LEC 15

CSE 123

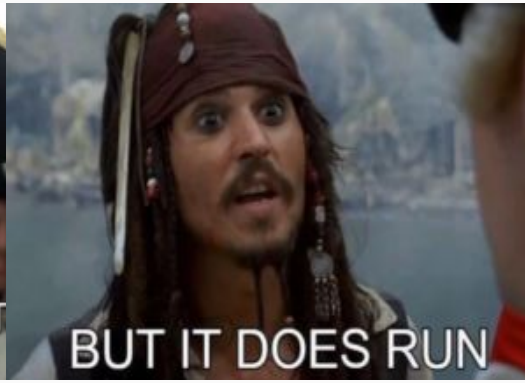
Hashing

Questions during Class?

Raise hand or send here

sli.do #cse123

BEFORE WE START



Music: 123 24su Lecture Tunes ☀️

Instructor: Joe Spaniac

TAs: Andras, Daniel, Eric, Nicole, Sahej, Trien, Zach

Lecture Outline

- Announcements
- Hashing
- Programming HashSet
- Final Remarks

Announcements

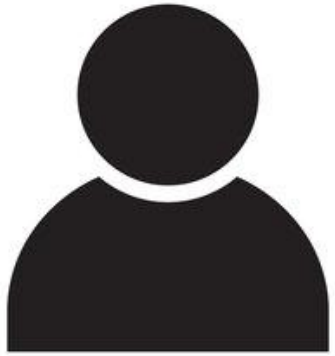
- P4 Spam Classifier Released!!!
  - Topics: Recursion, Machine Learning (AI)
  - PLEASE read the Specification (we can't emphasize this enough)
- Resubmission Period 6 closes tonight, 8/09 at 11:59pm

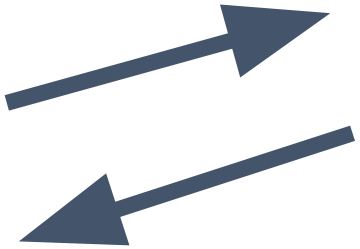
Upcoming…

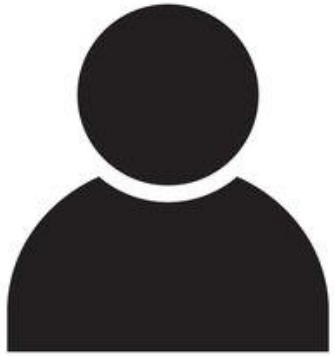
- Sections Next Tuesday/Thursday will be Exam Review!
- *Final Exam: Friday (8/16) 10:50-11:50am (GWN 301)*

