Lab#12 Prim's Algorithm (Minimal Spanning Tree)

1. **Input data 는 다음 그래프를 사용**
   - **<u>starting vertex(시작정점)은 정점 1,</u>**
   - Adjacency Matrix 로 저장 (프로그램에서)

2. **Output 은 다음과 같다.**

1) Weighted Graph(cost[v][i])를 다음과 같이 출력

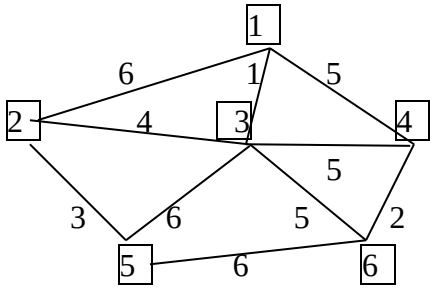
	v1	**v2**	**v3**	**v4**	**v5**	**v6**
v1	100	6	1	5	100	100
v2	6	100	4	100	3	100
v3	1	4	100	5	6	5
v4	5	100	5	100	100	2
v5	100	3	6	100	100	6
v6	100	100	5	2	6	100

2) Minimal Spanning Tree (v1 에서 시작하면)

1, 3 → 3, 2 → 2, 5 → 5, 6 → 6, 4 (Total Weight : 16)

Or

1, 3 → 3, 2 → 2 , 5 → 1, 4 → 4, 6 (Total Weight : 15)



3. **Prime Algorithm**

prim(int v) //starting vertex "v" {

1. cost matrix[시작정점 v] 에서 lowcost 로 해당 데이터 한줄 copy

   ex) lowcost[i] = cost[v][i]; //v 는 시작정점. Lowcost[]는 v 의 low data

2. mark **closest[]** for starting vertex // 방문한 시작 정점 마킹

3. Loop until n-1 edges

3.1 Select the lowest cost vertex from the lowcost // lowcost 서 가장 작은
and print Vi and Vj // 데이터 값 찾아서, 출력

3.2 total++; lowcost[v] = max; // 선택한 정점값을 max 로

3.3 low 의 나머지 데이터들도 update // greedy method 에서 벗어남

if (cost[k][j] < lowcost[j] && lowcost[j] < max) { // k= 선택된 정점.

//j = 2~maxn 까지 검사. //initial 은 max 값.

}

4. 출력 screen shot