



Heap structure and applications

Data Structures and Algorithms

Tran Ngoc Bao Duy

*Faculty of Computer Science and Engineering
Ho Chi Minh University of Technology, VNU-HCM*

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Overview

Heap structure

① Heap Definition

Tran Ngoc Bao Duy



② Heap Structure

Heap Definition

③ Basic Algorithms

Heap Structure

ReheapUp

Basic Algorithms

ReheapDown

ReheapUp

ReheapDown

④ Heap Data Structure

Heap Data Structure

⑤ Heap Operations

Heap Operations

Build a Heap

Build a Heap

Insert a Node

Insert a Node

Delete a Node

Delete a Node

⑥ Heap Applications

Heap Applications

Selection Algorithms

Selection Algorithms

Priority Queues

Priority Queues

Heap Sort

Heap Sort



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

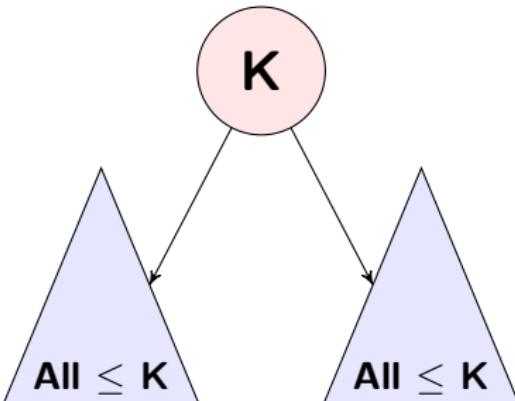
Heap Sort

Heap Definition

Definition

A **heap** (max-heap) is a binary tree structure with the following properties:

- ① The tree is complete or nearly complete.
- ② The key value of each node is **greater than or equal to** the key value in each of its descendants.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

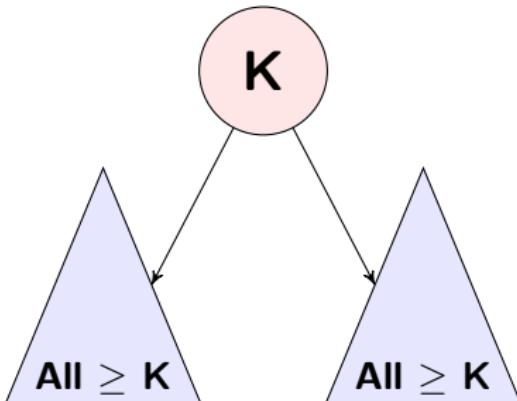


Heap Definition

Definition

A **min-heap** is a binary tree structure with the following properties:

- ① The tree is complete or nearly complete.
- ② The key value of each node is **less than or equal to** the key value in each of its descendants.



[Heap Definition](#)

[Heap Structure](#)

[Basic Algorithms](#)

[ReheapUp](#)

[ReheapDown](#)

[Heap Data Structure](#)

[Heap Operations](#)

[Build a Heap](#)

[Insert a Node](#)

[Delete a Node](#)

[Heap Applications](#)

[Selection Algorithms](#)

[Priority Queues](#)

[Heap Sort](#)



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

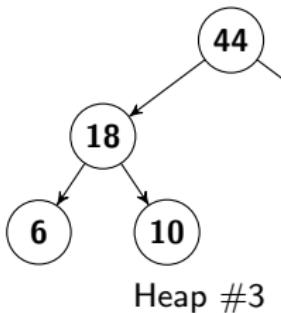
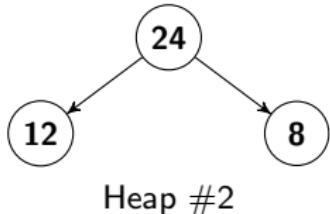
Heap Sort

Heap Structure

Heap trees

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

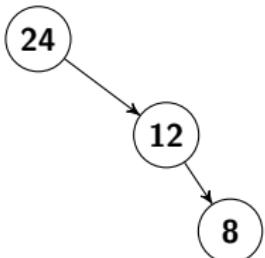
Selection Algorithms

Priority Queues

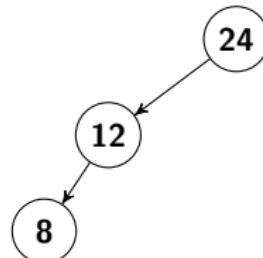
Heap Sort

Invalid Heaps

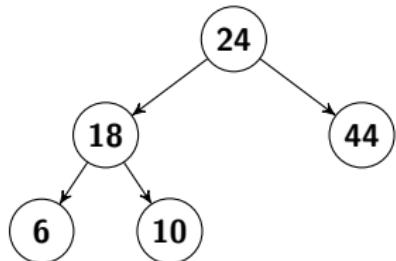
Heap structure



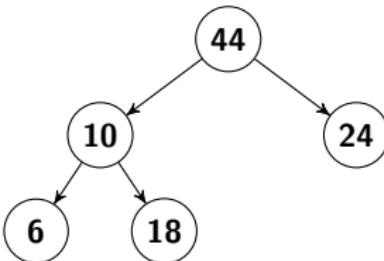
Not nearly complete



Not nearly complete



Root not largest



Subtree 10 not a heap

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

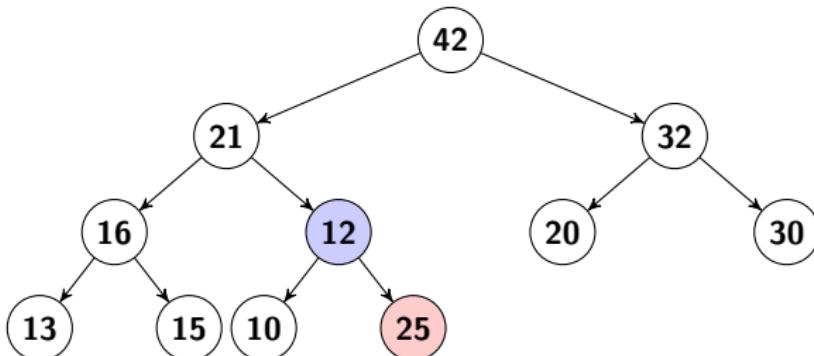
Priority Queues

Heap Sort

Basic Heap Algorithms

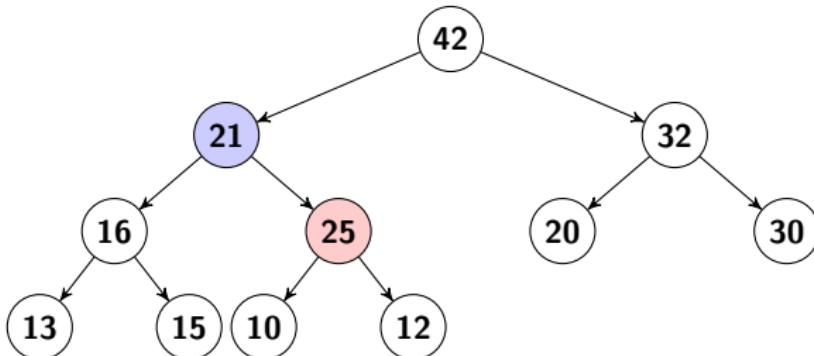
Definition

The [reheapUp](#) operation repairs a "broken" heap by [floating the last element up](#) the tree until it is in its correct location in the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

