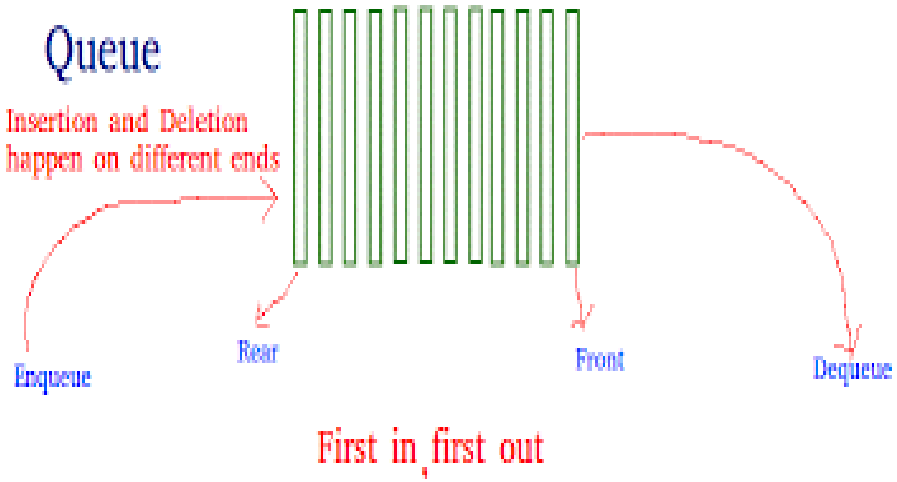
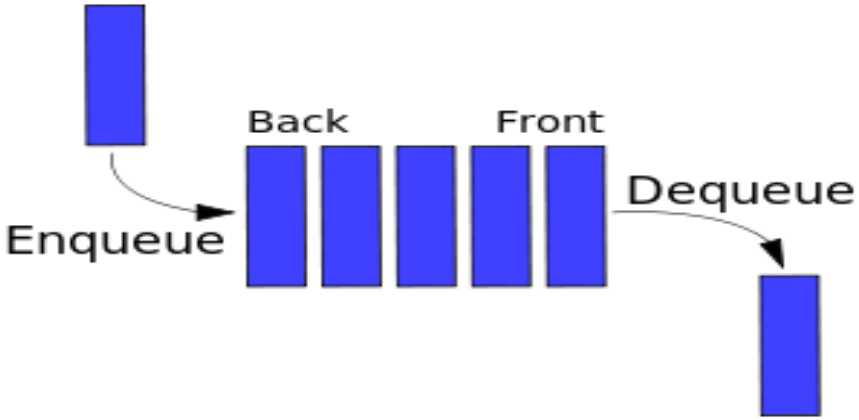
Priority Queues

An application of Heap

Examples for Queue



Operations on Queue



- Enqueue and Dequeue: Insert an element to the back/rear/tail and remove an element from front/head

Priority Queue (PQ)

- Similar to queue - difference that the logical order of elements in the priority queue depends on the priority of the elements.
- The element with highest priority will be moved to the front of the queue and one with lowest priority will move to the back of the queue.
- Hence, it is possible that when you enqueue an element at the back in the queue, it can move to front because of its highest priority
- Depending on the requirements in applications, there are 2 kinds of PQ: Max and Min Priority Queue

- Let's say we have an **array of 5 elements indicating jobs** {4, 8, 1, 7, 3} - Insert all the jobs in the max-priority queue.
- First as the priority queue is empty, so 4 is inserted initially.
- Now when 8 is inserted it will be moved to front as 8 is greater than 4.
- While inserting 1, as it is the current minimum element in the priority queue, it remains in the back of priority queue
- Now 7 is inserted between 8 and 4 as 7 is smaller than 8.
- Now 3 is inserted before 1 as it is the 2nd minimum element in the priority queue.

