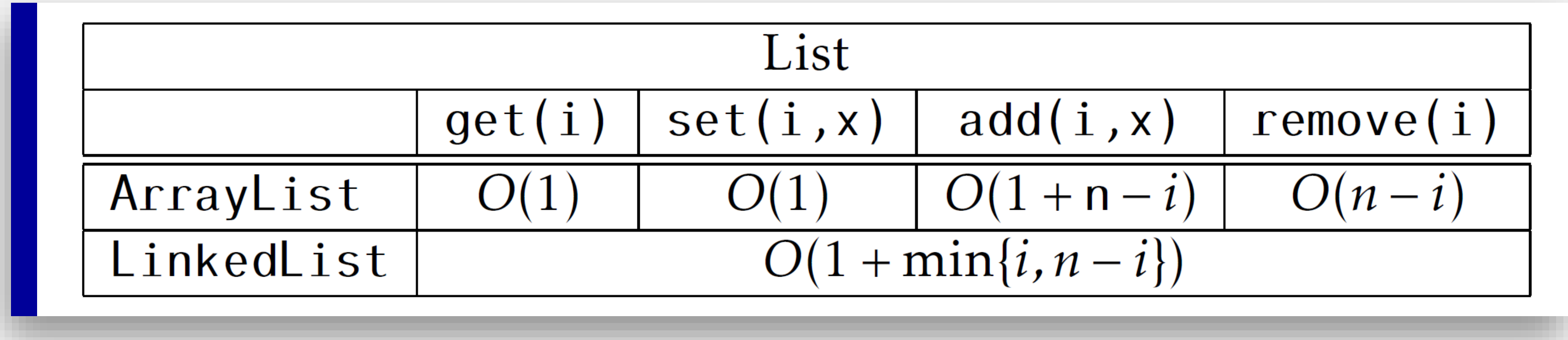
Alina Shaikhet

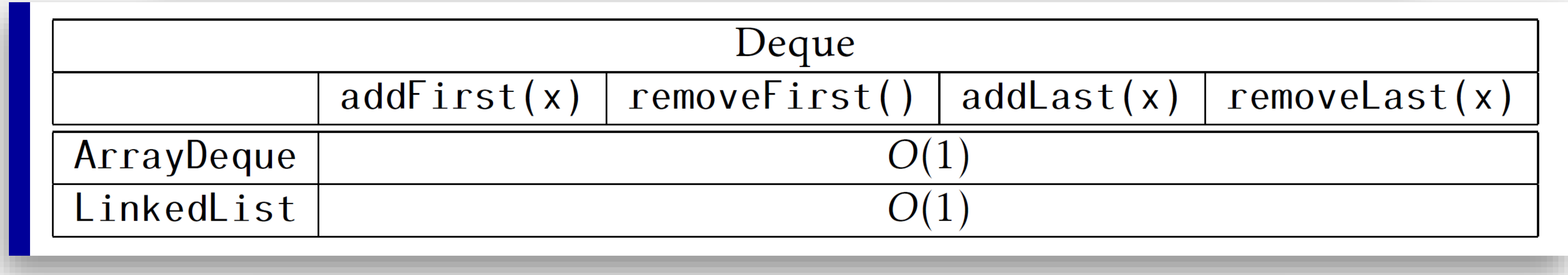
Carleton University

Implementation DEMO

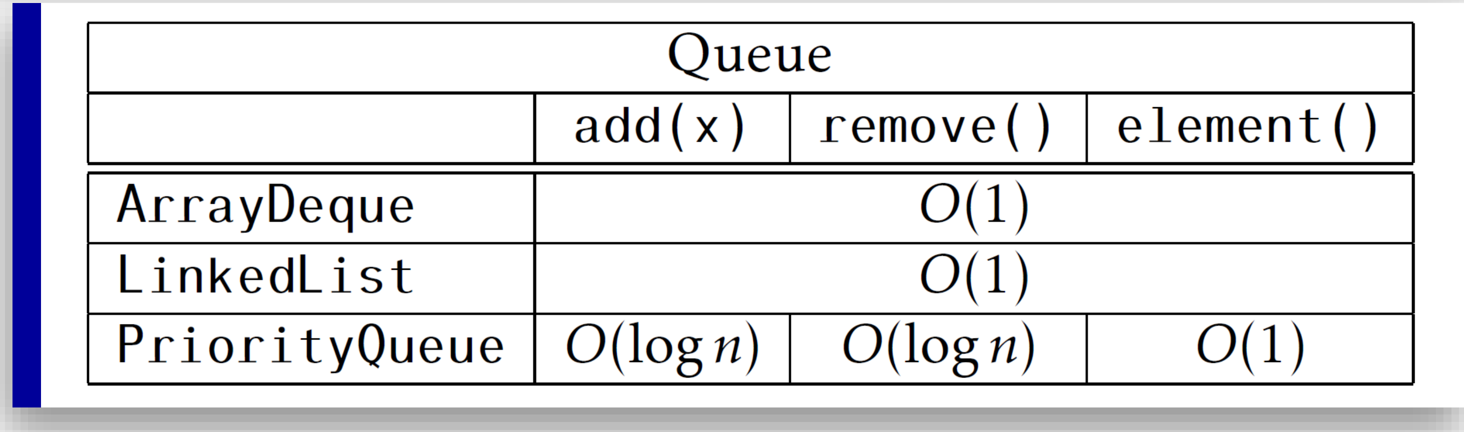
Java Collections Framework Reference



List				
	get(i)	set(i,x)	add(i,x)	remove(i)
