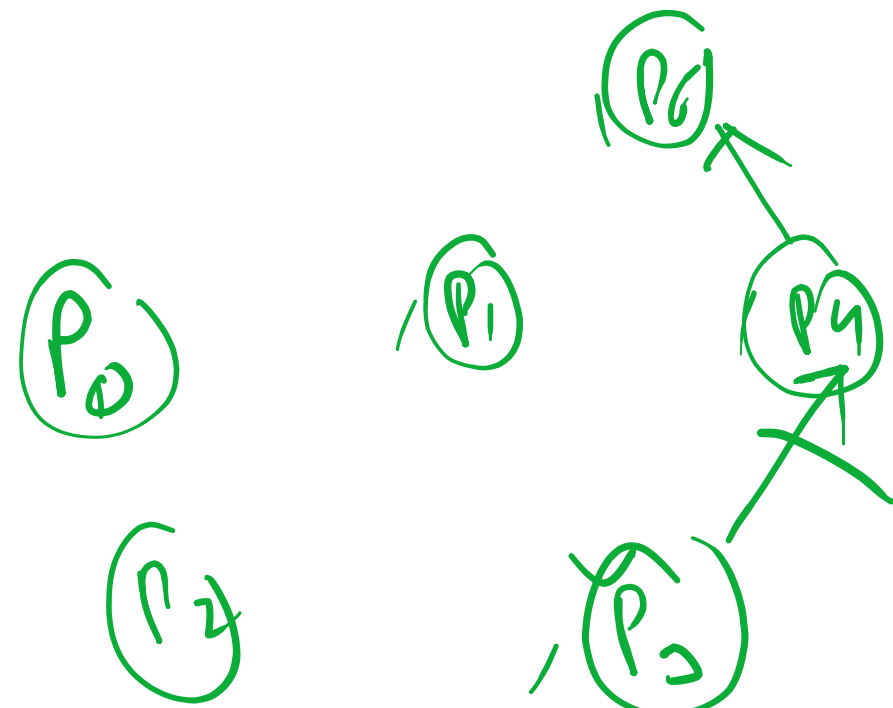
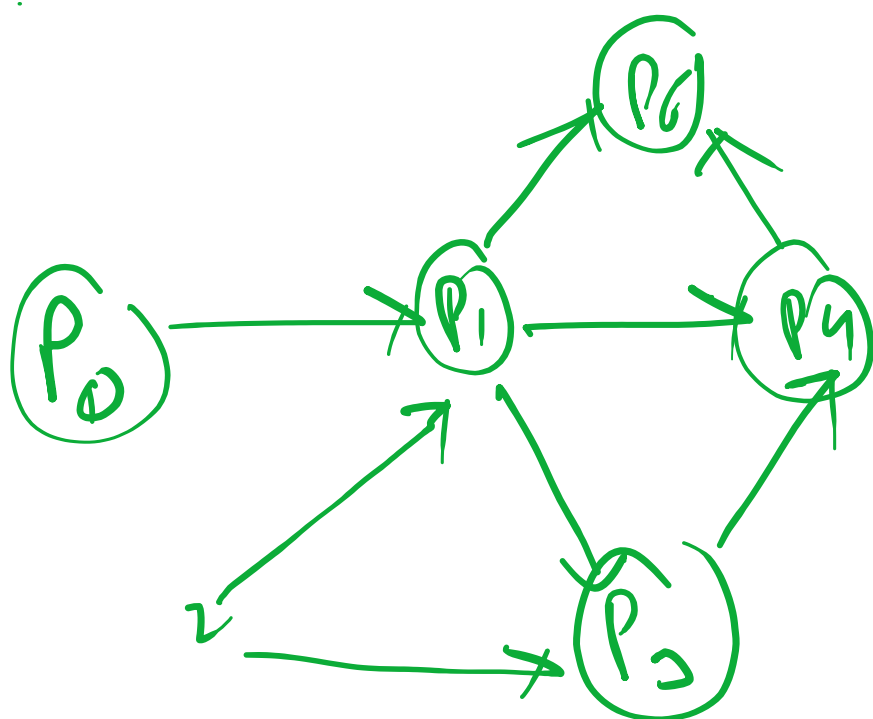
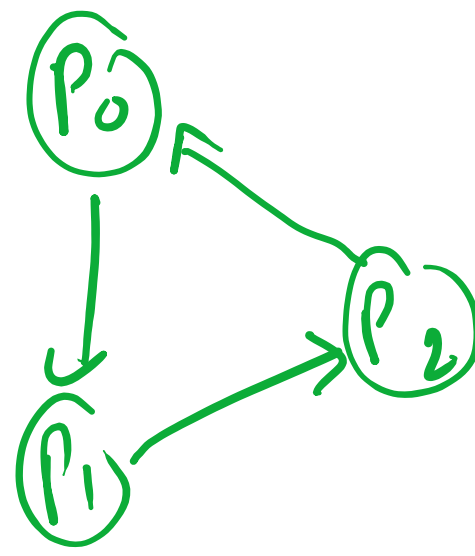


