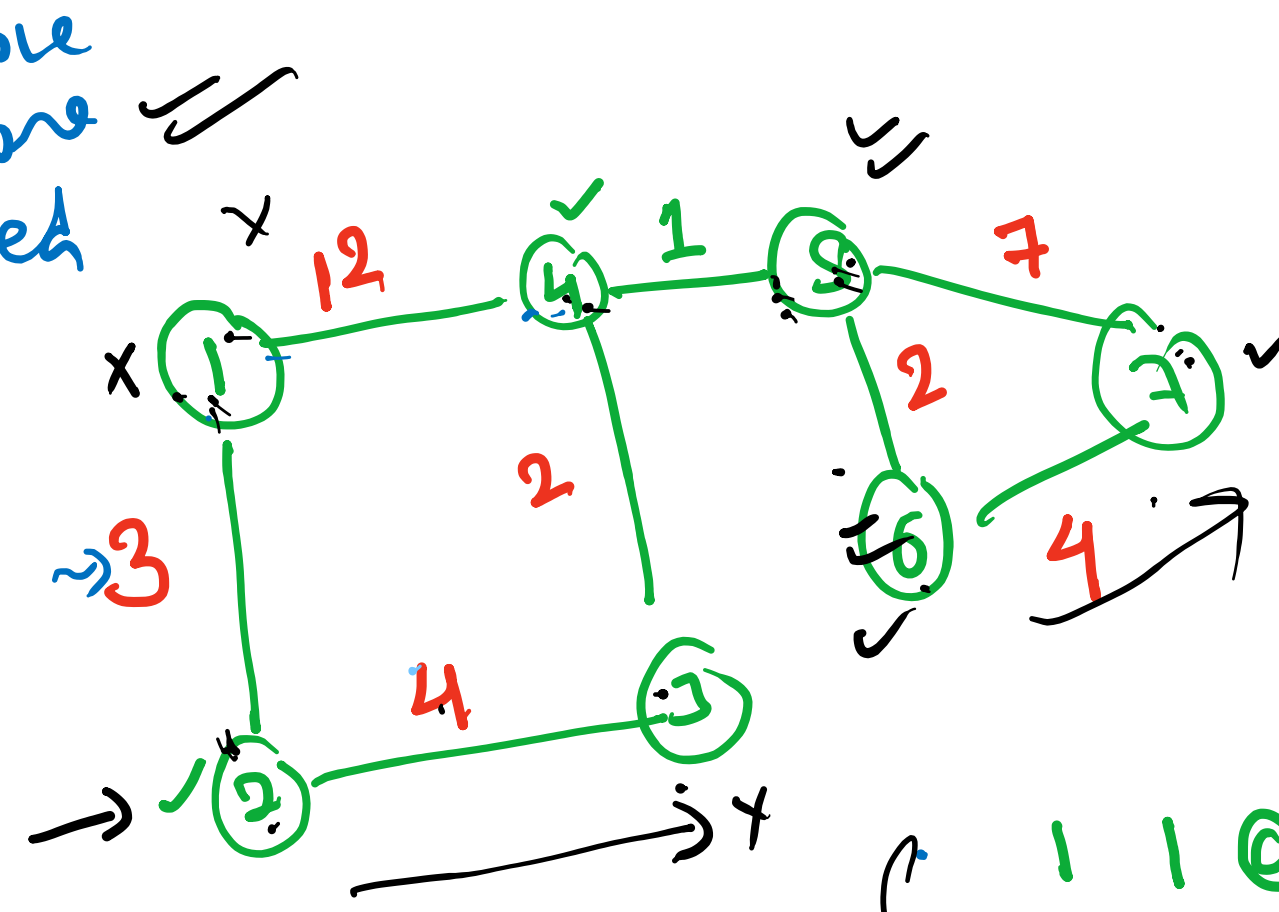


1. remove
2. ignore
3. visited
4. self
5. Add



dist

1-2	✓
1-3	✓
1-4	✓
1-5	✓
1-6	✓
1-7	✓

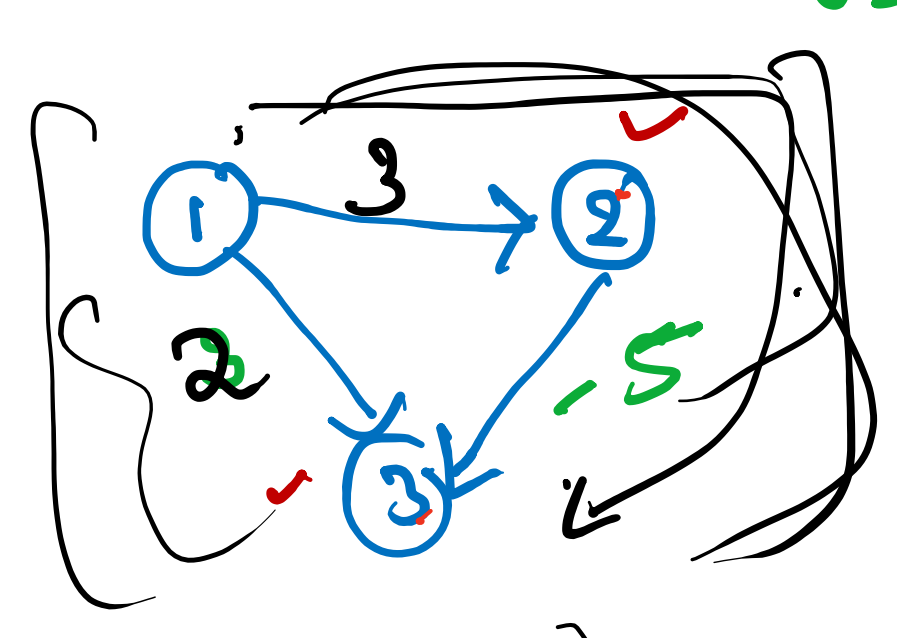