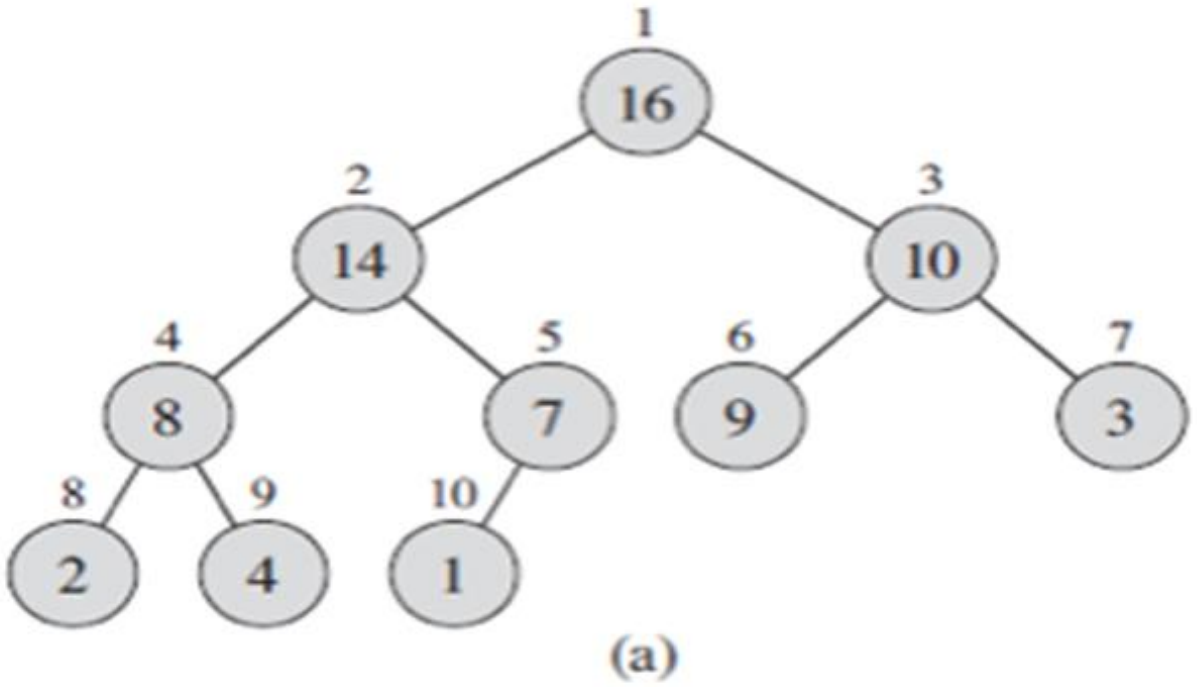
Priority Queue (PQ) & Operations

- PQ - Data structure for maintaining a set A of elements, each with an associated value called a key (priority).

Operations on Max-priority queue:

- MAXIMUM(A): returns the element of A with the largest key.
- EXTRACT-MAX(A): removes and returns the element of A with the largest key.
- INCREASE-KEY(A, x, k) increases the value of element x's key to the new value k, which is assumed to be at least as large as x's current key value.
- INSERT(A,x) : inserts the element x into the set A, which is equivalent to the operation $A = A \cup \{x\}$.