Lecture Outline

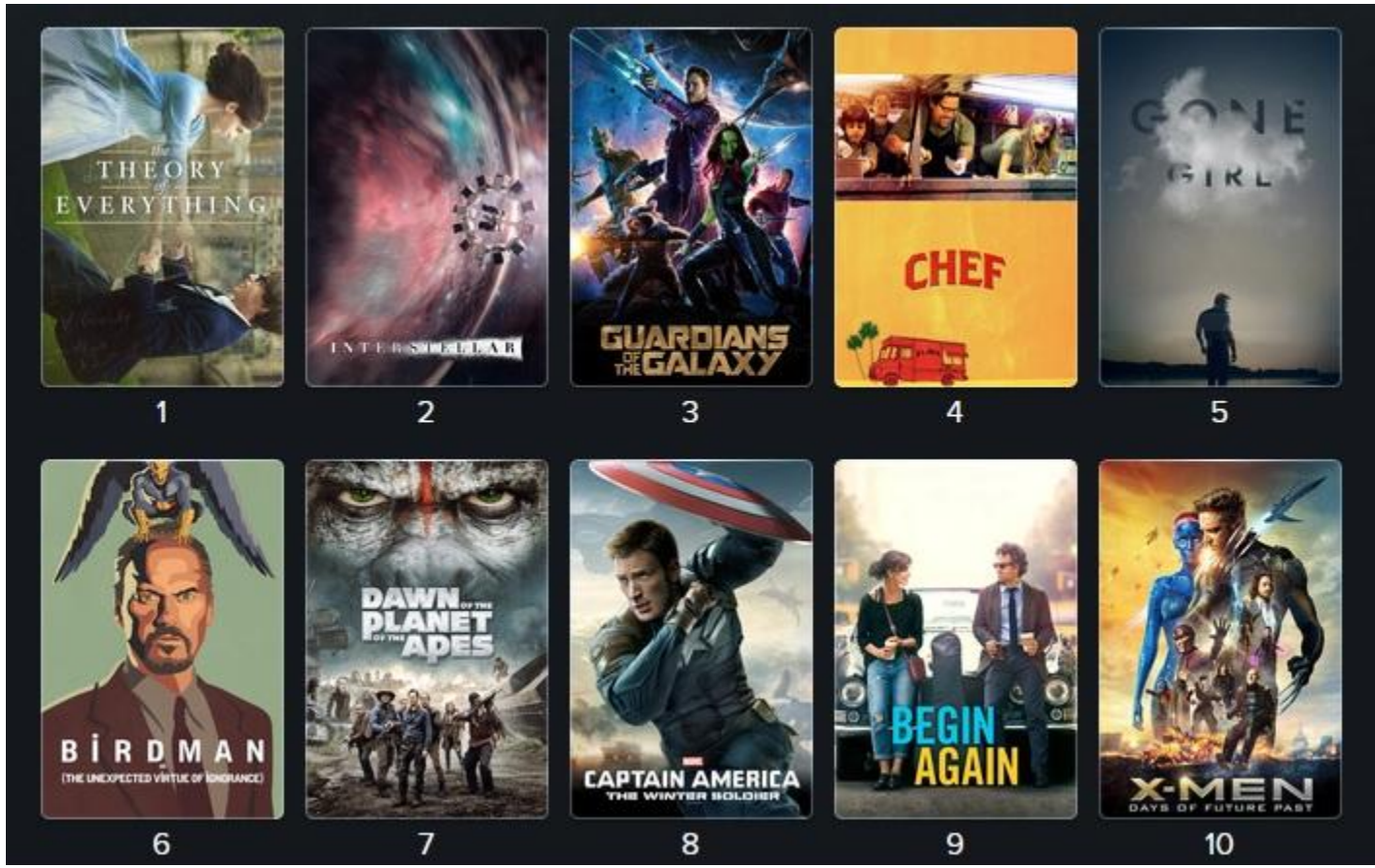
- Announcements
- **Hashing**
  - Introduction
  - Rules of Hash Codes
  - What is it and how do we use it?
  - Example
- Collisions and Resolving Them
- Programming HashSet
- Final Remarks








MATTHEW McCONAUGHEY
ANNE HATHAWAY
JESSICA CHASTAIN
MICHAEL CAINE
"THE END OF EARTH WILL NOT BE THE END OF US."
INTERSTELLAR
IN 70MM IMAX, 70MM FILM AND 35MM FILM
ON NOVEMBER 5
EVERYWHERE
NOVEMBER 7


THEORY of EVERYTHING
1
INTERSTELLAR
2
GUARDIANS OF THE GALAXY
3
CHEF
4
GONE GIRL
5
BIRDMAN
(THE UNEXPECTED VIRTUE OF IGNORANCE)
6
DAWN of the PLANET of the APES
7
CAPTAIN AMERICA
THE WINTER SOLDIER
8
BEGIN AGAIN
9
X-MEN
DAYS OF FUTURE PAST
10

HashMaps and Hash Sets

- We can store our values here in a Hash Table
- These efficiently store and retrieve data
- How do we make Hash Table operations fast?
- Through the use of *Hashing*

Operation	ArrayList	LinkedList	HashTable
contains(x)	O(n)	O(n)	
add(x)	O(1)*	O(1)	
remove(x)	O(n)	O(n)	

** take higher level CS courses if you're curious about the ambiguity*

HashMaps and Hash Sets

- We can store our values here in a Hash Table
- These efficiently store and retrieve data
- How do we make Hash Table operations fast?
- Through the use of *Hashing*

