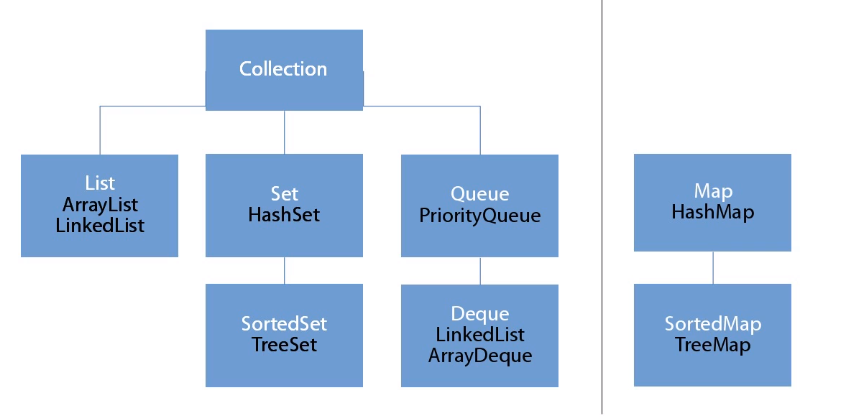
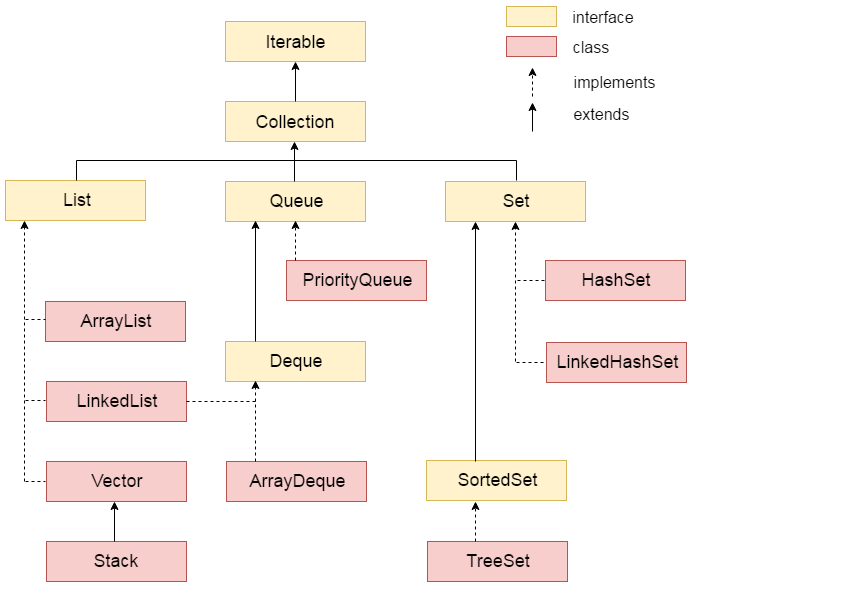
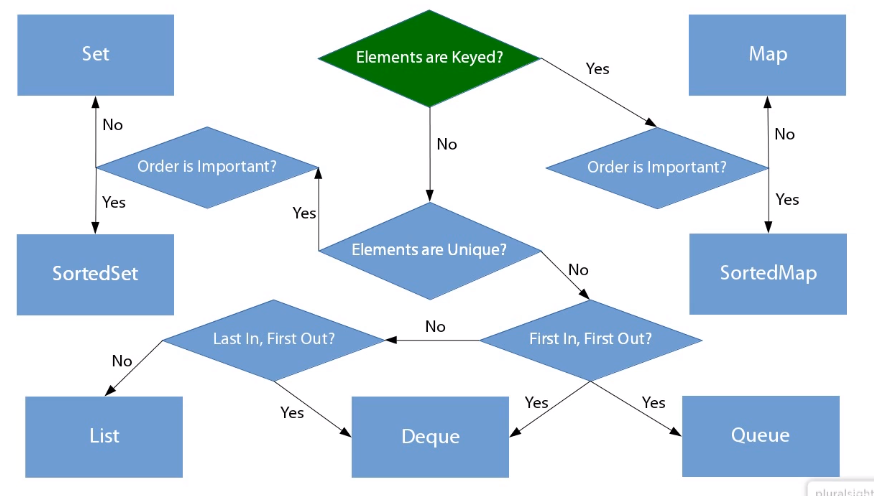
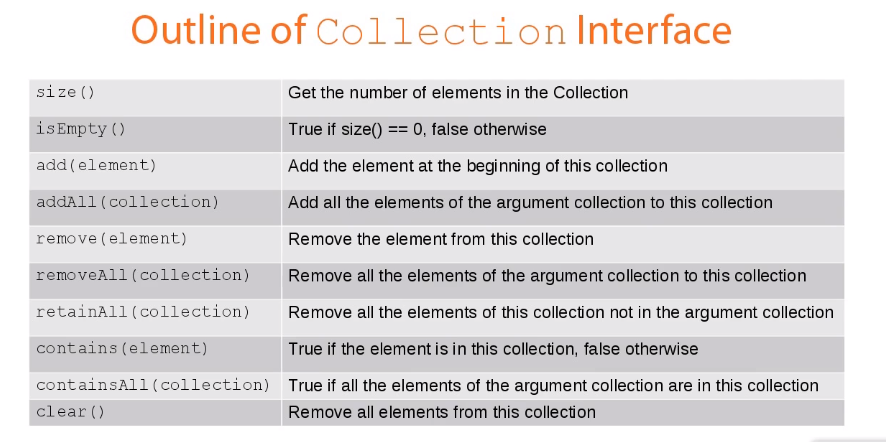
**JAVA FUNDAMENTALS: COLLECTIONS**

**Defining and iterating collections**

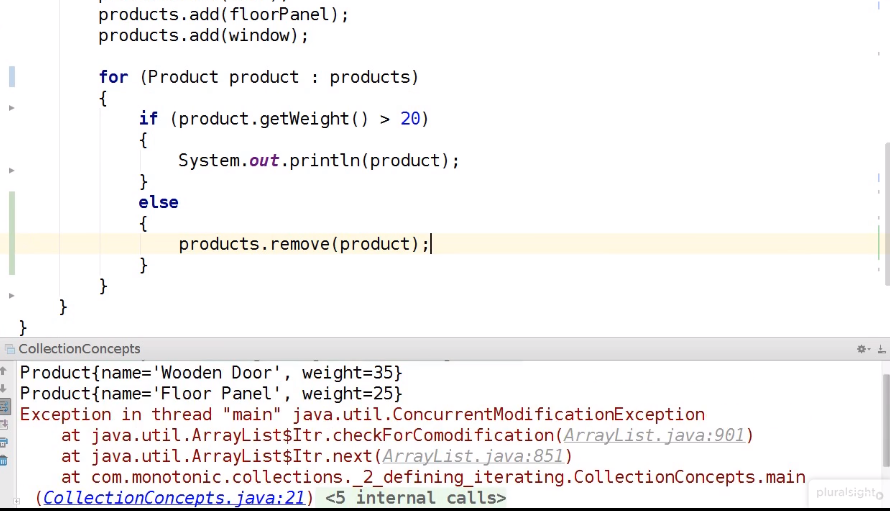
* Collection of collections



* using collection approriately
* Collection Behaviors

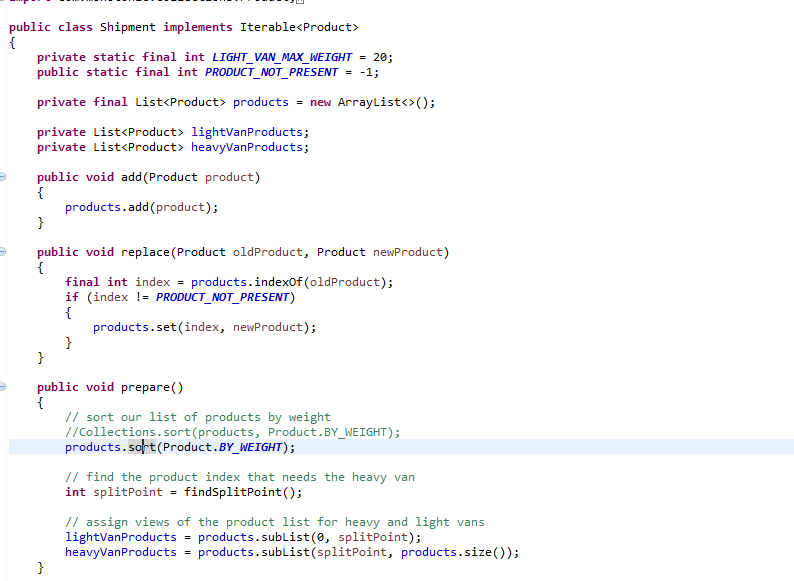


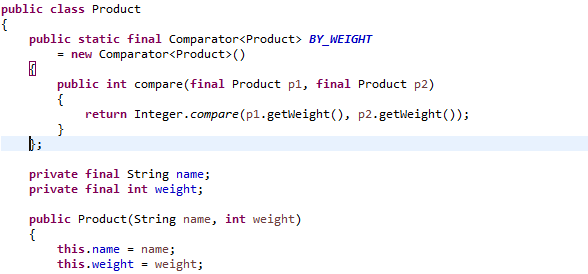
* When looping over a collection with a for loop, do not modify it (when want to modify it use iterator.)



