**Problem Statement : PS6 - [Hospital Emergency] - AS2\_PS6\_HE\_GROUP126**

**Design Documentation**

**Hospital -to- Airport**  with shortest distance & time

**Objective**:

**Part 1**: Shortest route to reach the airport from the hospital? The hospital and airport nodes ``should be taken as an input and is not fixed to the nodes mentioned in the graph above.

**Part 2**:How long would it take for the ambulance to reach the airport if the ambulance travels at an average speed of 80 km/hr.

**To achieve below the objective of problem statement we need to design the solution in the below steps :**

**Step 1:Description of Efficient Algorithm and Design Strategy :**

* Maintained Two sets mapping - **V**  - list of vertices included while doing STP ( shorted path traversal ) and one set of vertices yet not included in STP
* Maintain a list - **E -** contains Edges ( in form of start-end vertex & weight )
* Used BFS ( Breadth Fast Search ) for traversing all the vertices of graph
* Use a min-heap to include the next vertices not included in the tree/graph.

1. Initialize Min Heap with source vertex as root (the distance value assigned to source vertex is 0). The distance value assigned to all other vertices is INF (infinite).
2. While Min Heap is not empty, do the following
3. Extract the vertex with a minimum distance value node from Min Heap. Let the extracted vertex be u.
4. For every adjacent vertex v of u, check if v is in Min Heap. If v is in Min Heap and distance value is more than the weight of u-v plus distance value of u, then update the distance value of v.

**Example: Hospital - a - to - Airport - I, Algo as follows :**

1. -



1. -



1. -



**Step 2: Time Complexity and Efficiency Analysis :**

Our Algorithm having a complexity of **O(ELogV)** is an efficient one as compare to other solutions having O(V^2) because our case has observed the approach as follows :

* We can observe that the statements in the inner loop are executed O(V+E) times (similar to BFS).
* The inner loop has decrease keyecreaseKey() operation which takes O(LogV) time.
* So overall **time complexity is O(E+V)\*O(LogV) which is O((E+V)\*LogV) = O(ELogV)** - **Our Algo with time-efficient complexity vs other approach**

**Reference:**

* Algorithms Design: Foundations, Analysis and Internet Examples Michael T. Goodrich, Roberto Tamassia, 2006, Wiley (Students Edition)
* https://www.geeksforgeeks.org/dijkstras-algorithm-for-adjacency-list-representation-greedy-algo-8/