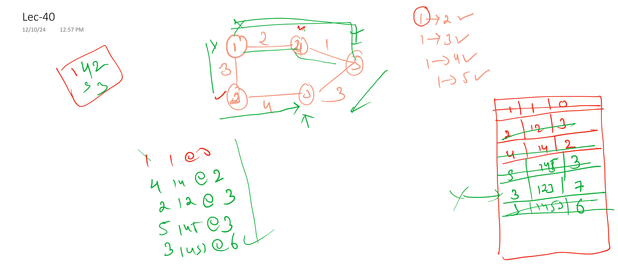
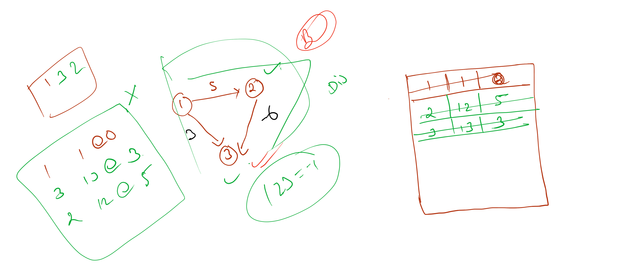
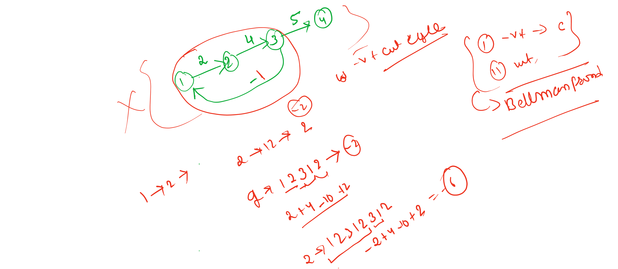
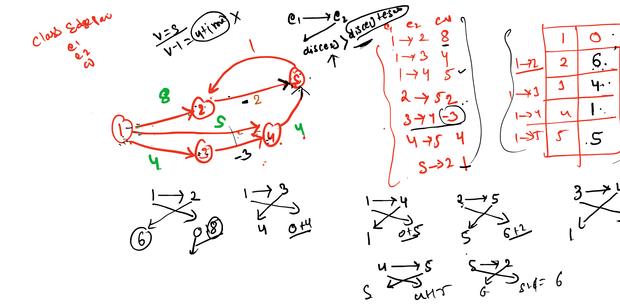
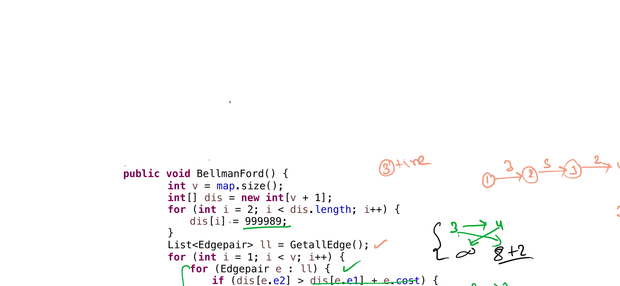


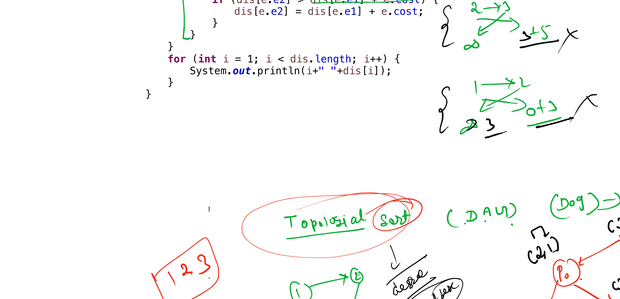


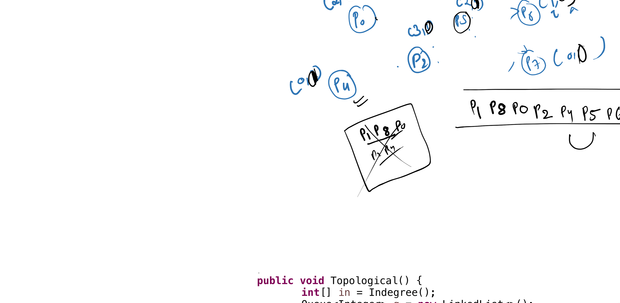






```
public void BellmanFord() {
        int v = map.size();
        int[] dis = new int[v + 1];
        for (int i = 2; i < dis.length; i++) {
            dis[i] = 999989;
        }
        List<Edgepair> ll = GetallEdge();
        for (int i = 1; i < v; i++) {
            for (Edgepair e : ll) {
                if (dis[e.e2] > dis[e.e1] + e.cost) {
                    dis[e.e2] = dis[e.e1] + e.cost;
                }
            }
        }
        for (int i = 1; i < dis.length; i++) {
            System.out.println(i+" "+dis[i]);
        }
    }
```





```
public void Topological() {
        int[] in = Indegree();
        Queue<Integer> q = new LinkedList<>();
        for (int i = 0; i < in.length; i++) {
            if (in[i] == 0) {
                q.add(i);
            }
        }
        while(!q.isEmpty()) {
            //1. remove
            int r=q.poll();
            System.out.print(r+" ");
            for(int nbrs:map.get(r).keySet()) {
                in[nbrs]--;
                if(in[nbrs]==0) {
                    q.add(nbrs);
                }
            }
        }
    }
```