1	2	3
4	5	6
7		

1	1	1	0
4	1	4	12
2	1	2	3
3	1	2	7
4	1	2	9
5	1	2	10
6	1	2	12
7	1	2	17
7	1	2	16

1	1	0	0
2	1	2	3
3	1	2	7
4	1	2	9
5	1	2	10
6	1	2	12
7	1	2	16

1-2 → 3  
1-3 → -2

Dijkstra



using dijkstra

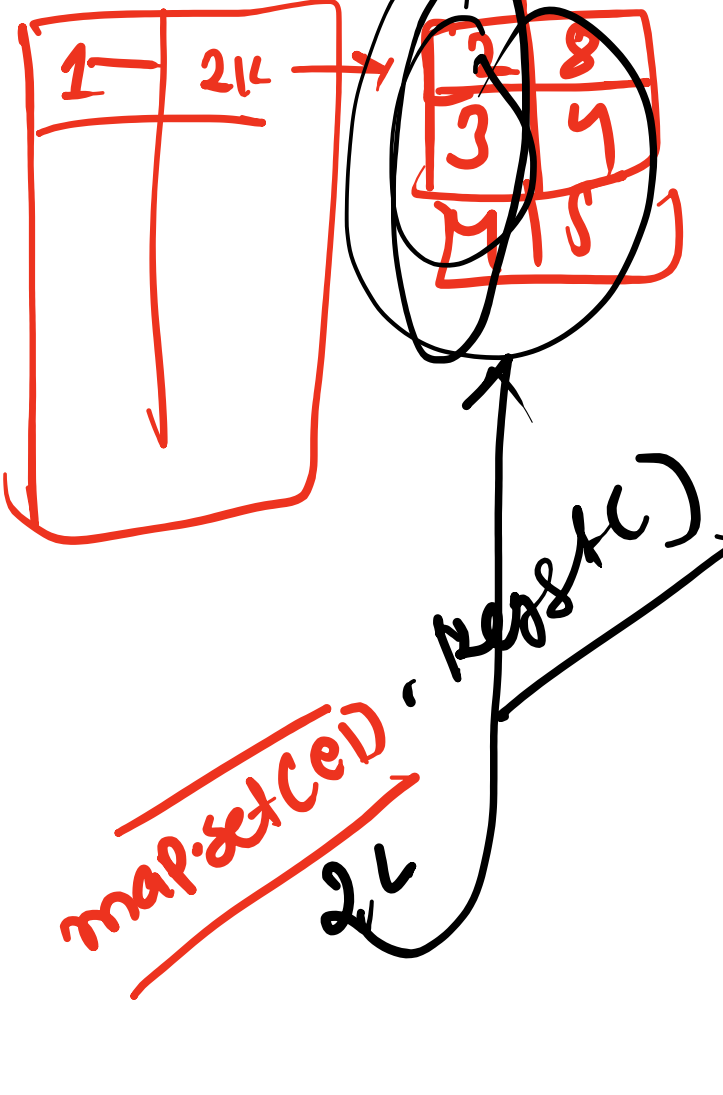
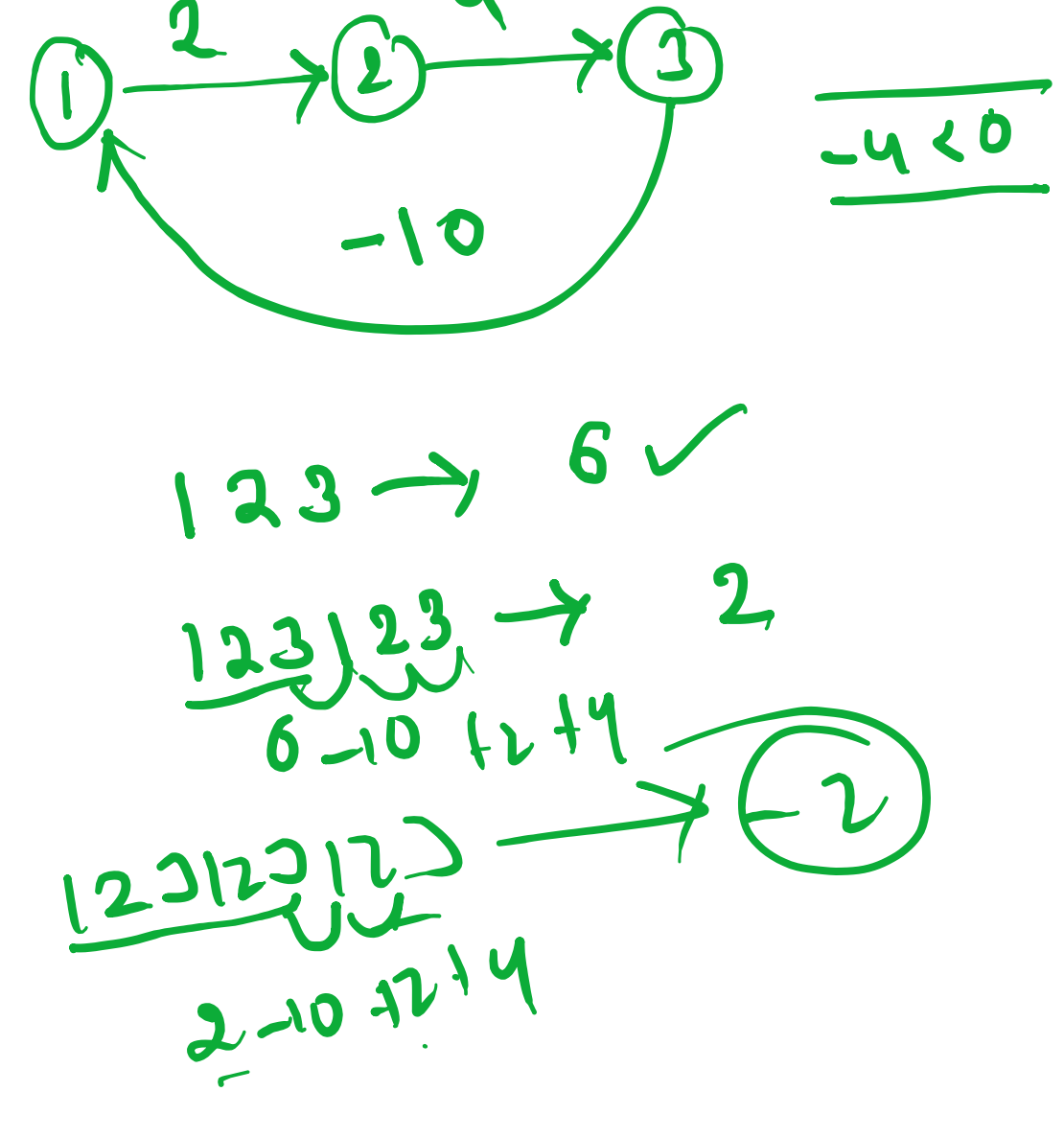
1	1	0	0
3	1	3	0
2	1	2	3

