

Principles of Computer Science II

Abstract Data Types

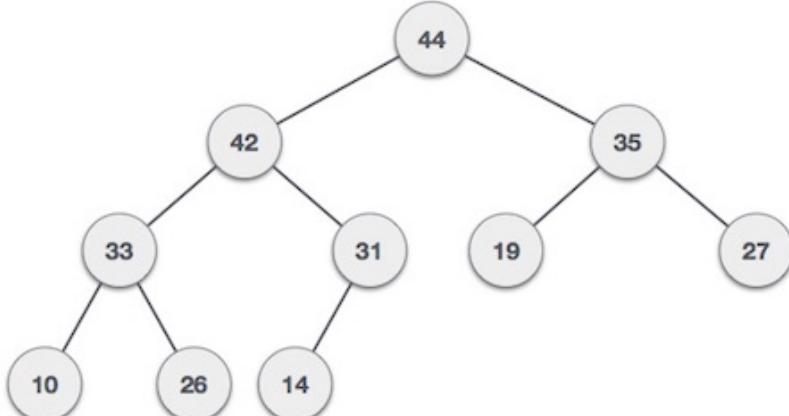
Ioannis Chatzigiannakis

Sapienza University of Rome

Lecture 11



Max Heap



Heap Data Structures

Heap is a special case of balanced binary tree data structure where the root-node key is compared with its children and arranged accordingly.

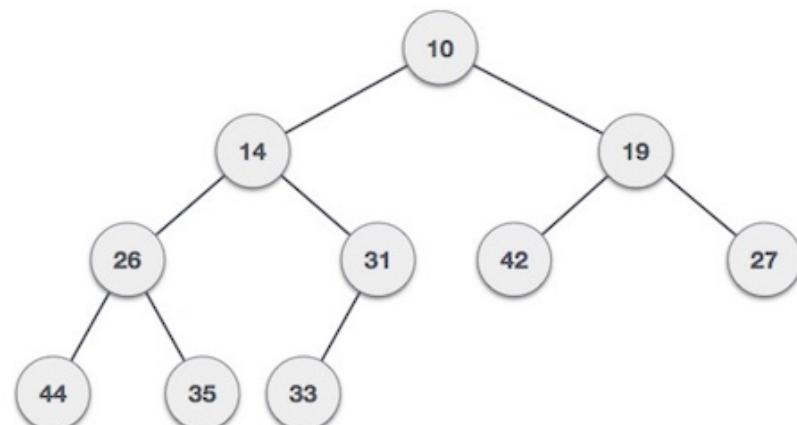
If α has child node β then

$$\text{key}(\alpha) \geq \text{key}(\beta)$$

- As the value of parent is greater than that of child, this property generates **Max Heap**.
 - By changing the criterion we can also get a **Min Heap**.



Min Heap



Max Heap – Add

1. Create a new node at the end of heap.
2. Assign new value to the node.
3. Compare the value of this child node with its parent.
4. If value of parent is less than child, then swap them.
5. Repeat step 3 & 4 until Heap property holds.

Note – In Min Heap construction algorithm, we expect the value of the parent node to be less than that of the child node.

Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31

35



Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31

35



Max Heap – Add Example

Input **35** 33 42 10 14 19 27 44 26 31



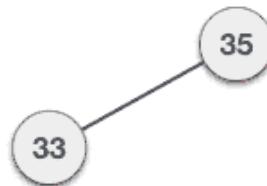
Max Heap – Add Example

Input **35 33** 42 10 14 19 27 44 26 31



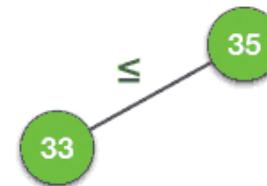
Max Heap – Add Example

Input **35 33** 42 10 14 19 27 44 26 31



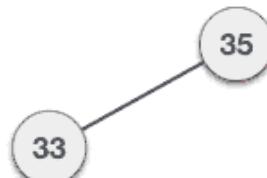
Max Heap – Add Example

Input **35 33** 42 10 14 19 27 44 26 31



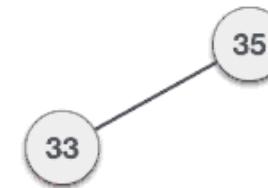
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



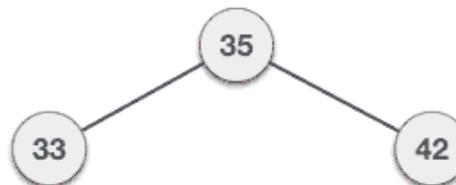
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



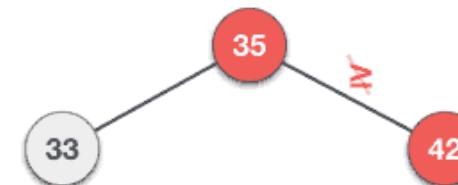
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



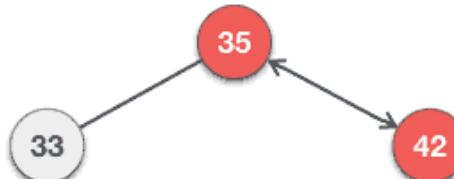
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



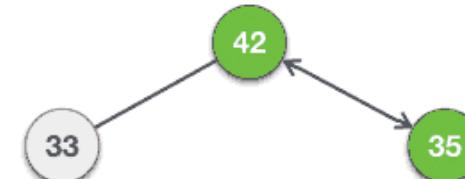
Max Heap – Add Example

Input 35 33 **42** 10 14 19 27 44 26 31



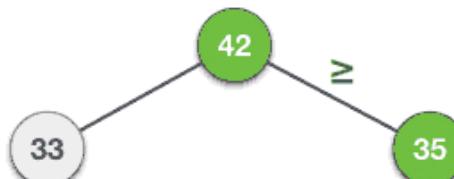
Max Heap – Add Example

Input 35 33 **42** 10 14 19 27 44 26 31



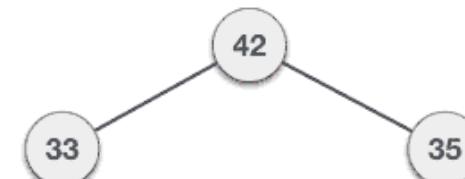
Max Heap – Add Example

Input 35 33 **42** 10 14 19 27 44 26 31



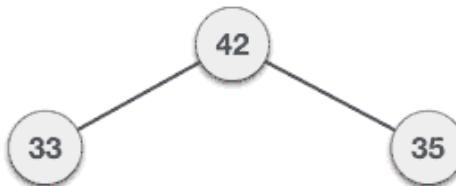
Max Heap – Add Example

Input 35 33 **42** 10 14 19 27 44 26 31



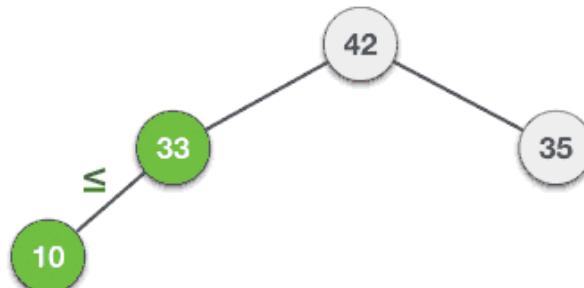
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



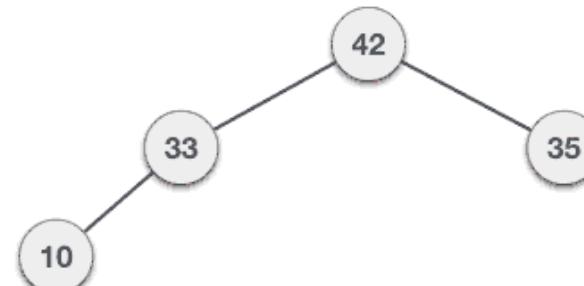
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



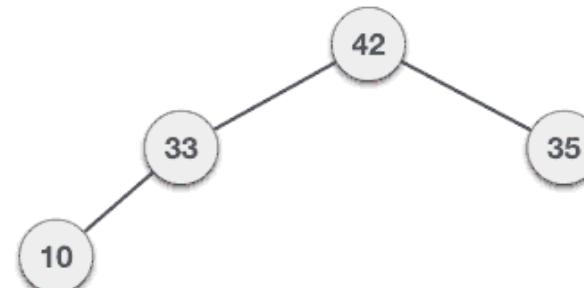
Max Heap – Add Example

Input 35 33 42 **10** 14 19 27 44 26 31



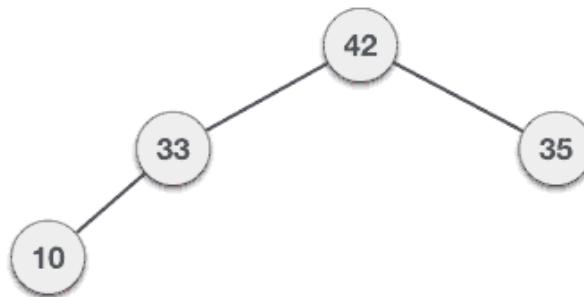
Max Heap – Add Example

Input 35 33 42 **10** 14 19 27 44 26 31



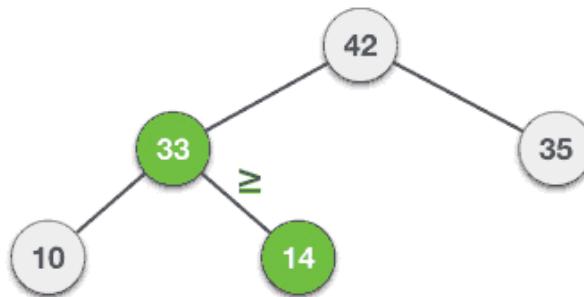
Max Heap – Add Example

Input 35 33 42 10 **14** 19 27 44 26 31



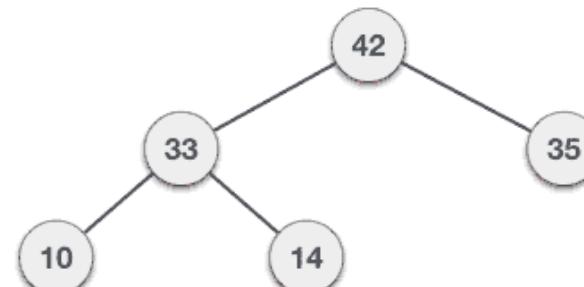
Max Heap – Add Example

Input 35 33 42 10 **14** 19 27 44 26 31



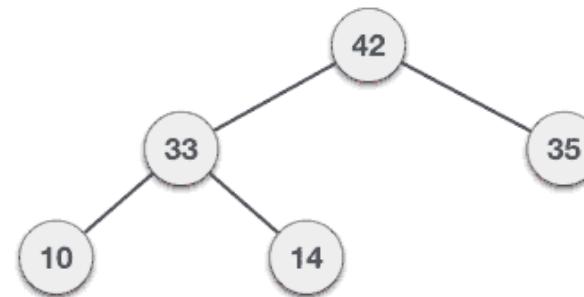
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



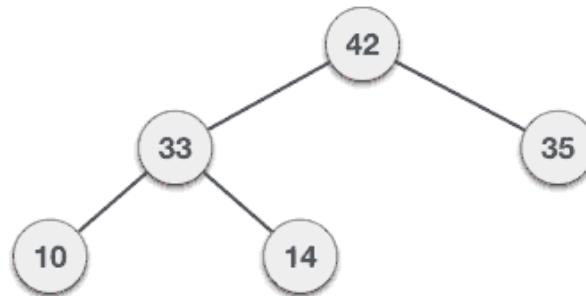
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31