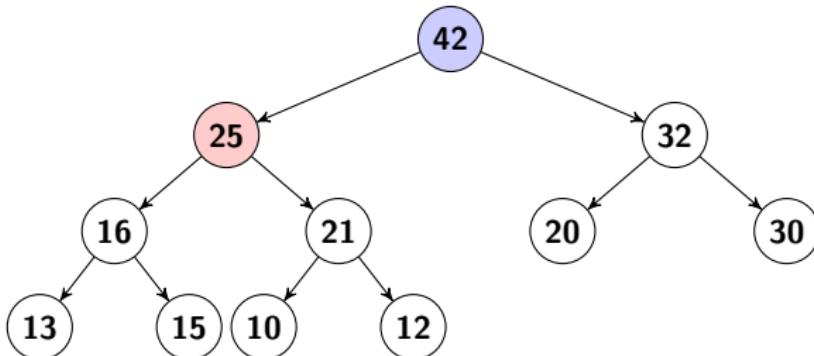
Definition

The [reheapUp](#) operation repairs a "broken" heap by [floating the last element up](#) the tree until it is in its correct location in the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

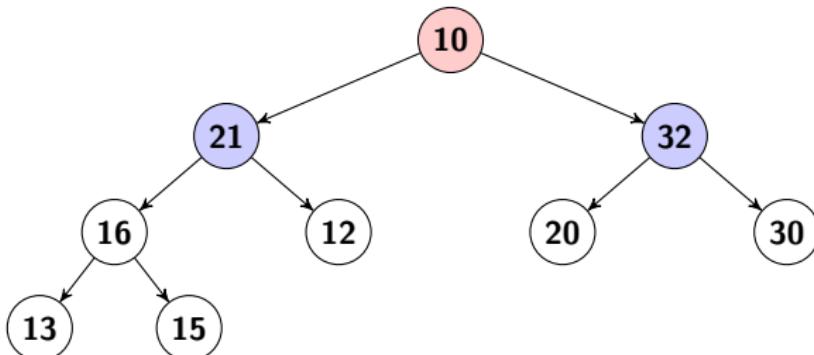
Definition

The [reheapUp](#) operation repairs a "broken" heap by [floating the last element up](#) the tree until it is in its correct location in the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

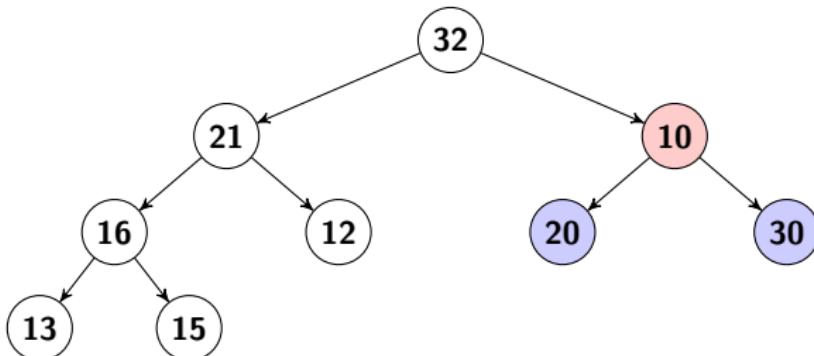
Definition

The [reheapDown](#) operation repairs a "broken" heap by [pushing the root down](#) the tree until it is in its correct location in the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

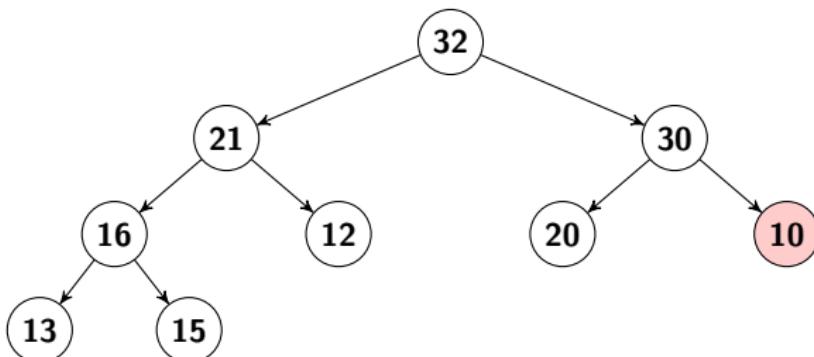
Definition

The [reheapDown](#) operation repairs a "broken" heap by [pushing the root down](#) the tree until it is in its correct location in the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Definition

The [reheapDown](#) operation repairs a "broken" heap by [pushing the root down](#) the tree until it is in its correct location in the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Heap Data Structure

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Properties of Heaps

- A complete or nearly complete binary tree.
- If the height is h , the number of nodes N is between 2^{h-1} and $2^h - 1$.
- **Complete tree:** $N = 2^h - 1$ when last level is full.
- **Nearly complete:** All nodes in the last level are on the left.

→ Heap can be represented in an array.

Heap in arrays

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

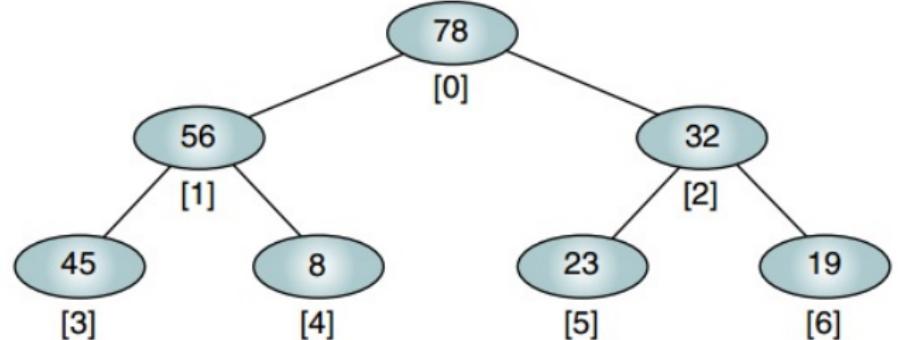
Delete a Node

Heap Applications

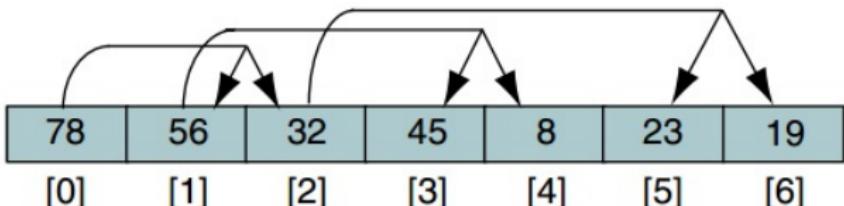
Selection Algorithms

Priority Queues

Heap Sort



(a) Heap in its logical form



(b) Heap in an array

(Source: Data Structures - A Pseudocode Approach with C++)

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Heap Data Structure

The relationship between a node and its children is fixed and can be calculated:

- ① For a node located at index i , its children are found at $2i + 1$ (left child) and $2i + 2$ (right child).
- ② The parent of a node located at index i is located at $\left\lfloor \frac{i - 1}{2} \right\rfloor$.
- ③ Given the index for a left child, j , its right sibling, if any, is found at $j + 1$. Conversely, given the index for a right child, k , its left sibling, which must exist, is found at $k - 1$.
- ④ Given the size, N , of a complete heap, the location of the first leaf is $\left\lfloor \frac{N}{2} \right\rfloor$.
- ⑤ Given the location of the first leaf element, the location of the last nonleaf element is 1 less.