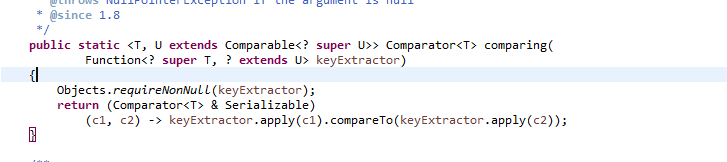
**Collections with iteration order: Lists**

* Sort object by attribute(Java Fundamental Collection demo 03)



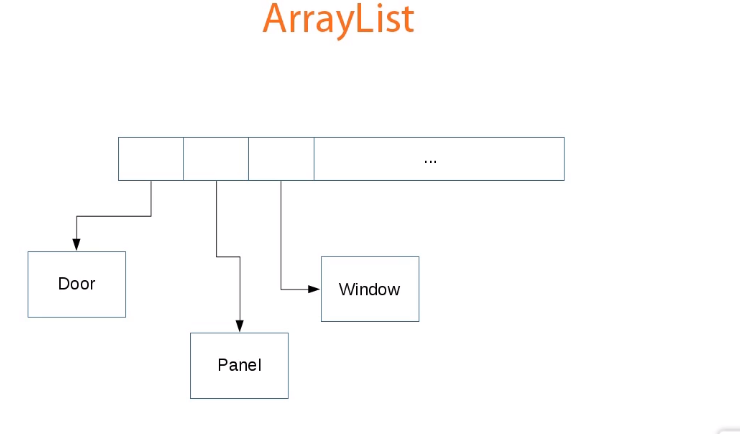
 

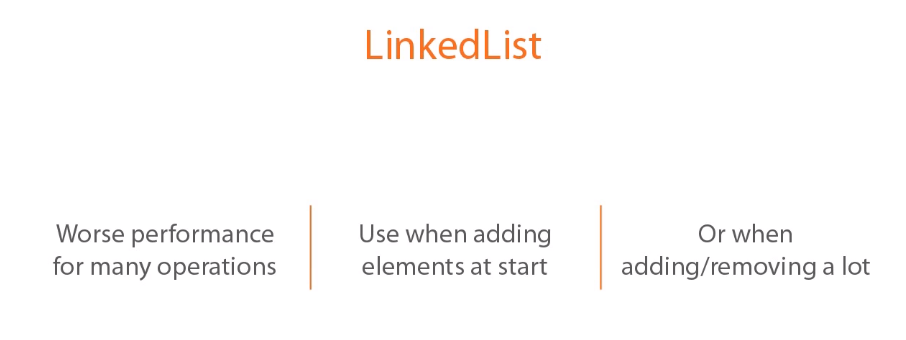


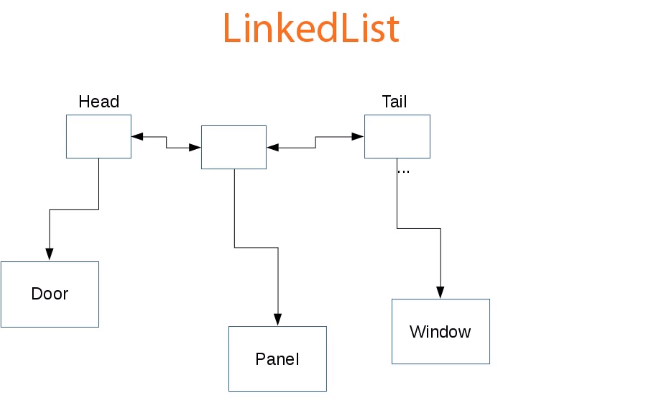


* Result sort ascending product by weight
* List implementations





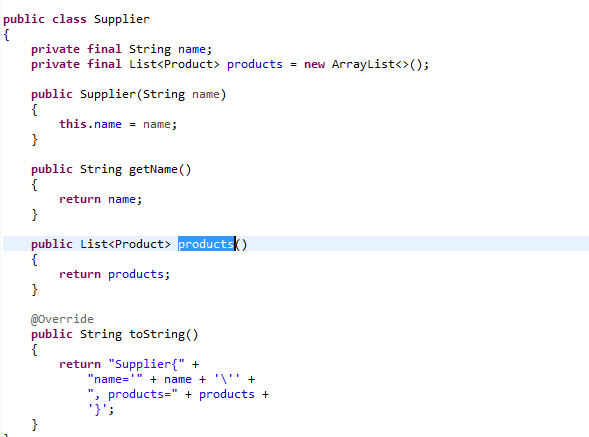


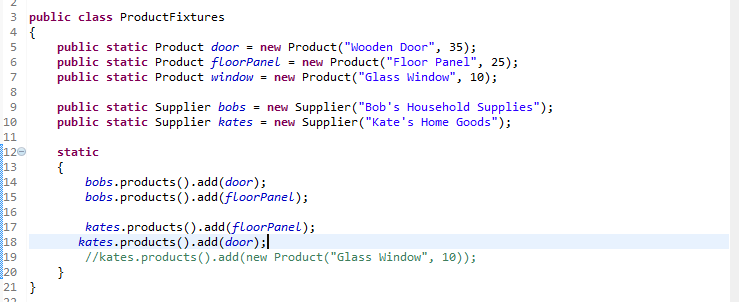


**Collection with uniqueness: sets**

* Set are collections of distinct elements. There no duplicates.
* Product colotuge demo(04)

Add product from suppler into catalogue, but just one entry if mulltibe suppler prove one product





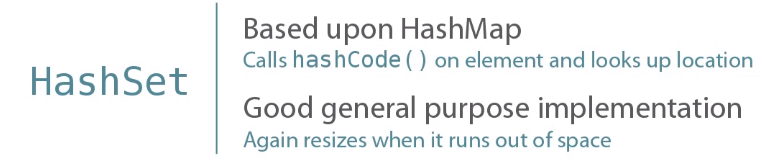
Bobs, kates add list

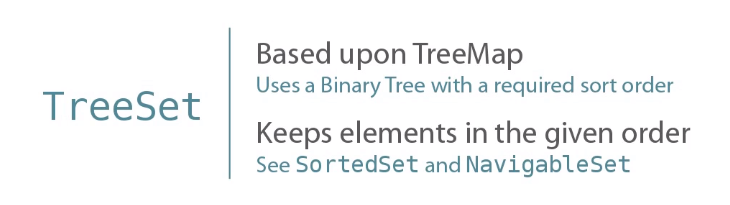


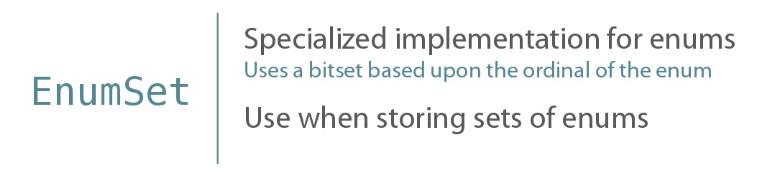
Usint set to entry one if duplication

Get list from supplier and add into Set

* Set implementation







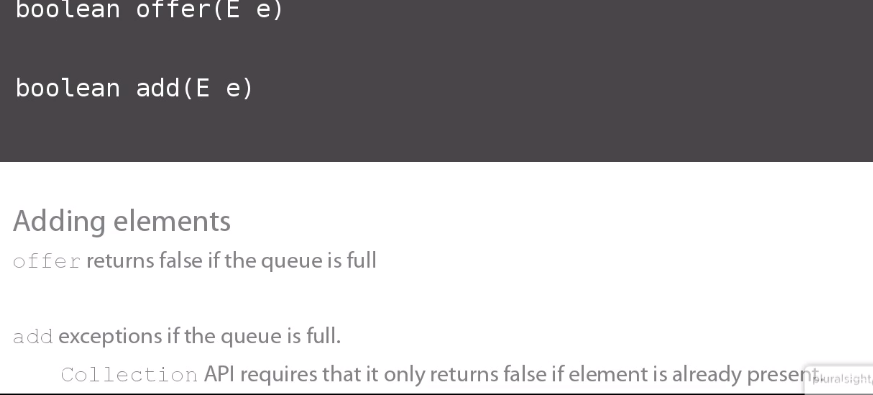
**Collections with Modification Order: queues, deques, and Stacks**