1	1	0
2	1	3
3	1	2

1	3	2
---	---	---

**Dijkstra**  
directed  
undirected  
-ve wt → edge  
+ve wt ✓  
is with cycle sumco  
graph is with cycle X

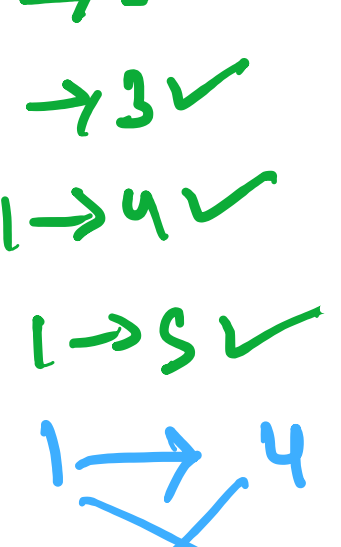
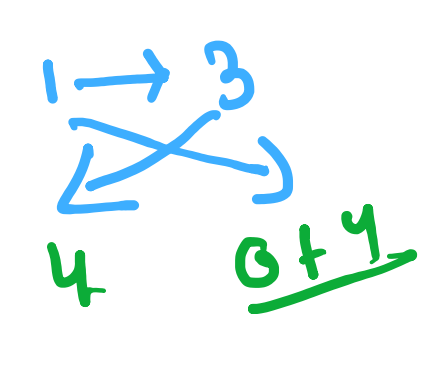
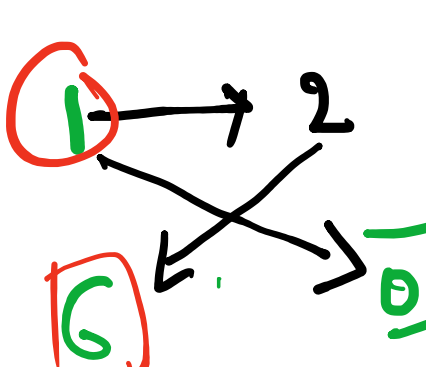
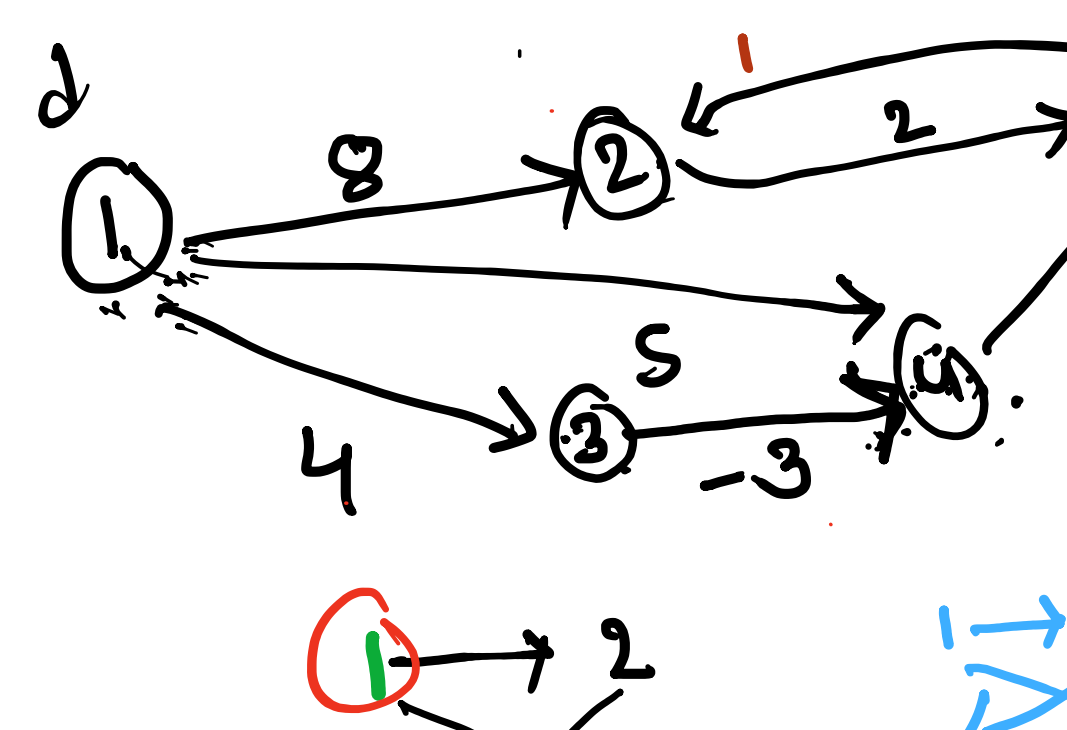
**Bellman**  
directed  
undirected  
-ve wt edge ✓  
+ve ✓



