Week 3 Day 2

Advanced Java Topics

Maps

REVATURE

Maps store values in key/value pairs

- Not technically part of the Collections API because it doesn't implement Iterable
- Can still obtain iterators for its values
  - .entrySet(), .keySet(), .values()

Maps: HashSet

- Elements stored as key/value pairs
- Insertion and retrieval is fast
- Does not maintain insertion order
- Null keys and values

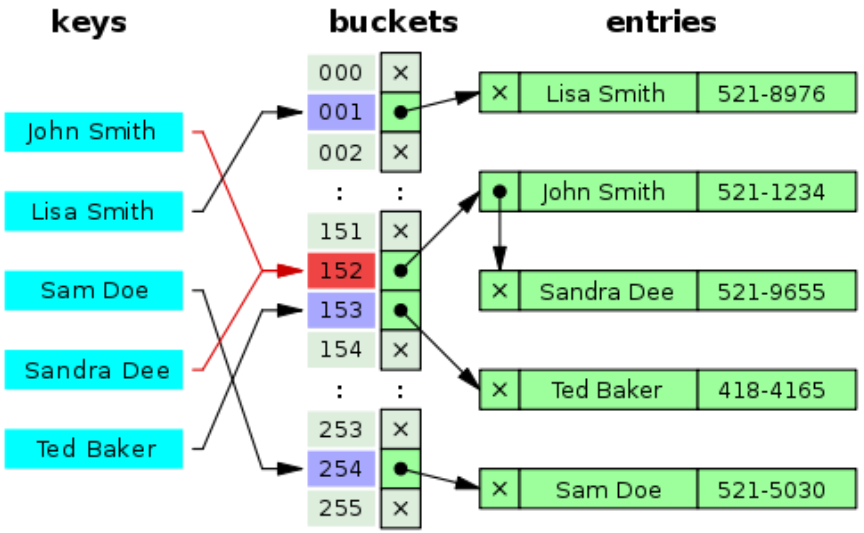


Image Source: https://javahungry.blogspot.com/2013/08/hashing-how-hash-map-works-in-java-or.html

Maps: TreeMap and HashTable

REVATURE

- TreeMap
- Keys stored in a sorted tree
- Insertion and retrieval slower
- No null keys
  - Null values allowed

- HashTable
- Older threadsafe implementation of HashMap
- No null keys
- No null values

Lists: Vector and Stack

REVATURE

- Vector is an older implementation of an arraylist
  - Doubles in size when the array is resized
- Stack is an older implementation of the stack datastructure
  - First in last out
  - ArrayDeque is now preferred

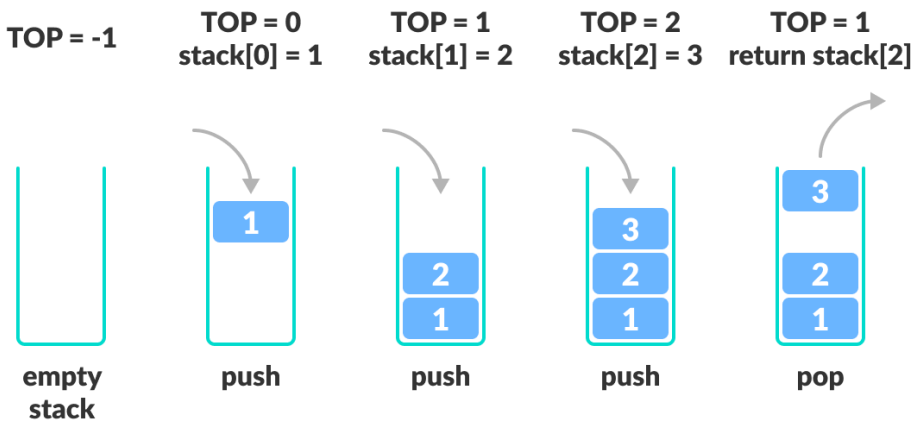


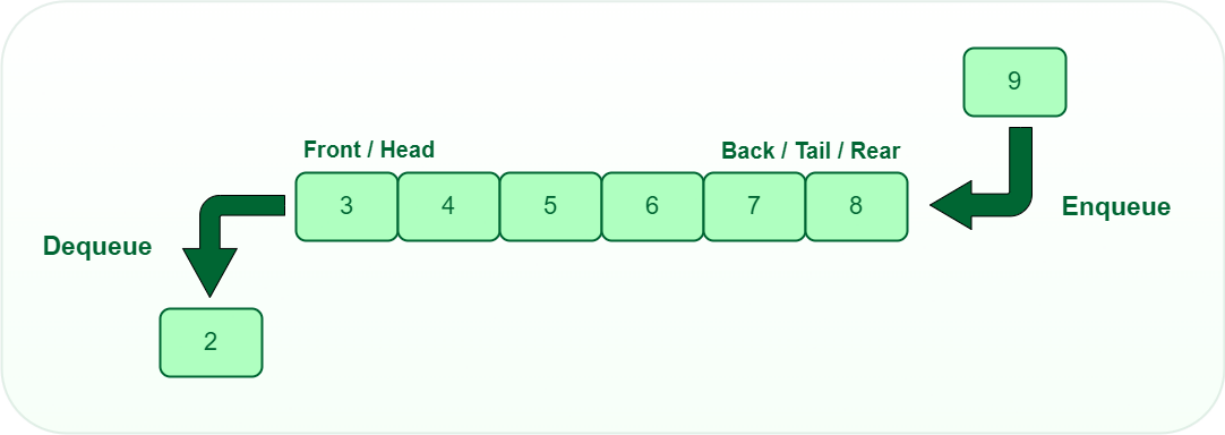
Image Source: https://www.programiz.com/dsa/stack

