

# Talent Boost – Data Structures and Algorithms course

*Hash Table, Hash Functions,  
Hash Code*

# Introduction

# What is complexity?

*The amount of resources required to run an algorithm.*

# How many types of complexity we use?

- Time complexity
- Space complexity

# Which are the basic operations supported by most data structures?

- *Create*
- *Search*
- *Delete*

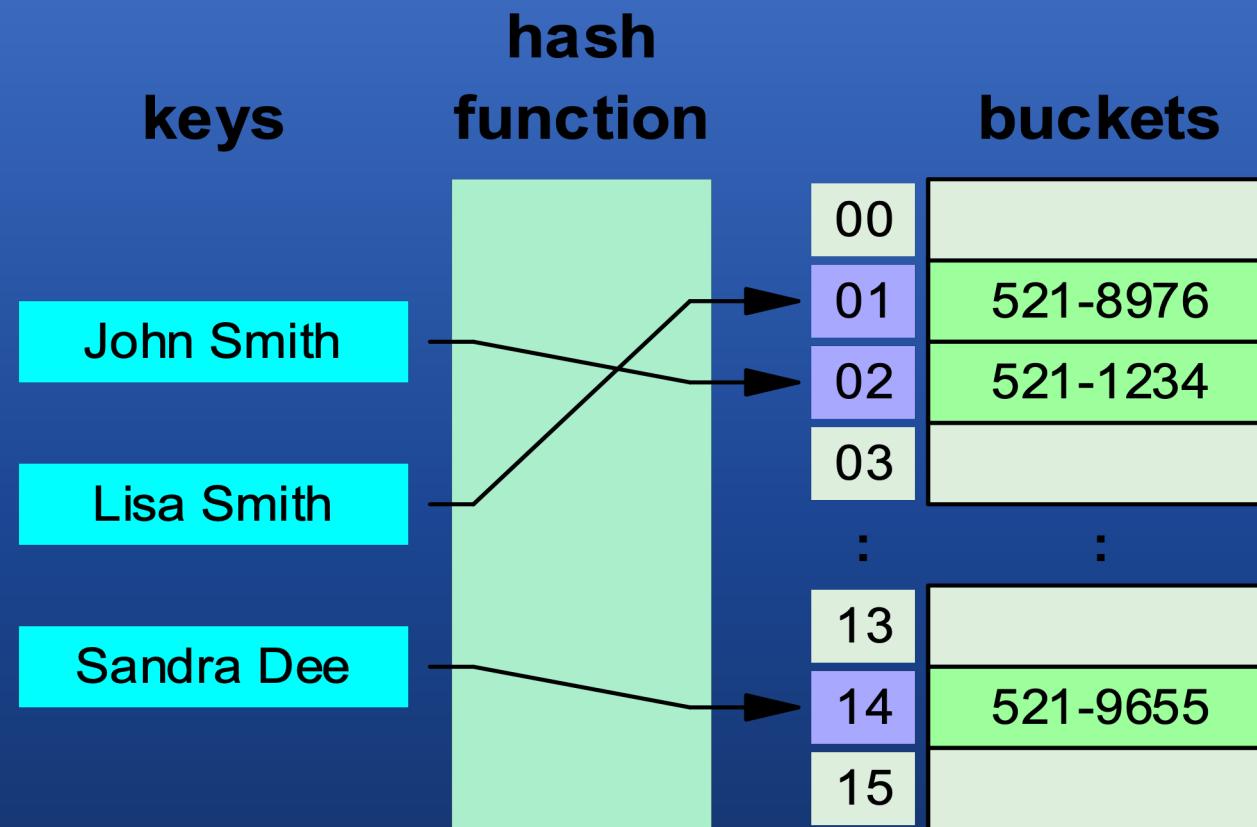
# What is the complexity of ...

- Array
- Linked List
- Self-Balancing Tree

# Can we do better?



# Hash



A close-up photograph of a person's hands interacting with a tablet screen. The person is wearing a light-colored shirt. The tablet has a white frame and is resting on a dark wooden surface. In the background, another person's hands are visible, also interacting with a tablet. The overall scene suggests a technology-themed environment.

There are many things behind  
the term *Hash*

*Hash Function*

*Hash Table*

*Hash code*

# What is Hash Table?

*Data structure with direct access to a desired element, **regardless** of the key type.*

How do we achieve this fast and direct access to a desired element in the table?

*Hash Function* - Function that maps for each element his index in the hash table uniquely using *Hash Code*.

What will happen if two  
elements have the same Hash  
Code?