(i) Bellman Ford

(ii) How Find  
-ve wt cycle  
Time Replx  
→ explain

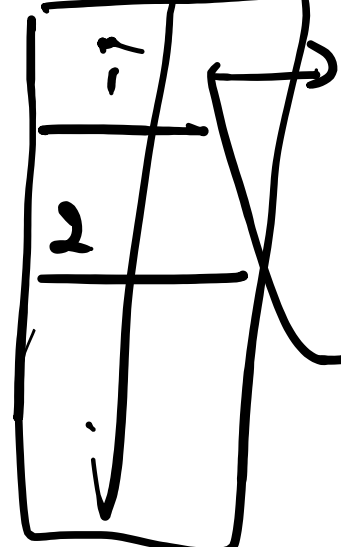
(iii) counting sort



1	0	✓
1-2	2	6
1-3	3	4
1-4	4	+1
1-5	5	5

```

public List<EdgePair> getAllEdge() {
    List<EdgePair> ll = new ArrayList<>();
    for (int e1 : map.keySet()) { // [1,2,3,4,5]
        for (int e2 : map.get(e1).keySet()) {
            int cost = map.get(e1).get(e2);
            ll.add(new EdgePair(e1, e2, cost));
        }
    }
    return ll;
}
  
```



1	2	8
2	1	4

1	3	4
2	1	4

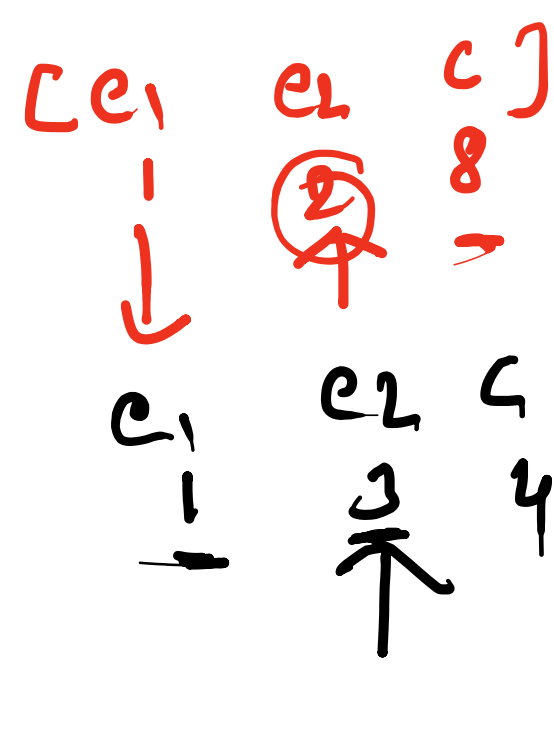
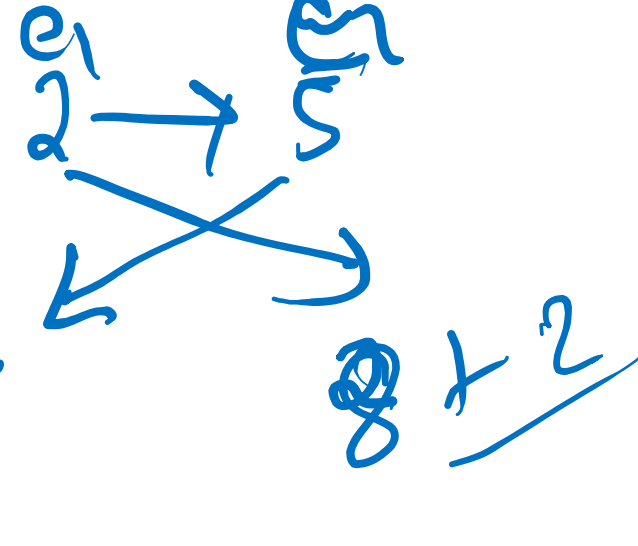
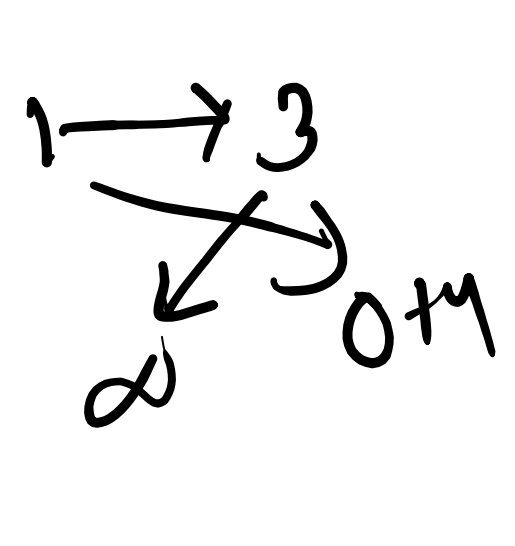
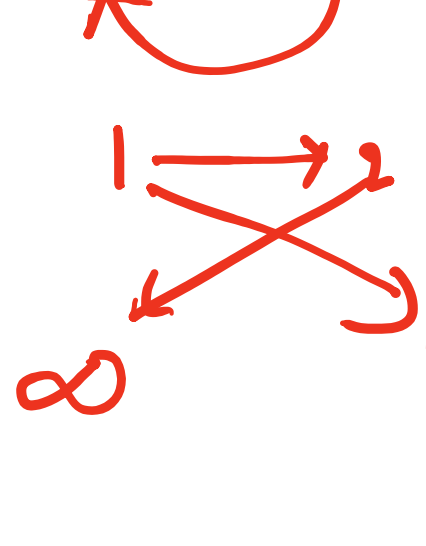
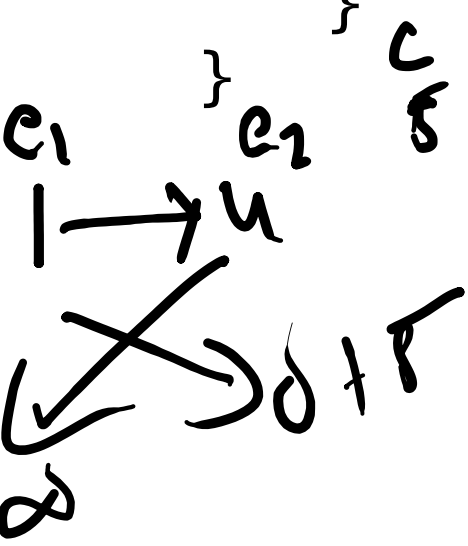
0	1	2	3	4	5
---	---	---	---	---	---