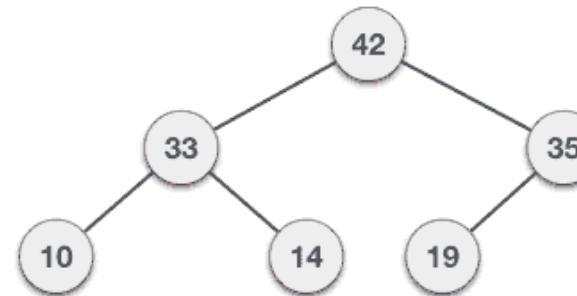
Max Heap – Add Example

Input 35 33 42 10 14 **19** 27 44 26 31



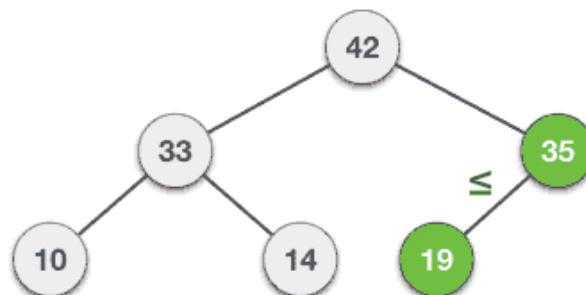
Max Heap – Add Example

Input 35 33 42 10 14 **19** 27 44 26 31



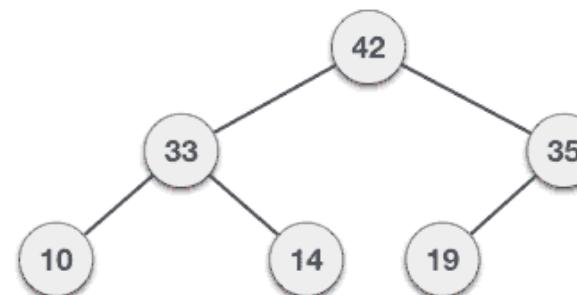
Max Heap – Add Example

Input 35 33 42 10 14 **19** 27 44 26 31



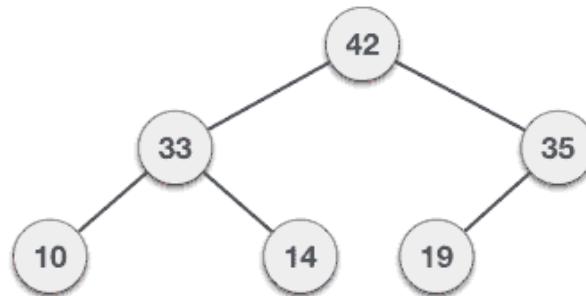
Max Heap – Add Example

Input 35 33 42 10 14 **19** 27 44 26 31



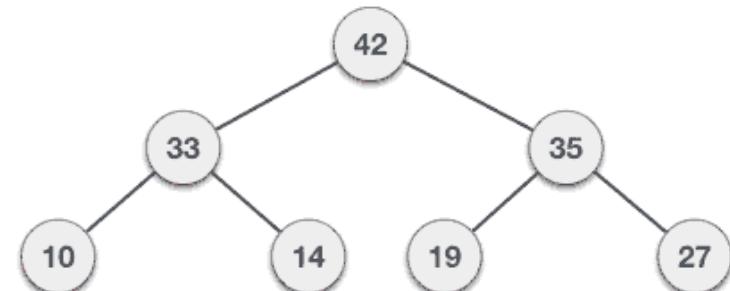
Max Heap – Add Example

Input 35 33 42 10 14 19 **27** 44 26 31



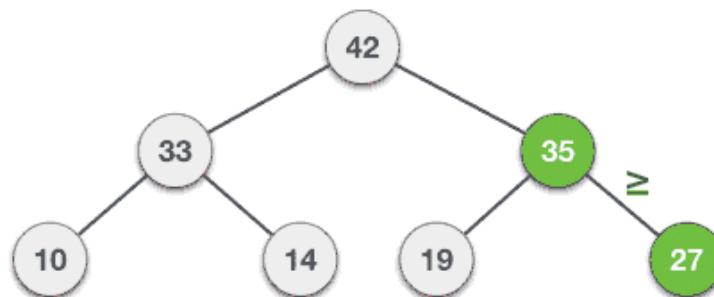
Max Heap – Add Example

Input 35 33 42 10 14 19 **27** 44 26 31



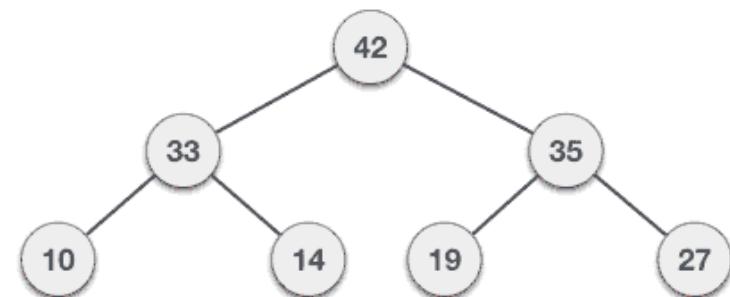
Max Heap – Add Example

Input 35 33 42 10 14 19 **27** 44 26 31



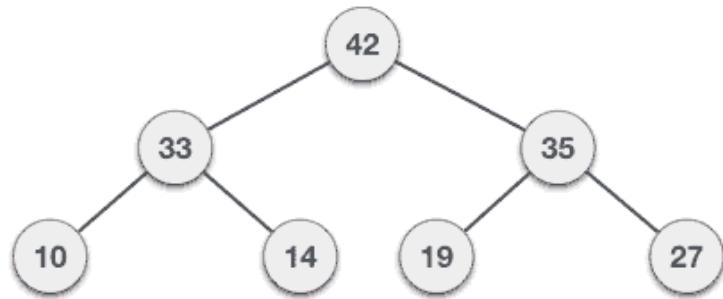
Max Heap – Add Example

Input 35 33 42 10 14 19 **27** 44 26 31



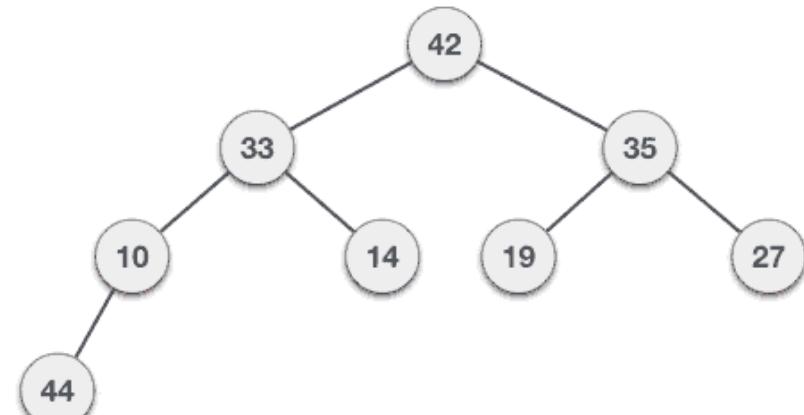
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31



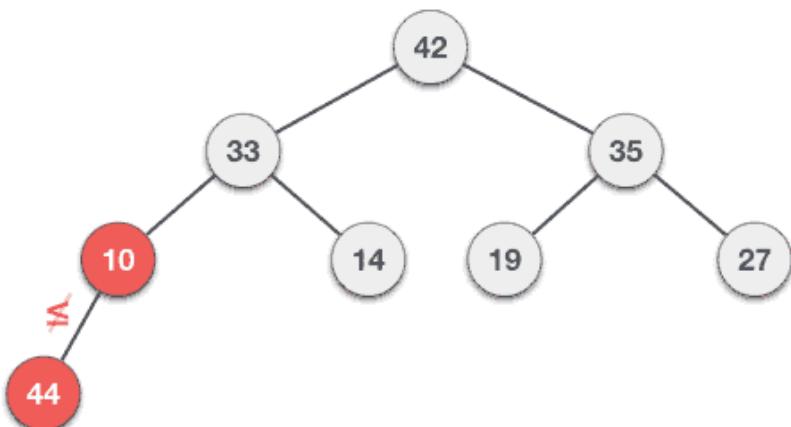
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31



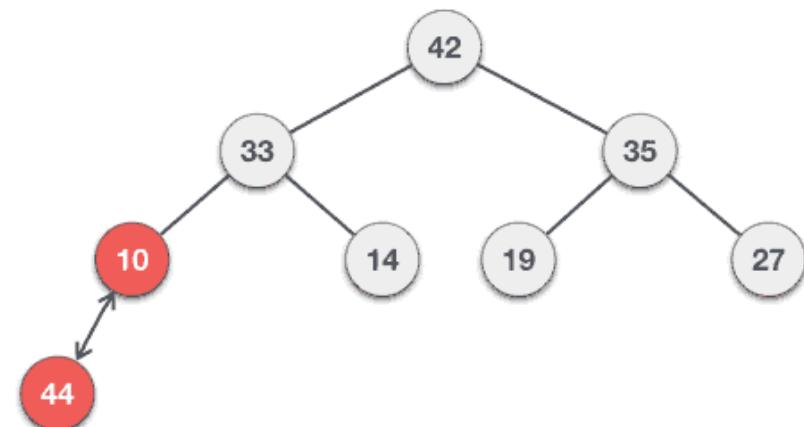
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31