For \Rightarrow DPA

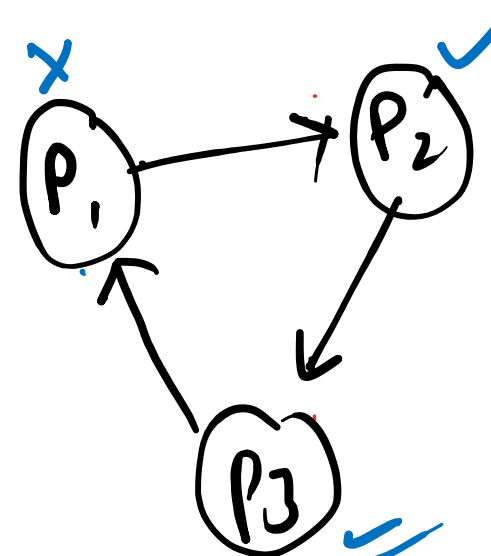
Topological DPA

① \rightarrow ind
[Indegree
Out degree] \rightarrow Topology



P0 P2 P1 P3 P4

P1



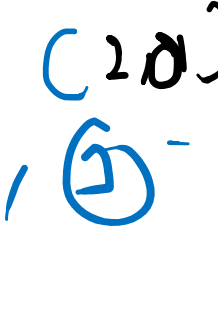
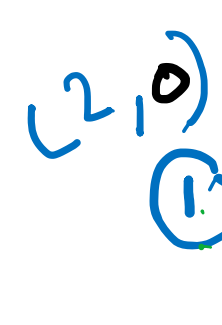
P1 P2 P3

ndc 2+

① \rightarrow [0, 1, 2, 3, 4] \rightarrow [0, 1, 2, 3, 4] ...