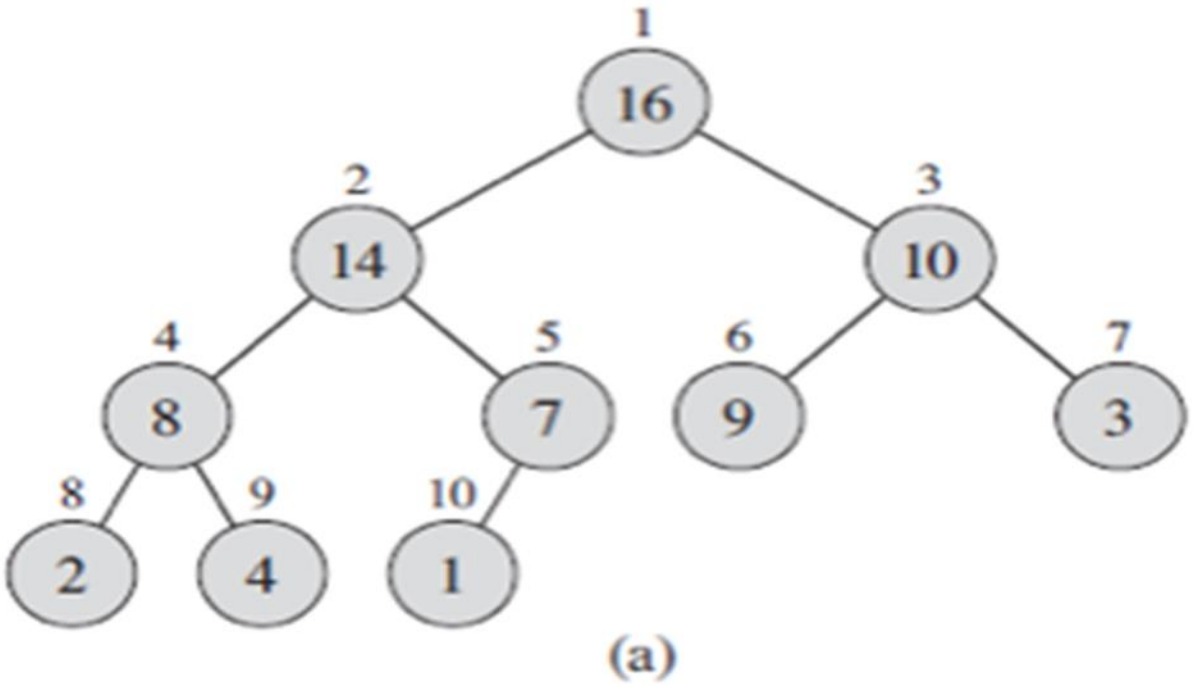
MAXIMUM(A) ?



HEAP-MAXIMUM(A)

return A[1]

Extract-Max (A) ?



How do we extract/ remove the max value from the heap and still maintain the heap property?

Extract-Max operations

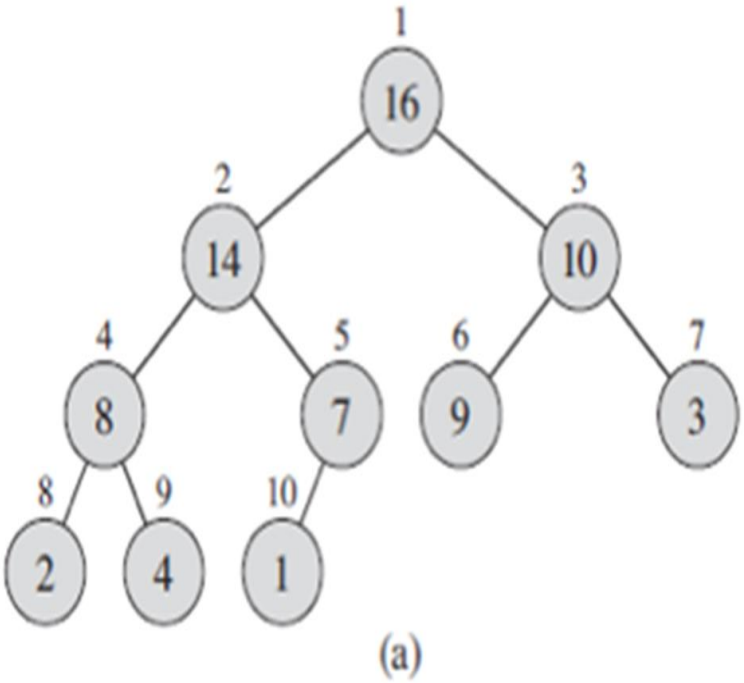
HEAP-EXTRACT-MAX(A)

```
1 if A.heap-size < 1
2error "heap underflow"
3 max = A[1]
4 A[1] = A[A.heap-size]
5 A.heap-size = A.heap-size - 1
6 MAX-HEAPIFY(A, 1)
7 return max
```