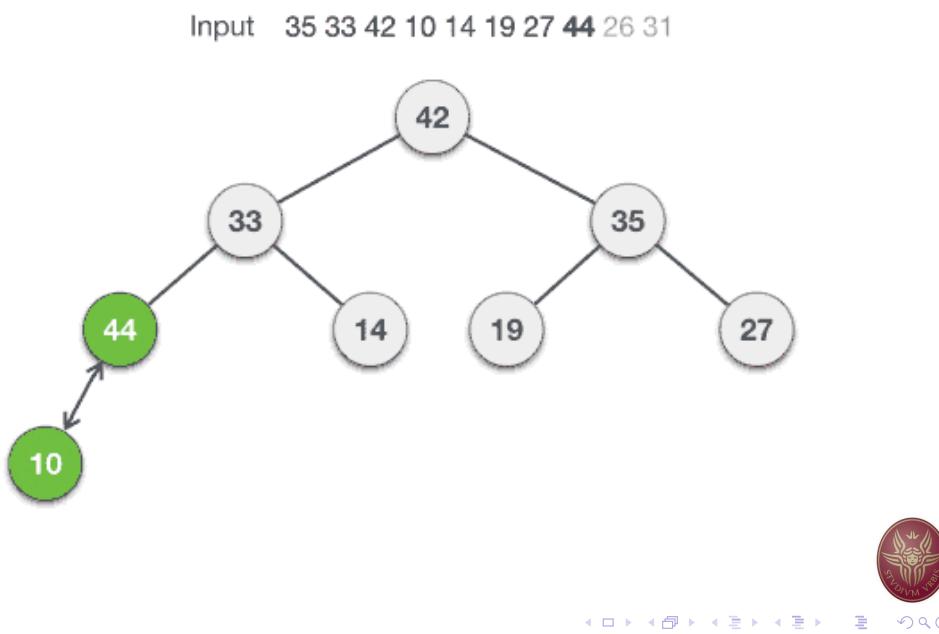


Max Heap – Add Example

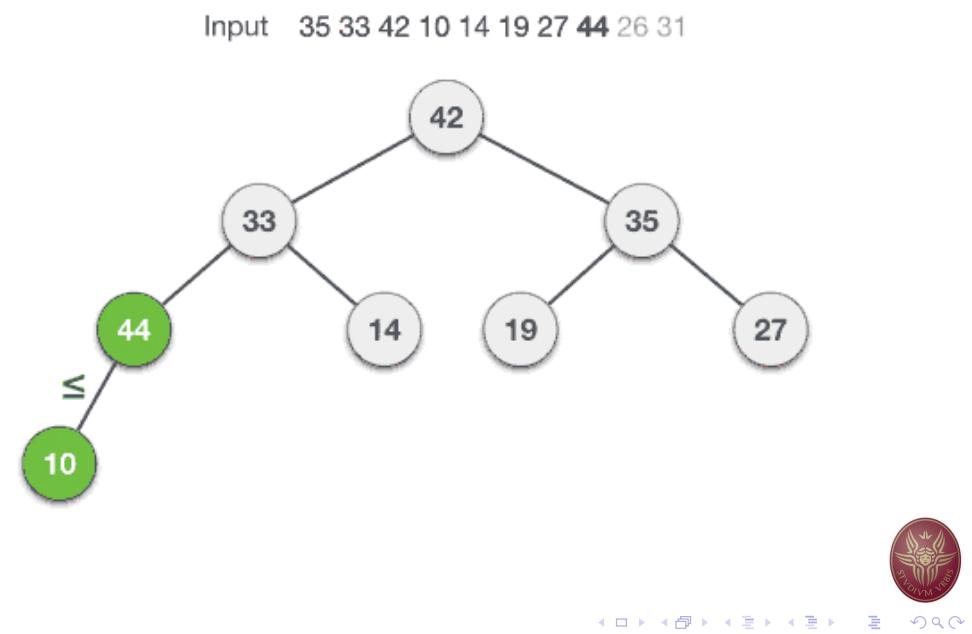
Input 35 33 42 10 14 19 27 **44** 26 31



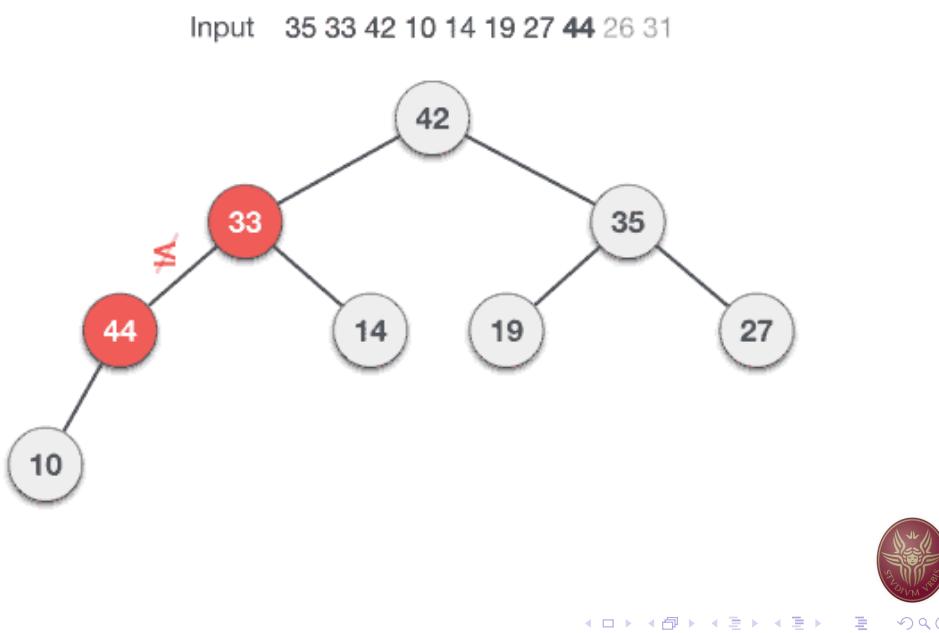
Max Heap – Add Example



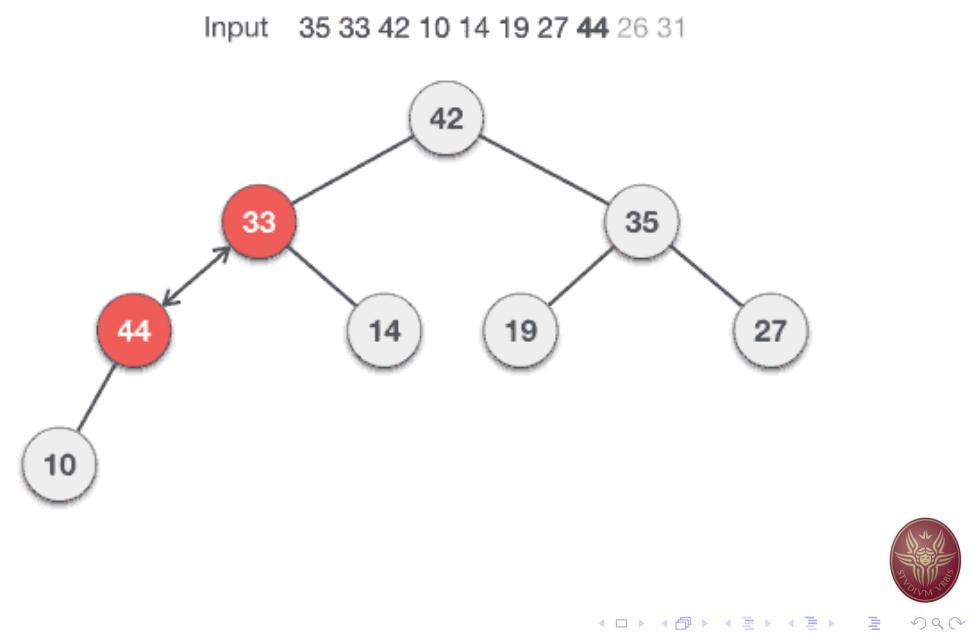
Max Heap – Add Example



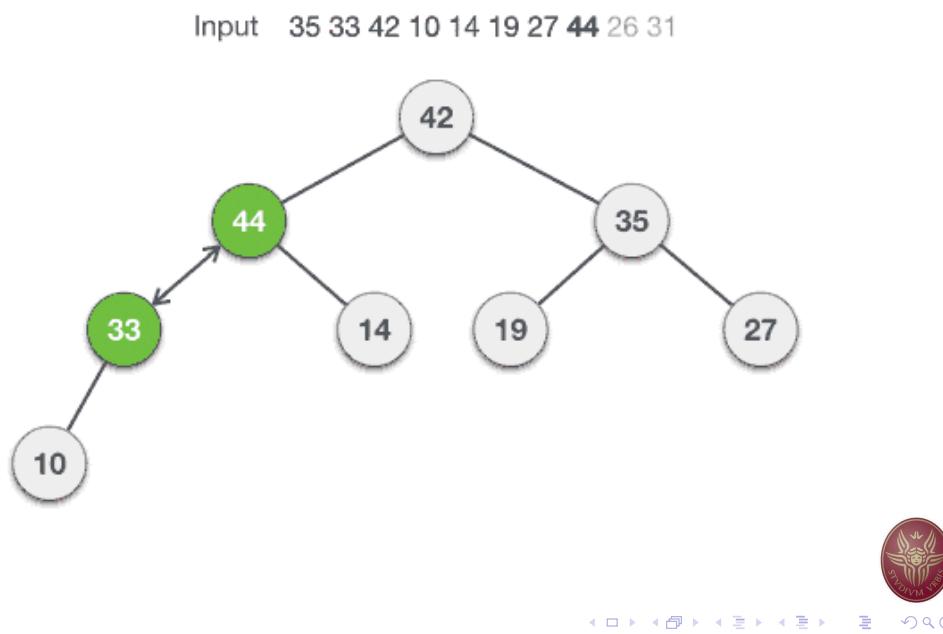
Max Heap – Add Example



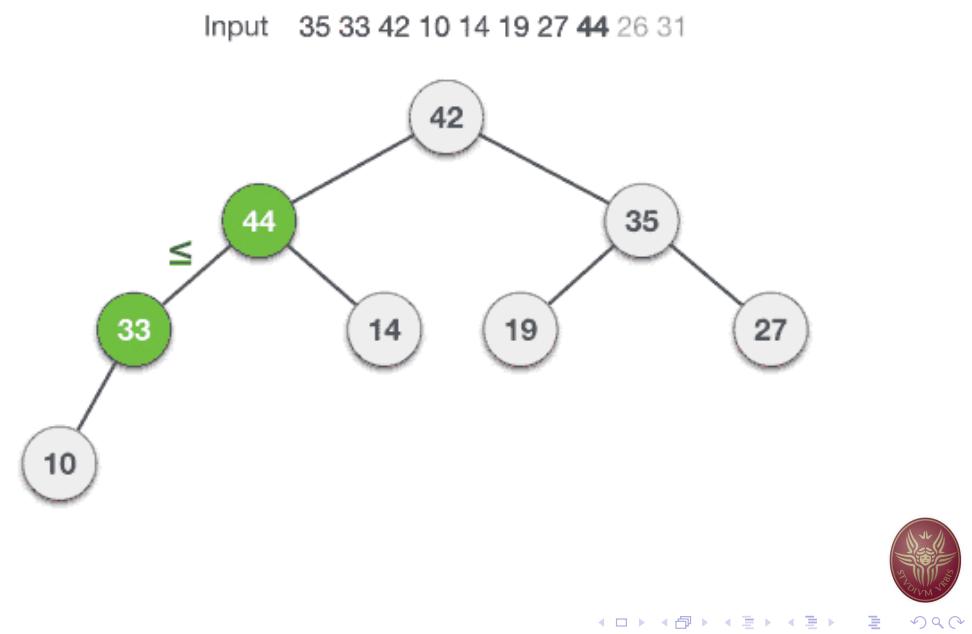
Max Heap – Add Example



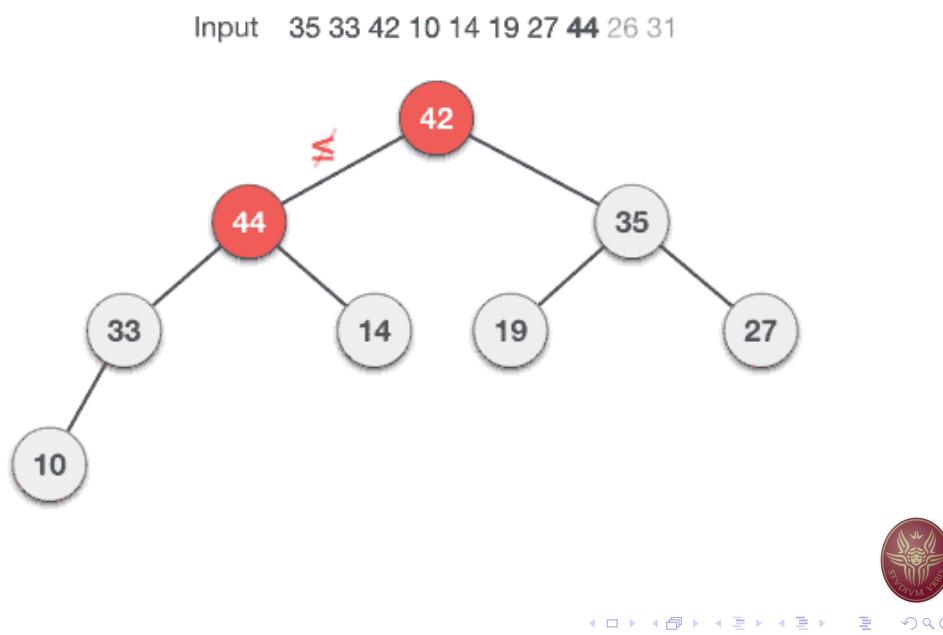
Max Heap – Add Example



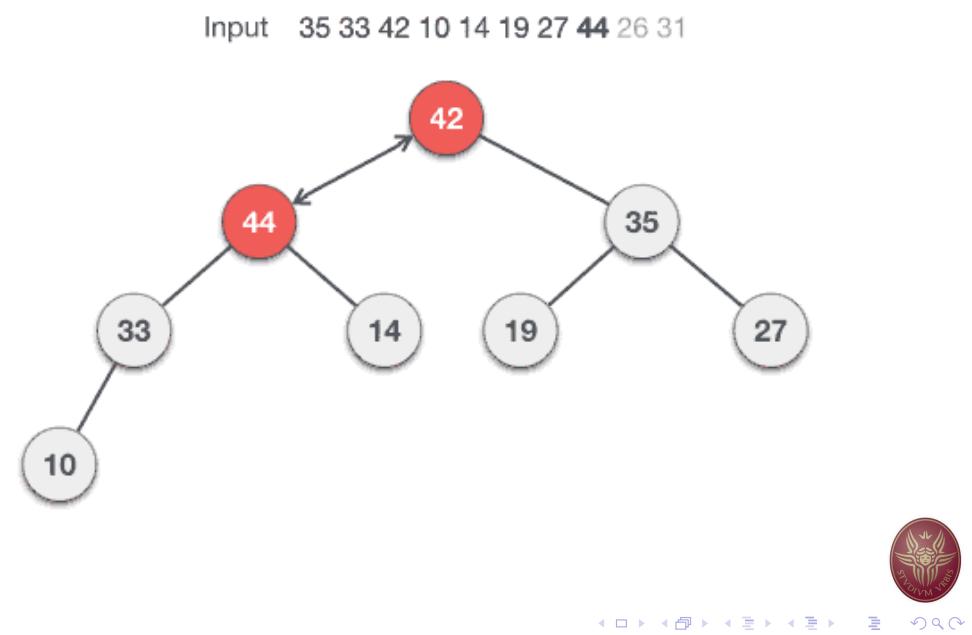
Max Heap – Add Example



Max Heap – Add Example

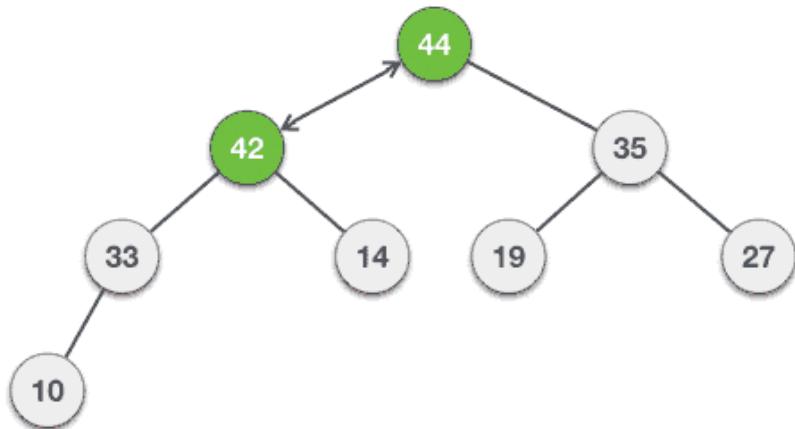


Max Heap – Add Example



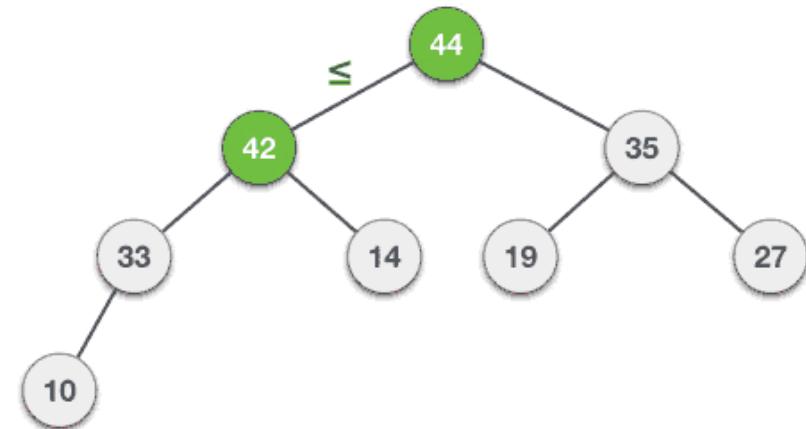
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31



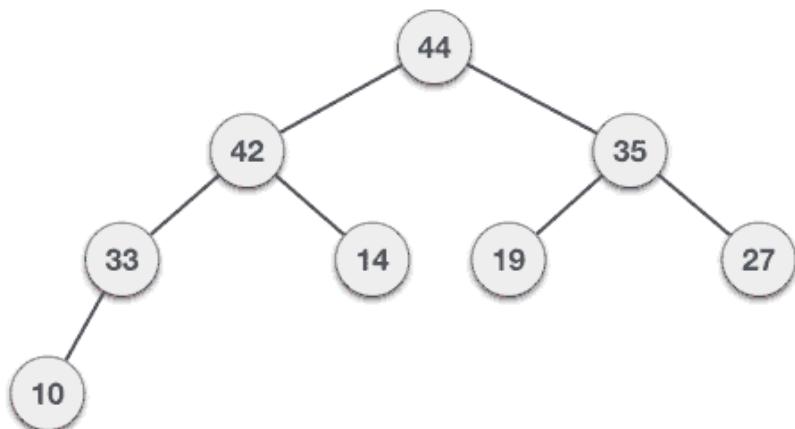
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31



