



## Lecture 12:

### Sorting and Searching Continued

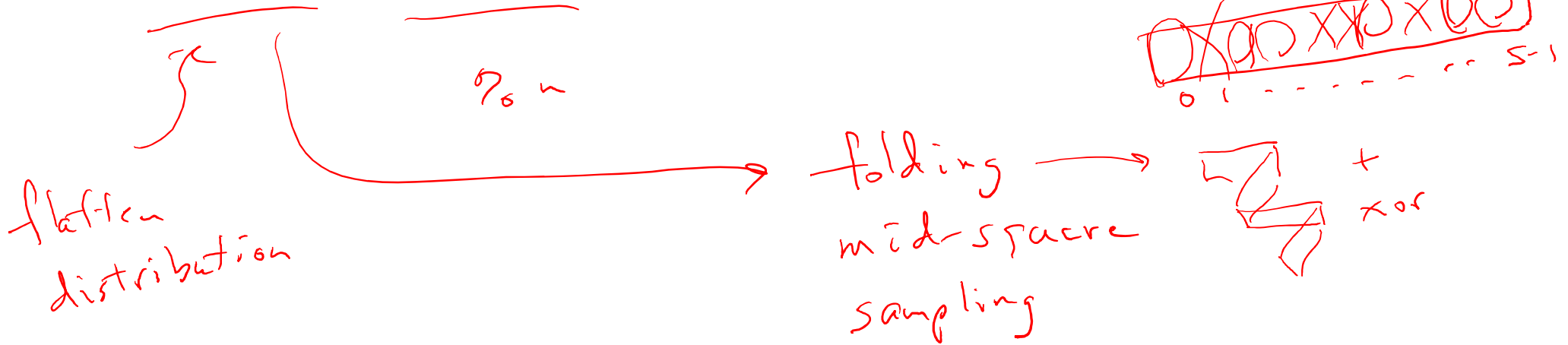
# Objectives

- Be able to define the terms hashing, hash table, perfect hash function, uniform key distribution
- Be able to comment on how hash tables achieve a constant-time ( $O(1)$ ) performance for the search problem
- Be able to talk about various scrambling strategies for hash functions
- Be able to discuss hash table sizes and how they relate to key distributions
- Be able to define what a probe is, and to discuss probe sequences
- Be able to define and illustrate linear open addressing
- Be able to define quadratic open addressing
- Be able to do an analysis of hash table efficiency, based on load factor  $\lambda$ , of a linear open addressing hash table and of a separate chaining hash table
- Be able to discuss how *dict* and *set* achieve their ability to store any Python object

# Hashing

$n = \text{size of table}$   
(# of slots)

- Scrambling + addressing = hashing



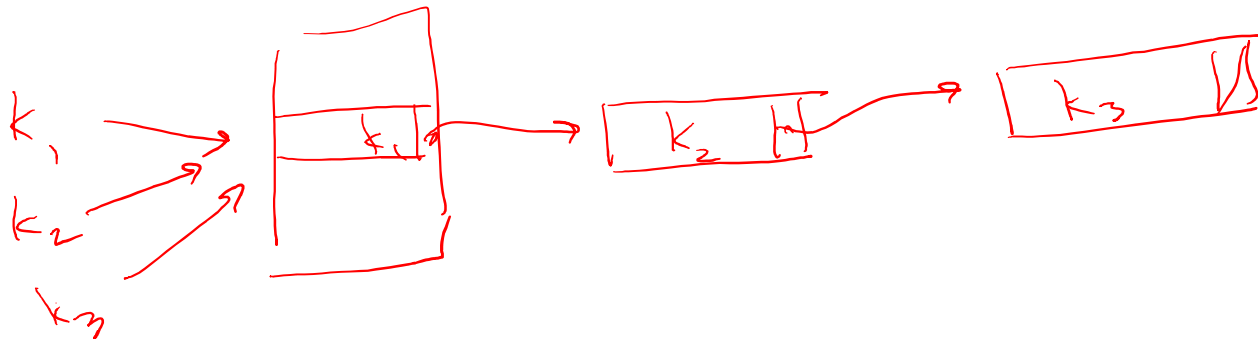
Chaining refers to using pointers to linked stored items

## Collisions and Collision Resolution

- There are two general approaches:

- Open Addressing – Store the collided key in the primary table area

- Separate Chaining – Store the collided key elsewhere and link it to the original hashed location