full n=8

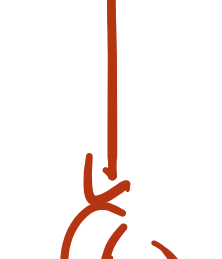
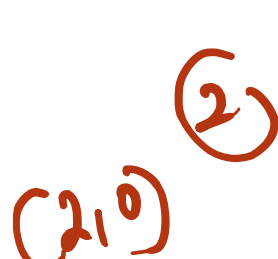
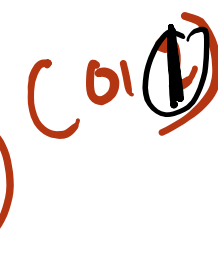
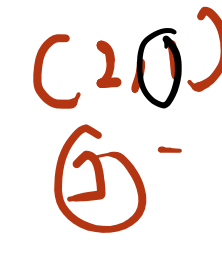
```
public void TopologicalSort() {
    Queue<Integer> q = new LinkedList<>();
    int[] in = indegree();
    for (int i = 0; i < in.length; i++) {
        if (in[i] == 0) {
            q.add(i);
        }
    }
    while (!q.isEmpty()) {
        int rv = q.poll();
        System.out.print(rv + " ");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



1 2 3 4 5 6 7 8

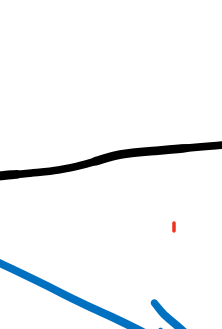