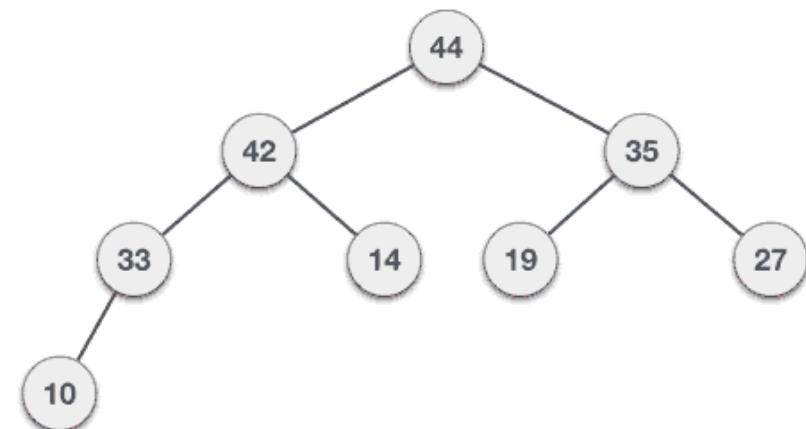
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31



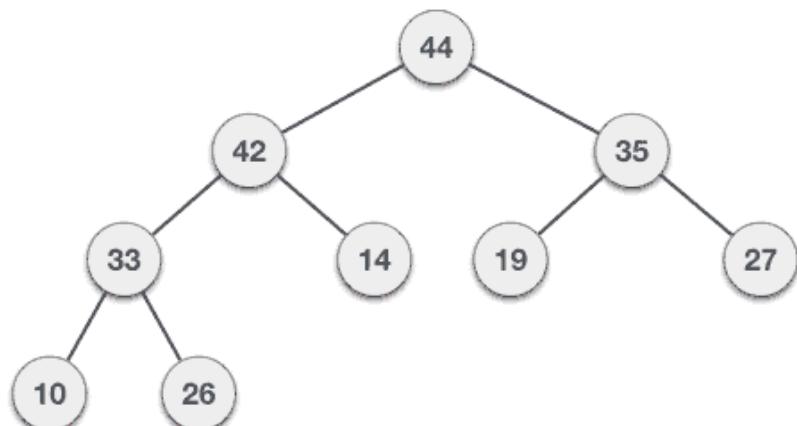
Max Heap – Add Example

Input 35 33 42 10 14 19 27 **44** 26 31



Max Heap – Add Example

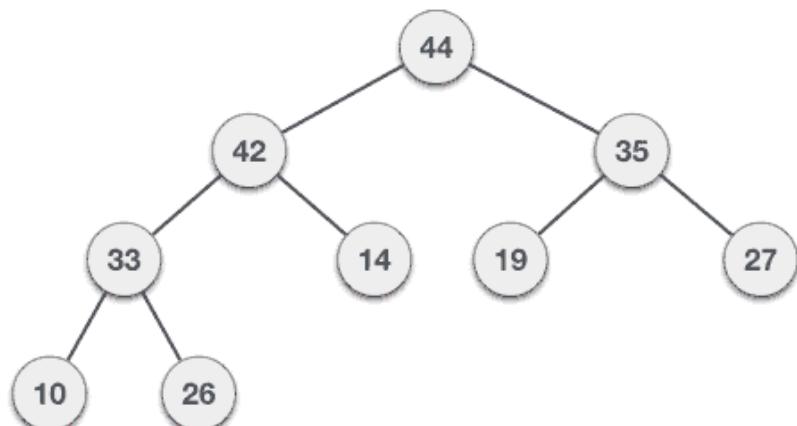
Input 35 33 42 10 14 19 27 44 **26** 31



A set of small, semi-transparent navigation icons located at the bottom of the slide, including arrows for navigation, a magnifying glass for search, and other symbols for specific functions.

Max Heap – Add Example

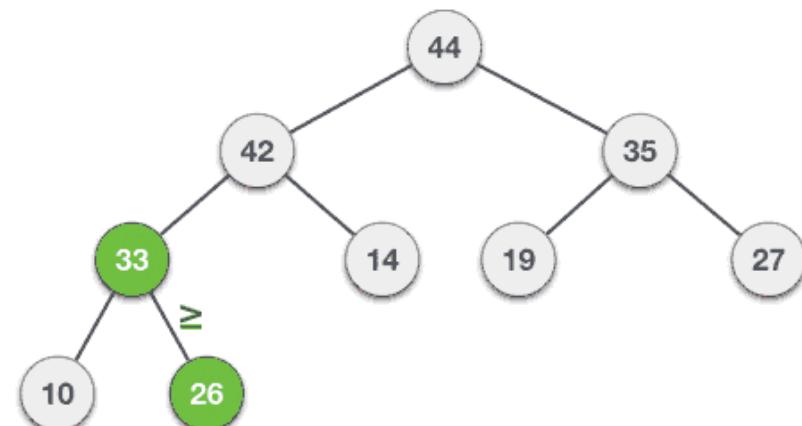
Input 35 33 42 10 14 19 27 44 **26** 31



A set of small, semi-transparent navigation icons located at the bottom of the slide, including arrows for navigation, a magnifying glass for search, and other symbols for specific functions.

