

National University of Computer and Emerging Sciences



Lab Manual
for
Data Structure

Course Instructor	Ms. Abeeda Akram
Lab Instructor(s)	Mr. Sohaib Ahmad Ms. Ammara
Section	BCS-3G
Semester	FALL 2022

Department of Computer Science
FAST-NU, Lahore, Pakistan

Lab Manual 10

Objectives:

After performing this lab, students shall be able to revise:

- ✓ Minheap

Problem 1

```
template <typename T>
class minHeap{
public:
    minHeap( ); // default constructor

    minHeap(T* arr, int N); // parameterized constructor that will take an array of
    random numbers and its size in parameters and initialize the heap with random
    values. It will call the buildMinHeap() function to convert the random values into
    a heap.

    void buildMinHeap() // It will generate heap from random values stored in the
    object.

    void insert(const T & x); // Inserts the key value in the heap array such that, the
    resultant heap tree is a complete binary tree and it follows min heap order.

    bool isEmpty() const; // returns true if it is empty

    const T & getMin() const; //returns minimum value this operation should be
    performed in O(1)

    void deleteMin(); // deletes minimum value this operation should be performed in
    O(logN)

    bool deleteAll(T key); //remove all occurrences of key value from the heap
    and update the heap accordingly.

private:
    vector<T> _vector;

    void bubble_up(int i); // A recursive method to heapify a subtree with the root at
    given index. It maintains heap property during insertion

    void bubble_down(int i); // It maintains heap property during deletion

};
```

Your task is to create a template-based minHeap class with the functions mentioned above and write main to perform the following tasks;

1. Insert following items in heap; 10, 40, 50, 5, 60, 15, 20
2. Find first, second and third minimum in min heap

```
int main()
{
    int array[] = {10, 4, 5, 30, 3, 300};

    minHeap obj(array, 6);

    for(int i=0; i<3; ++i)
    {
        cout << obj.getMin()<< " ";
        obj.deleteMin();
    }

    return 0;
}
```

3. Implement a non-member function Heapsort (T^* arr, int size, int sorting_order) that will take an array of random numbers, and its size in parameters from the user, along with the order of sorting 1 means ascending and 0 means descending. You may need to use the following functions to implement this.
 - I. Parameterized constructor
 - II. getMin()
 - III. deleteMin()