Operation	ArrayList	LinkedList	HashTable
contains(x)	O(n)	O(n)	**O(1) (average), O(n) (worst)**
add(x)	O(1)*	O(1)	**O(1) (average), O(n) (worst)**
remove(x)	O(n)	O(n)	**O(1) (average), O(n) (worst)**

** take higher level CS courses if you're curious about the ambiguity*

Basic Idea of Hashing

- Converting data (our values or objects) into a **Hash Code**
- **Hash Code** is basically like a *"fingerprint"*
- We store our data under the **Hash Code** (*fingerprint*)
- Use this **Hash Code** to retrieve our data
- Motivation: Makes searching, storing, and getting values easier, especially in large databases

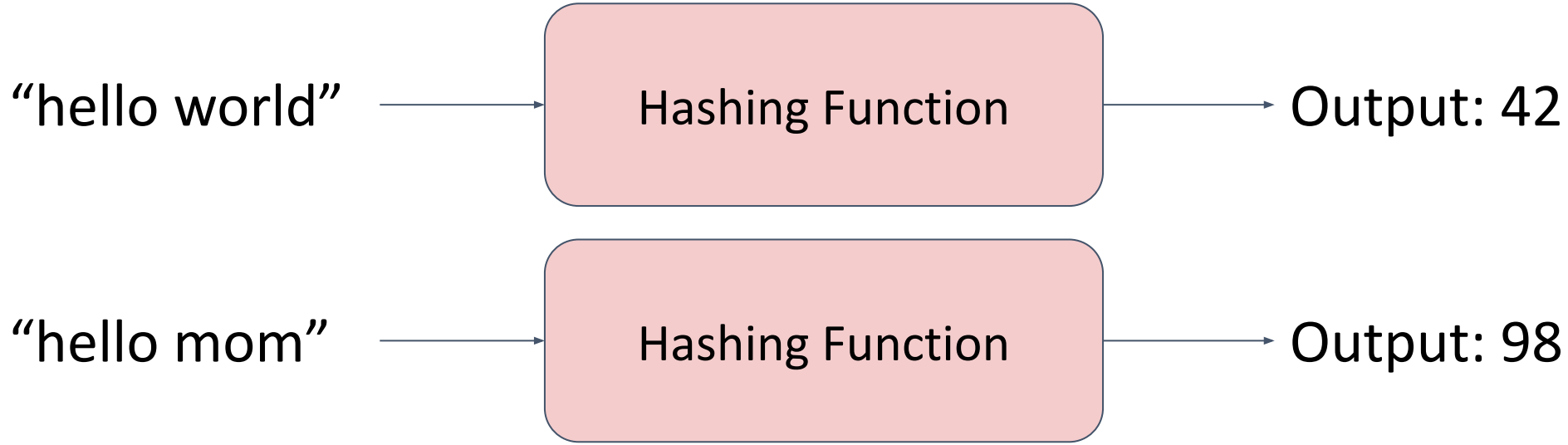
What is a Hash Function?

Hash Function - An algorithm that produces an "fingerprint" of the original message

- Other names: Hash or **Hash Code**, Fingerprint, Checksum, Digest, …

What makes a good Hash Function?