ArrayList	$O(1)$	$O(1)$	$O(1+\mathrm{n}-i)$	$O(n-i)$
LinkedList	$O(1+\min\{i,n-i\})$			

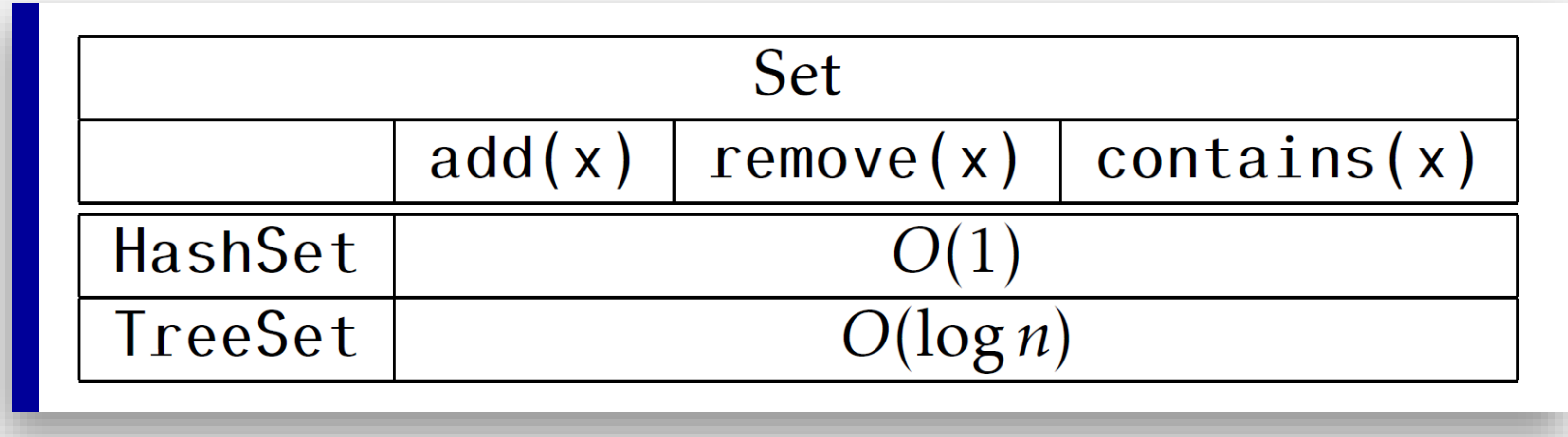


Deque				
	addFirst(x)	removeFirst()	addLast(x)	removeLast(x)
ArrayDeque	$O(1)$			
LinkedList	$O(1)$			

