**Assignment No.: 09**

**Title: Heap Data Structure**

**Course Outcome: CO1(C214447.1):** Analyze algorithms and to determine algorithm correctness and time efficiency class.

**CO2(C214447.2):** Implement abstract data type (ADT) and data structures for given application.

**CO4(C214447.4**):Solve problems using algorithmic design techniques and data structures.

**Date of Completion: 13/12/2021**

**Assessment Grade / Marks:**

**Assessor’s Sign with Date:**

**Assignment No.: 9**

**Title:** Heap Data Structure

**Aim:** To study Heap Data Structure **Problem Statement:**

Implement Heap sort to sort a given set of values using  Max heap or  Min heap.

**Requirements:**  Visual studio code

**Theory / Procedure / Diagram / Circuits:**

Heap Data Structure:

A Heap is a special Tree-based data structure in which the tree is a complete binary tree.

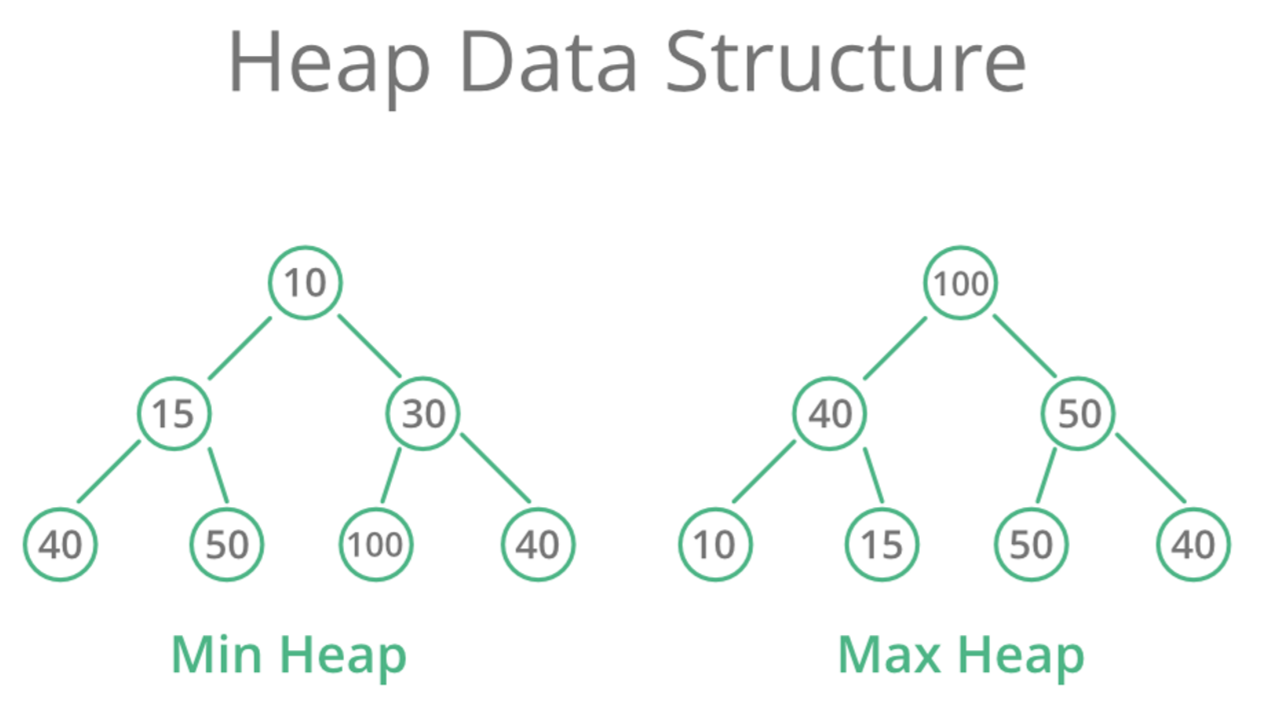
Generally, Heaps can be of two types:

1. Max-Heap:

In a Max-Heap the key present at the root node must be greatest among the keys present at all of its children. The same property must be recursively true for all sub-trees in that Binary Tree.

1. Min-Heap:

In a Min-Heap the key present at the root node must be minimum among the keys present at all of its children. The same property must be recursively true for all sub-trees in that Binary Tree.



**Algorithms / Methods / Steps:**

1. **Algorithm for Insert Operation on Heap Data Structure:**

* Step 1: Add the element to the bottom level of the heap at the most left.
* Step 2: Compare the added element with its parent, if they are in the correct order, stop.
* Step 3: If not, swap the element with its parent and return to the previous step.

1. **Algorithm for the Delete Operation on Heap Data Structure:**

* Step 1: Replace the root of the heap with the last element on its last level.
* Step 2: Compare the new root with its children; if they are in the correct order, stop.
* Step 3: If not, swap the element with one of its children and return to the previous step. (Swap with its smaller child in a min-heap and its larger child in a max-heap).

1. **Algorithm for Heap Sort:**

* Step 1: Construct a Binary Tree with given list of Elements.
* Step 2: Transform the Binary Tree into Min-Heap.
* Step 3: Delete the root element from Min-Heap using Heapify method.
* Step 4: Put the deleted element into the Sorted list.
* Step 5: Repeat the same until Min-Heap becomes empty.
* Step 6: Display the sorted list.

**Input:**

Enter the number of values: 9

Enter the 1th value: 33

Enter the 2th value: 23

Enter the 3th value: 26

Enter the 4th value: 12

Enter the 5th value: 65

Enter the 6th value: 33

Enter the 7th value: 65

Enter the 8th value: 31

Enter the 9th value: 67

**Output:**

\*\*\*\*\*\*\*\*\*\*Sorted Values\*\*\*\*\*\*\*\*\*\*

12

23

26

31

33

33

65

65

67

Minimum Heap: 12

Maximum Heap: 67