Max Heap – Add Example

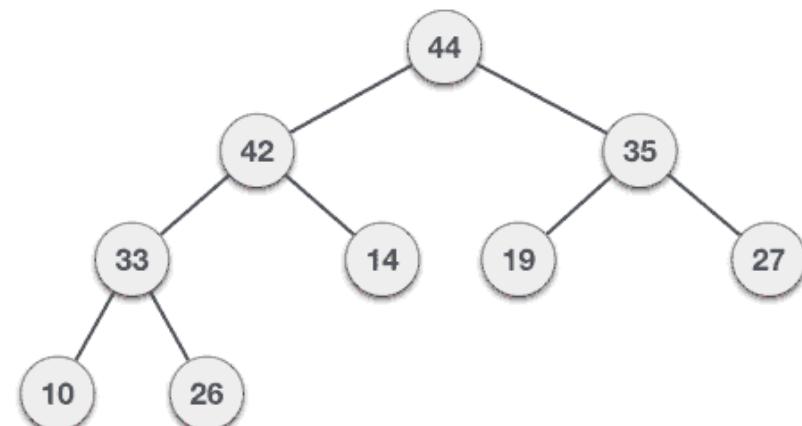
Input 35 33 42 10 14 19 27 44 **26** 31



A set of small, semi-transparent navigation icons located at the bottom of the slide, including arrows for navigation, a magnifying glass for search, and other symbols for specific functions.

Max Heap – Add Example

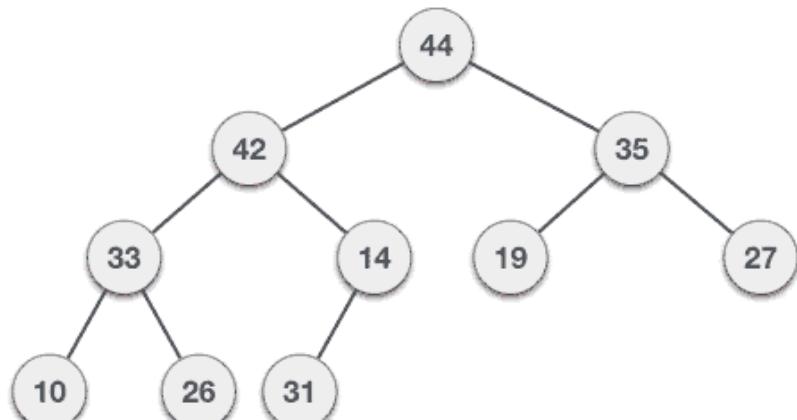
Input 35 33 42 10 14 19 27 44 26 **31**



A set of small, light-blue navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and table of contents.

Max Heap – Add Example

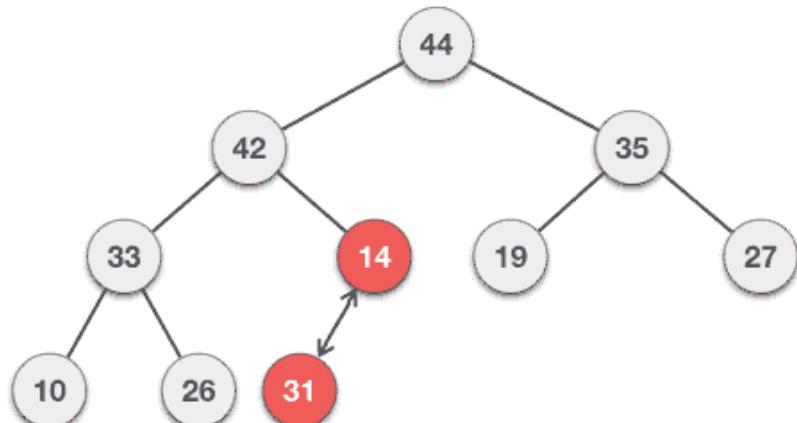
Input 35 33 42 10 14 19 27 44 26 31



A set of small, light-blue navigation icons typically found in presentation software like Beamer. The icons include symbols for back, forward, search, and other document-related functions.

Max Heap – Add Example

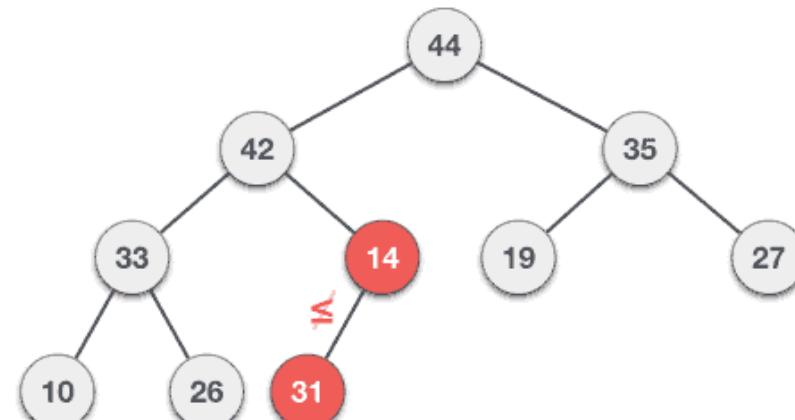
Input 35 33 42 10 14 19 27 44 26 31



A set of small, light-gray navigation icons typically found in presentation software like Beamer. From left to right, they include: a left arrow, a square, a right arrow, a double left arrow, a double square, a double right arrow, a triple left arrow, a triple right arrow, a horizontal ellipsis, a circular arrow, a magnifying glass, and a refresh/circular arrow icon.

Max Heap – Add Example

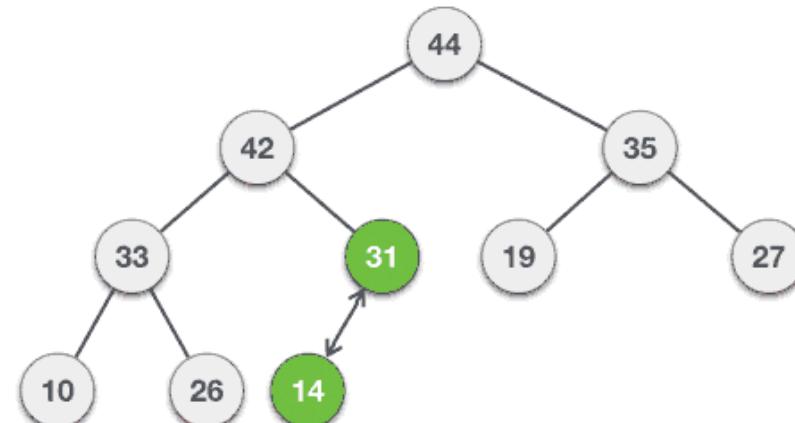
Input 35 33 42 10 14 19 27 44 26 **31**



A set of small, light-blue navigation icons typically found in presentation software like Beamer. The icons include symbols for back, forward, search, and other document-related functions.

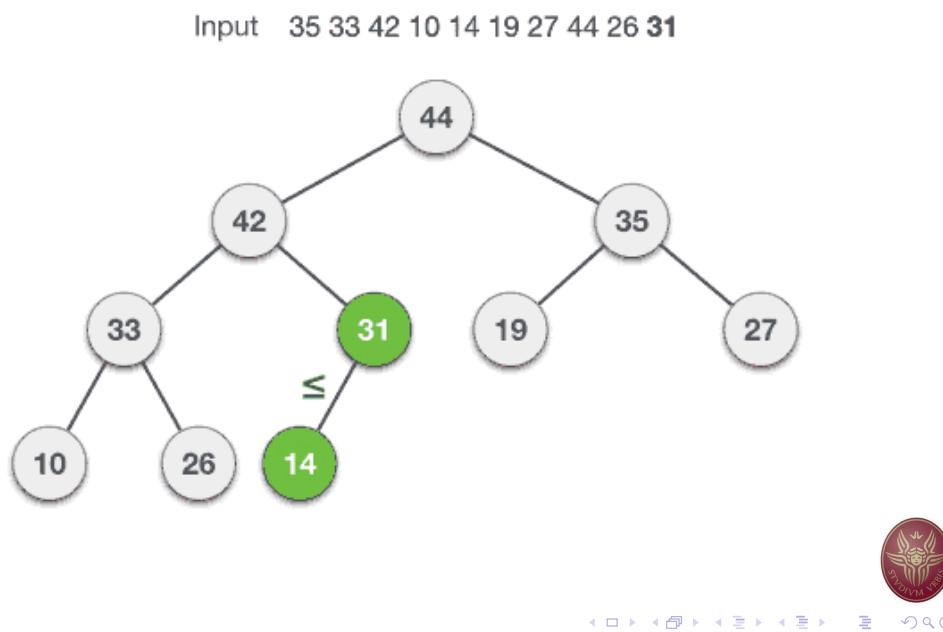
Max Heap – Add Example

Input 35 33 42 10 14 19 27 44 26 31

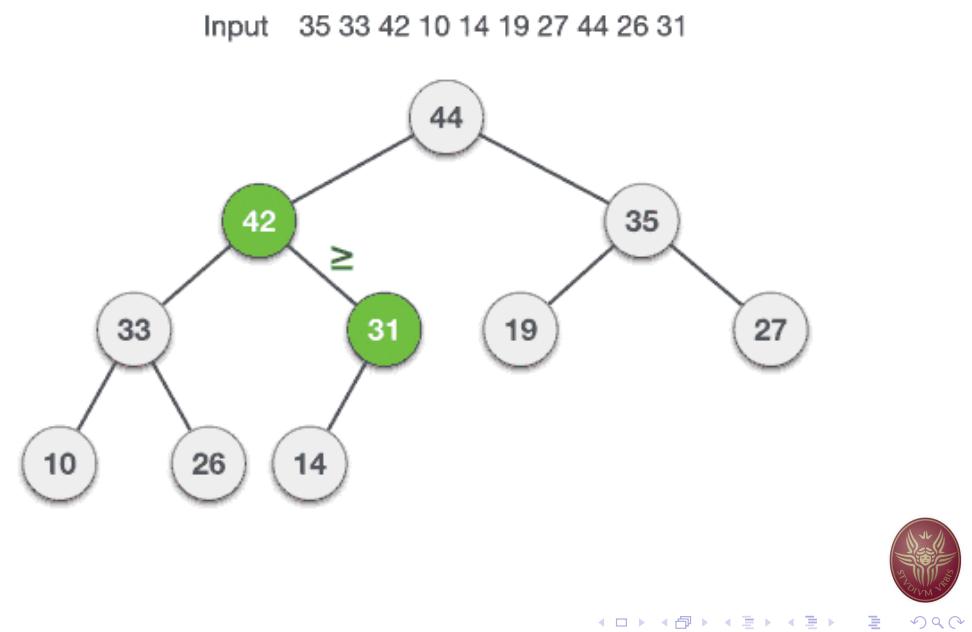


A set of small, light-blue navigation icons typically found in presentation software like Beamer. They include symbols for back, forward, search, and table of contents.

