Ayan Das

1BM18CS022

Date: ______ Page: ______

Write-up

→ Dijsktra's Algorithm

```
class Main
{
    private static void getRoute(int [] prev, int i, List <Integer> route)
    {
        if (i >= 0)
        {
            getRoute ( prev, prev [i], route);
            route. add[i];
        }
    }
    public static void shortestPath(Graph graph, int source, int N)
    {
        priority Queue <Node> minHeap;
        minHeap = new PriorityQueue < >(Comparator. Comparing Int
                                    (node -> node . weight) );
        minHeap. add (new Node(Source, 0));
        List <Integer> dist = new ArrayList <> (Collections. nCopies
                                    (N, Integer. MAX_ VALUE)).
        dist. set (source, 0);
        boolean () done = new boolean [N],
        done [source] = true,
        int () prev = new int [N];
        prev [source] = -1;
        List < Integer> route = new ArrayList <> ();
```

@yan...

Date: ________ Page: ______

```
while (! minHeap.isEmpty())
{
    Node node = minHeap.poll();
    int u = node.Vertex();
    for (Edge edge: graph.adjList.get(u))
    {
        int v = edge.dest;
        int weight = edge.weight;
        if (! done[v] && (dist.get(u) + weight) < dist.get(v))
        {
            dist.set(v, dist.get(u) + weight);
            prev[v] = u;
            minHeap.add(new Node(v, dist.get(v)));
        }
    }
    done[u] = true;
}
 for (int i = 1; i < N; i++)
 {
    if (i != source && dist.get(i) != Integer.MAX_VALUE)
    {
       getRoute(prev, i, route);
       System.out.printf("Path(%d → %d): Min Cost = %d,
             Route = %s \n", source, i, dist.get(i), route);
       route.clear();
    }
 }
}
```

@yam...