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opeartions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Heap Opeartions



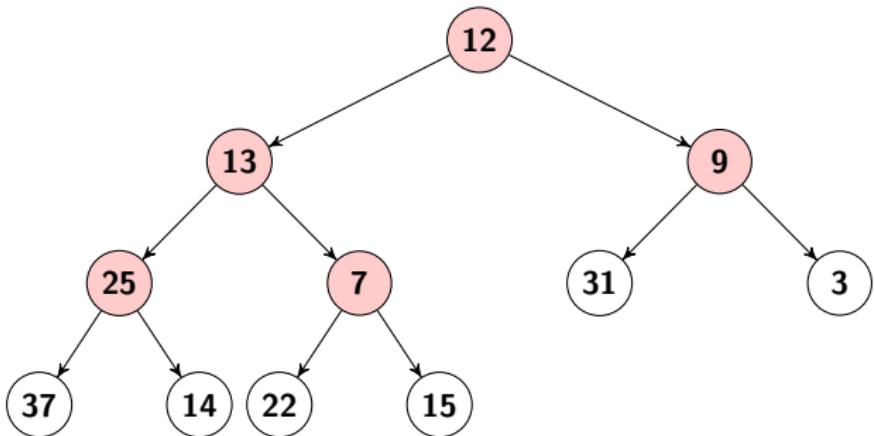
- Given a filled array of elements in random order, to build the heap we need to rearrange the data so that each node in the heap is greater than its children.
- We begin at the first non-leaf node, reheap down until the root. After that, we complete building a heap based on filled array.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	13	9	25	7	31	3	37	14	22	15
----	----	---	----	---	----	---	----	----	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

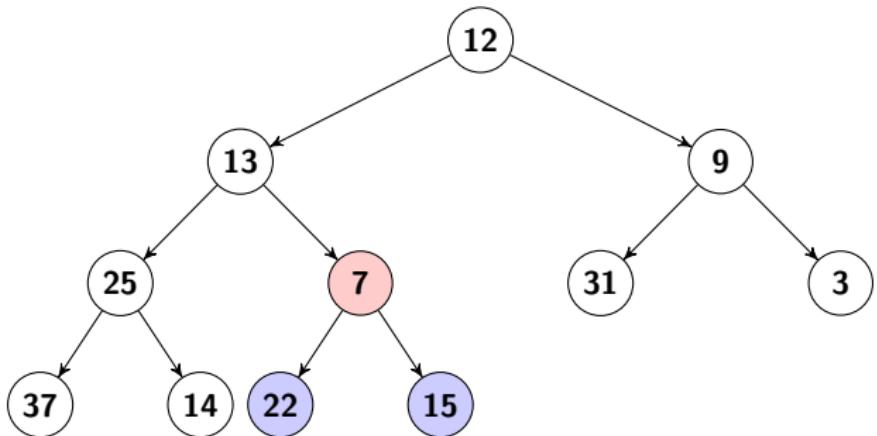
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	13	9	25	7	31	3	37	14	22	15
----	----	---	----	---	----	---	----	----	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

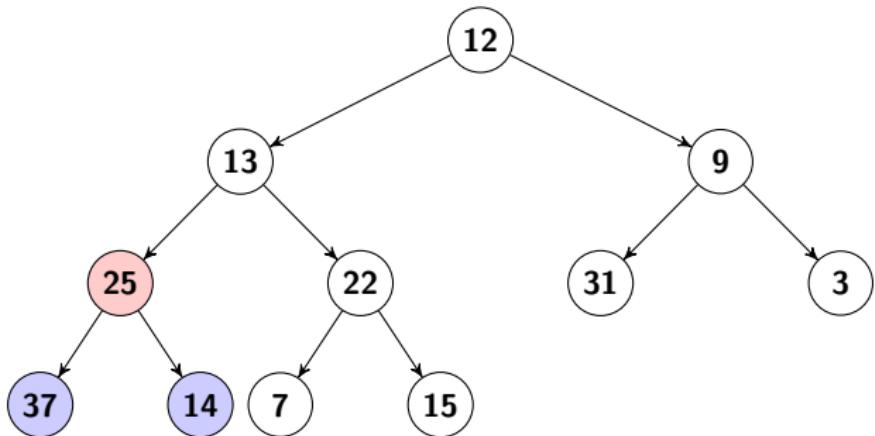
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	13	9	25	22	31	3	37	14	7	15

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

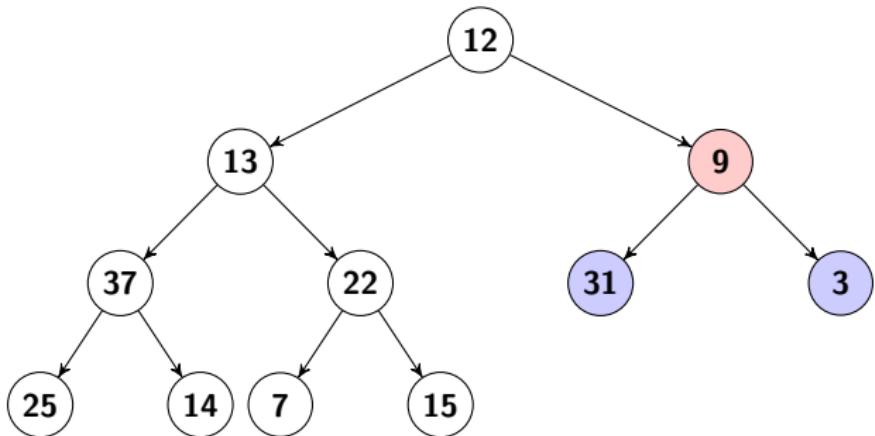
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	13	9	37	22	31	3	25	14	7	15
----	----	---	----	----	----	---	----	----	---	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

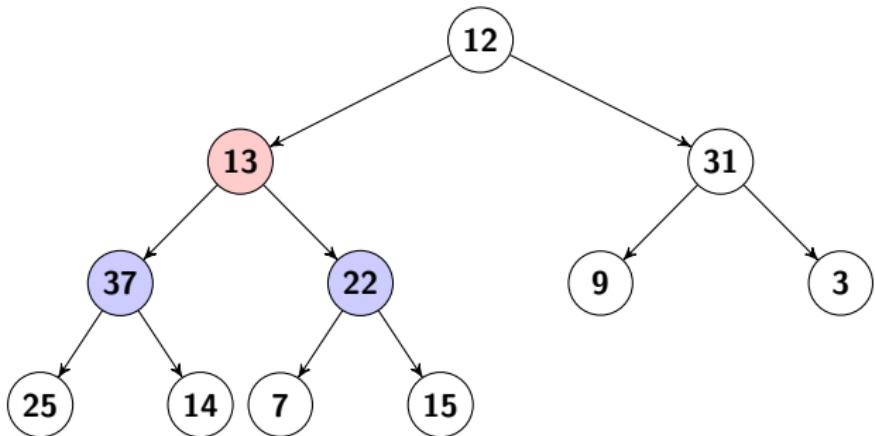
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	13	31	37	22	9	3	25	14	7	15
----	----	----	----	----	---	---	----	----	---	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