2	4	4	0	1	1	1	1	1	2
---	---	---	---	---	---	---	---	---	---

0	1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---	---

```

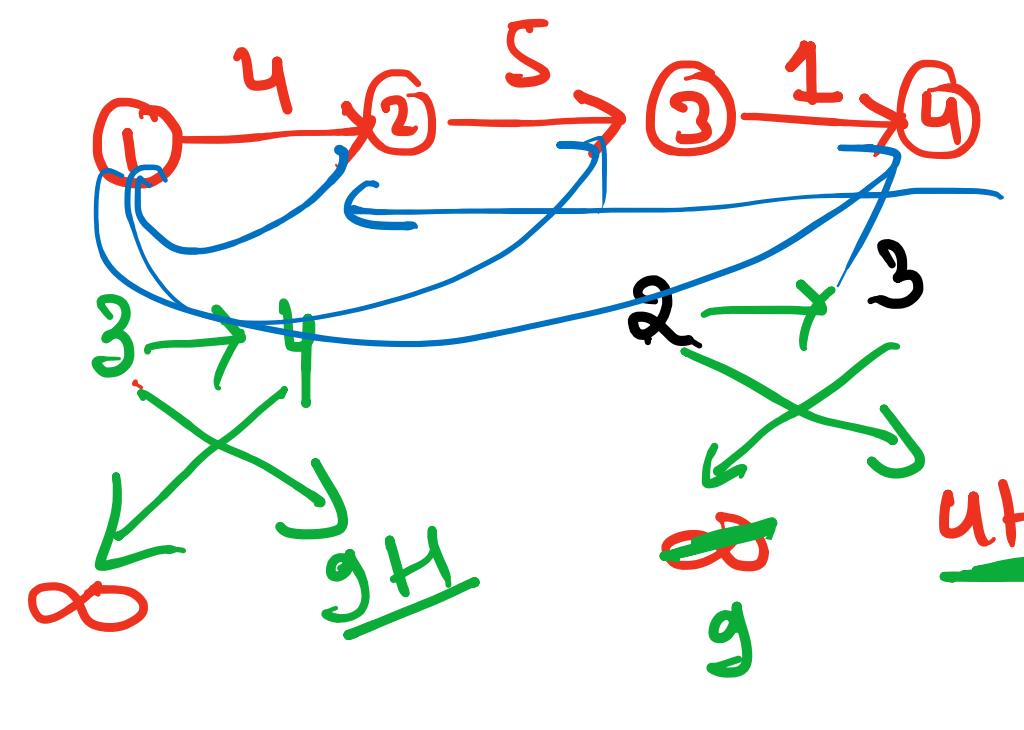
for (EdgePair e : ll) {
    if (dis[e.e2] > dis[e.e1] + e.cost) {
        dis[e.e2] = dis[e.e1] + e.cost;
    }
}
  