# Collision Handling

## Open addressing

- Linear probing
- Quadratic probing
- Double Hashing

### Linear Probing Example

Insert (76)	Insert (93)	Insert (40)	Insert (47)	Insert (10)	Insert (55)
$76 \% 7 = 6$	$93 \% 7 = 2$	$40 \% 7 = 5$	$47 \% 7 = 5$	$10 \% 7 = 3$	$55 \% 7 = 6$

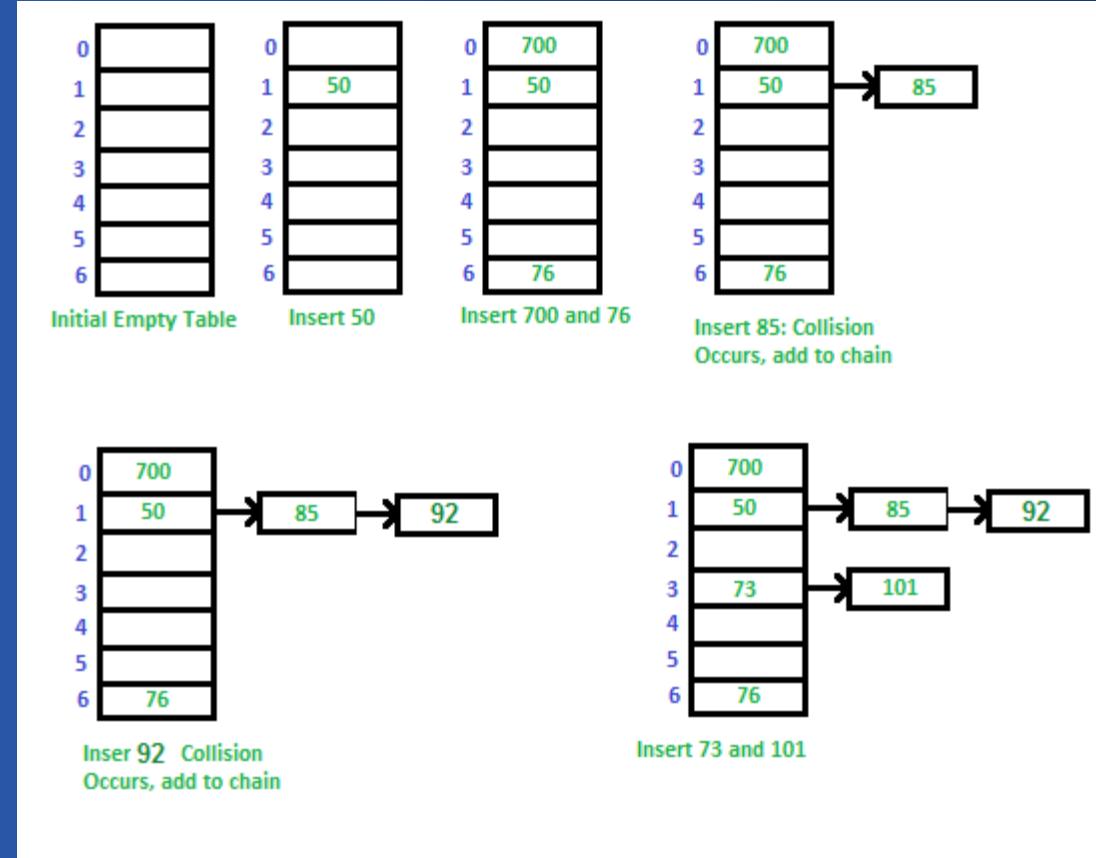
0	0	0	0	0	0
1	1	1	1	1	1
2	93	2	93	2	93
3	3	3	3	3	10
4	4	4	4	4	40
5	5	40	5	40	5
6	76	6	76	6	76

# Collision Handling

## Separate chaining

Implementation using Linked List

Implementation using Self-Balancing Tree



The background of the slide features a high-angle aerial photograph of lush green rice terraces winding through a dense tropical landscape. In the upper left foreground, there is a large, semi-transparent dark blue graphic element containing the word "BREAK".

BREAK

# The Design by Contract

The contract specifies what that component expects of clients and what clients can expect of it.