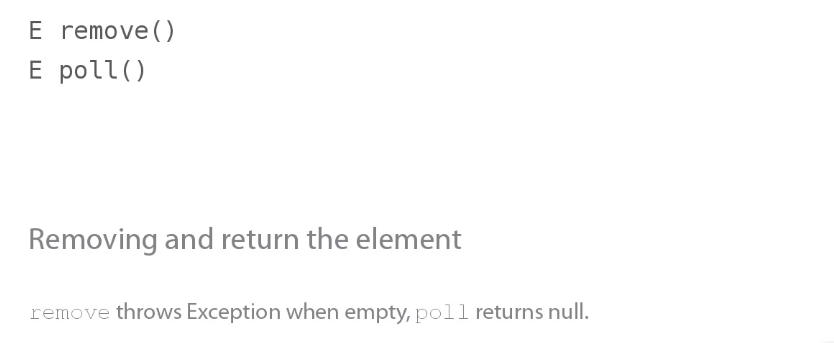
* Queues are a first in, first out data structure

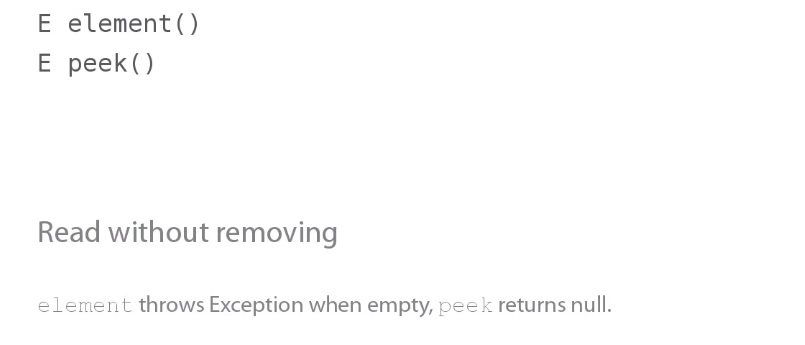
remove method that pull the enquiries off one by one

poll method

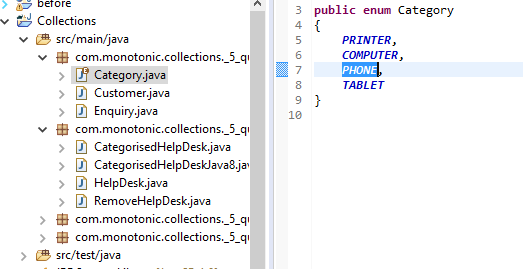
* Queues(05)

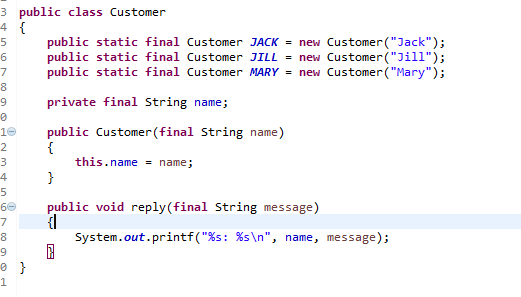
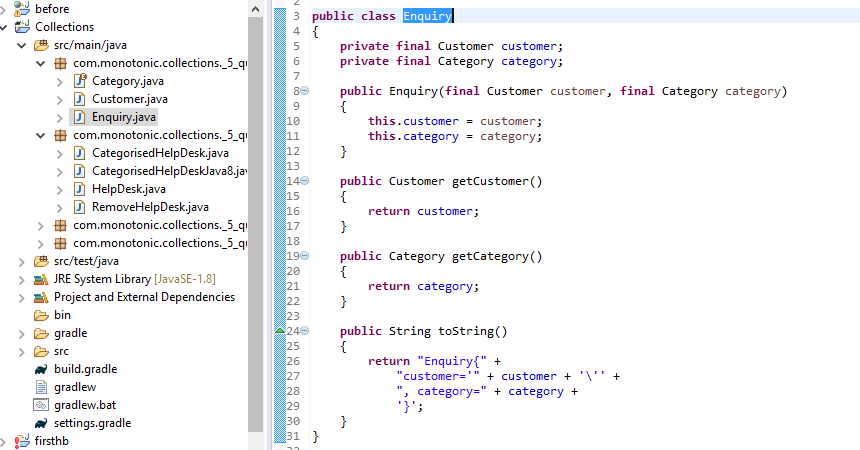


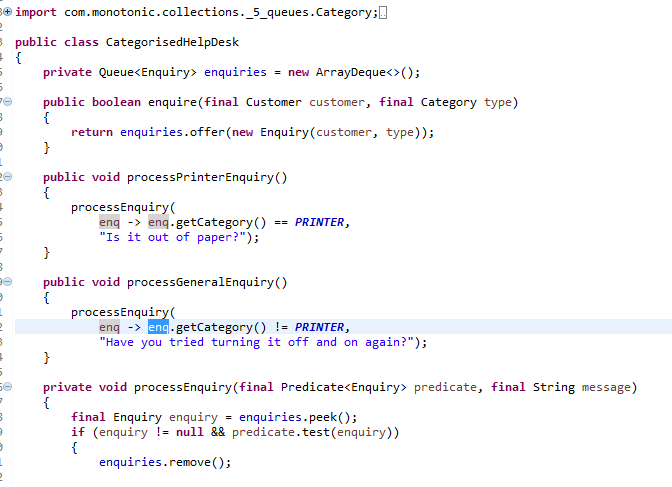


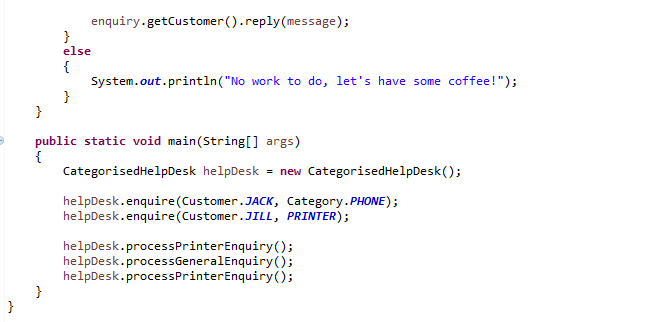


Demo

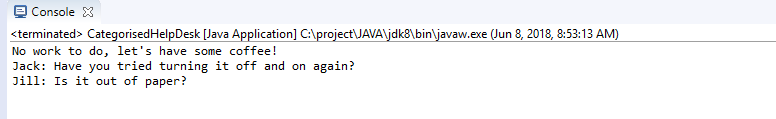






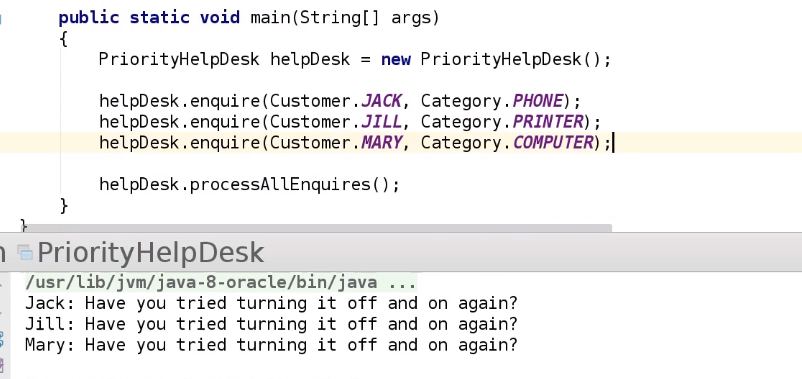


Result

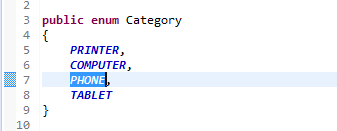


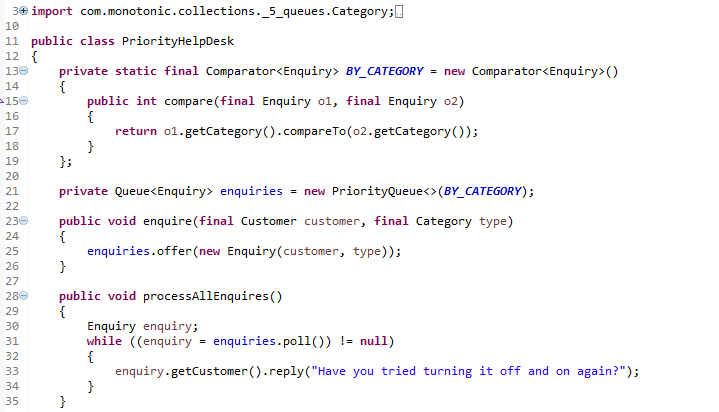
* Priority queues(05)
* First in, first out