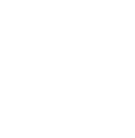
2 10 4 3 5 7 6

1 2 3 4 5 6 7 8

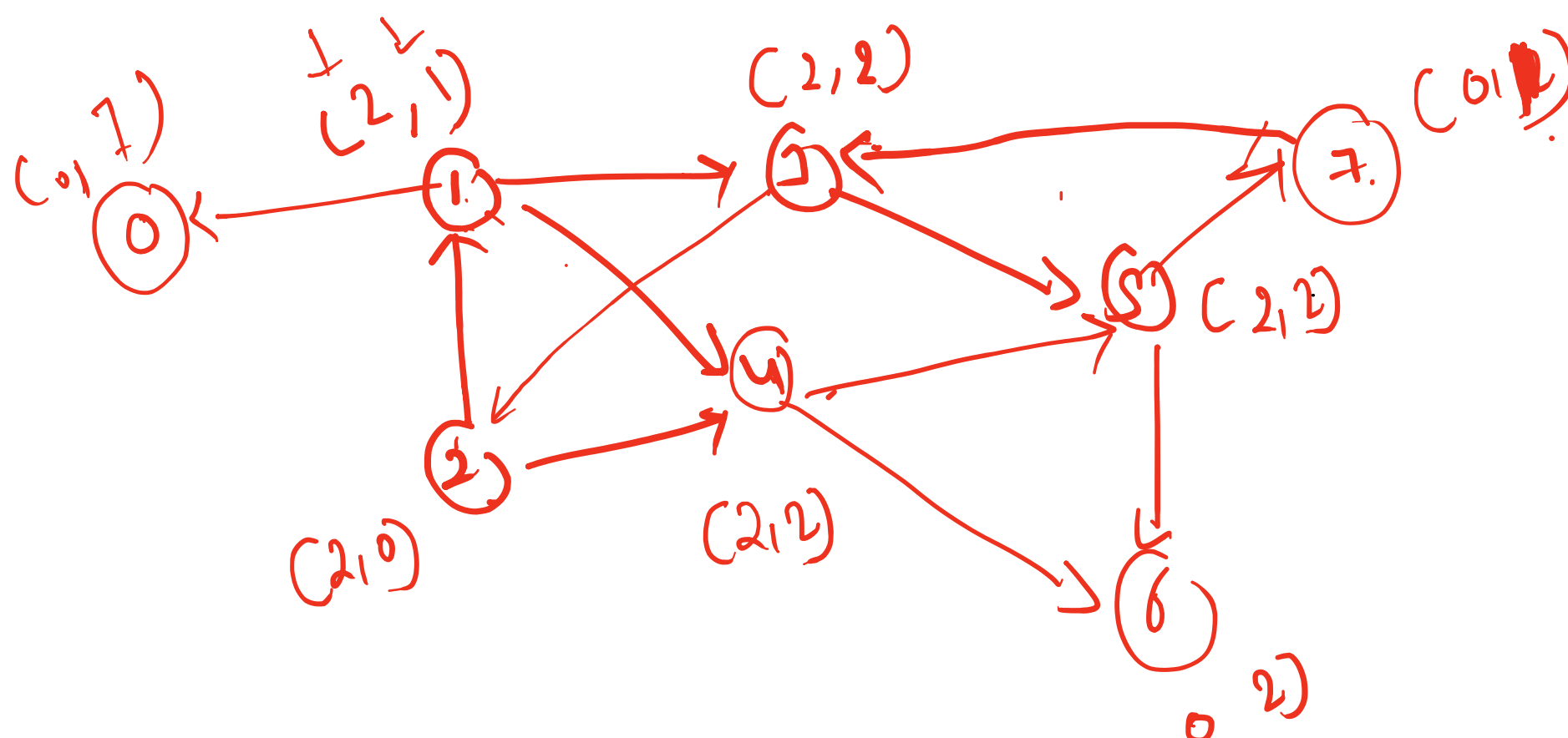


2

2 1 0

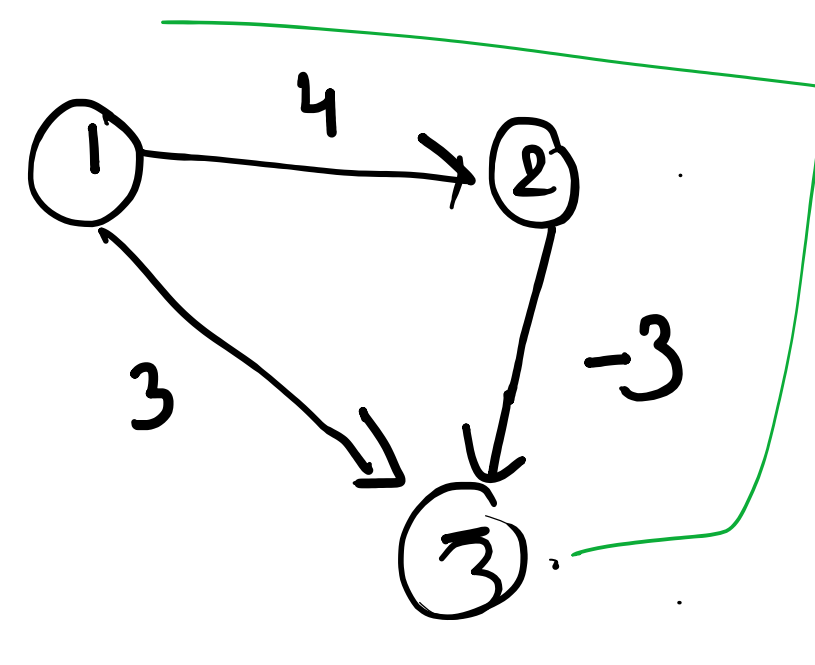


1 2 3 4 5 6 7 8



1-2

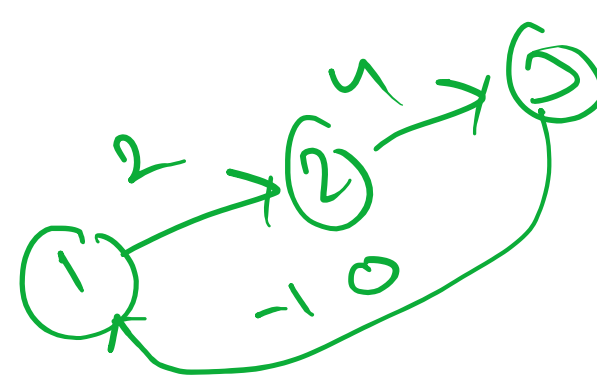
1 1 0 0
3 0 0 3
2 1 2 0 4



1	1	0	0
2	1	2	4
3	1	3	3

1 2 3 \Rightarrow ①

1 2 3 6



6-10-2-4/12+4
 $=$ ② \rightarrow 10/14 $=$ ②
Bellman Ford

1 2 3 2 3 1 2
6

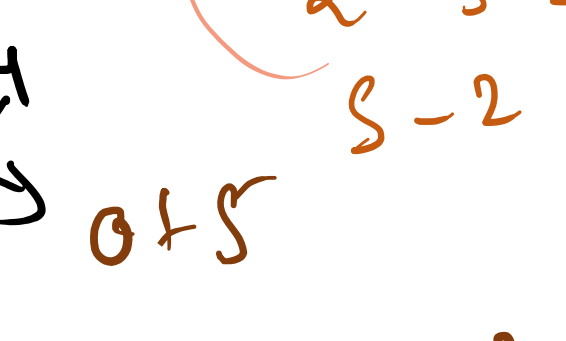
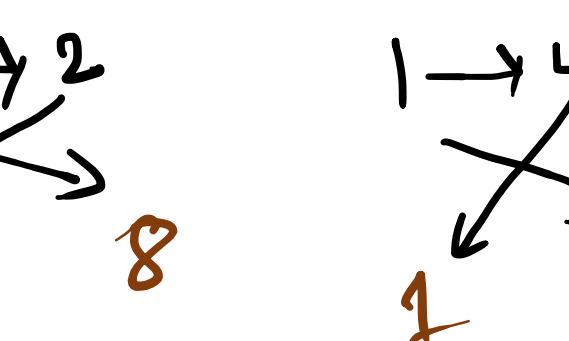
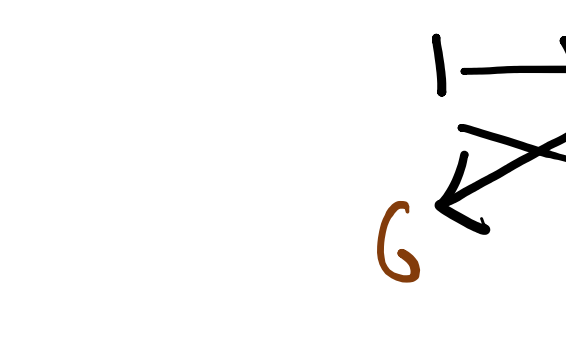
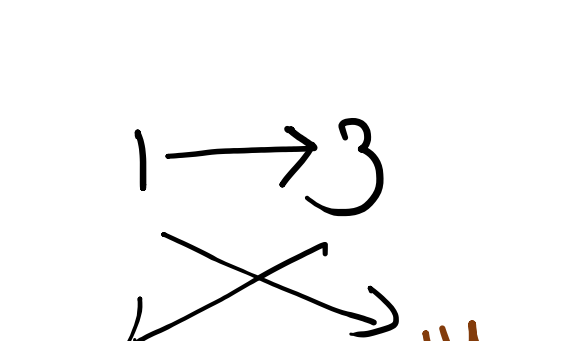
Bellman

1-2 8
1-3 4
1-4 5
3-4 -3