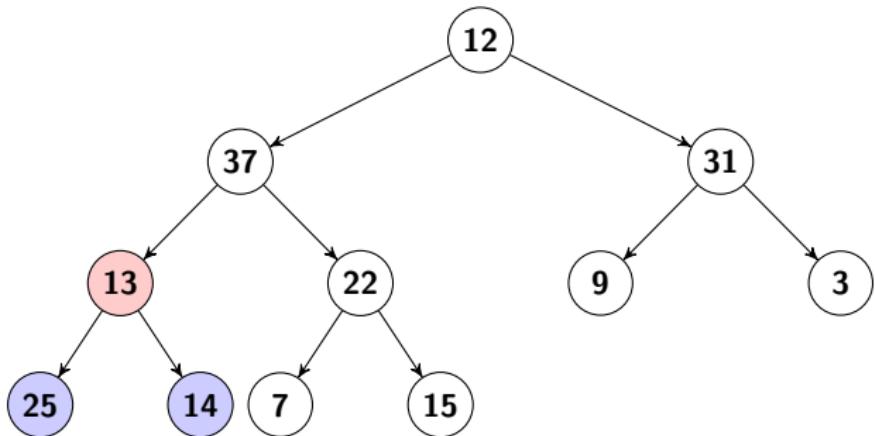
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	37	31	13	22	9	3	25	14	7	15

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

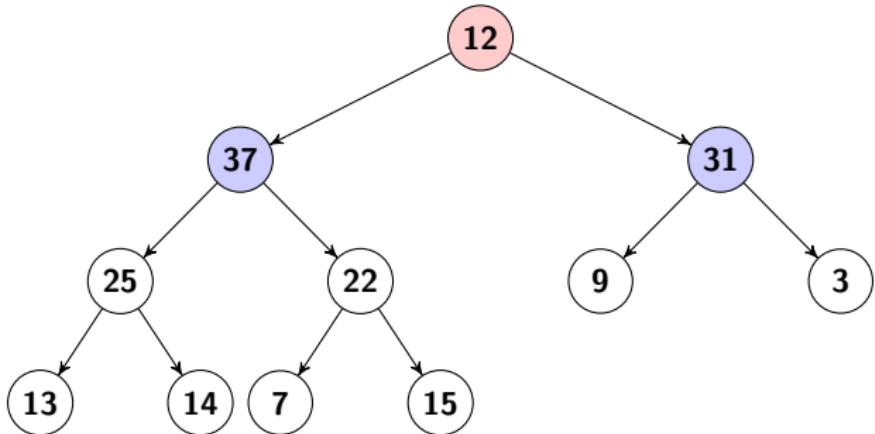
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



12	37	31	25	22	9	3	13	14	7	15
----	----	----	----	----	---	---	----	----	---	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

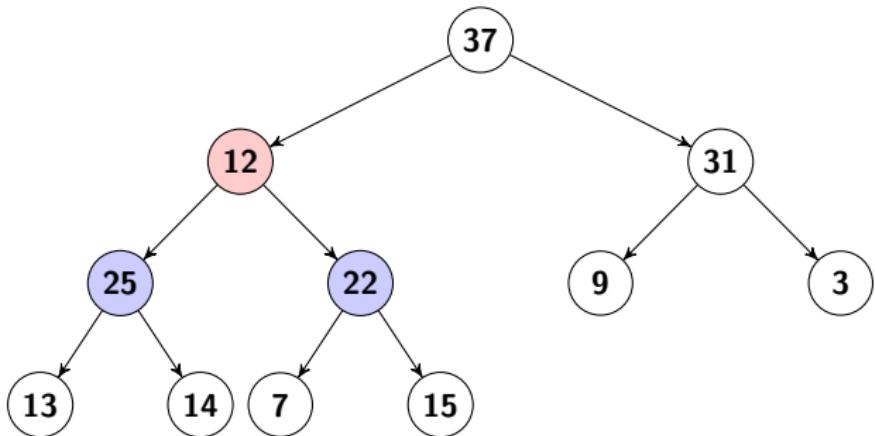
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



37	12	31	25	22	9	3	13	14	7	15
----	----	----	----	----	---	---	----	----	---	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

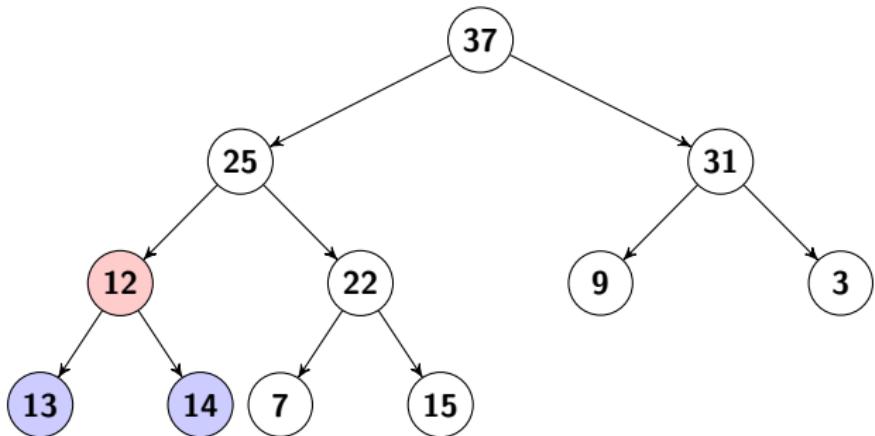
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



37	25	31	12	22	9	3	13	14	7	15
----	----	----	----	----	---	---	----	----	---	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

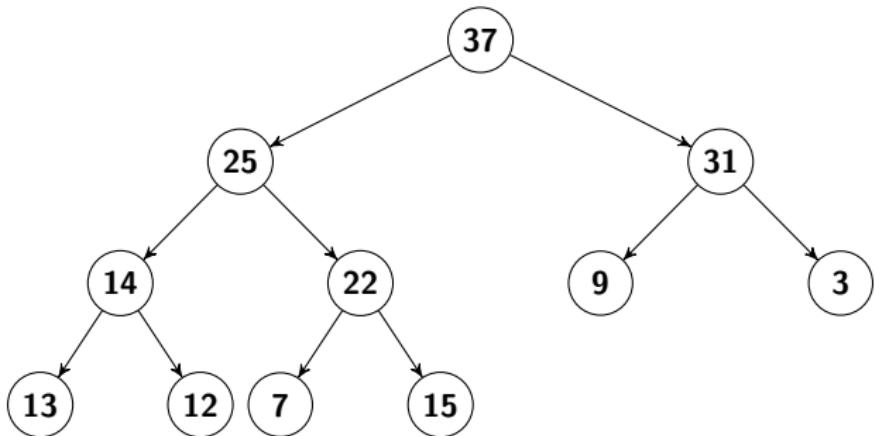
Priority Queues

Heap Sort

Build a Heap

Heap structure

Tran Ngoc Bao Duy



37	25	31	14	22	9	3	13	12	7	15
----	----	----	----	----	---	---	----	----	---	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Insert a Node into a Heap