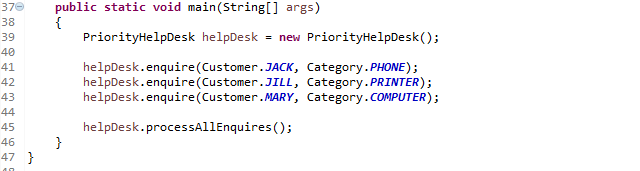
Normal



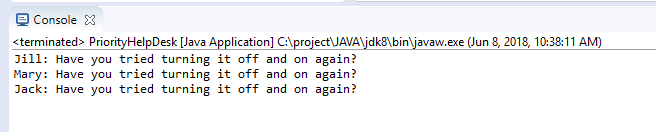
Using priority queues







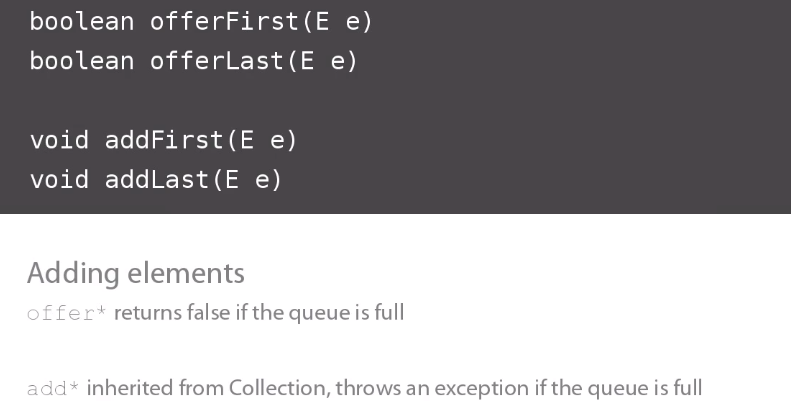
Result

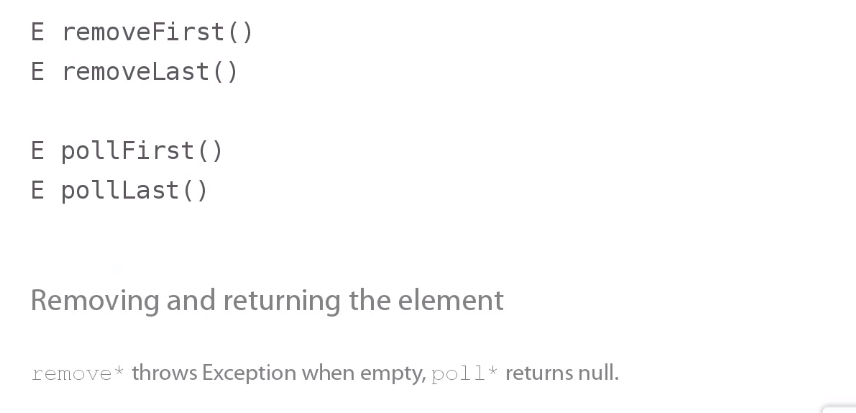


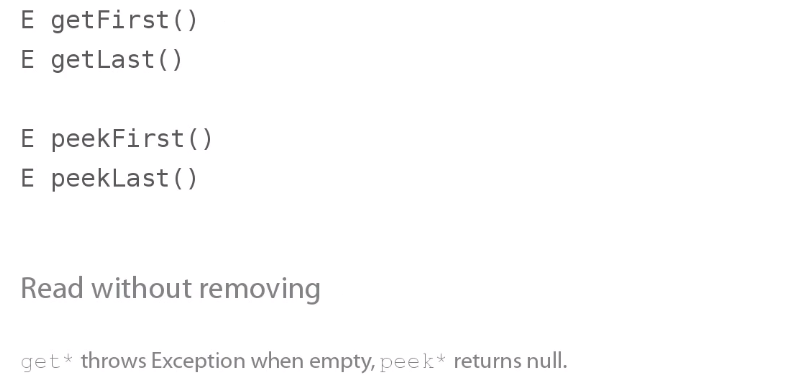
Print according to class Category

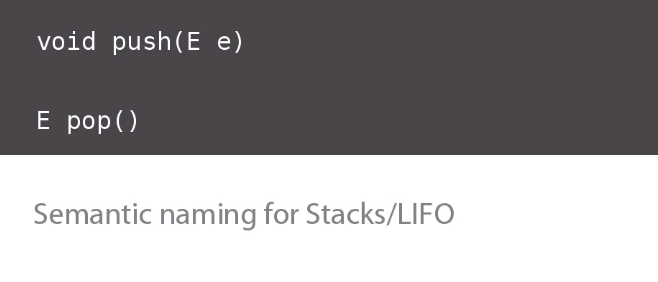
* Stacks and Deques

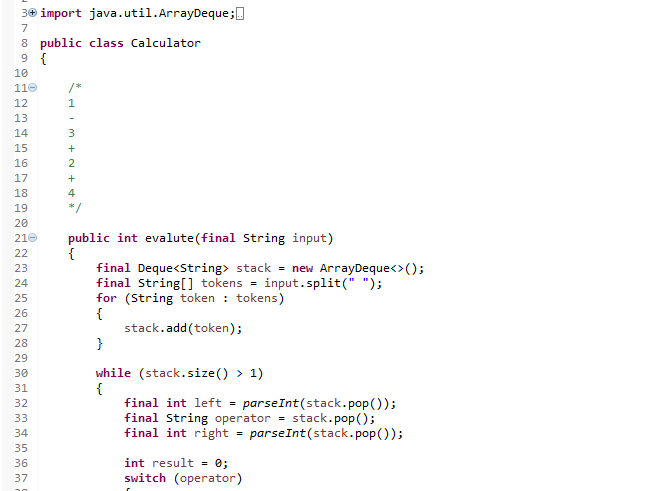
Last in ,first out

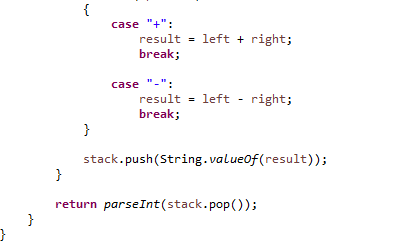








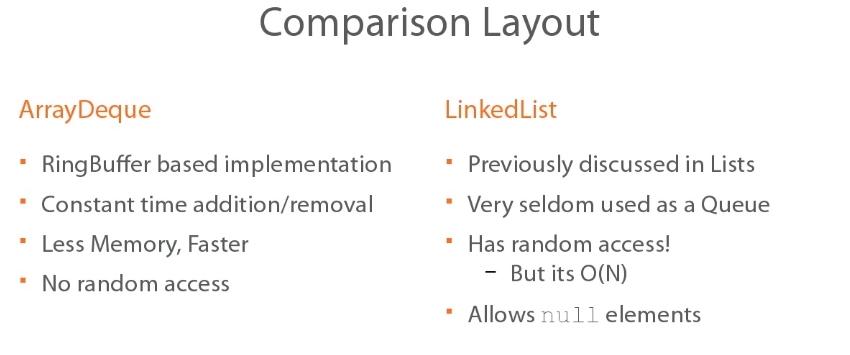




Convert result to int

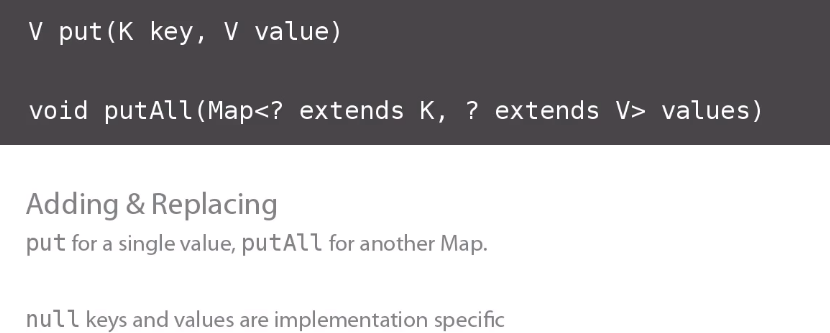
Convert result to String and push on top