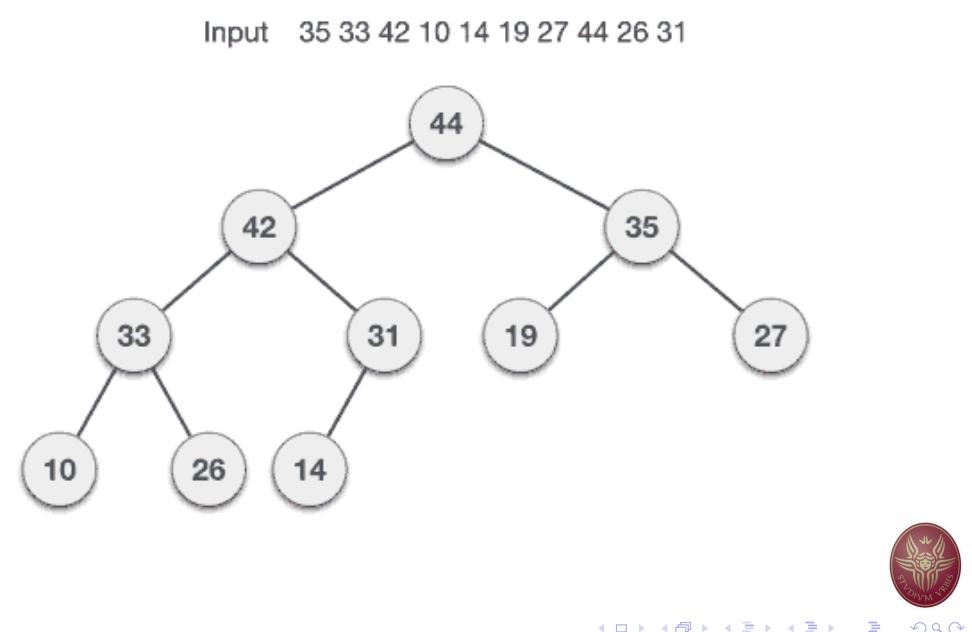
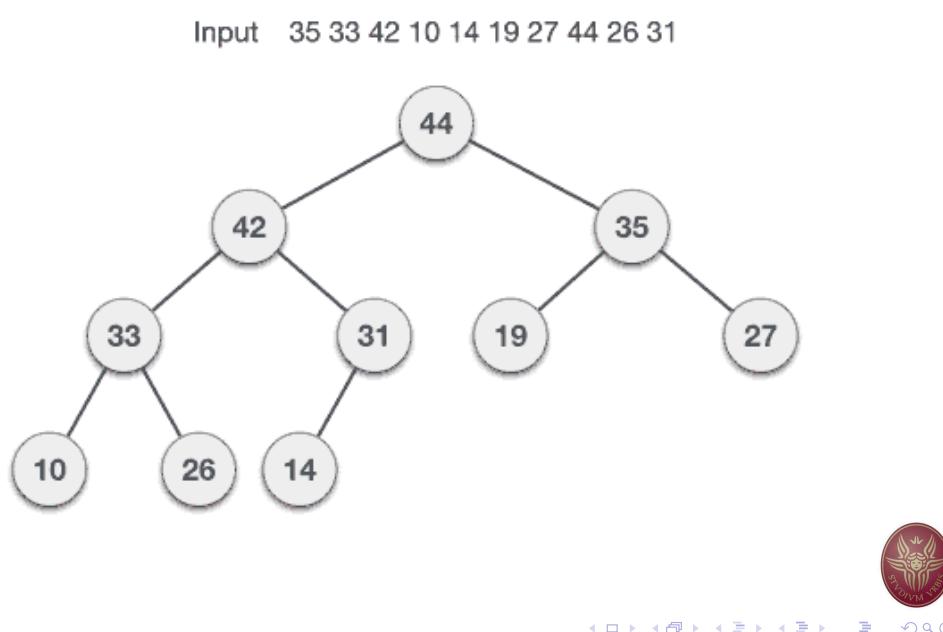
Max Heap – Add Example



Max Heap – Add Example

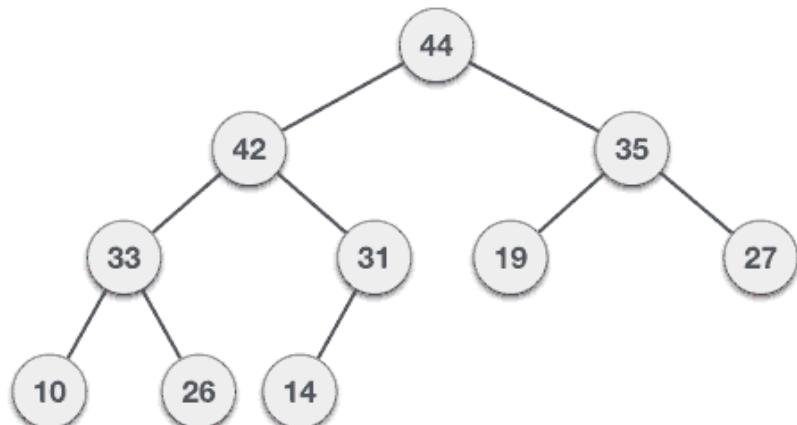


Max Heap – Add Example



Max Heap – Add Example

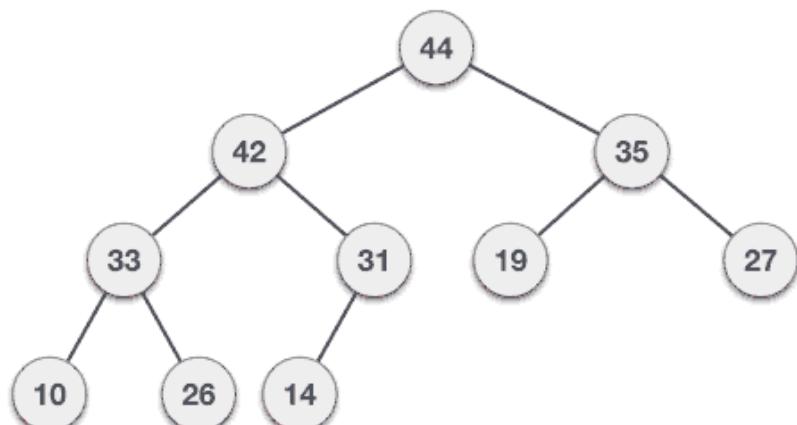
Input 35 33 42 10 14 19 27 44 26 31



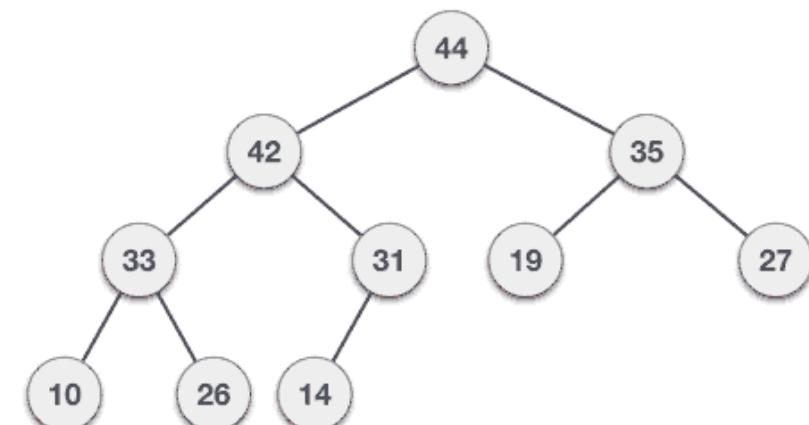
Max Heap – Deletion

1. Remove root node.
2. Move the last element of last level to root.
3. Compare the value of this child node with its parent.
4. If value of parent is less than child, then swap them.
5. Repeat step 3 & 4 until Heap property holds.

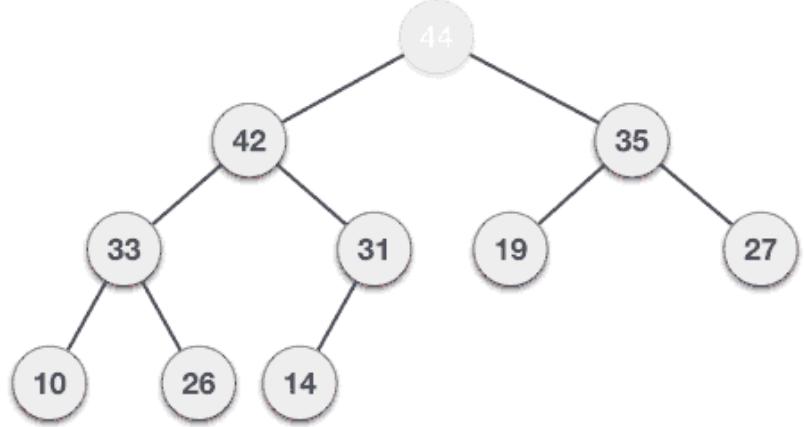
Max Heap – Deletion Example



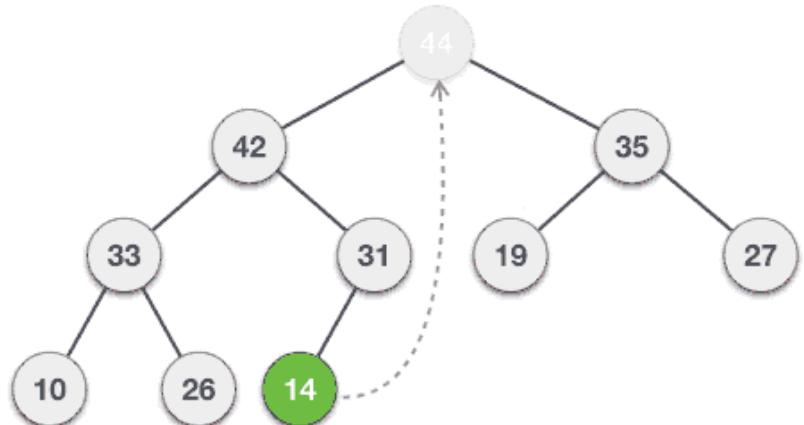
Max Heap – Deletion Example



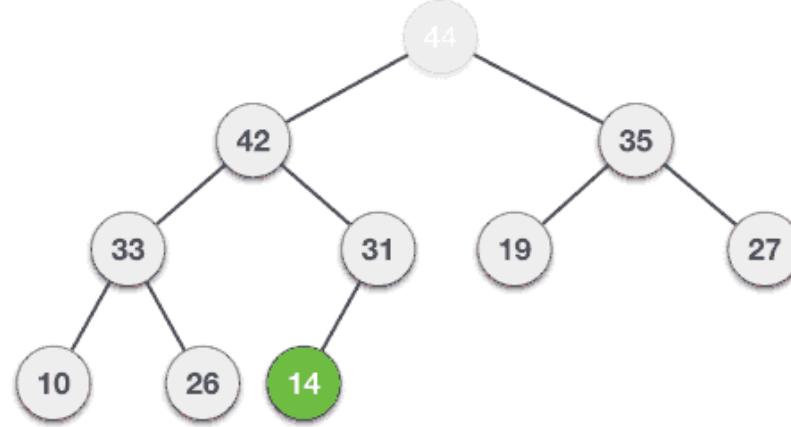
Max Heap – Deletion Example



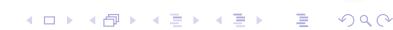
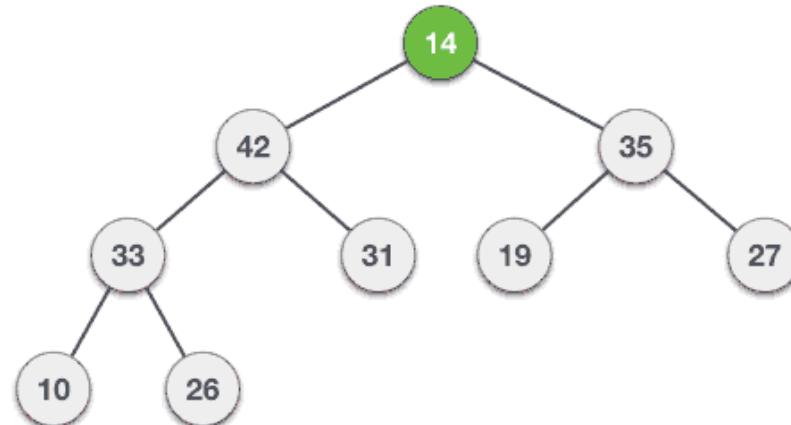
Max Heap – Deletion Example



