|  |  |  |
| --- | --- | --- |
| **Lab 11** | |  |
| **Topic** | * Min Heap * Max Heap * Heap Sort |
| **Objective** | * The basic purpose of this lab is to implement ADT of Min Heap, Max Heap, Heap Sort and test its applications. |

**Instructions:**

* Indent your code.
* Comment your code.
* Use meaningful variable names.
* Plan your code carefully on a piece of paper before you implement it.
* Name of the program should be same as the task name. i.e. the first program should be Task\_1.cpp

# void main() is not allowed. Use int main()

* **You have to work in multiple files. i.e separate .h and .cpp files**

# You are not allowed to use system("pause")

* **You are not allowed to use any built-in functions**

# You are required to follow the naming conventions as follow:

* + **Variables:** firstName; (no underscores allowed)
  + **Function:** getName(); (no underscores allowed)
  + **ClassName:** BankAccount (no underscores allowed)

# Students are required to complete the following tasks in lab timings.

**Heap**

**Max Heap:**

A max heap is a data structure in which each child node is less than or equal to its parent node. In a Max Heap, the largest element is stored at the root node

**Min Heap:**

A min heap is a similar type of data structure where each child node is greater than its parent node. In a Min Heap, the smallest element is located at the root node.

**Task 1**

Create a C++ generic abstract class named as **Heap**, with the following:

**Attributes:**

1. int \*heapArray
2. int heapSize
3. int maxSize

**Functions:**

**virtual void insert(int val) = 0;**

* + Should add a new element in the heap.

**virtual void display () = 0;**

* + Should display elements of heap.

**Task 2**

**Now use the above class to make another derived class named as MinHeap having the following additional functionalities:**

**Implement both pure virtual functions ‘insert(int)’ and ‘display()’ declared in base class, in MinHeap.**

1. Implement a ‘**deleteMin’** function to which should delete the smallest element in the heap.

**int deleteMin()**

1. Implement a ‘**HeapifyUp’** function to maintain the Min Heap property in case a new element is inserted in the heap.

**void HeapifyUp(int index)**

1. Implement a ‘**HeapifyDown’** function to maintain the Min Heap property in case an element is deleted from the heap.

**void HeapifyDown(int index)**

1. Implement a ‘**HeapSort’** function to sort the Min Heap array in descending order

**void HeapSort()**

**Task 3**

**Now use the above class to make another derived class named as MaxHeap having the following additional functionalities:**

**Implement both pure virtual functions ‘insert(int)’ and ‘display()’ declared in base class, in MaxHeap.**

1. Implement a ‘**deleteMax’** function to which should delete the largest element in the heap.

**int deleteMax()**

1. Implement a ‘**HeapifyUp’** function to maintain the Max Heap property in case a new element is inserted in the heap.

**void HeapifyUp(int index)**

1. Implement a ‘**HeapifyDown’** function to maintain the Max Heap property in case an element is deleted from the heap.

**void HeapifyDown(int index)**

1. Implement a ‘**HeapSort’** function to sort the Max heap array in ascending order.

**void HeapSort()**

**Now create menu based program to perform the following operations**

1. Press 1 to add element in minHeap.
2. Press 2 to delete smallest element in minHeap.
3. Press 3 to add element in maxHeap
4. Press 4 to delete largest element in maxHeap.
5. Press 5 to print elements of heap.
6. Press 6 to heapifyUp in min Min Heap.
7. Press 7 to heapifyUp in min Max Heap.
8. Press 8 to heapifyDown in min Min Heap.
9. Press 9 to heapifyDown in min Max Heap.
10. Press 10 to heap sort min Heap.
11. Press 11 to heap sort Max Heap.
12. Press 0 to exit.