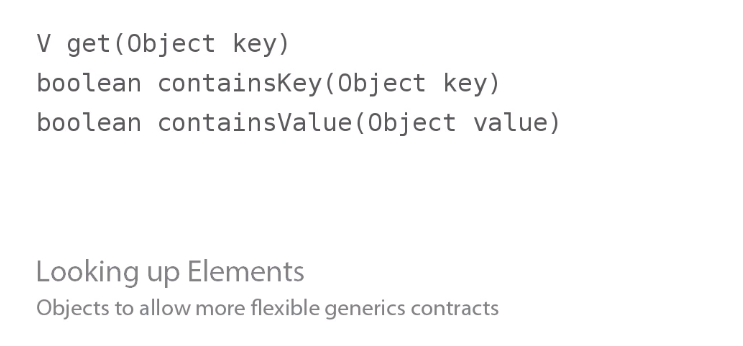
* Implementation and summary



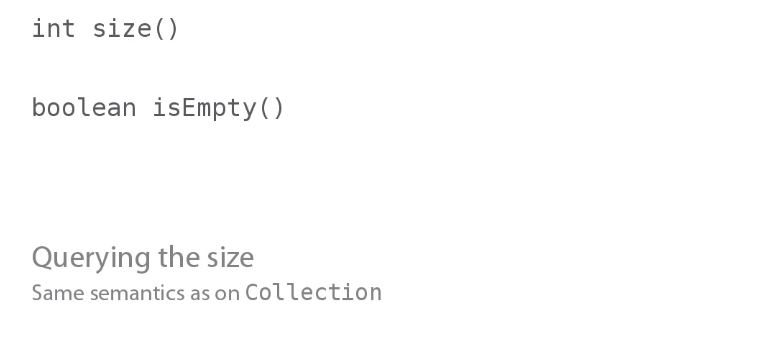
**Collections of Pairs: Maps**

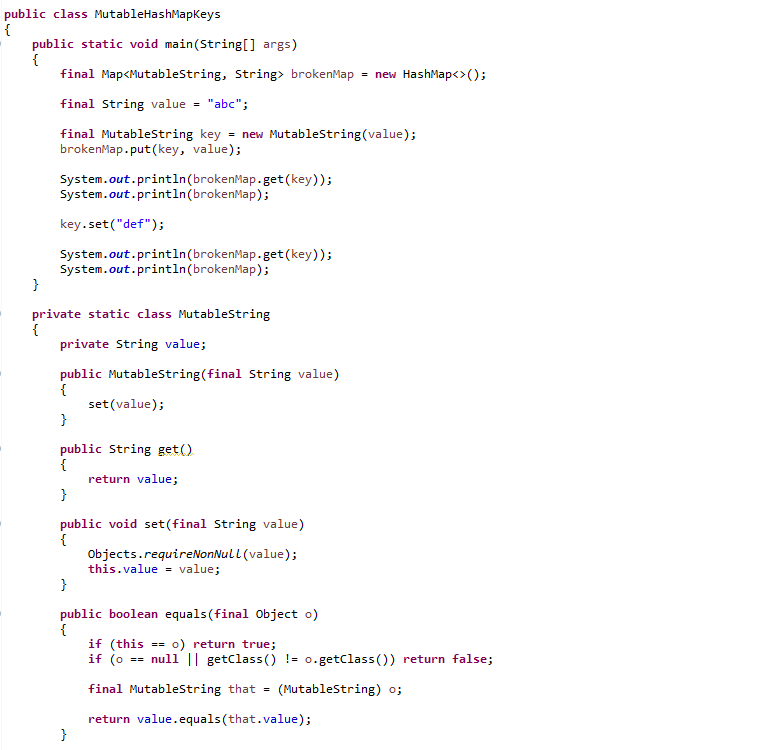
* Map

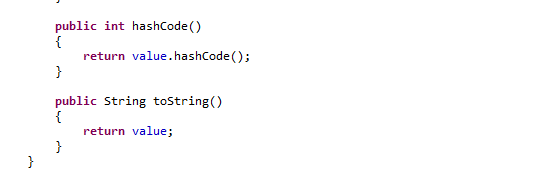






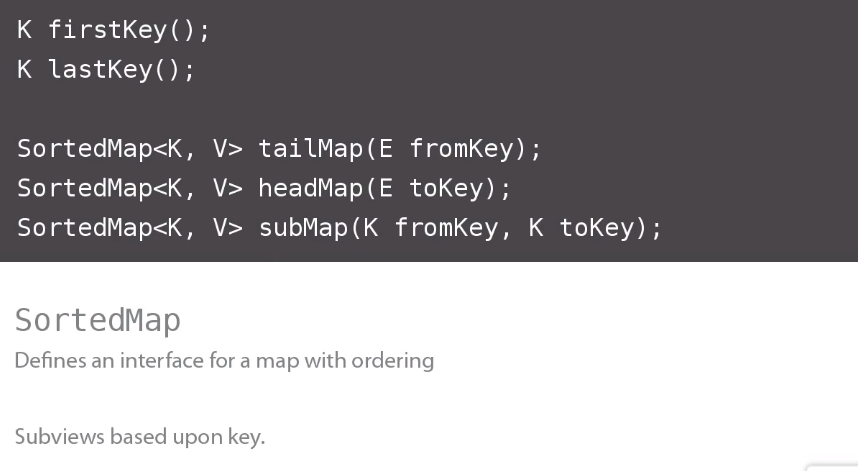


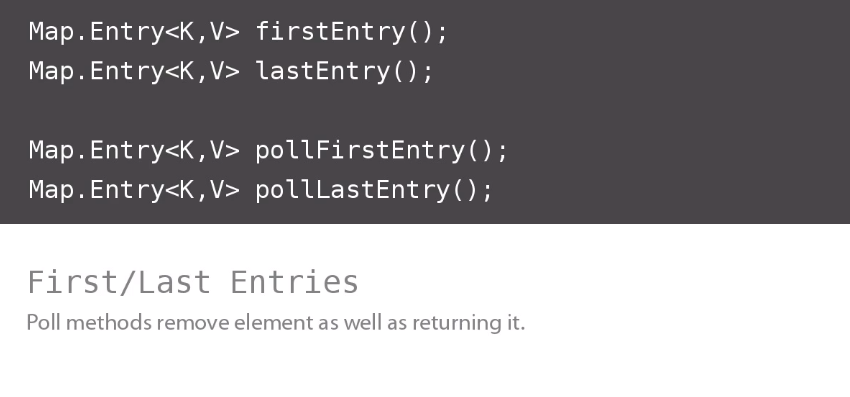


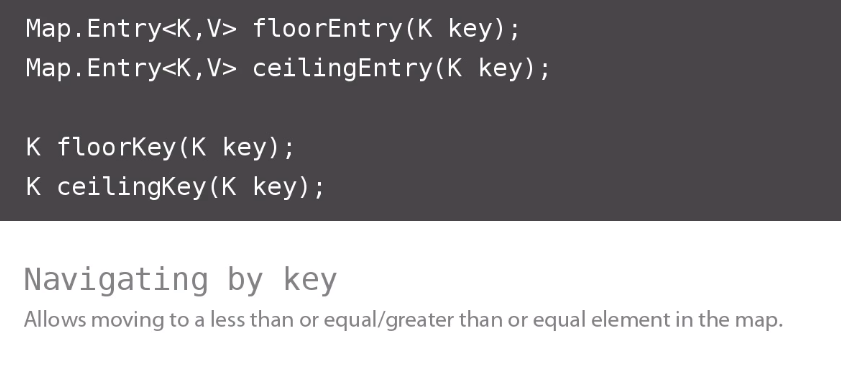


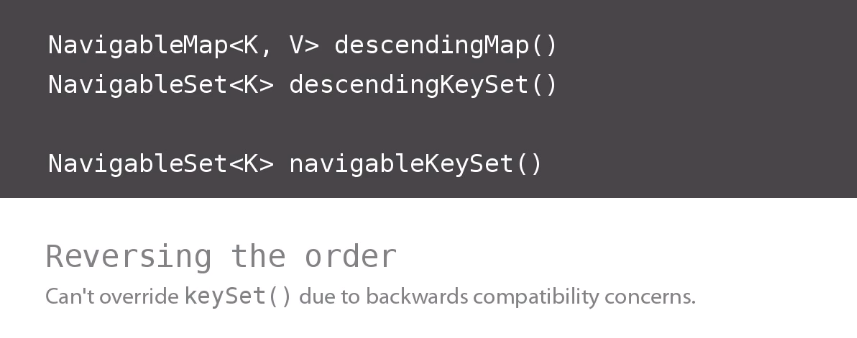
Result

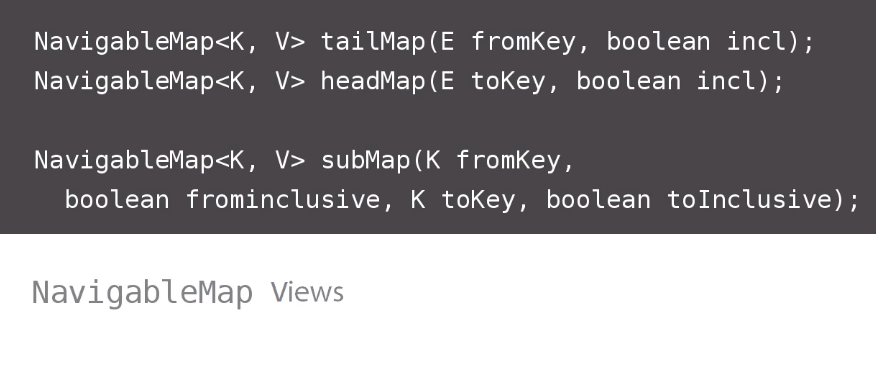