Equal “Objects” should **always** result in the same Hash Code

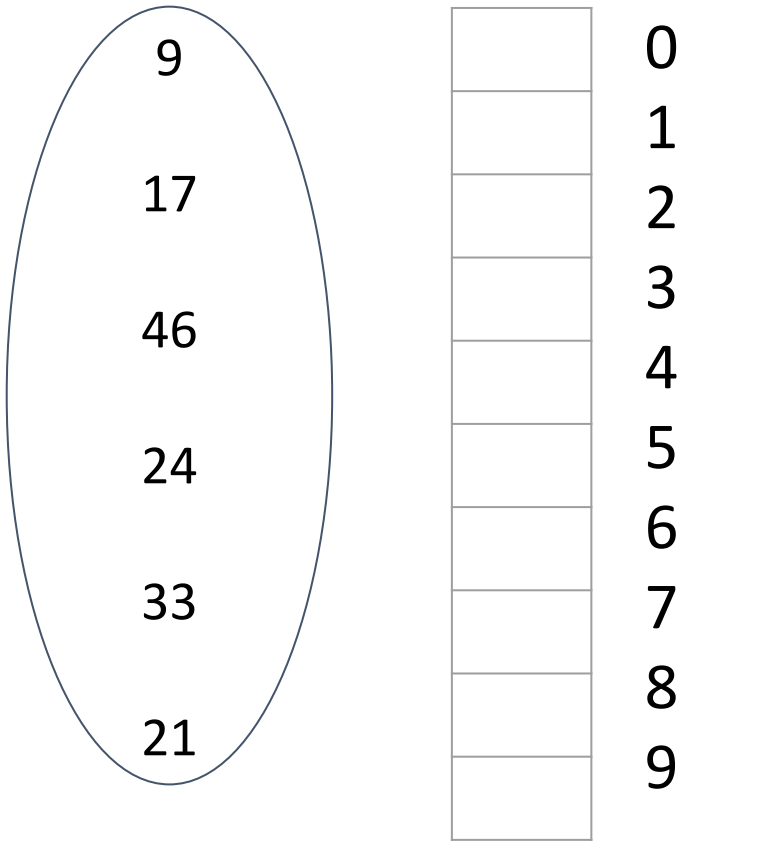


* Can’t be random

What is an Example Hash Function?

For Integers

h(x) = x % size



Recap

Let's Store our value "What is the meaning of life"

User: *"Hash Function, what is my Hash Code?"*

Computer: *"Your Hash Code is: "* _____

User: *"Ok, let's store our value under this fingerprint"*

Computer: *"Sure can do Chief"*

Recap

Let's Store our value "What is the meaning of life"

User: *"Can I get my value back?"*

Computer: *"No"*

User: *"???"*

User: *"Why not?!?"*

Computer: *"What is the hash code?"*

User: *"42"*

Computer: *"Okie, here you go!"*

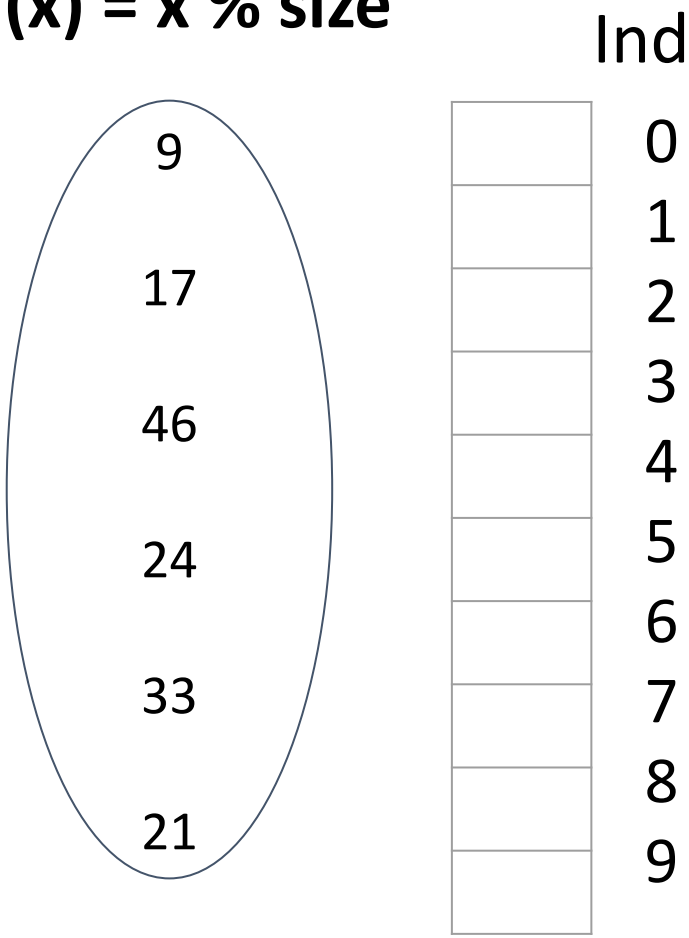
Returns: "What is the meaning of life"

User: *"Why are you so difficult* 😭*"*

What is an Example Hash Function?