# hashCode()

*The general contract of hashCode is: Whenever it is invoked on the same object more than once during an execution of a Java application, the hashCode method must consistently return the same integer, provided no information used in equals comparisons on the object is modified.*

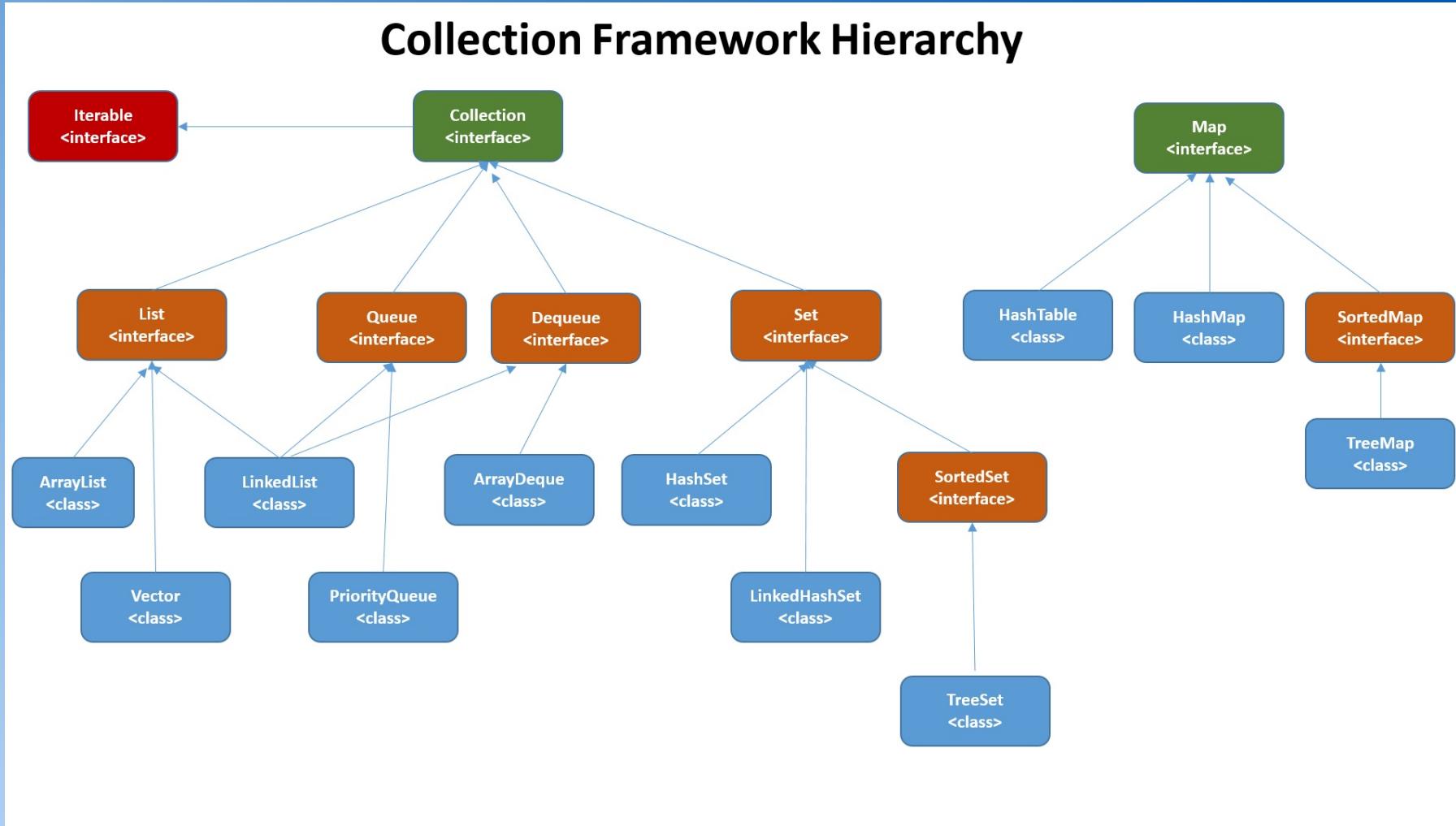
# Equals()

*If two objects are equal according to the equals(Object) method, then calling the hashCode method on each of the two objects must produce the same integer result.*

# Load Factor

The *load factor* is a measure of how full the hash table is allowed to get before its capacity is automatically increased.

# Collections



# Map

## *Hashtable*

- is a legacy class
- doesn't allow any null key or value

## *HashMap*

- store values based on keys
- allows duplicate values
- contains unique keys

## *LinkedHashMap*

- maintains the insertion order

An aerial photograph of a lush green landscape featuring numerous winding rice terraces. The fields are filled with water and planted in different colors of green and yellow. A small cluster of buildings and a colorful umbrella are visible near the top left. The terrain is rugged and mountainous.

