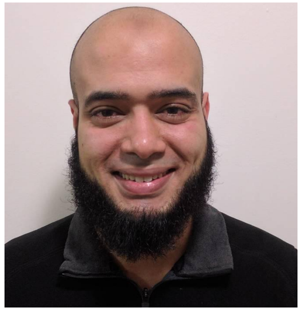
Data Structures

Chaining

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Smartphone Contacts Application

- Given a list of contact information (name, phone number), we need to be able to insert/remove/check each contact in some data-structure.
    - Target speed and efficiency!
- Let’s create a class that can hash its objects based on specific key(s)
    - Here we use the **name** as the main entry to search/remove

Data and Collisions

- Assume that we're using these entries, and have computed their respective final hash functions

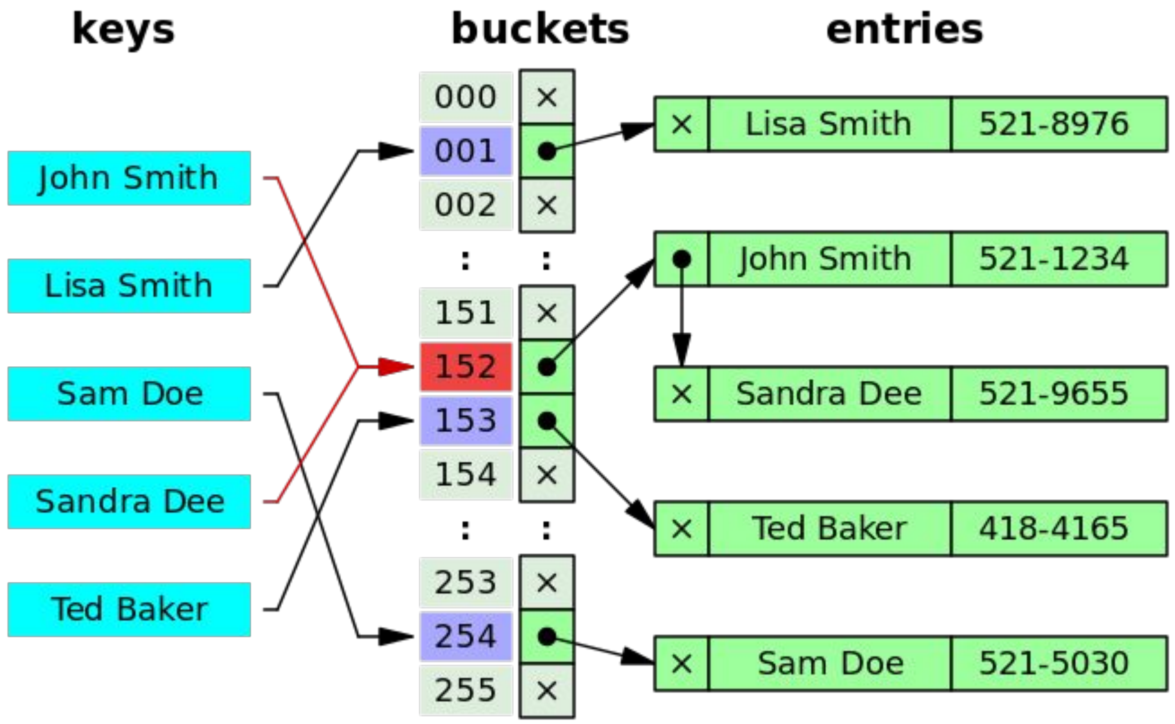
Name (as search key)	**Attached data (phone #)**	**Hash Function code**
John Smith	5211234	152
Lisa Smith	5218976	1
Sam Doe	5215030	254
Sandata Dee	5219655	152 (collision)
Ted Baker	4184165	153

Chaining

- This is a very simple idea to understand and implement
- Our array (aka table), will have in each index (aka bucket) another data structure that inserts/removes/searches for th**e items with the same** key
  - Array of linked-lists
  - Array of AVL tree
  - Array of vector
- The implementation may vary, but we have our eventual idea

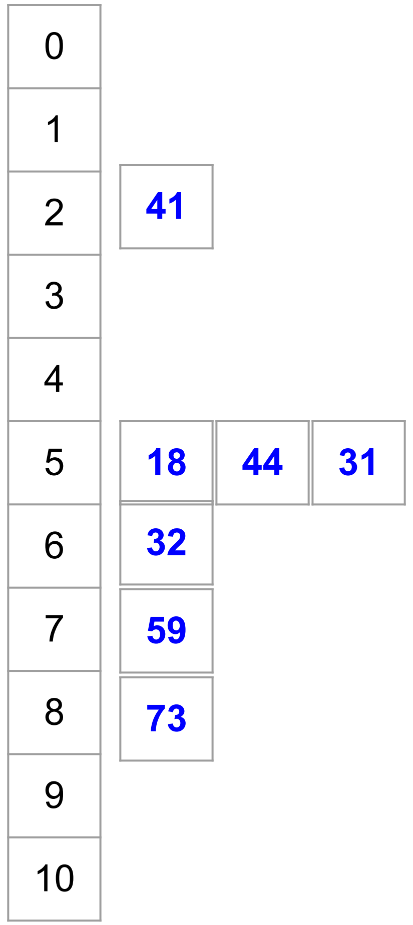
Chaining

- x (None) means there are no elements so far at this key
- Otherwise, each key is a linked list
- Key 152: linked list of 2 items (John, Sandra)



Integers example

- Assume we have the following integers
- [**18, 41, 22**, 44, 59, 32, 31, 73] and their respective hash indices: [5 , 2 , 9 , 5 , 7 , 6 , 5 , 8]
- Assume our table has 11 buckets



"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."