

# CS213M: Assignment 4

## Problem 1: Implementation of a Min-Max Heap

Due Date: 13/03/2015

---

We have to implement a min-max heap in this assignment. It supports the the following API. You have to implement the data structure in a class named **MinMaxHeap**. You have to submit two files, **minMaxHeap.hpp** and **minMaxHeap.cpp**. Submit an empty .cpp file if you want to define all the functions in the header file as this is a template class.

**Note:** Do not change the signatures of the functions below. You can use only STL vectors amongst the STL containers for this assignment.

### Functions to be defined

1. `MinMaxHeap();`

This is a constructor for your class. Initialise your member variables here if needed.

2. `void insert(T elem);`

Insert an object in the heap. This operation should preferably take  $O(\log n)$  time where  $n$  is the current number elements in the data structure.

3. `void deleteMin();`

Delete the minimum of the objects currently in the heap. This operation should preferably take  $O(\log n)$  time where  $n$  is the current number elements in the data structure.

4. `void deleteMax();`

Delete the maximum of the objects currently in the heap. This operation should preferably take  $O(\log n)$  time where  $n$  is the current number elements in the data structure.

5. `T getMin();`

Return the minimum of the objects currently in the heap. This operation should preferably be a constant time operation.

6. `T getMax();`

Return the maximum of the objects currently in the heap. This operation should preferably be a constant time operation.

7. `void deleteElems(pred predObject);`

The `pred` class defines a function `toDelete` with the signature `bool toDelete (T)`. The function `deleteElems`, then, will delete, from the min-max heap, all the elements such that `toDelete` returns true when called on them. You'll thus have to implement the `pred` class when you wish to use the `MinMaxHeap`.

**Note:** Use the less than operator to perform **all** the comparisons on objects in the heap, so that just defining one for the template type `T` in the scope of compilation works.