```



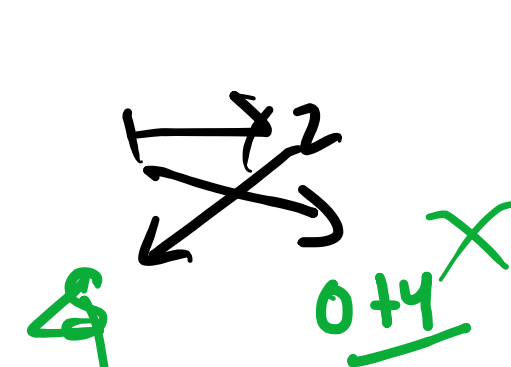
```

public void BellManFord() {
    int v = map.size();
    int dis[] = new int[v + 1];
    List<EdgePair> ll = getAllEdge();
    for (int i = 2; i <= dis.length; i++) {
        dis[i] = 87687679;
    }
    for (int i = 1; i <= v - 1; 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.print(dis[i] + " ");
    }
}
  
```

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

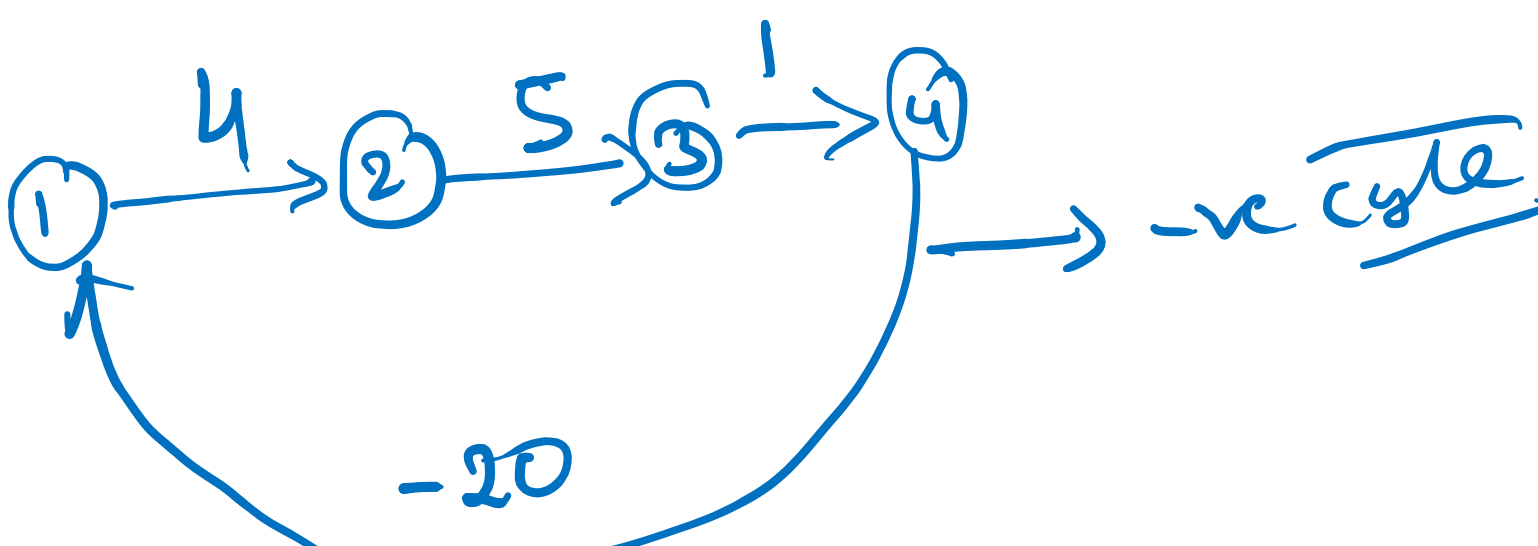


3-4 1 ✓  
2-3 5 ✓  
1-2 4



0	4	9	10
0	1	2	3

-wt cycle



Counting sort → Comparison X

$$F(i) = F(i-1) + F(i)$$

2	4	4	0	1	1	1	1	1	2
0	1	2	3	4	5	6	7	8	9
0	1	6	10	10	13	13	14	14	
0	1	2	3	4	5	6	7	8	9

{ 2, 1, 1, 0, 1, 2, 5, 4, 0, 2, 8, 7, 9, 2, 6, 1, 9 };

0	0	1	1	1	1	2	2	2	2	5	4	6	7	8	9
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15