* We usea **Binary Heap** for [**priority queue**](https://en.wikipedia.org/wiki/Priority_queue)in **Minimum spanning tree** to make efficient time complexity .
* **primMST** function’s time complexity (E Log V).
* inHeap is a bool array to check if the node is in the minimum heap or not.
* Key is an array to store the currently minimum distances.
* create heapNode array for all the vertices from HeapNode class so that each element in the array have key and vertex.
* Loop for initialization.

**A screenshot of a social media post

Description automatically generated**

* We create object from Minheap class.
* Loop to add all the vertices to minheap with complexity exact (V).
* Initialize the first node with no parent.
* initialize the first node with no parent and with zero distance.

**A screenshot of a cell phone

Description automatically generated**

* *there are 2 nested loops :*
* the first is while loop -> we move on minimum heap with complexity upper (Log V).
* then extract the min node from minheap.
* Take the node’s vertex.
* Mark this vertex not exist in the heap.

*A screenshot of a cell phone

Description automatically generated*

* The second loop to move on the neighbours of the extracted vertex with complexity upper (V).
* Then calculate the distance from the extracted vertex and its neighbours.
* If the neighbour exist: .
* Then see if we can update the new key or not.
* Use function decreasekey to put the new key in its right position with complexity upper(Log E).
* If we can -> then update parent node for destination.

*A screenshot of a cell phone

Description automatically generated*