10		
11	ZINC	ORE
12	OVER	PAR

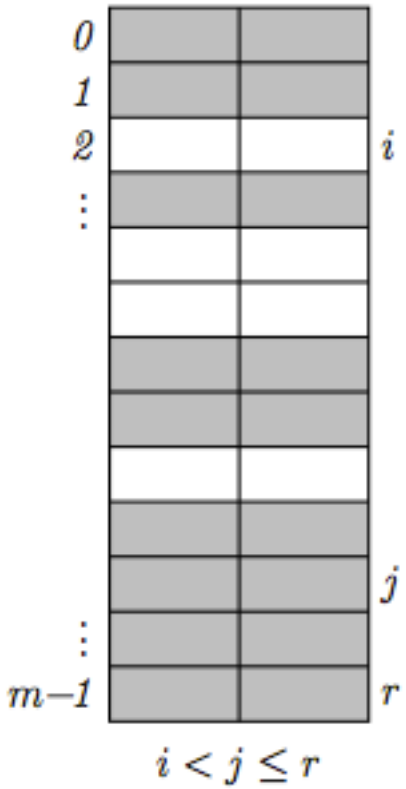
(d)

Addressing Deletions

- When **NOT** to reposition a key







Thank you
for your attention.

Questions ...