Heap structure

Tran Ngoc Bao Duy



[Heap Definition](#)

[Heap Structure](#)

[Basic Algorithms](#)

ReheapUp

ReheapDown

[Heap Data Structure](#)

[Heap Opearitions](#)

Build a Heap

Insert a Node

Delete a Node

[Heap Applications](#)

Selection Algorithms

Priority Queues

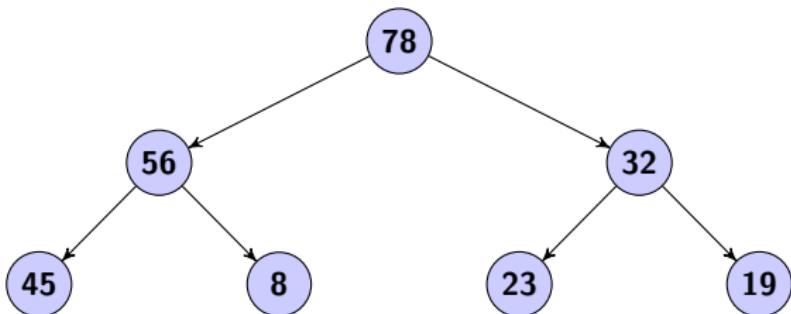
Heap Sort

- To insert a node, we need to locate the first empty leaf in the array.
- We find it immediately after the last node in the tree, which is given as a parameter.
- To insert a node, we move the new data to the first empty leaf and reheap up.

Insert a Node into a Heap

Heap structure

Tran Ngoc Bao Duy



78	56	32	45	8	23	19
----	----	----	----	---	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

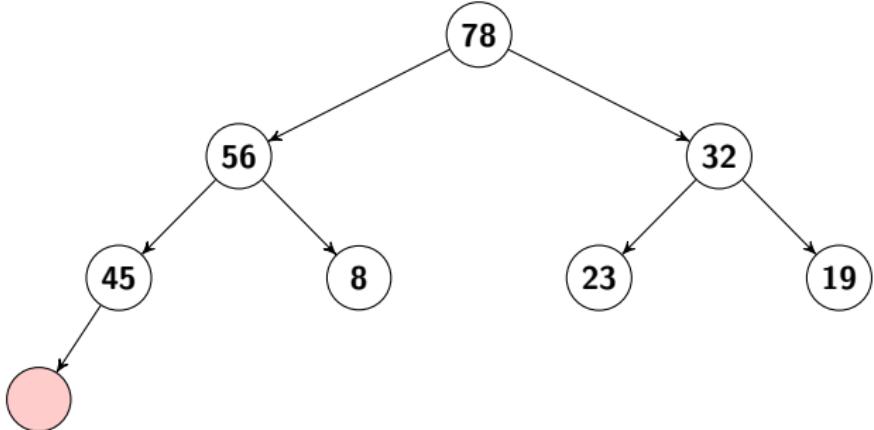
Delete a Node

Heap Applications

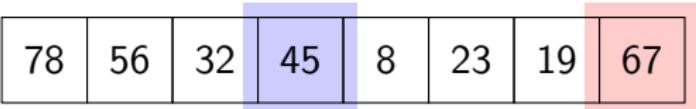
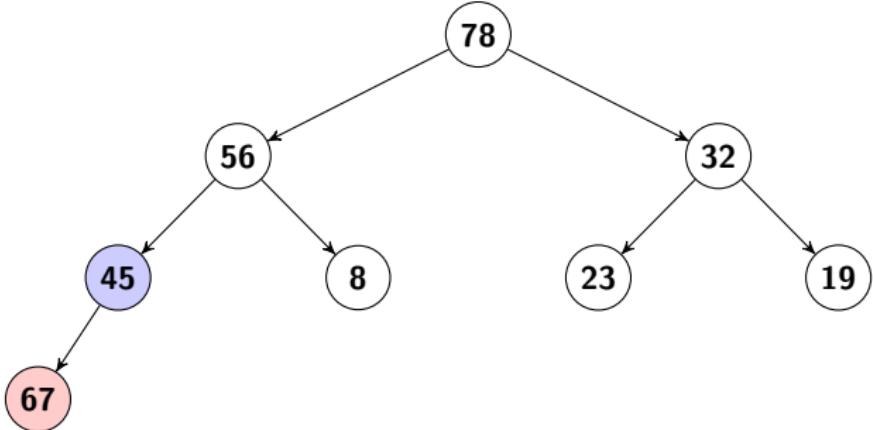
Selection Algorithms

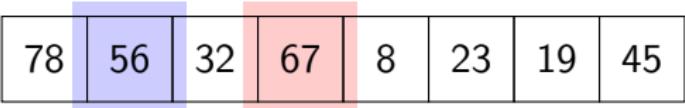
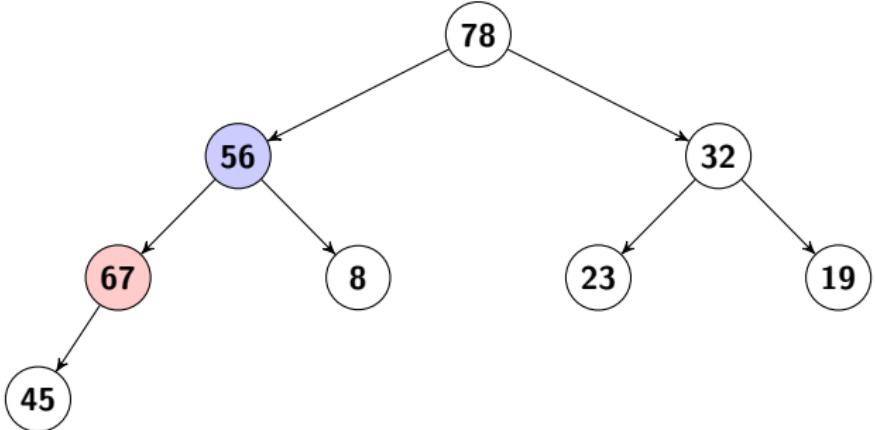
Priority Queues

Heap Sort

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

78	56	32	45	8	23	19	
----	----	----	----	---	----	----	--

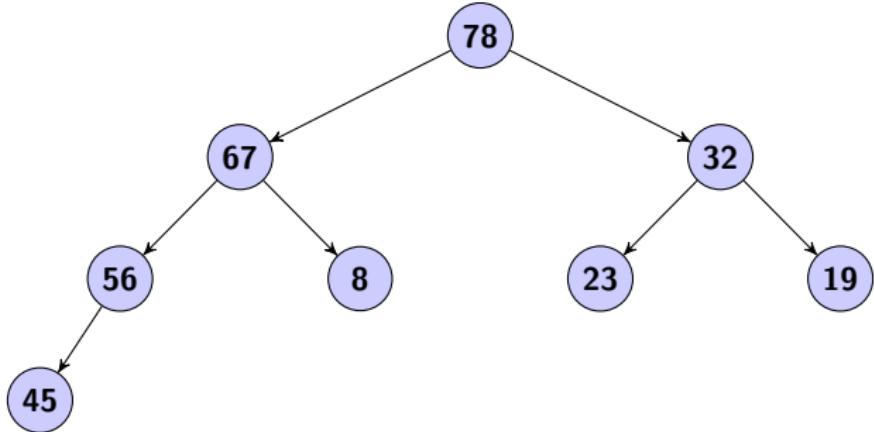
[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Insert a Node into a Heap