Let's say for Integer **x**

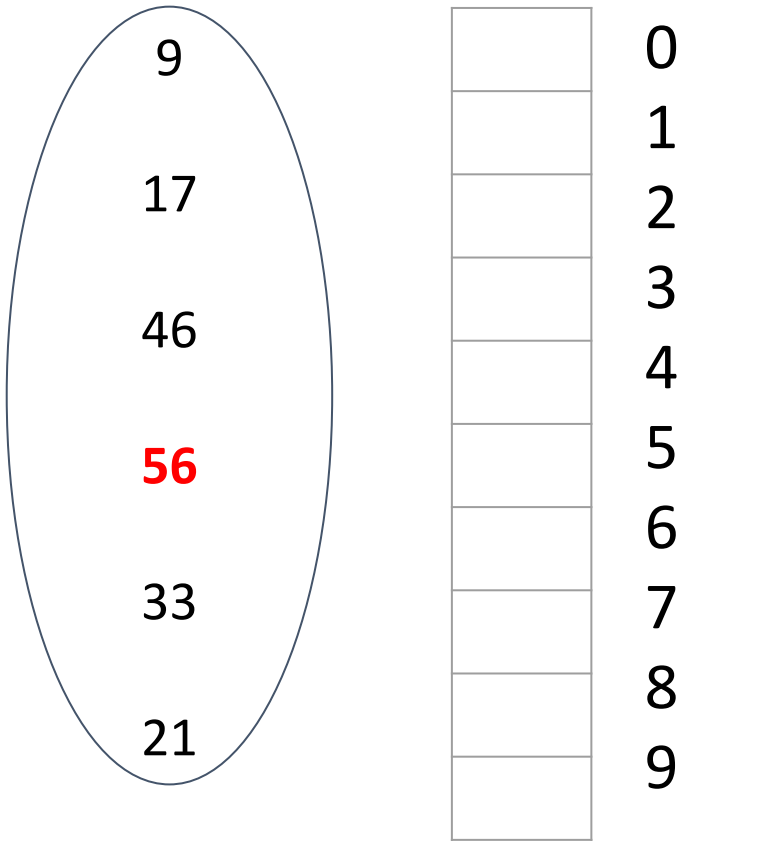
h(x) = x % size



What about this Example?

Let's say for Integer **x**

h(x) = x % size



Lecture Outline

- Announcements
- Hashing
- **Collisions and Resolving Them**
  - Linear Probing
  - Quadratic Probing
  - ⭐Chaining
- Programming HashSet
- Final Remarks

What is Linear Probing?