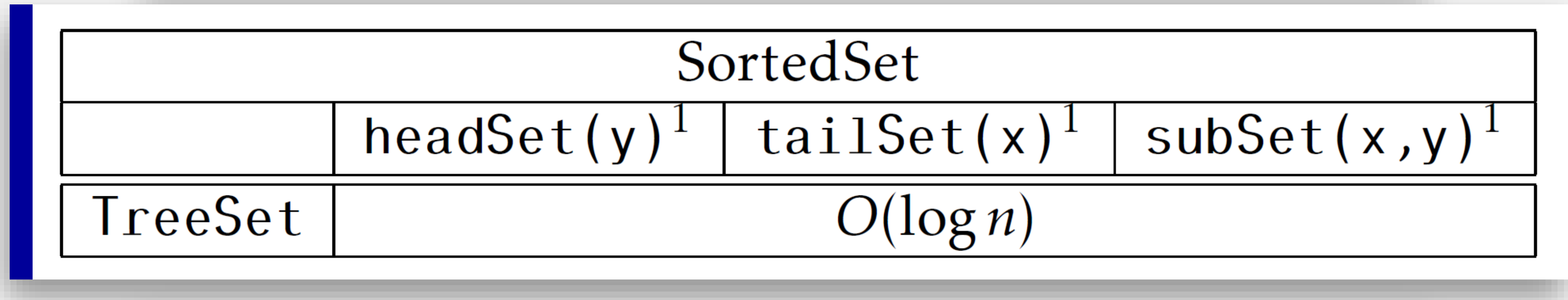


Queue			
	add(x)	remove()	element()
ArrayDeque	$O(1)$		
LinkedList	$O(1)$		
PriorityQueue	$O(\log n)$	$O(\log n)$	$O(1)$