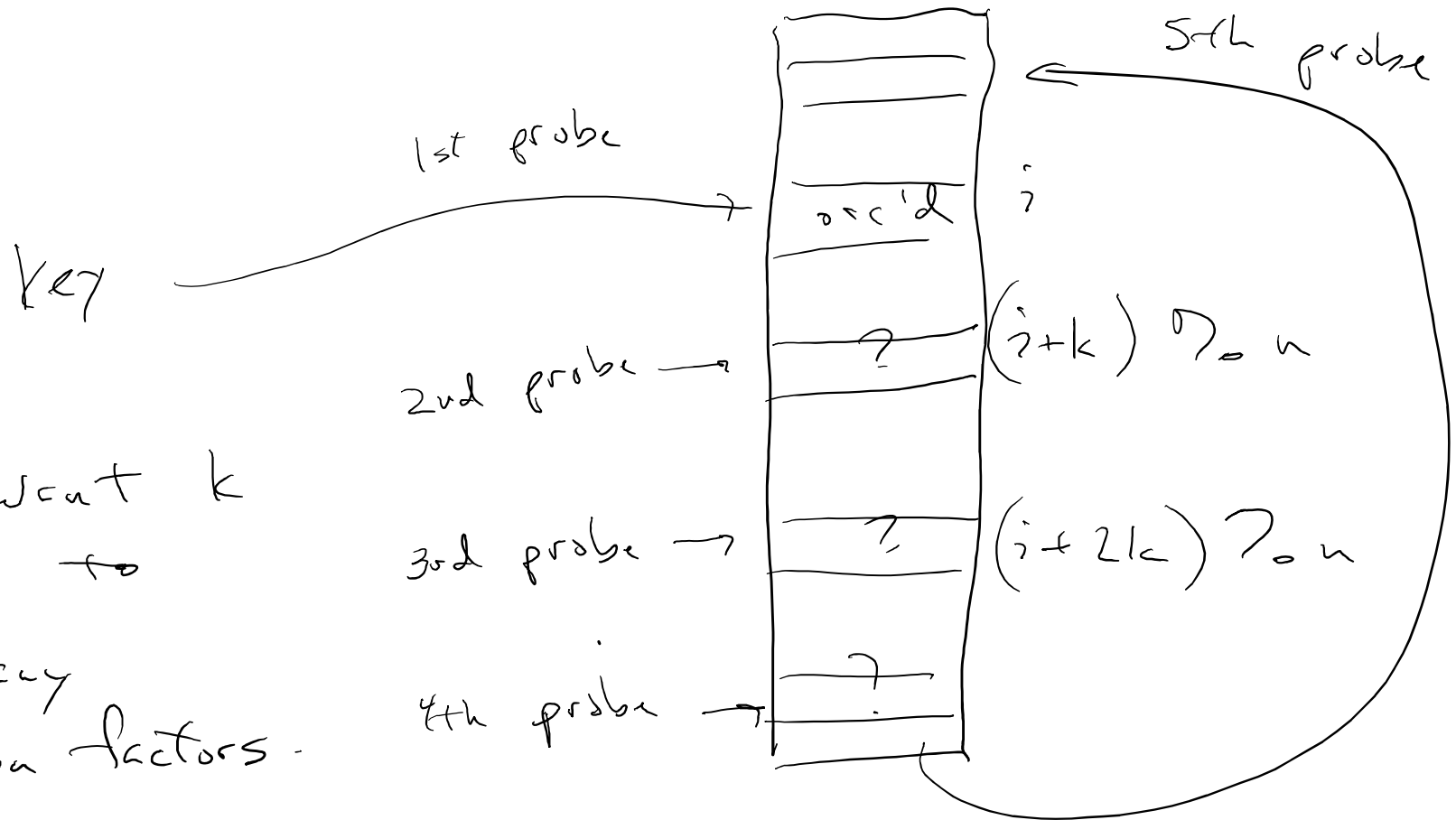
## Probes and Probe Sequences

Clustering is where probe sequences become unduly long.  
Degrades efficiency

A probe is an investigation of a hash table position or an item in a chained list. It can either be (1) investigating whether the slot is occupied & finding out it is not, or (2) a check of the value of what is stored there. To find a collided value will necessitate, potentially, a whole sequence of such probes.

# Linear Open Addressing

$$j^{\text{th}} \text{ probe} \quad (i + (j-1) \underbrace{k}_{\text{constant}}) \% n$$



Don't want  $k$   
&  $n$  to  
have any  
common factors.

Example: Load a table of size 19,  
using  $k = 5$

- Keys 34, 56, 88,  
23, 92, 77, 6, 87,  
32, 70, 30, 49

|    |    |
|----|----|
| 0  |    |
| 1  | 77 |
| 2  |    |
| 3  |    |
| 4  | 23 |
| 5  |    |
| 6  | 6  |
| 7  |    |
| 8  |    |
| 9  |    |
| 10 |    |
| 11 |    |
| 12 | 87 |
| 13 | 88 |
| 14 |    |
| 15 | 34 |
| 16 | 92 |
| 17 | 32 |
| 18 | 56 |

Deleting from a hash table



# Primary and Secondary Clustering

- Primary clustering occurs when many keys hash to the same address
- Secondary clustering occurs when probe sequences merge in other ways

Why does the book advocate making the size of the table a prime number?

# Quadratic Open Addressing

# Separate Chaining

# Hash Table Analysis

# dict and set

- Variable-sized hash tables