A way to resolve collisions by adding the element in the next available spot

Regular Hash Function

h(x) = x % size

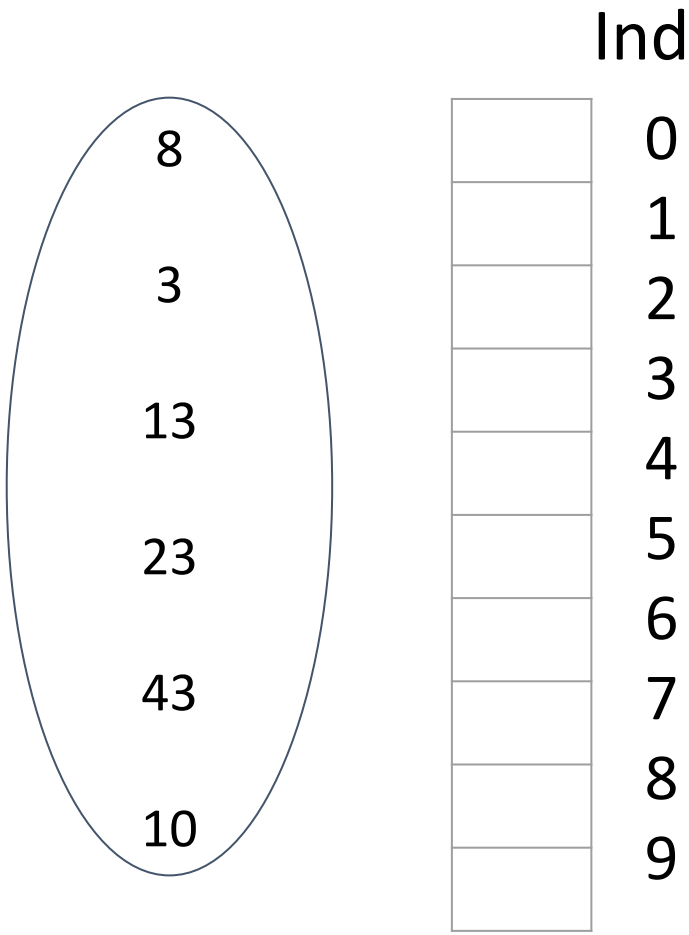
Hash Function (if collision)

h'(x) = [h(x) + f(i)] % size

f(i) = i

What is Linear Probing?

h(x) = x % size **h'(x) = [h(x) + f(i)] % size** **f(i) = i**



What is Quadratic Probing?

A way to resolve collisions by adding the element in the next available spot (quadratically)

Regular Hash Function

h(x) = x % size

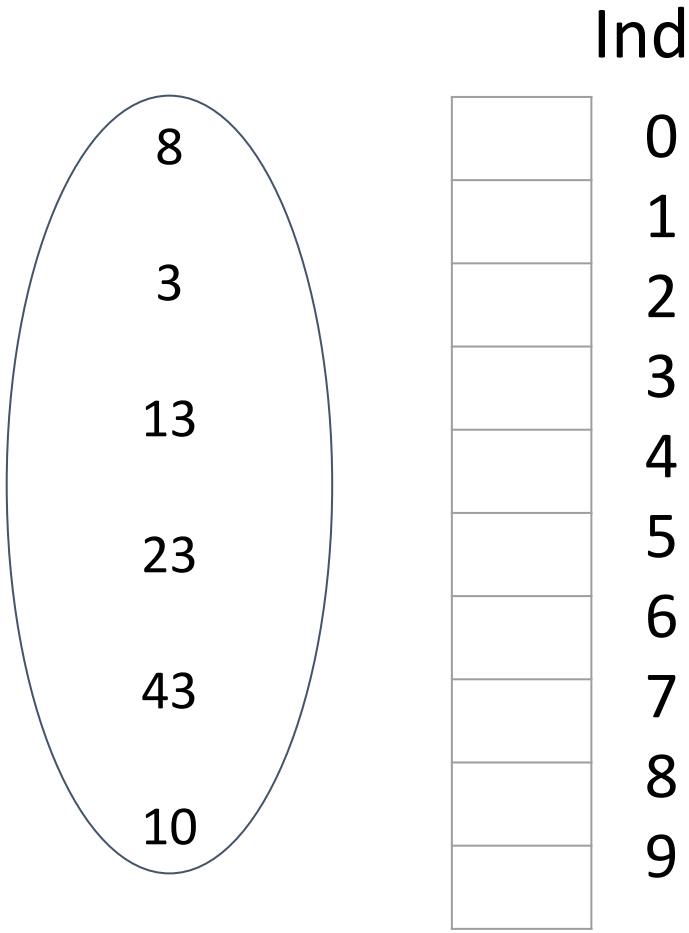
Hash Function (if collision)

h'(x) = [h(x) + f(i)] % size

f(i) = i^2

What is Quadratic Probing?