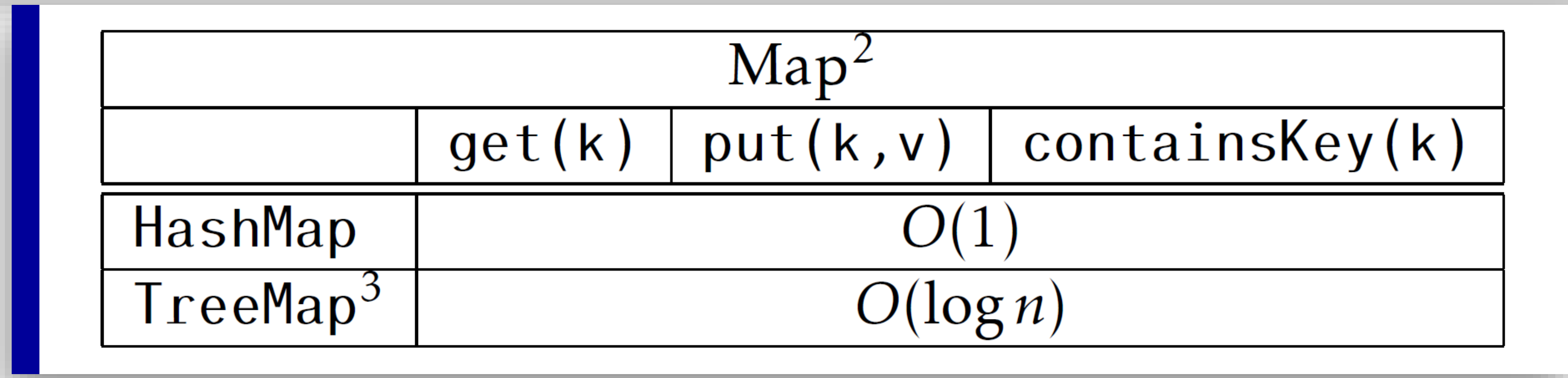
Java Collections Framework Reference



Set			
	add(x)	remove(x)	contains(x)
HashSet	$O(1)$		
TreeSet	$O(\log n)$		



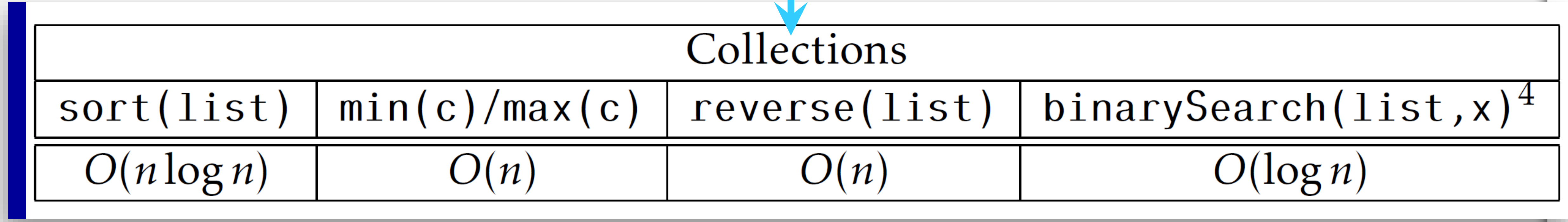
SortedSet			
	headSet(y)[1]	tailSet(x)[1]	subSet(x,y)[1]
TreeSet	$O(\log n)$		



Map[2]			
	get(k)	put(k,v)	containsKey(k)
HashMap	$O(1)$		
TreeMap[3]	$O(\log n)$		

Java Collections Framework Reference

class



Collections			
sort(list)	min(c)/max(c)	reverse(list)	binarySearch(list,x)[4]
$O(n \log n)$	$O(n)$	$O(n)$	$O(\log n)$

Collections.sort(mylist); ← Sorts the specified list into ascending order

https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html

ListDemo.java & SetDemo.java

```
> javac ListDemo.java
> java ListDemo
```

to compile

to run

With sets order is not guaranteed.

Demonstration

```
> javac ListDemo.java
> java ListDemo
```

to compile

to run

attribute

```
> java ListDemo 100000000
        Adding 100000000 values...done (2.732694899s)

 > java ListDemo 100000

        Adding 100000 values...done (0.006679900000000001s)
        Adding 100000 values at the front...done (0.322008199s)

 > java ListDemo 1000000
          Adding 1000000 values...done (0.027967501000000002s)
          Adding 1000000 values at the front...done (65.229541s)
```

Huge difference!

Now change implementation from ArrayList to LinkedList

```
> java ListDemo 1000000
          Adding 1000000 values...done (0.11051920000000001s)
          Adding 1000000 values at the front...done (0.12018820000000001s)

> java ListDemo 100000
           Using 200000 calls to contains()...done (35.2039442s)
```

Very slow ☹

Demonstration

```
> java SetDemo 100000

        Adding 100000 elements to HashSet ...done (0.024065100000000002s)
        Searching 200000 elements...done (0.011080300000000001s)
        Adding 100000 elements to TreeSet ...done (0.0842638s)
        Searching 200000 elements...done (0.052904400000000004s)

> java SetDemo 10000000

        Adding 10000000 elements to HashSet ...done (1.098738s)
        Searching 20000000 elements...done (0.3798946s)
        Adding 10000000 elements to TreeSet ...done (2.7329163000000003s)
        Searching 20000000 elements...done (2.5870101s)
```

WordFun.java

This program opens a file and parses it into words.

It returns the content of the file as a giant list of strings.

```
> java WordFun TreasureIsland.txt
```

the book is copyright free so we can freely use it as test data

shell command to count # of words:

```
> wc -w TreasureIsland.txt
> cat TreasureIsland.txt | Measure-Object -Word
```

PowerShell

Basic PowerShell Commands

cat	Get-Content	Gets the contents of a file.
cd	Set-Location	Sets the current working location to a specified location.
chdir	Set-Location	Sets the current working location to a specified location.
clc	Clear-Content	Deletes the contents of an item, but does not delete the item.
clear	Clear-Host	Clears the display in the host program.
cp	Copy-Item	Copies an item from one location to another.
ls	Get-ChildItem	Gets the files and folders in a file system drive.
man	help	Displays information about Windows PowerShell commands and concepts.
md	mkdir	Creates a new item.
rm	Remove-Item	Deletes files and folders.