Heap structure

Tran Ngoc Bao Duy



78	56	32	67	8	23	19	45
----	----	----	----	---	----	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Delete a Node from a Heap

Heap structure

Tran Ngoc Bao Duy



[Heap Definition](#)

[Heap Structure](#)

[Basic Algorithms](#)

ReheapUp

ReheapDown

[Heap Data Structure](#)

[Heap Opearitions](#)

Build a Heap

Insert a Node

[Delete a Node](#)

[Heap Applications](#)

Selection Algorithms

Priority Queues

Heap Sort

- When deleting a node from a heap, the most common and meaningful logic is to delete the root.
- After it has been deleted, the heap is thus left without a root.
- To reestablish the heap, we move the data in the last heap node to the root and reheap down.

Delete a Node from a Heap

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

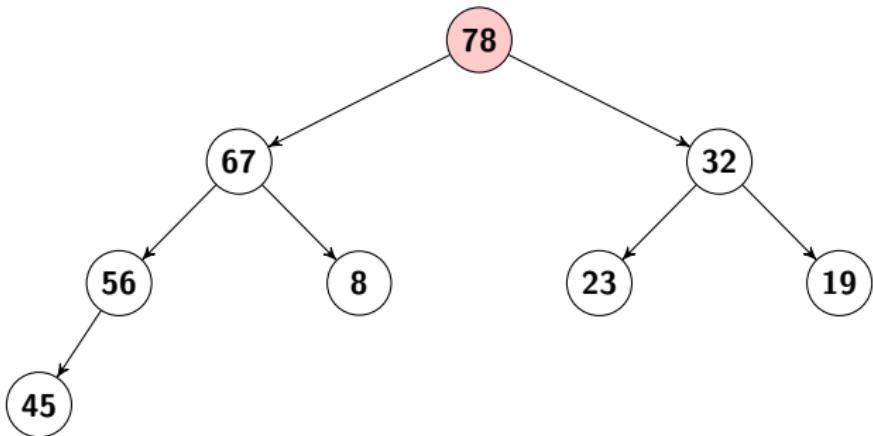
Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

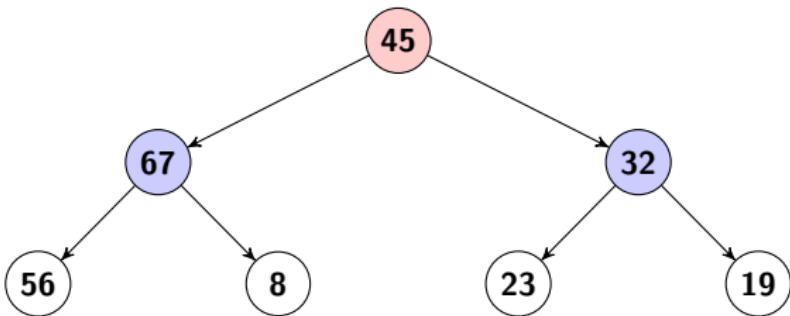


78	67	32	56	8	23	19	45
----	----	----	----	---	----	----	----

Delete a Node from a Heap

Heap structure

Tran Ngoc Bao Duy



45	67	32	56	8	23	19
----	----	----	----	---	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

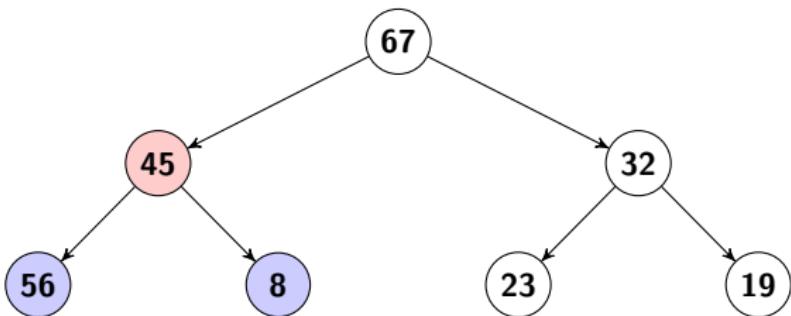
Priority Queues

Heap Sort

Delete a Node from a Heap

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

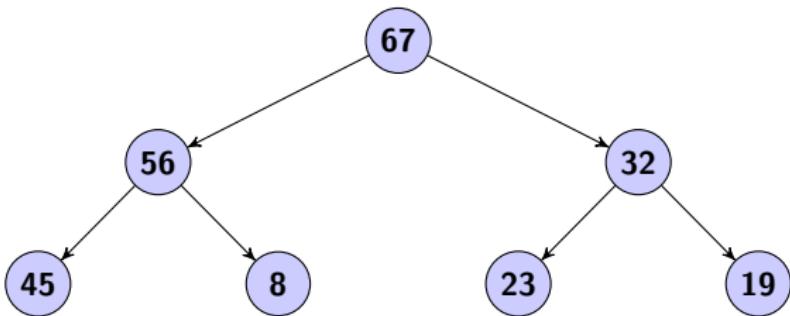
Priority Queues

Heap Sort

Delete a Node from a Heap

Heap structure

Tran Ngoc Bao Duy



67	56	32	45	8	23	19
----	----	----	----	---	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Complexity of Binary Heap Operations

Heap structure

Tran Ngoc Bao Duy



- ReheapUp: $O(\log_2 n)$
- ReheapDown: $O(\log_2 n)$
- Build a Heap: $O(n)$.
- Insert a Node into a Heap: $O(\log_2 n)$
- Delete a Node from a Heap: $O(\log_2 n)$

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Complexity of Binary Heap Operations

Heap structure

Tran Ngoc Bao Duy



- ReheapUp: $O(\log_2 n)$
- ReheapDown: $O(\log_2 n)$
- Build a Heap: $O(n)$. Why?
- Insert a Node into a Heap: $O(\log_2 n)$
- Delete a Node from a Heap: $O(\log_2 n)$

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

Heap Applications



Three common applications of heaps are:

- ① selection algorithms,
- ② priority queues,
- ③ and sorting.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Problem

Determining the k^{th} element in an unsorted list.

Two solutions:

- ① Sort the list and select the element at location k . The complexity of a simple sorting algorithm is $O(n^2)$.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Opearitions](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Problem

Determining the k^{th} element in an unsorted list.

Two solutions:

- ① Sort the list and select the element at location k . The complexity of a simple sorting algorithm is $O(n^2)$.
- ② Create a heap and delete $k - 1$ elements from the heap, leaving the desired element at the top. The complexity is $O(n \log_2 n)$.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Problem

Determining the k^{th} element in an unsorted list.

Two solutions:

- ① Sort the list and select the element at location k . The complexity of a simple sorting algorithm is $O(n^2)$.
- ② Create a heap and delete $k - 1$ elements from the heap, leaving the desired element at the top. The complexity is $O(n \log_2 n)$.