* Sorted and Navigable Maps

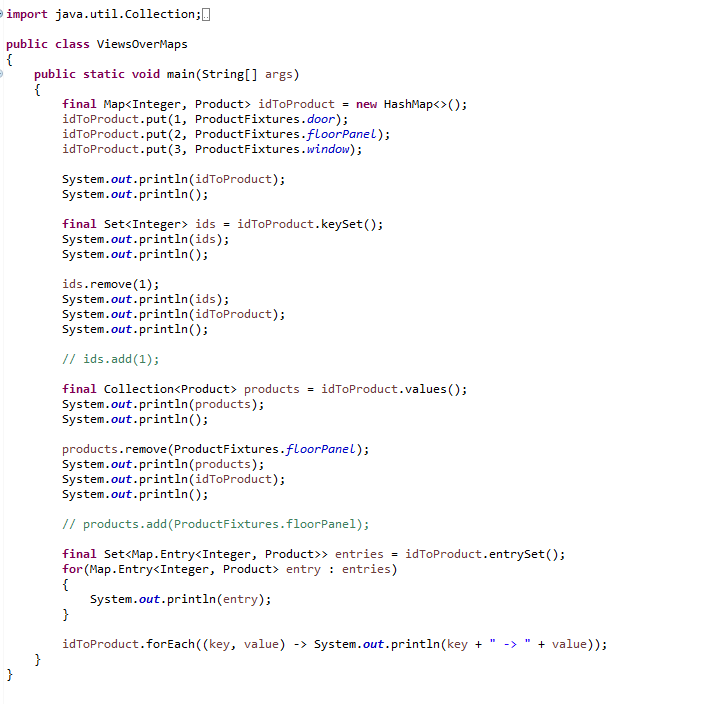


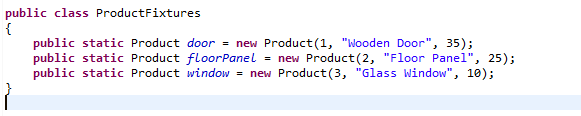




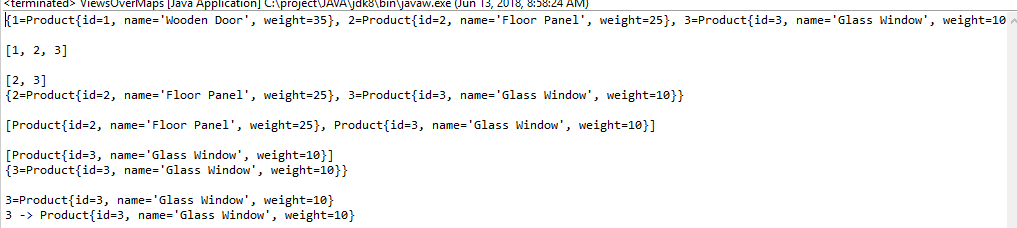




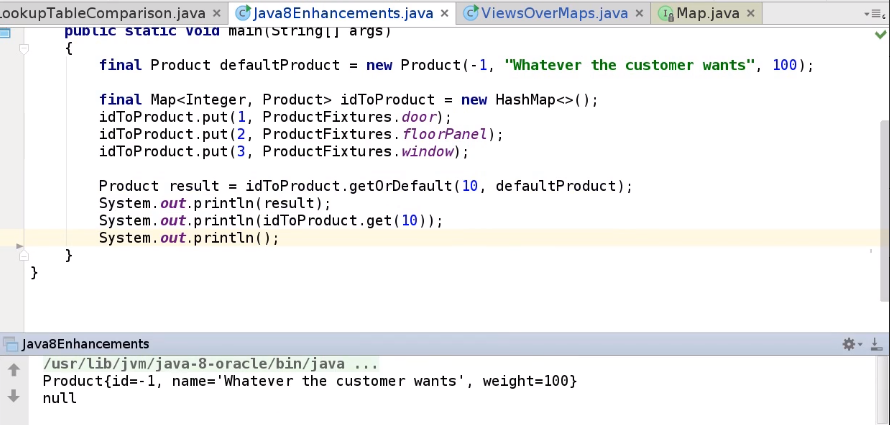




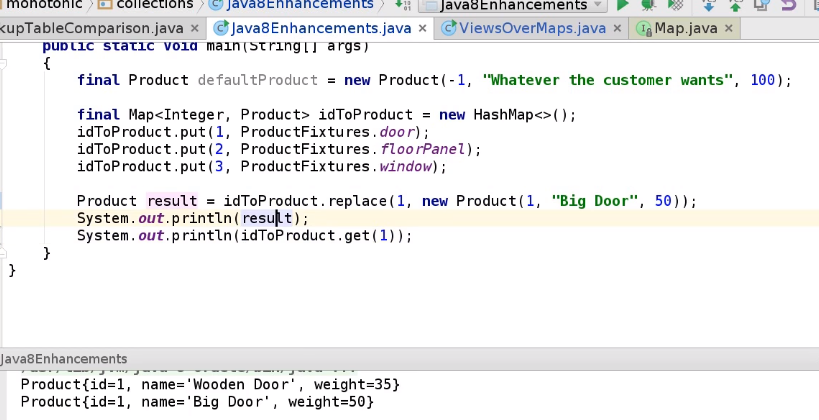
Result



* JAVA 8 enhancement
* Default value



* Update value

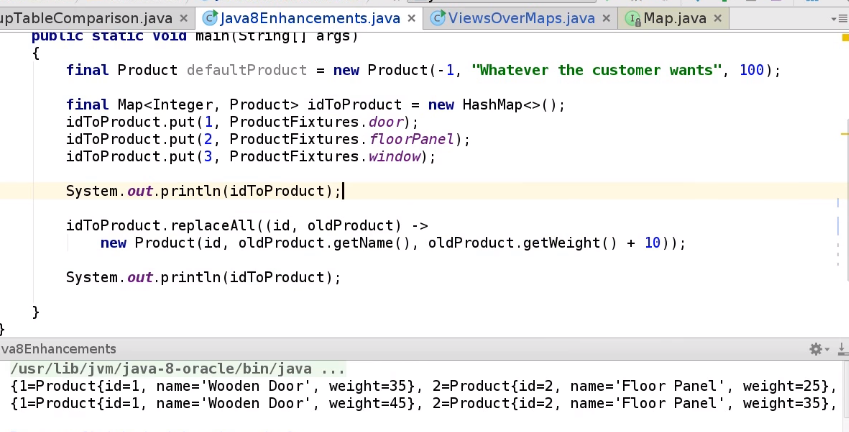


Reulst is old value

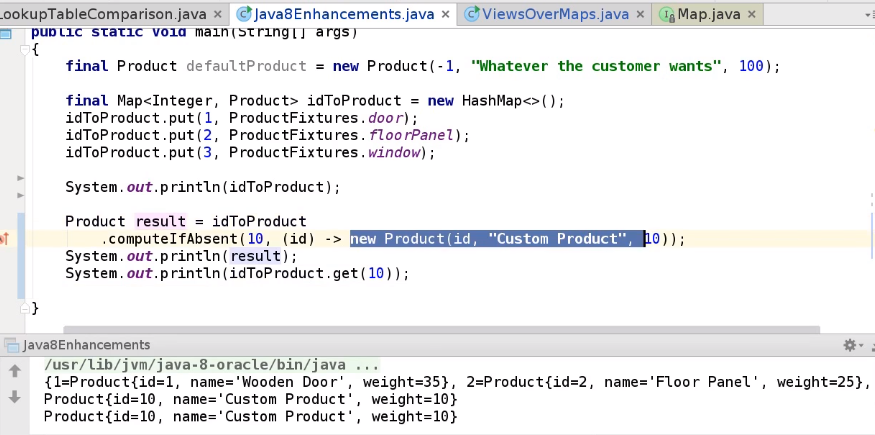
* won’t update value if key don’t exist