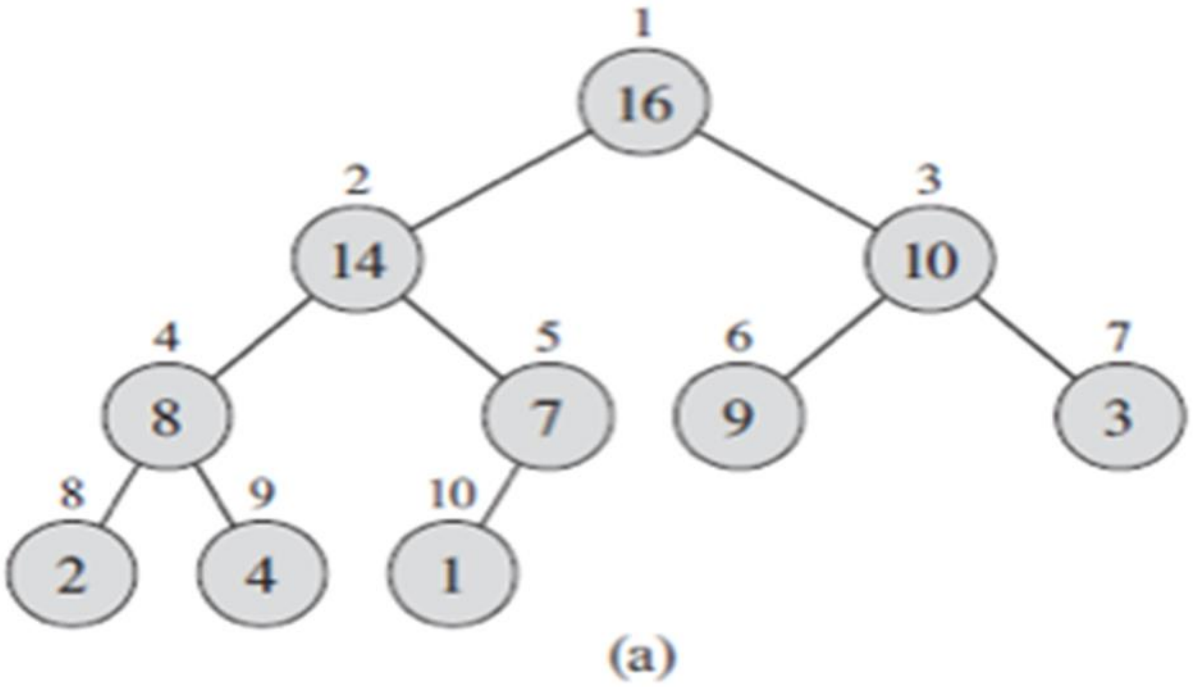
Trace the working of Extract-Max

HEAP-EXTRACT-MAX(A)

```
1  if A.heap-size < 1
2error "heap underflow"
3 max = A[1]
4  A[1] = A[A.heap-size]
5  A.heap-size = A.heap-size - 1
6  MAX-HEAPIFY(A, 1)
7  return max
```



HEAP-INCREASE-KEY (A,i,key) ?



- Suppose we have to increase the key of the last node (currently the key is 1) to 15
- How do we proceed ?

HEAP-INCREASE-KEY

The procedure HEAP-INCREASE-KEY implements the INCREASE-KEY operation.

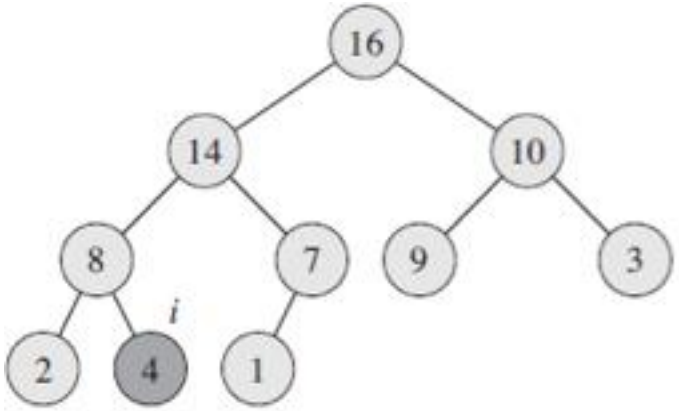
An index i into the array identifies the priority-queue element whose key to be increased.