Rather than simply discarding the elements at the top of the heap, a better solution would be to place the deleted element at the end of the heap and reduce the size by 1.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Problem

Determining the k^{th} element in an unsorted list.

Two solutions:

- ① Sort the list and select the element at location k . The complexity of a simple sorting algorithm is $O(n^2)$.
- ② Create a heap and delete $k - 1$ elements from the heap, leaving the desired element at the top. The complexity is $O(n \log_2 n)$.

Rather than simply discarding the elements at the top of the heap, a better solution would be to place the deleted element at the end of the heap and reduce the size by 1.

After the k^{th} element has been processed, the temporarily removed elements can then be inserted into the heap.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)



Priority Queues

The heap is an excellent structure to use for a priority queue.

Example

Assume that we have a priority queue with three priorities: high (3), medium (2), and low (1).

Of the first five customers who arrive, the second and the fifth are high-priority customers, the third is medium priority, and the first and the fourth are low priority.

Arrival	Priority	Priority
1	low	$1999 \ (1 \ \& \ (1000 - 1))$
2	high	$3998 \ (3 \ \& \ (1000 - 2))$
3	medium	$2997 \ (2 \ \& \ (1000 - 3))$
4	low	$1996 \ (1 \ \& \ (1000 - 4))$
5	high	$3995 \ (3 \ \& \ (1000 - 5))$

(Source: Data Structures - A Pseudocode Approach with C++)

[Heap Definition](#)

[Heap Structure](#)

[Basic Algorithms](#)

ReheapUp

ReheapDown

[Heap Data Structure](#)

[Heap Operations](#)

Build a Heap

Insert a Node

Delete a Node

[Heap Applications](#)

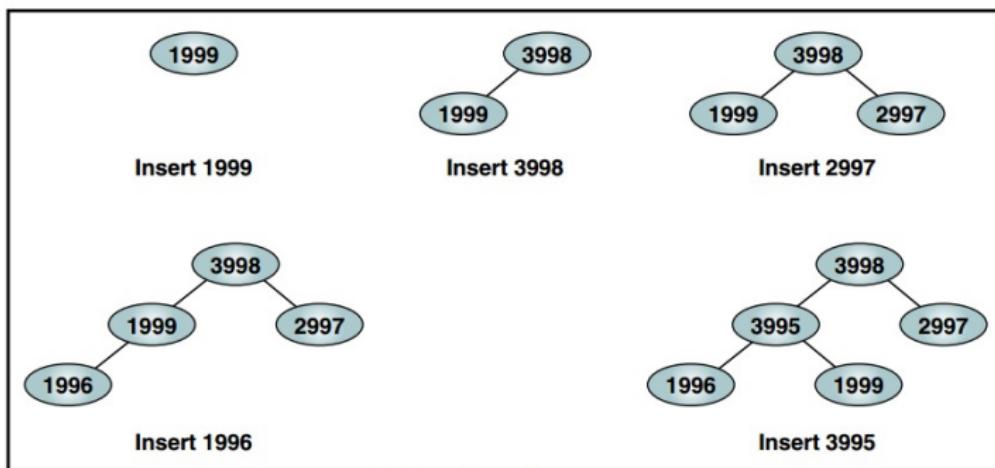
Selection Algorithms

Priority Queues

Heap Sort

Priority Queues

The customers are served according to their priority and within equal priorities, according to their arrival. Thus we see that customer 2 (3998) is served first, followed by customer 5 (3995), customer 3 (2997), customer 1 (1999), and customer 4 (1996).



(a) Insert customers

(Source: Data Structures - A Pseudocode Approach with C++)



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

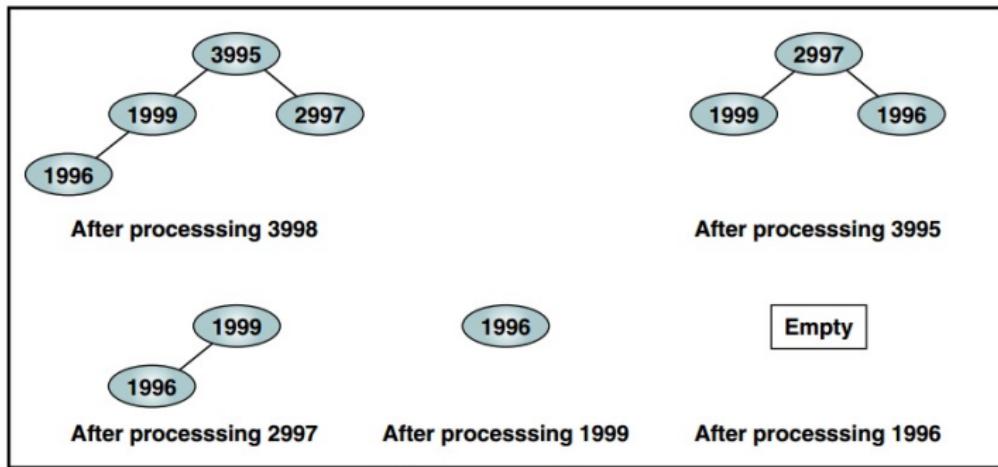
Priority Queues

Heap Sort

Priority Queues

Heap structure

Tran Ngoc Bao Duy



(b) Process customers

(Source: Data Structures - A Pseudocode Approach with C++)

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort



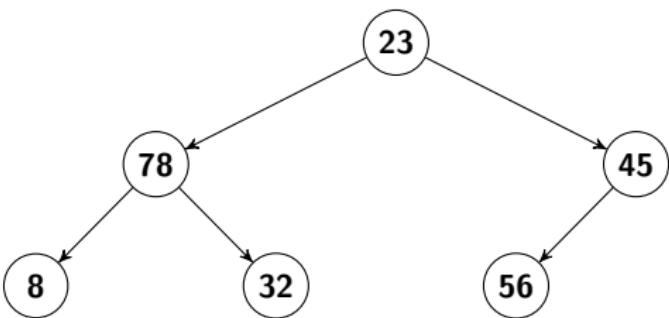
- The unsorted sublist is organized into a **heap** (building a heap).
- In each pass, in the unsorted sublist, the largest element is **selected** and **exchanged** with the last element (delete a node from heap).
- Then the heap is **reheaped**.