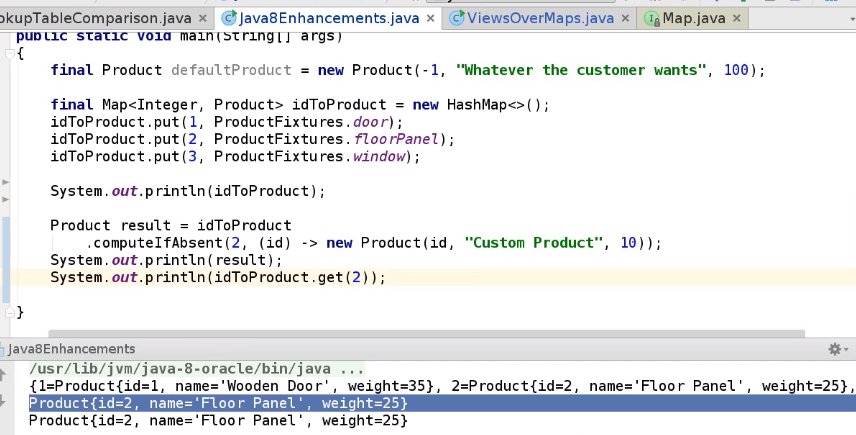
* replace all product



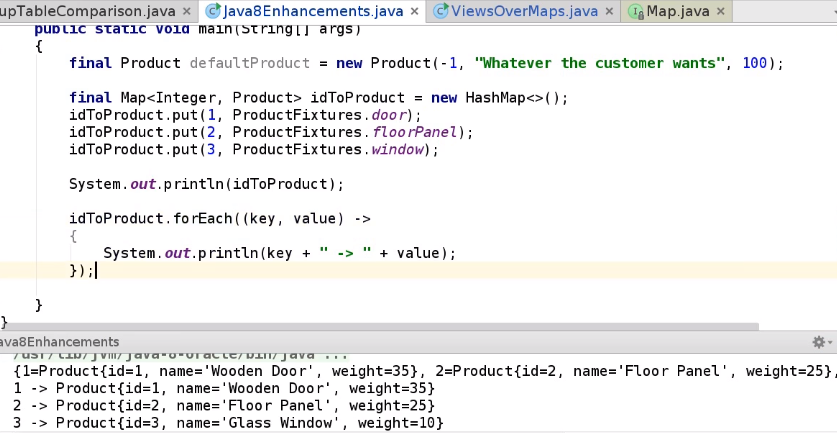
* method computeIfAbsent()



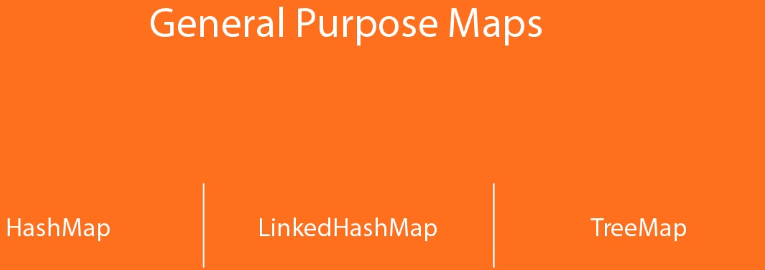
Just when missing element

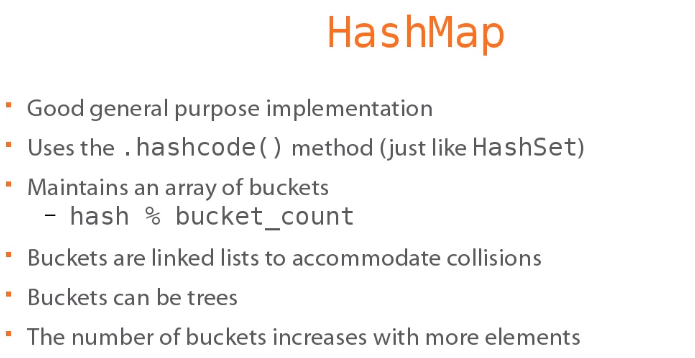


* loop and prind value



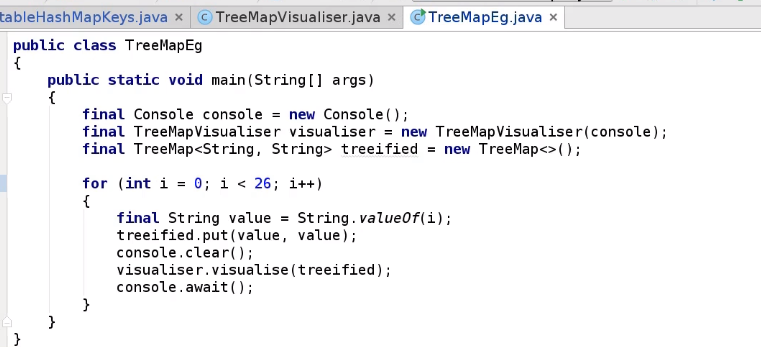
* implementations



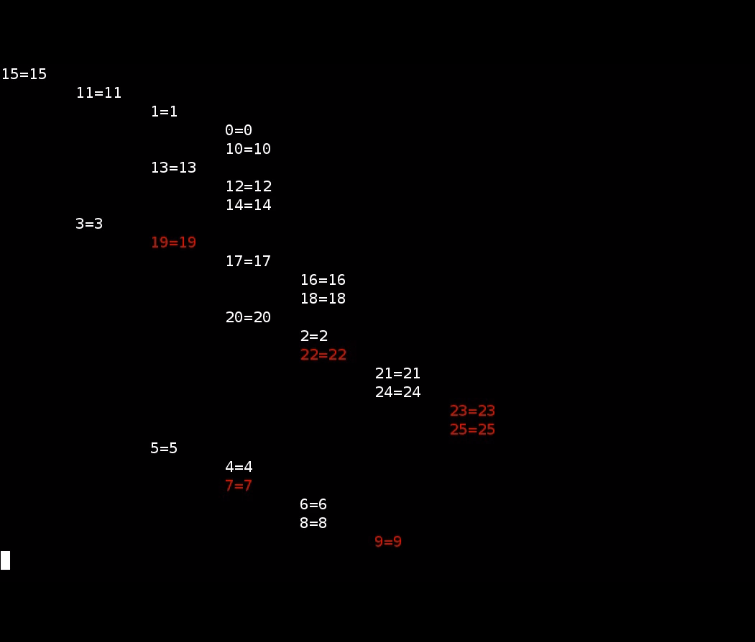


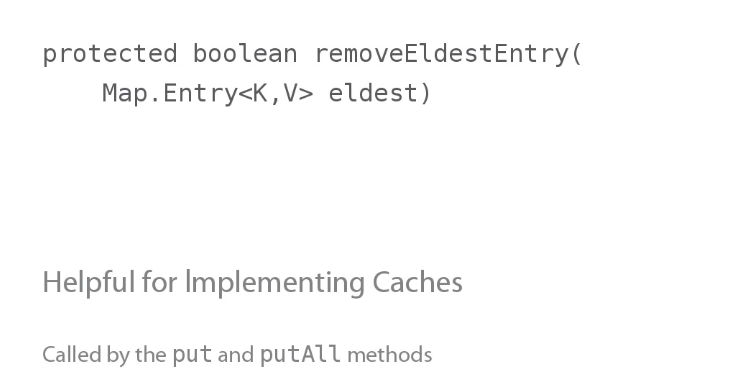
* TreeMap, linkedHashMap, and WeakHashMap