Queues: ArrayDeque and PriorityQueue

REVATURE

- ArrayDeque
- Double ended queue
- Can be used as a queue or stack
- Items stored in a resizable array

- PriorityQueue
- Orders elements by their natural order
- Uses a comparator for natural ordering



Comparable vs Comparator

REVATURE

Comparable Interface

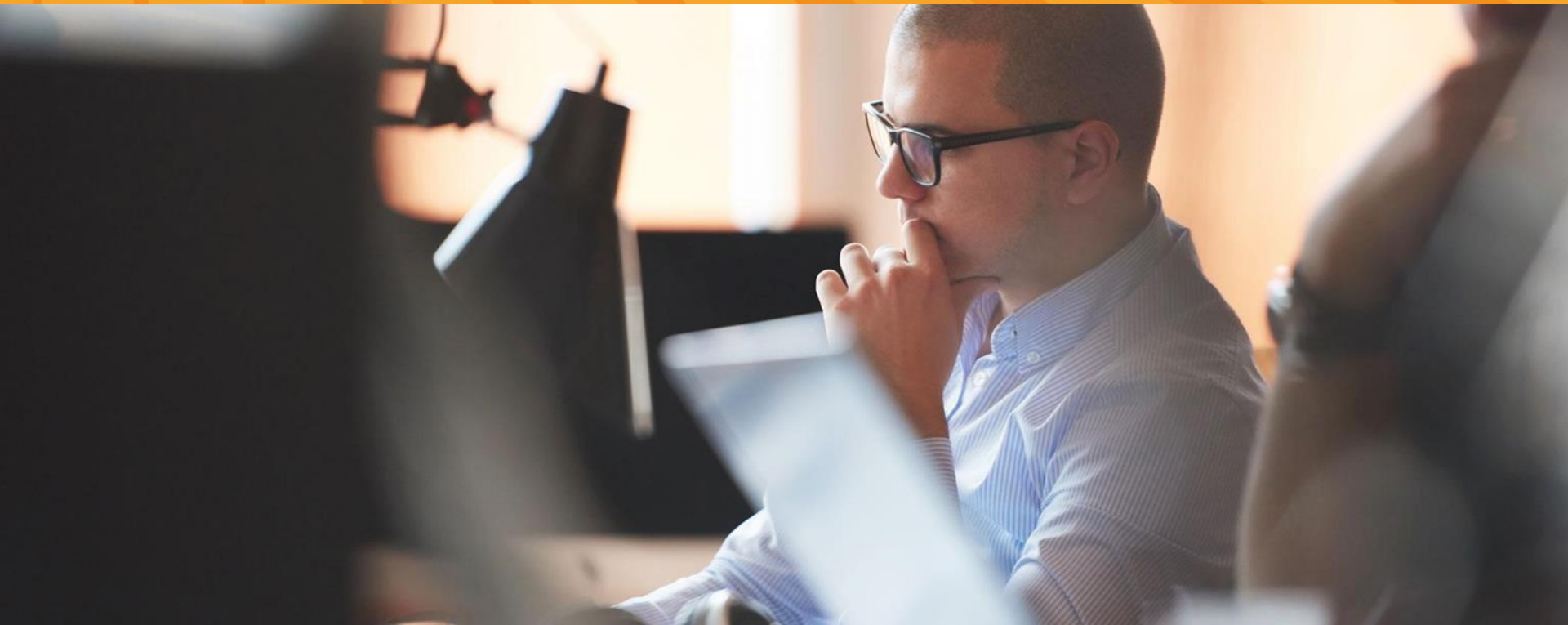
- Defines the natural ordering for the class
- Implement Comparable
- Override compareTo()

Comparator Interface

- Defines total ordering on some collection of objects
- Implement Comparator
- Override compare()

REVATURE

Comparator vs Comparable DEMO



Java 8 Features

REVATURE

- Functional Interfaces
- Lambda Expressions
- Stream API
- Reflections API
- Date and Time API
- Optional Classes
- Predicates

Java 8: Lamda and Functional Interfaces

REVATURE

Functional interfaces are interfaces with only one method

- Implicitly created by lambda functions
- Can be explicitly created for use by lambdas

Lamdas

- Allow for the creation of disembodied methods
  - parameters(s) -> expression

Java 8: Date and Time API

REVATURE

Unified date and time package including:

- LocalDate
- LocalTime
- LocalDateTime
- ZonedDateTime
- Period
- Duration
- DateTimeFormatter

Java 8: Optional Class

REVATURE

- Introduced to reduce null checking
  - Object could optionally have a value or be empty

```
public class OptionalExample {
  public Optional<String>
  getAmbiguousString(boolean b) {
    if (true) {
      return Optional.of("awesome
      string!");
    } else {
      return Optional.empty();
    }
  }
  public static void main(String[] args)
  {
    Optional<String> optString =
    getAmbiguousString(false);
    String theString = optString.orElse
    (""); // specify a fallback value
    System.out.println(theString);
    // we can use the String without
    fear of NullPointerException now
  }
```

Java 8: Streams

REVATURE

- Introduced more functional programming to java
  - Operate on a stream of elements
- An abstraction which are lazily loaded and do not modify the source
  - Do not store data, simply transform
- Two types of streams
  - Intermediate
  - Terminal

REVATURE

Java 8 Features DEMO