4-5 4
2-5 2
5-2 1

1	0
2	6
3	4
4	2
5	5

1-2 8
3-4 -3
4-5 4
2-5 2
5-2 1

max sel(c) for 2

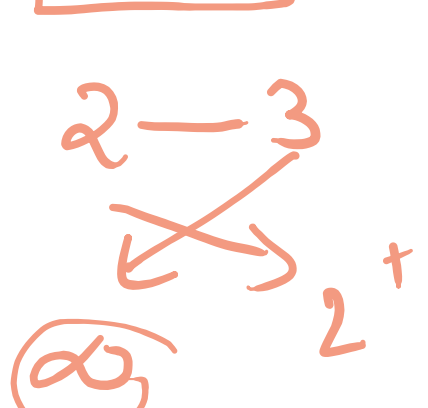


```
public void Bellmanford() {
    int v = map.size();
    int[] dis = new int[v + 1];
    for (int i = 2; i < dis.length; i++) {
        dis[i] = 9798999;
    }
    List<EdgePair> ll = getAllEdges();
    for (int i = 1; i < v; i++) {
        for (EdgePair e : ll) {
            if (dis[e.e2] > dis[e.e1] + e.cost) {
                dis[e.e2] = dis[e.e1] + e.cost;
            }
        }
    }
}
```



3-4 5
2-3 1
1-2 2

1	0
2	2
3	3
4	8



1-2 2
2-3 1
3-4 5

X