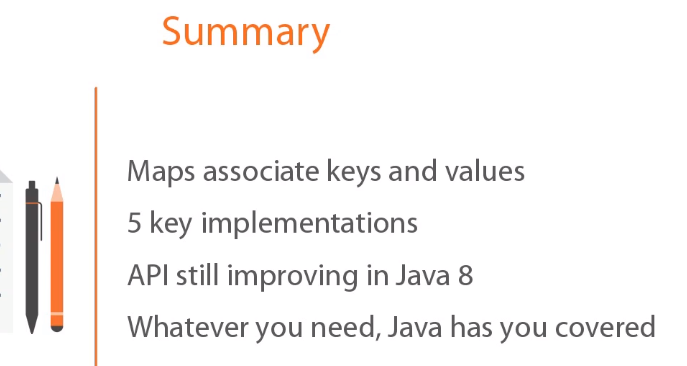
Result







**Enum and summary**

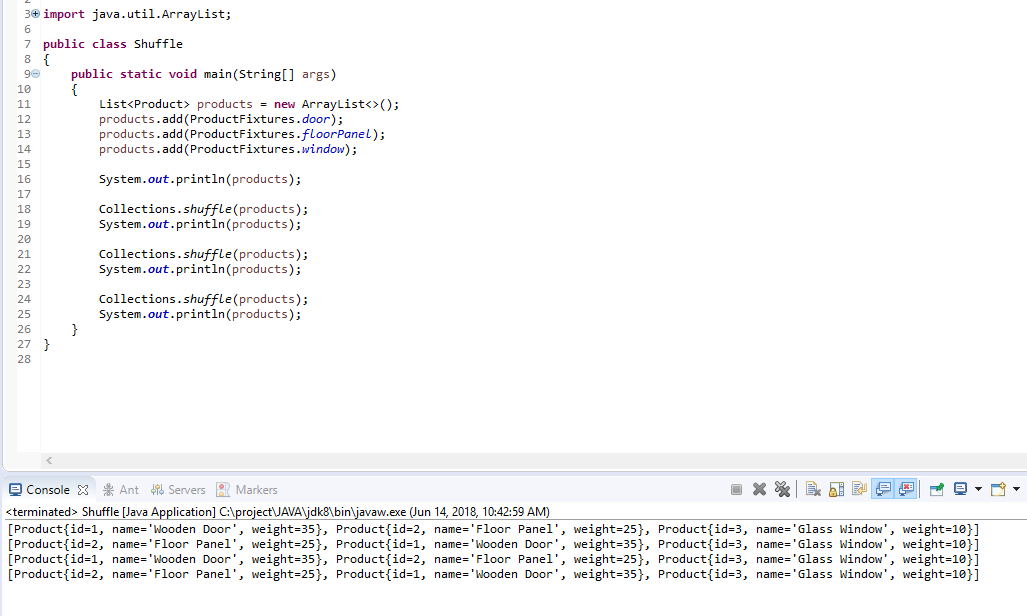


**Collection operations**

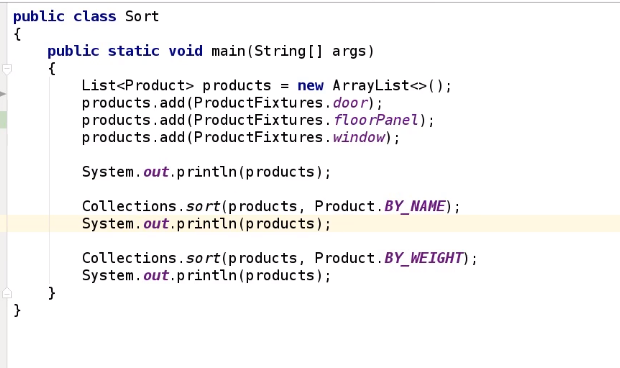
* Collection algorithms
* Rotate() it take the last element from the list and moves it to beginning , and then shuffles all of them up by one



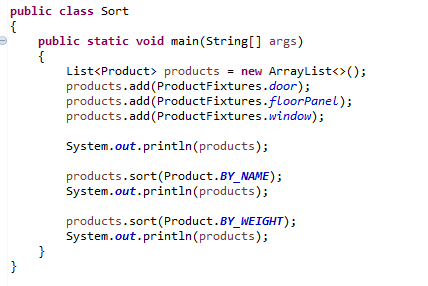
* Shuffling() rearranging order randomly

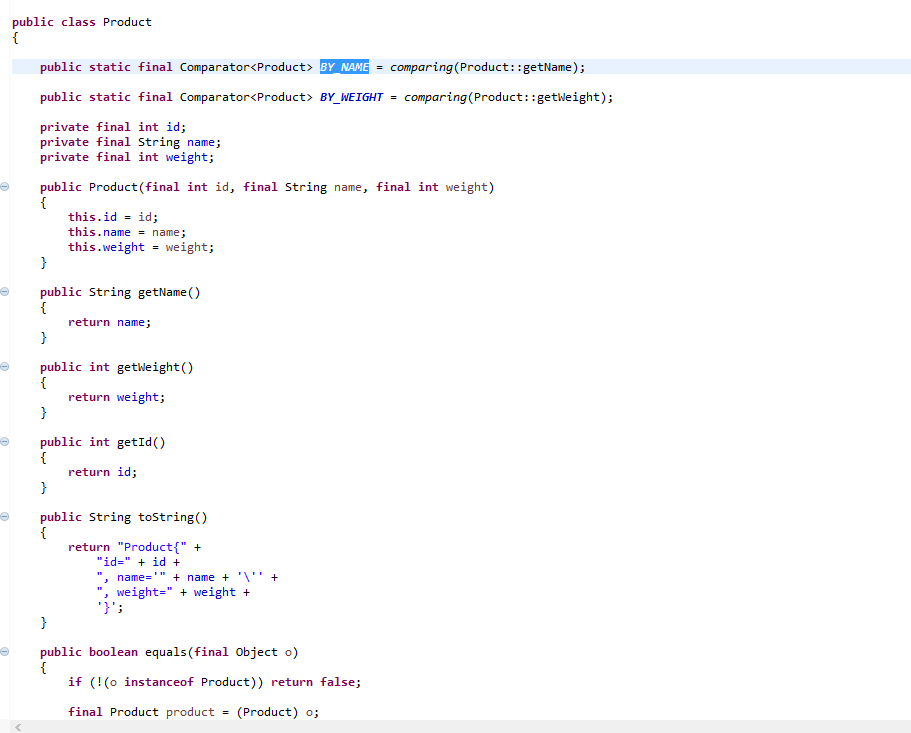


* Sort() sorting take a list in a given order

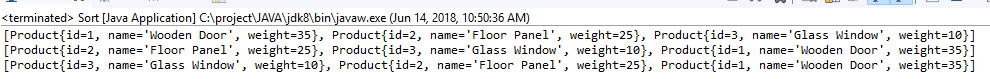


In java 8



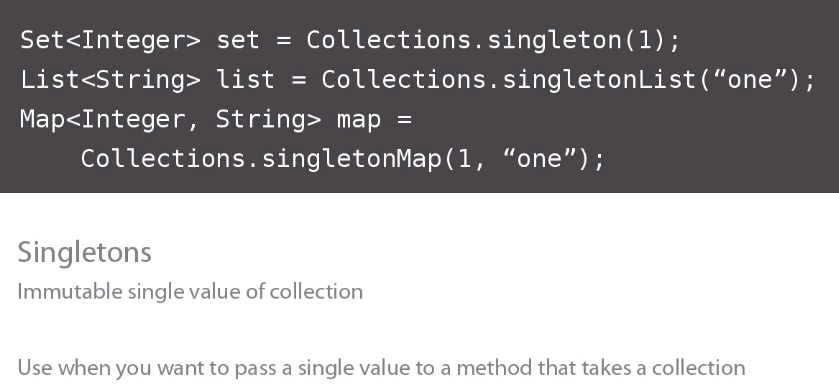


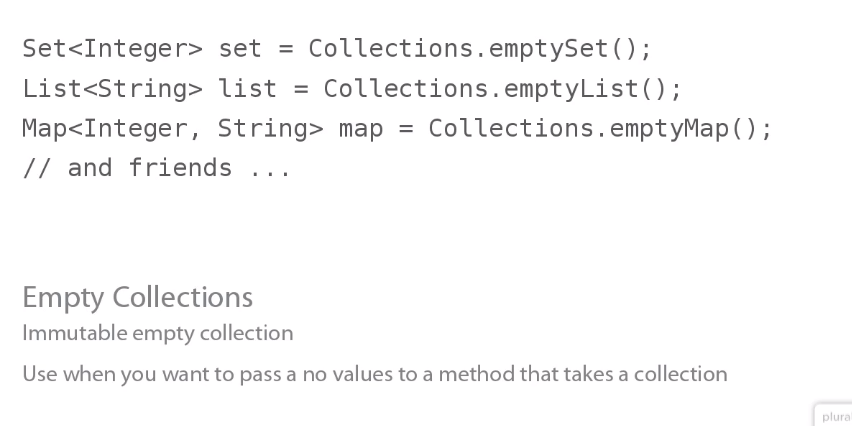
Result



**Collection operations**

* Collections factories





* Collection Utilities

