# Heaps

## Chapter 23

*Data Structures and Abstractions with Java, 4e, Global Edition*
Frank Carrano

# Heaps

- Definition: Complete binary tree whose nodes contain `Comparable` objects and are organized as follows.
  - Each node contains an object no smaller/larger than objects in its descendants
  - Maxheap: object in node greater than or equal to its descendant objects
  - Minheap: object in node less than or equal to its descendant objects

# Heaps

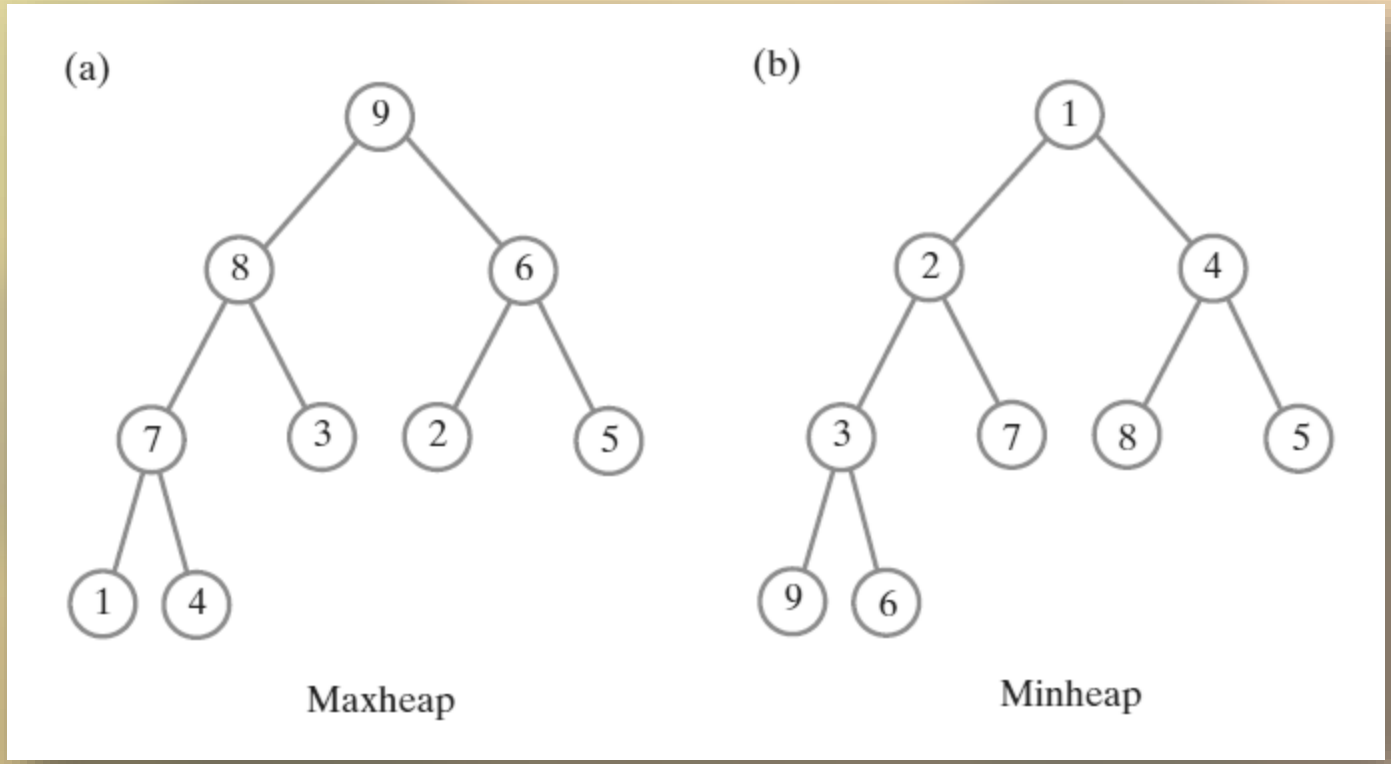


FIGURE 23-21 (a) A maxheap and
(b) a minheap that contain the same values

# Heaps

```
public interface MaxHeapInterface<T extends Comparable<? super T>>
{
   /** Adds a new entry to this heap.
       @param newEntry  An object to be added. */
   public void add(T newEntry);

   /** Removes and returns the largest item in this heap.
       @return  Either the largest object in the heap or,
                if the heap is empty before the operation, null. */
   public T removeMax();
```

LISTING 23-6 An interface for a maxheap

# Heaps

```
12      /** Retrieves the largest item in this heap.
13          @return  Either the largest object in the heap or,
14                   if the heap is empty, null. */
15      public T getMax();
16
17      /** Detects whether this heap is empty.
18          @return  True if the heap is empty, or false otherwise. */
19      public boolean isEmpty();
20
21      /** Gets the size of this heap.
22          @return  The number of entries currently in the heap. */
23      public int getSize();
24
25      /** Removes all entries from this heap. */
26      public void clear();
27  } // end MaxHeapInterface
```

LISTING 23-6 An interface for a maxheap

# Heaps

```
public final class HeapPriorityQueue<T extends Comparable<? super T>>
                   implements PriorityQueueInterface<T>
{
   private MaxHeapInterface<T> pq;

   public Heap PriorityQueue()
   {
      pq = new MaxHeap<>();
   } // end default constructor

   public void add(T newEntry)
   {
      pq.add(newEntry);
   } // end add
   < Implementations of remove, peek, isEmpty, getSize, and clear are here. >
   . . .
} // end HeapPriorityQueue
```

LISTING 23-7 The beginning of the class **HeapPriorityQueue**

# End

Chapter 23