KEY POINTS

- PQs are just like queue ADT but some elements have greater priority, which are removed first.
- The elements must be comparable so that data can be ordered.
- We use a key value pair to help order things.
- We relate KVPs by using a total order relation in order to allow two distinct entries to have the same key.

Entry ADT

A KVP with methods for getKey() and getValue().

Comparator ADT

- Typically kept seperate from elements.
- Used to compare two obj. from a total order.

compare(a, b)

$i \leq 0$ if $a \leq b$
$i = 0$ if $a = b$
$i \geq 0$ if $a \geq b$
} same sign

NAME/DATE/SUBJECT

Priority Queue

NOTES

ADT Functions:

insert(k, v)

removeMin() } we return null if we run into an exception.

min() ⇒ like peek()

Least → highest:
The smaller numbers will have greater priority so will be removed first.

Lexicographic 2D comp. (a, b) (c, d) (x, y)

$a \neq c$	$b \neq d$	$x = y$
$x > y$ iff $b > d$	$x > y$ iff $b > d$	
if →	else if →	else

You are not guaranteed an ordered list! The PQ will simply return the next object with highest priority.

Total Order Relation: $\leq$

- Either $x \leq y$ or $y \leq x$ (comparability)
- $x \leq y$ and $y \leq x \Rightarrow x = y$ (antisymmetry)
- $x \leq y$ and $y \leq z \Rightarrow x \leq z$ (transitive)
- ($x_{min} \leq y$) is a representation of the reflexive relation.

SUMMARY

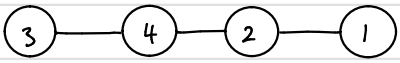
KEY POINTS

NAME/DATE/SUBJECT

Sequence based priority queue.

NOTES

Unsorted list

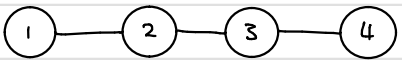


Insert O(1)

remove Min O(n)

Min

Sorted list



Insert O(n)

remove Min O(1)

Min

O(1) operations can either insert or remove from the front.

O(n) operations must traverse each element in the list.

Selection sort - $O(n^2)$

- Relies on an unsorted queue
- You can insert elements with O(n) time.
- Removing objects has running time

$$O(n) + O(n-1) + \cdots O(1) = O(n^2)$$

Insertion sort - $O(n^2)$

- Relies on a sorted queue.

SUMMARY