[Heap Definition](#)[Heap Structure](#)[Basic Algorithms](#)[ReheapUp](#)[ReheapDown](#)[Heap Data Structure](#)[Heap Operations](#)[Build a Heap](#)[Insert a Node](#)[Delete a Node](#)[Heap Applications](#)[Selection Algorithms](#)[Priority Queues](#)[Heap Sort](#)

Heap Sort

Heap structure

Tran Ngoc Bao Duy



23	78	45	8	32	56
----	----	----	---	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

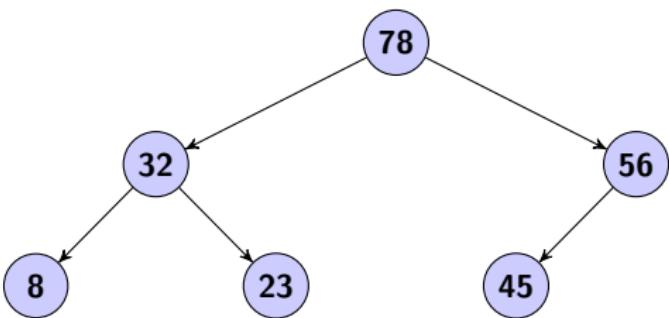
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



78	32	56	8	23	45
----	----	----	---	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

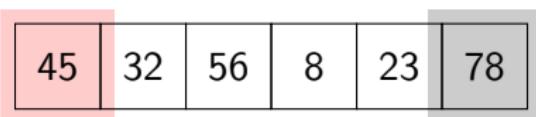
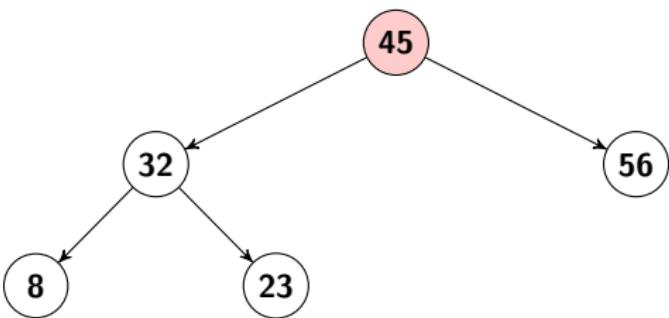
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

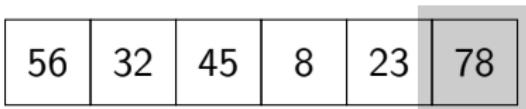
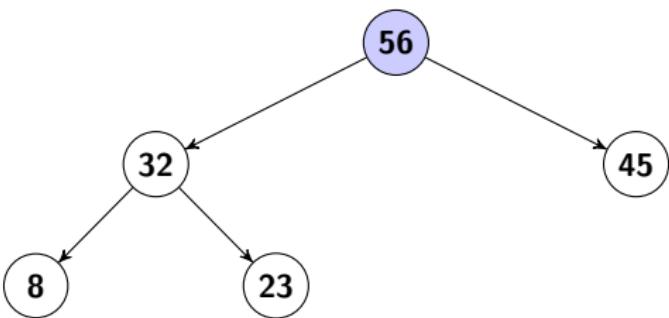
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

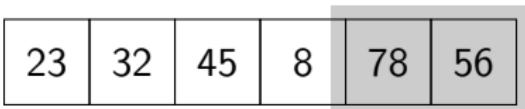
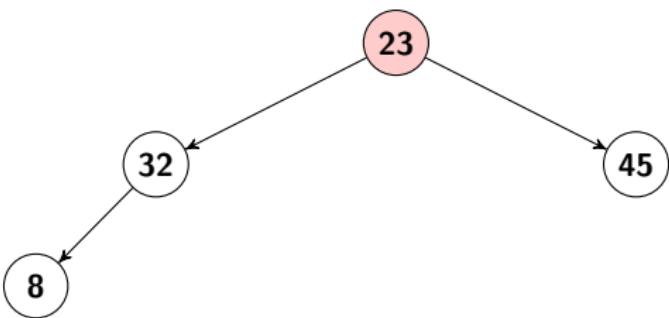
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

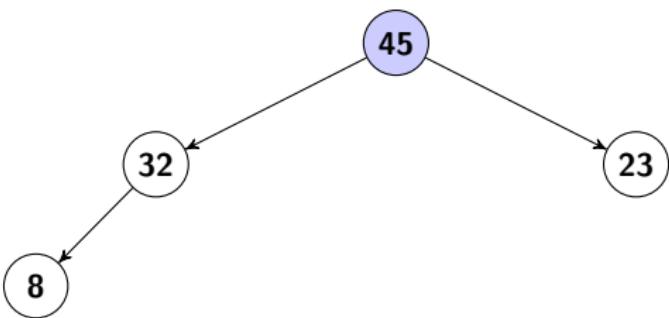
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



45	32	23	8	78	56
----	----	----	---	----	----

Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

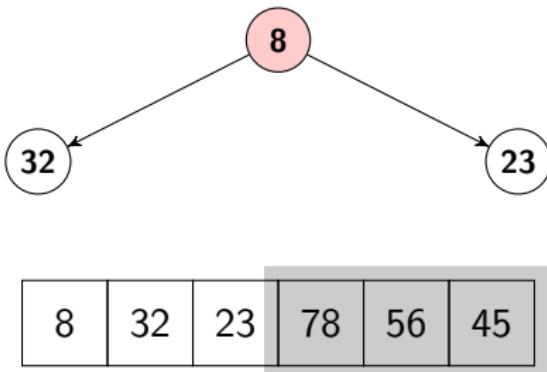
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

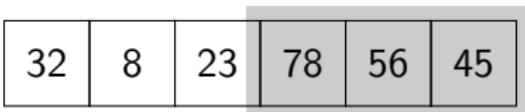
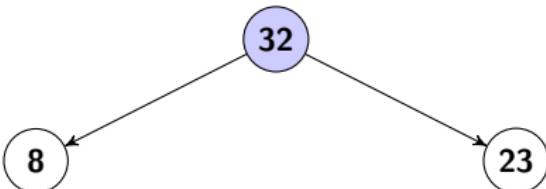
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Operations

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

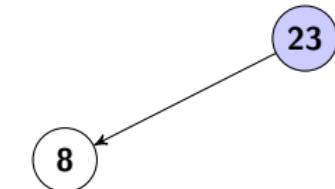
Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



23	8	78	56	45	32
----	---	----	----	----	----

[Heap Definition](#)

[Heap Structure](#)

[Basic Algorithms](#)

ReheapUp

ReheapDown

[Heap Data Structure](#)

[Heap Operations](#)

Build a Heap

Insert a Node

Delete a Node

[Heap Applications](#)

Selection Algorithms

Priority Queues

Heap Sort

Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

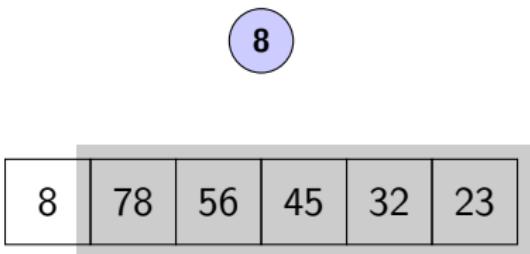
Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort



Heap Sort

Heap structure

Tran Ngoc Bao Duy



Heap Definition

Heap Structure

Basic Algorithms

ReheapUp

ReheapDown

Heap Data Structure

Heap Opearitions

Build a Heap

Insert a Node

Delete a Node

Heap Applications

Selection Algorithms

Priority Queues

Heap Sort

78	56	45	32	23	8
----	----	----	----	----	---



THANK YOU.

[Heap Definition](#)

[Heap Structure](#)

[Basic Algorithms](#)

[ReheapUp](#)

[ReheapDown](#)

[Heap Data Structure](#)

[Heap Opearitions](#)

[Build a Heap](#)

[Insert a Node](#)

[Delete a Node](#)

[Heap Applications](#)

[Selection Algorithms](#)

[Priority Queues](#)

[Heap Sort](#)