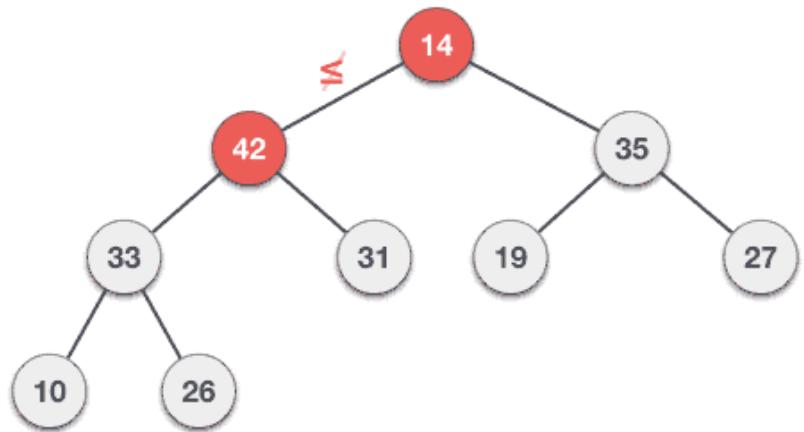
Max Heap – Deletion Example



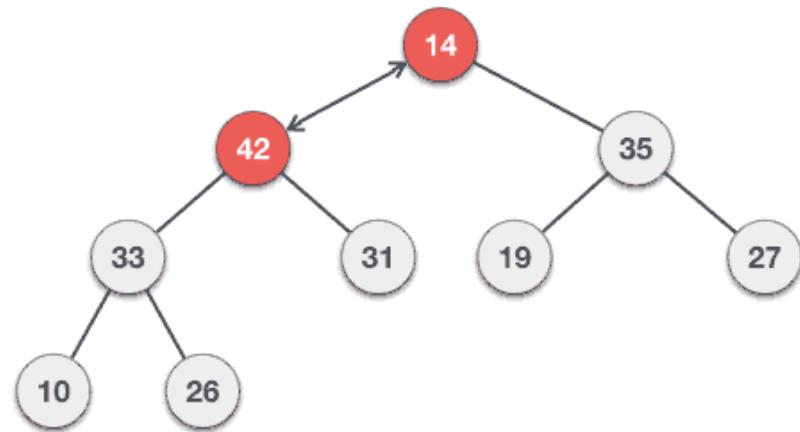
Max Heap – Deletion Example



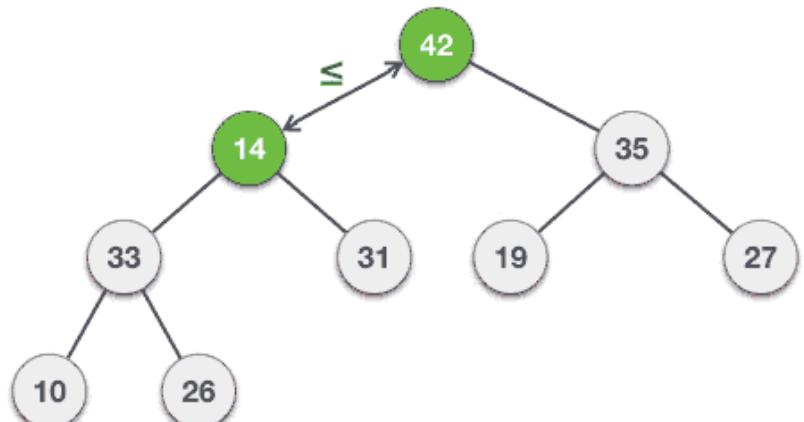
Max Heap – Deletion Example



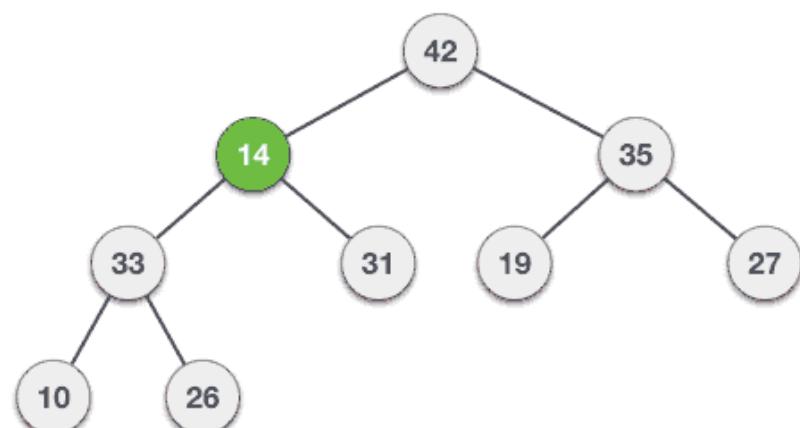
Max Heap – Deletion Example



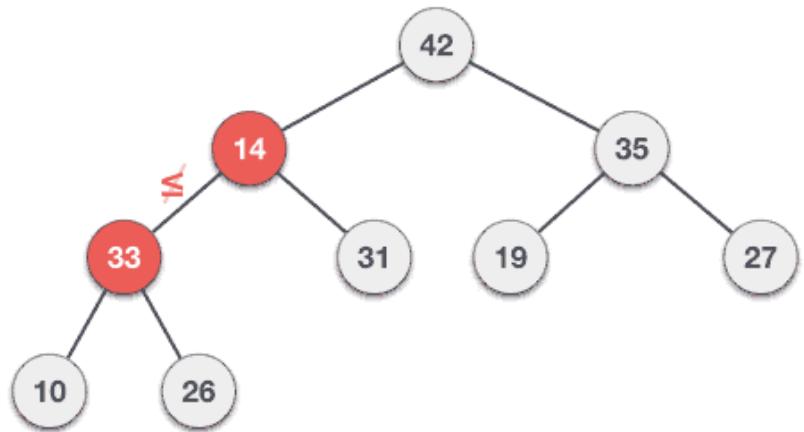
Max Heap – Deletion Example



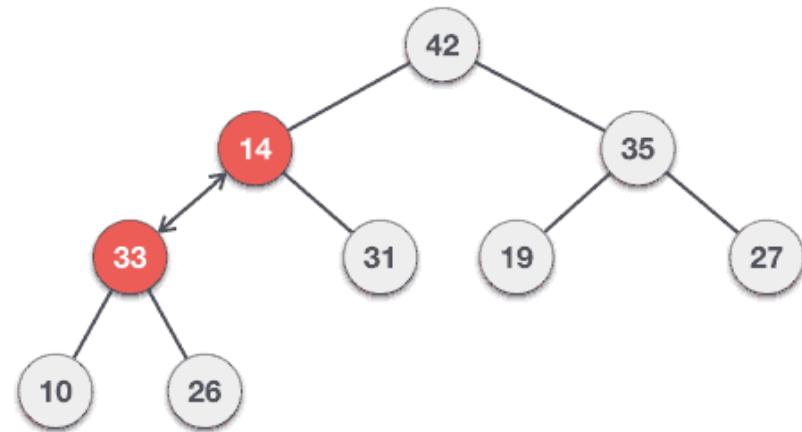
Max Heap – Deletion Example



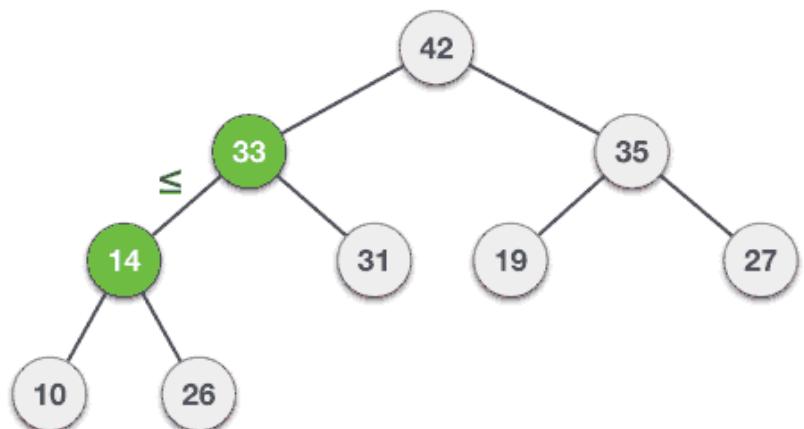
Max Heap – Deletion Example



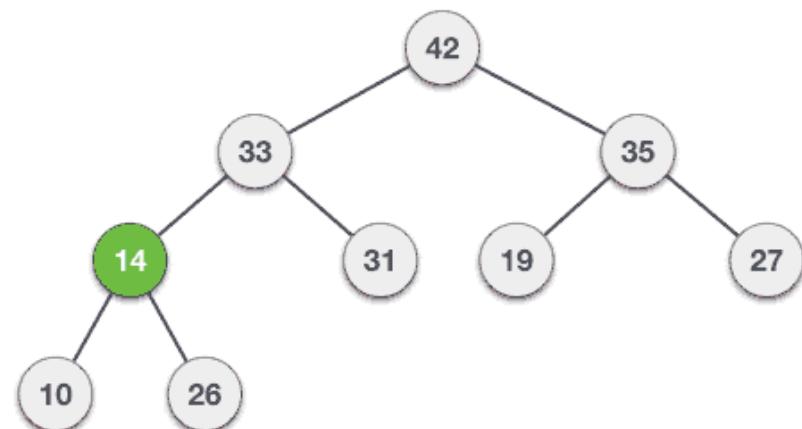
Max Heap – Deletion Example



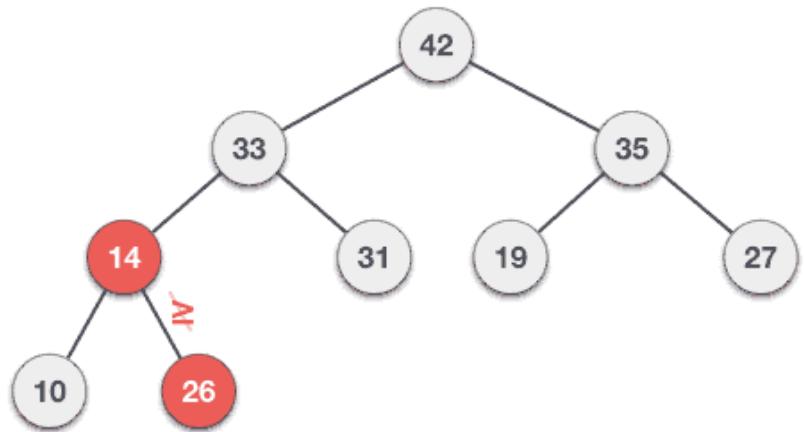
Max Heap – Deletion Example



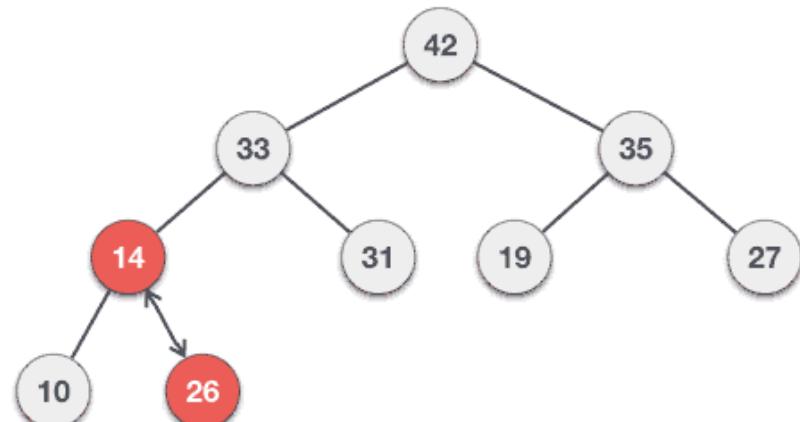
Max Heap – Deletion Example



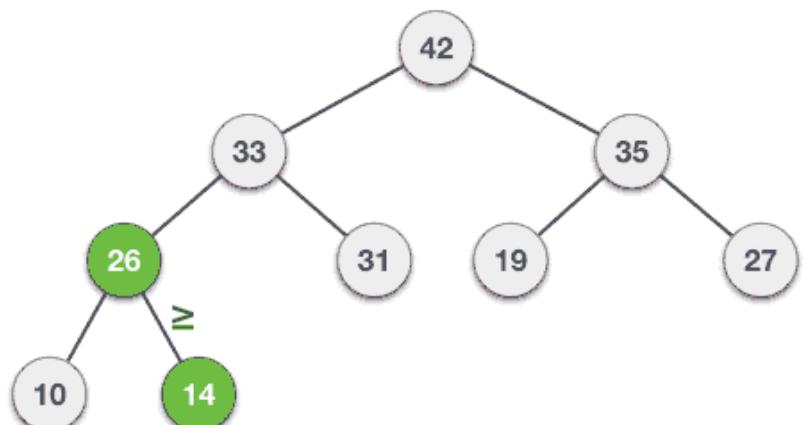
Max Heap – Deletion Example



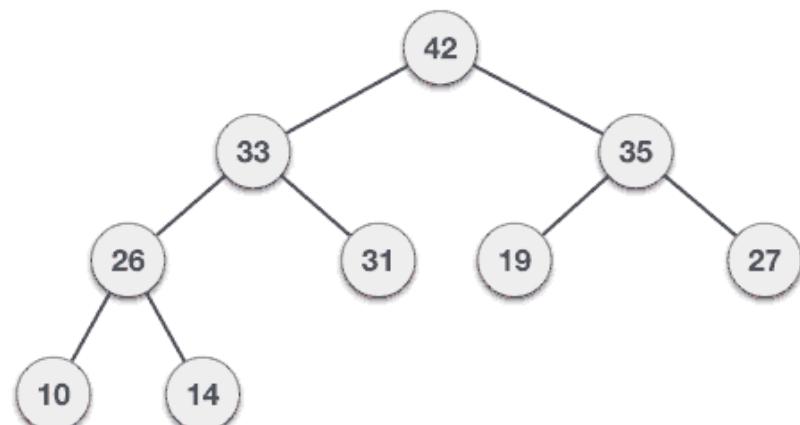
Max Heap – Deletion Example



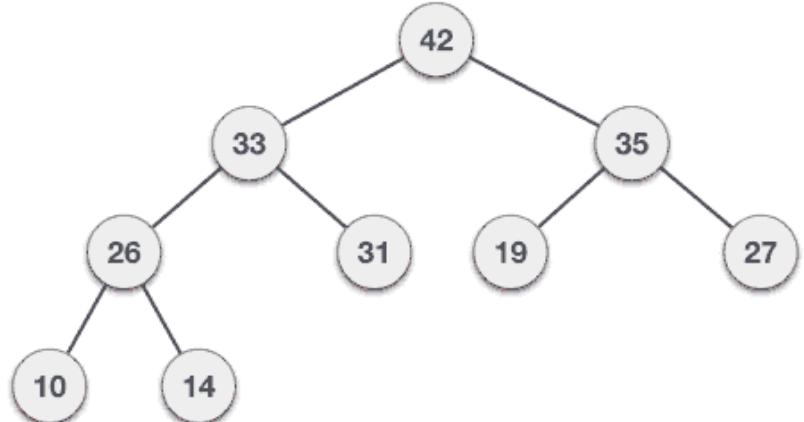
Max Heap – Deletion Example



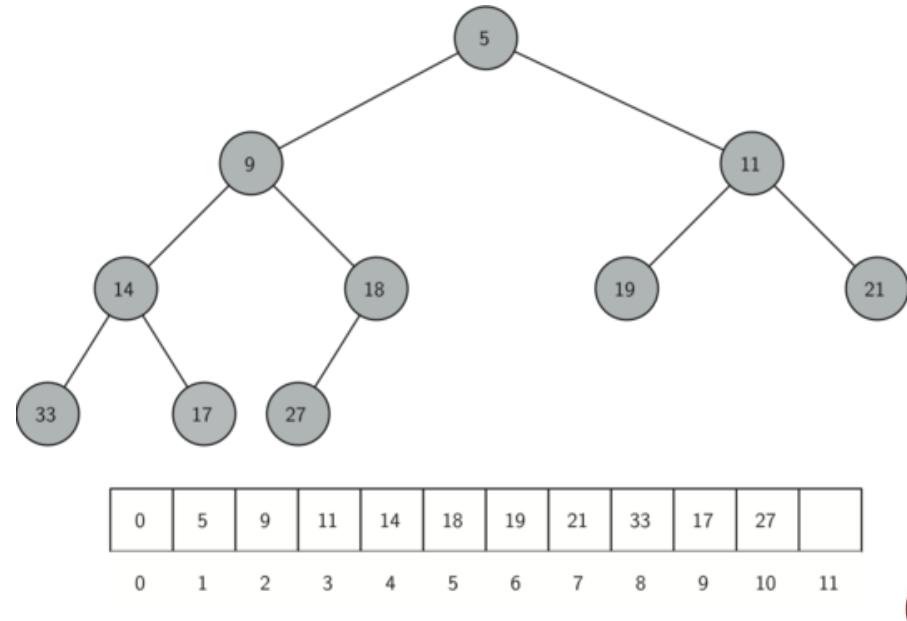
Max Heap – Deletion Example



Max Heap – Deletion Example



Heap Implementation using Arrays

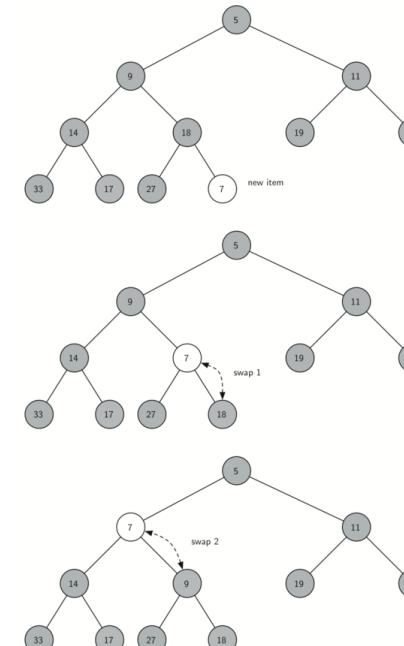


Binary Heap

```
1 class BinHeap:
2     def __init__(self):
3         self.heapList = [0]
4         self.currentSize = 0
```



Binary Heap – Insertion

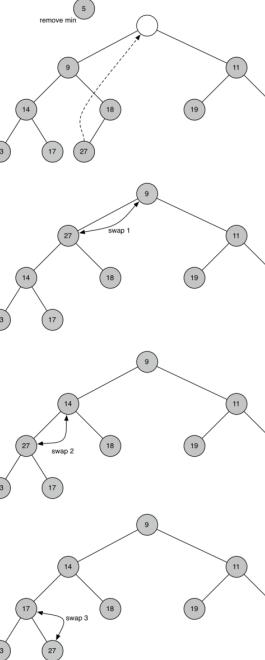


Binary Heap – Insertion

```
1 def percUp(self, i):
2     while i // 2 > 0:
3         if self.heapList[i] < self.heapList[i // 2]:
4             tmp = self.heapList[i // 2]
5             self.heapList[i // 2] = self.heapList[i]
6             self.heapList[i] = tmp
7         i = i // 2
8
9 def insert(self, k):
10    self.heapList.append(k)
11    self.currentSize = self.currentSize + 1
12    self_percUp(self.currentSize)
```



Binary Heap – Remove



Binary Heap – Remove

```
1 def percDown(self, i):
2     while (i * 2) <= self.currentSize:
3         mc = self.minChild(i)
4         if self.heapList[i] > self.heapList[mc]:
5             tmp = self.heapList[i]
6             self.heapList[i] = self.heapList[mc]
7             self.heapList[mc] = tmp
8         i = mc
9
10 def minChild(self, i):
11     if i * 2 + 1 > self.currentSize:
12         return i * 2
13     else:
14         if self.heapList[i*2] < self.heapList[i*2+1]:
15             return i * 2
16         else:
17             return i * 2 + 1
```

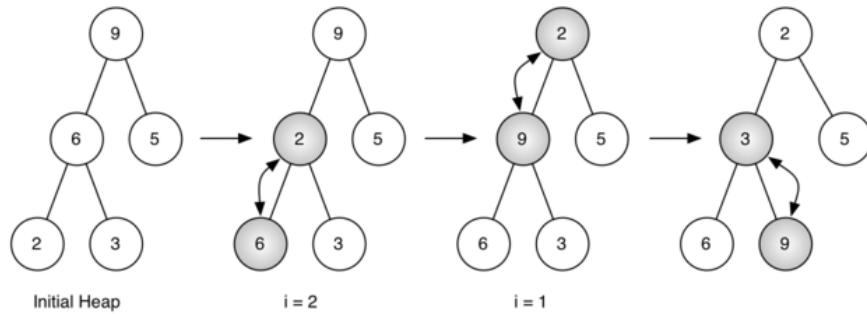


Binary Heap – Remove

```
1 def delMin(self):
2     retval = self.heapList[1]
3     self.heapList[1] = self.heapList[self.currentSize]
4     self.currentSize = self.currentSize - 1
5     self.heapList.pop()
6     self_percDown(1)
7     return retval
```



Binary Heap – Build from List



Binary Heap – Build from List

```
1 def buildHeap(self, alist):
2     i = len(alist) // 2
3     self.currentSize = len(alist)
4     self.heapList = [0] + alist [:]
5     while (i > 0):
6         self_percDown(i)
7         i = i - 1
```