BREAK

# Set

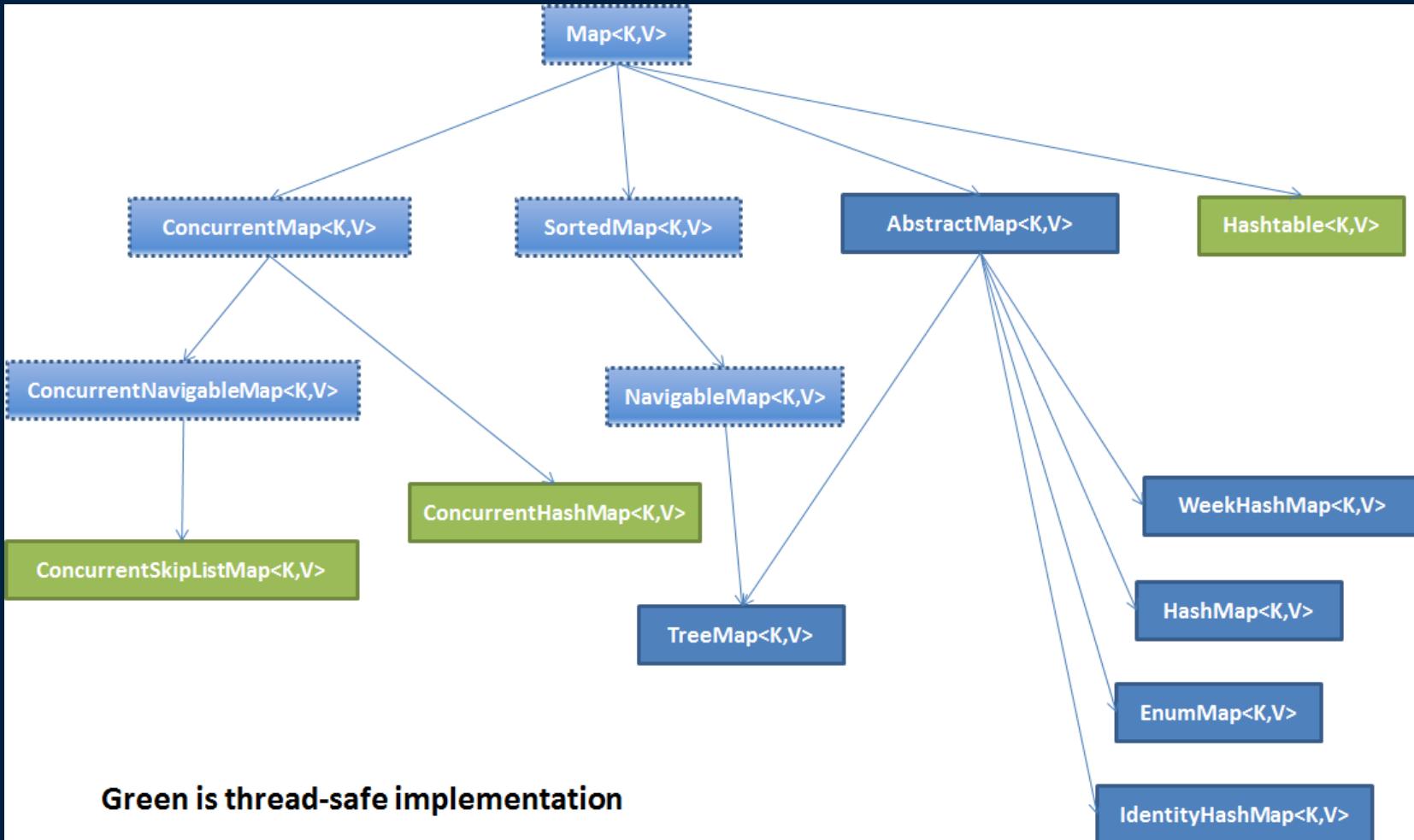
## *HashSet*

- Internally uses data structure like a **hash table** for storage.
- Always contains unique items.

## *LinkedHashSet*

- Store items in LinkedList.
- Maintains the insertion order.
- Always contains unique items.

# And many more ..



The background of the slide is a high-angle aerial photograph of a lush, green landscape. It features numerous winding, terraced fields, likely rice paddies, which are filled with water and show distinct patterns of cultivation. Interspersed among the fields are clusters of tropical trees, including palm trees, and small buildings or huts. A paved path or road cuts through the lower part of the scene. The overall color palette is dominated by various shades of green and blue.

BREAK

# Summary

There is no optimal data structure for every problem!

We should ask ourselves:

- Which is the most common operation that we are going to perform with that data structure?
- Memory limit

Even for the Hash Table, there are different types of implementation, depending on the problem.

# Resources

- <https://javapapers.com/core-java/java-hashtable/>
- <https://www.geeksforgeeks.org/hashing-data-structure/>
- <https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html>
- <https://docs.oracle.com/javase/8/docs/api/java/util/Hashtable.html>
- <https://facingissuesonit.com/2019/10/15/java-collection-framework-hierarchy/>
- <https://www.interviewcake.com/concept/java/lru-cache>





# Thank You