The procedure first updates the key of element A[i] to its new value.

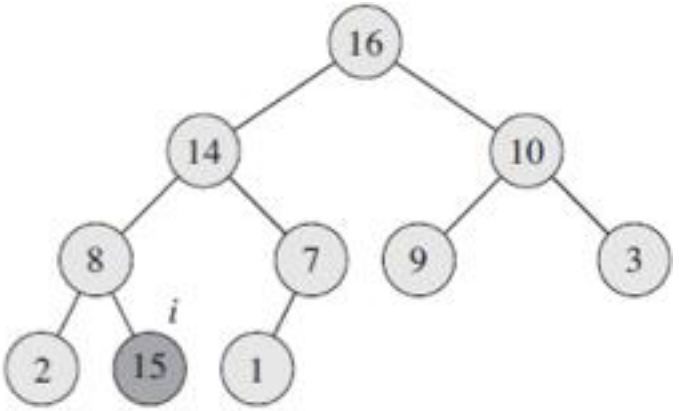
- Increasing the key of A[i] might violate the max-heap property. How do we handle this?

- Traverse a simple path from this node toward the root to find a proper place for the newly increased key.

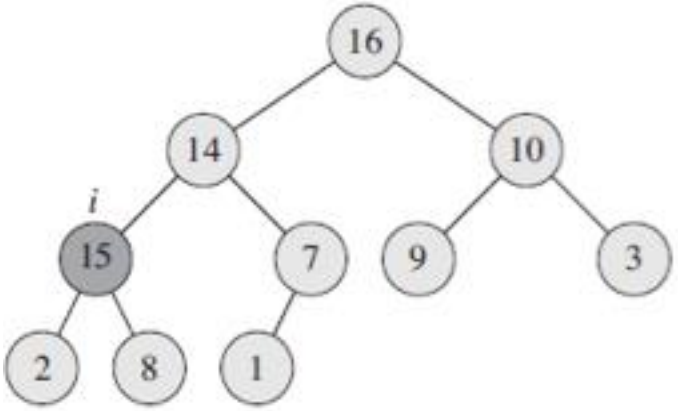
Example: Working of Heap-Increase key



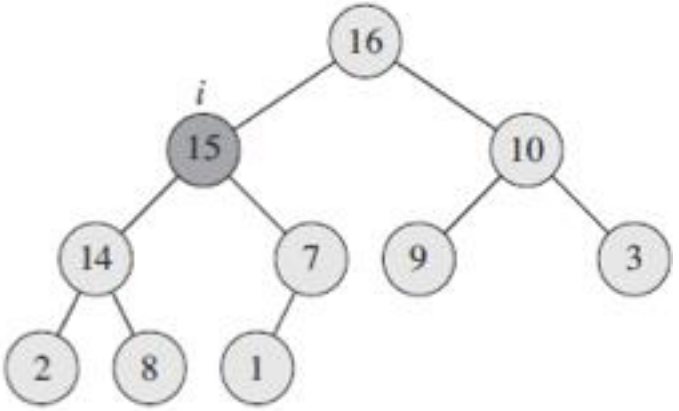
(a)



(b)



(c)



(d)