h(x) = x % size **h'(x) = [h(x) + f(i)] % size** **$f(i) = i^2$**



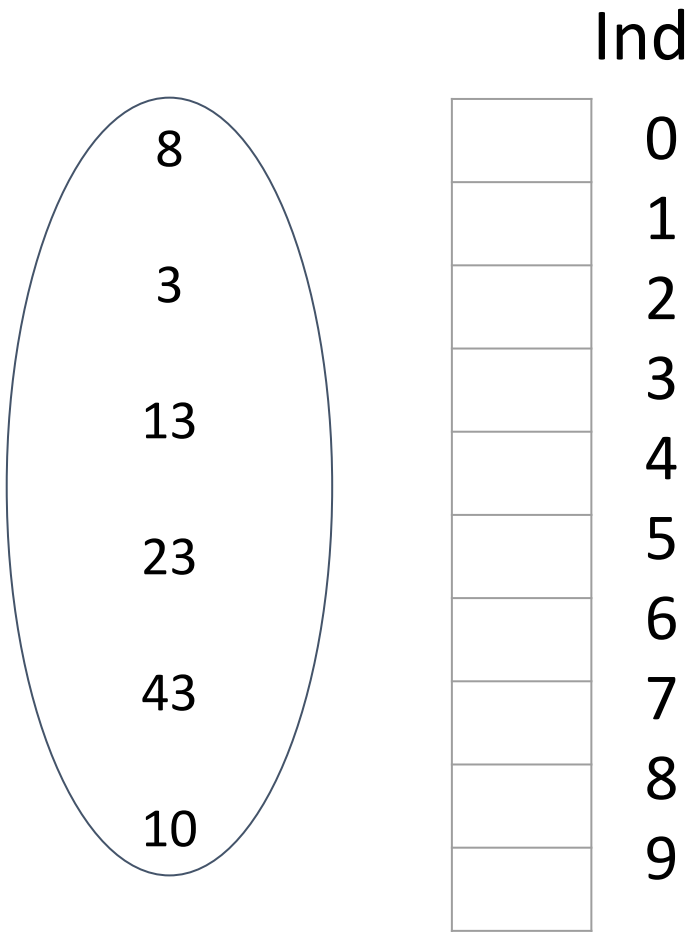
What is ⭐Chaining?

A way to resolve collisions by creating a LinkedList at that Index (also called a “bucket”)

- Combines both features of ArrayList Indexing and the ease of adding values using LinkedLists

What is ⭐Chaining?

h(x) = x % size



Recap (Comparison)

Linear Probing	Quadratic Probing	⭐Chaining
A way to resolve collisions by adding the element in the next available spot	A way to resolve collisions by adding the element in the next available spot (quadratically)	A way to resolve collisions by creating a LinkedList at that Index (also called a “bucket”)

Why ⭐Chaining?

Clustering - A tendency for data to clump together when using solutions to Collisions like Linear and Quadratic probing

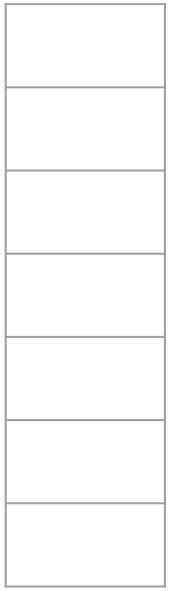
- Linear and Quadratic Probing often result in “Clustering”
- Inefficient use of space in the table
- This means the Runtimes will also be slower

1	2	3	4	5	6				7							8	9		

Why ⭐Chaining?

Operation	ArrayList	LinkedList	HashTable
contains(x)	O(n)	O(n)	**O(1) (average), O(n) (worst)**
add(x)	O(1)*	O(1)	**O(1) (average), O(n) (worst)**
remove(x)	O(n)	O(n)	**O(1) (average), O(n) (worst)**

- Hashing can reduce it down to O(1)
- "Load Factor" - lambda (λ)
  - the number of values in each LinkedList
- Finding the index in the Table is O(1)
- Finding value in LinkedList is O(λ) or essentially O(1)



Lecture Outline

- Announcements
- Hashing
- Collisions and Resolving Them
- **Programming HashSet** 
- Final Remarks

Lecture Outline

- Announcements
- Hashing
- Collisions and Resolving Them
- Programming HashSet
- **Final Remarks**