National University of Computer and Emerging Sciences

Laboratory Manual

for

Data Structures Lab

Course Instructor	Mr. Saad Farooq
Lab Instructor(s)	Husnain Iqbal Usama Hassan
Section	CS-E
Date	13-Dec-2021
Semester	Fall 2021

Department of Computer Science

FAST-NU, Lahore, Pakistan

Objectives:

In this lab, students will practice:

1. Implementation of Min Heap and Max Heap Using Arrays

Question 1:

a. Create a struct HeapItem as follows:

```
template <typename k, typename v>
struct HeapItem
{
	k key;
	v value;
};
```

b. Now create two classes MinHeap and MaxHeap which contains:

1. A pointer to HeapItem, "arr".
2. An int variable "capacity" which stores the total capacity of heap.
3. An int variable "totalItems" which contains the count of total number of items stored.

Provide the following member functions for your MinHeap class:

1. **Default constructors** which assigns nullptr to arr pointer. `MinHeap(), MaxHeap()`

2. **Overloaded constructors** which takes as argument the value of capacity and allocates the memory of the required capacity to arr pointer. `MinHeap(int _capacity), MaxHeap(int _capacity).`

3. An **insert function** for both classes which takes as argument a key value pair. It then inserts the key value pair in the heap array such that, the resultant heap tree is a complete binary tree and it follows min/max heap ordering. If totalItems==capacity, then double the capacity of heap array and insert the key value pair. There must not be any memory leaks. `void insert(k key, v value)`

4. A **getMin and getMax function** for both classes respectively which assigns the value of that HeapItem, whose key is minimum/maximum, to the parameter passed by reference. It does not delete that HeapItem from the heap. Use assert(totalItems>0) to throw an error if the heap is empty. `void getMin(v& _value)`

5. A **deleteMin & deleteMax functions** for both classes respectively which deletes the HeapItem which has the minimum/mximum key. The Heap must remain a complete binary tree and it must follow min heap ordering after deleteMin is called. User assert(totalItems>0) to throw an error if the heap is empty. `void deleteMin()`

6. A **heapify function** for both classes which will convert the input array into min/max heap. Heapify(T* arr, int size) in linear time

7. A **shrinkHeap** function for both classes which will shrink the heap by deleting extra space after deletion, if the elements of the heap reduces to 25 % of its original size then reduce its capacity to half

8. An **isEmpty function** for both classes which returns true if the heap has no element. `bool isEmpty() const`

9. A **destructor** for both classes

Question 2:

Insert: Inside main() function

- (9, 2.53), (254, 2.98), (111, 3.20), (6, 3.3), (5, 3.1), (4, 3.2), (176, 3.5), (101, 3.2), (153, 2.34), (10, 2.64), (16, 2.5), (300, 2.38), (43, 2.9), (15, 3.5)
-