Priority Queue

Min Heap

File name: PriorityQueueMin.h

In Assignment-3 you need to implement a priority queue by **min-heap** data structure of **Node** class where priority is set by the **key** attribute of a **Node**. A **Node** with less **key** value having higher priority (as it's a min heap). You need to implement the functionalities of priority queue in the "PriorityQueueMin.h" file. The description of the methods that need to be implemented by you are given below.

1. Private:: **void topToBottomAdjust(int i)** :
   The function compares the node at ith position with its parent node and swaps them if adjustment is needed and considers the position of the parent node as **i** in the next iteration. The iteration ends if **i** becomes the root node or no adjustment is needed at any step.

2. Private:: **void heapify(int i) :**
   The function compares the node at ith position with its child having maximum priority (minimum key value) and swaps them if adjustment is needed and considers the position of maximum-priority child as **i** in the next iteration. The iteration ends if the node at ith **position** becomes a leaf or no adjustment is needed at any step.

3. Public:: **bool push(Node n) :**
   The function inserts the node **n** in the min-heap maintaining the heap property and increases the heap size by 1 and returns true. The function returns false only if the heap size is maximum.

4. Public:: **bool pop() :**
   The function removes the node with highest priority (Minimum key value) from the heap and decreases the heap size by 1 and returns true. The function returns false only if the heap contains no element.

5. Public:: **Node top() :**
   The function returns the node with highest priority (Minimum key value) in the heap.

6. Public:: **bool isEmpty() :**
   The function returns true if the heap contains no element, otherwise returns false.

7. Public:: **bool size() :**
   The function returns the numbers of elements in the heap.

Minimum Spanning Tree

Prim's Algorithm

File name: MSTPrims.cpp

In Assignment-3 you need to implement the Prim's algorithm for finding the minimum spanning tree of a given weighted and connected graph. You will use your own priority queue that is implemented in the "PriorityQueueMin.h" file. The description of the methods for finding the minimum spanning tree is given below.

1. **void mst(int r) :**
   The function calculates the minimum spanning tree from a given cost matrix considering **r** as the id of the root node of the tree. Key value and parent of all nodes are set by this function.

2. **void showEdges() :**
   The function shows all the edges of the minimum spanning tree.

3. **int mstCost() :**
   The function returns the cost of the minimum spanning tree.