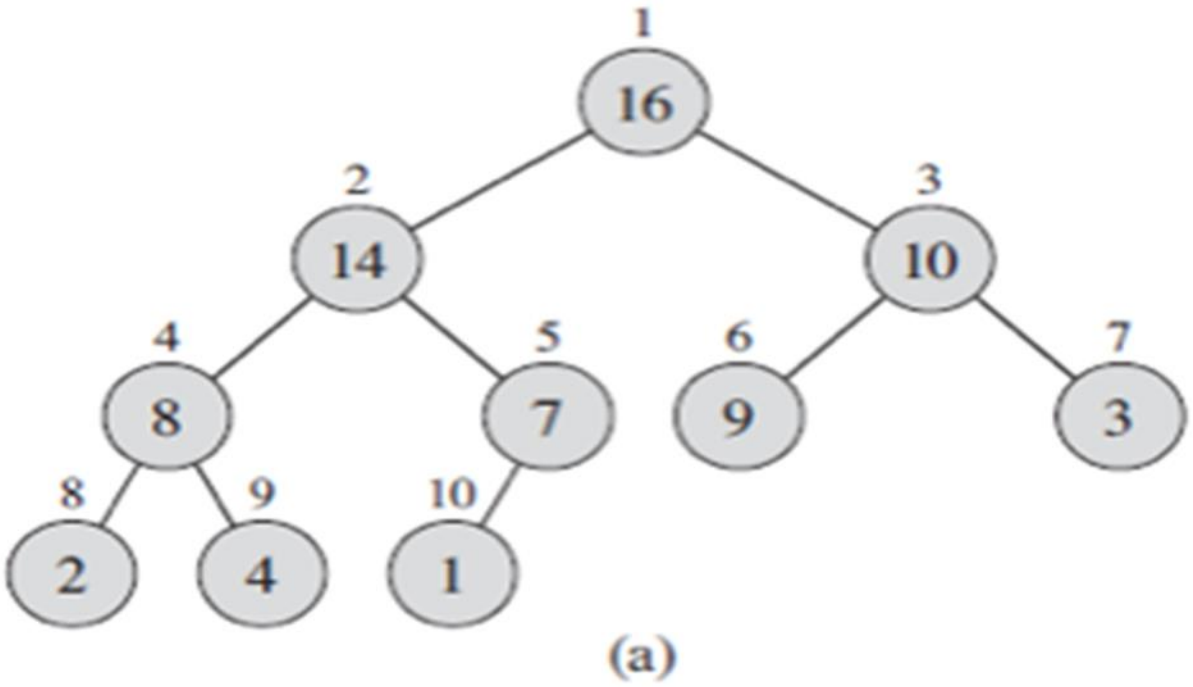
HEAP-INCREASE-KEY

```
HEAP-INCREASE-KEY(A, i, key)
1  if key < A[i]
2       error "new key is smaller than current key"
3  A[i] = key
4  while i > 1 and A[PARENT(i)] < A[i]
5       exchange A[i] with A[PARENT(i)]
6       i = PARENT(i)
```

Loop invariant: Heap-Increase-Key

At the start of each iteration of the **while** loop of lines 4–6, the subarray $A[1..A.heap\text{-}size]$ satisfies the max-heap property, except that there may be one violation: $A[i]$ may be larger than $A[\text{PARENT}(i)]$.

MAX-HEAP-INSERT (A, key)



- How do we insert a new key to a max-heap?
- Can we use the existing functions?

MAX-HEAP-INSERT

- The procedure MAX-HEAP-INSERT implements the INSERT operation.
- It takes as an input the key of the new element to be inserted into max-heap A.
- The procedure first expands the max-heap by adding to the tree a new leaf whose key is the minimum value.
- Then it calls HEAP-INCREASE-KEY to set the key of this new node to its correct value and maintain the max-heap property.

MAX-HEAP-INSERT

```
MAX-HEAP-INSERT(A, key)
1  A.heap-size = A.heap-size + 1
2  A[A.heap-size] = −∞
3  HEAP-INCREASE-KEY(A, A.heap-size, key)
```

Exercise: Write the functions Insert, Delete, Heap-Decrease-Key and Heap-extract-Min for Min-heap priority queue.

Application: Max-Priority Queue

- Max-priority queues - to schedule jobs on a shared computer.
- The max-priority queue keeps track of the jobs to be performed and their relative priorities.
- When a job is finished or interrupted, the scheduler selects the highest-priority job from among those pending by calling EXTRACT-MAX.
- The scheduler can add a new job to the queue at any time by calling INSERT and preserve the order of the priority of new job.

THANK YOU