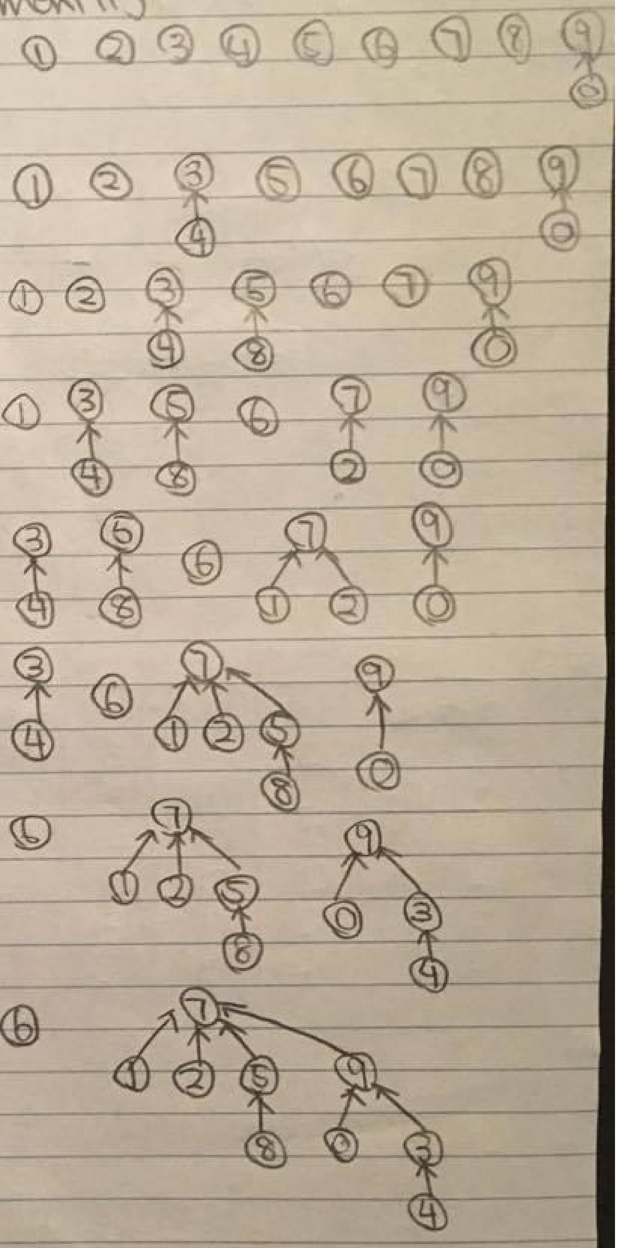
Jordan (Yu-Lin) Wang
V00786970

CSC226 Assignment #3

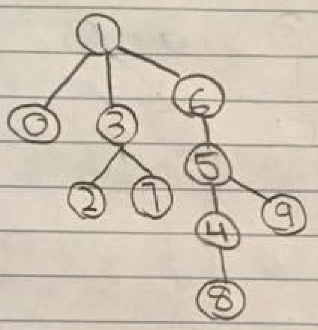
#1)

Parent p	q	0	1	2	3	4	5	6	7	8	9
9	0	9	1	2	3	4	5	6	7	8	9
3	4	9	1	2	3	3	5	6	7	8	9
5	8	9	1	2	3	3	5	6	7	5	9
7	2	9	1	7	3	3	5	6	7	5	9
2	1	9	7	7	3	3	5	6	7	5	9
5	7	9	7	7	3	3	7	6	7	5	9
0	3	9	7	7	9	3	7	6	7	5	9
4	2	9	7	7	9	3	7	6	7	5	7



#2)

i	0	1	2	3	4	5	6	7	8	9
id[i]	1	1	3	1	5	6	1	3	4	5

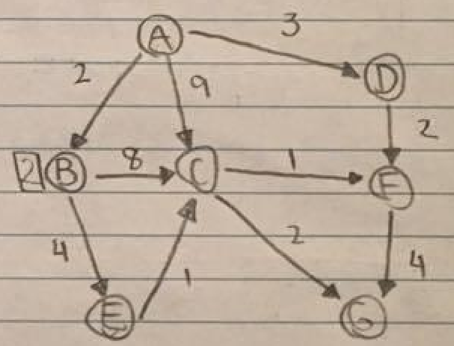


No, this result can not be the result of running weighted quick-union because to be running weighted, when linking tree 6 and tree 5 together, ideally 5 would be the root due to larger weight. This will result in tall tree, which is opposite for weighted quick union and result in longer running time.

#3)

	A	B	C	D	E	F	G
	+∞	+∞	+∞	+∞	+∞	+∞	+∞
1	0	2	9	3	+∞	+∞	+∞
2	0	2	9	3	6	5	11
3	0	2	7	3	6	5	9
4	0	2	7	3	6	5	9
5	0	2	7	3	6	5	9
6	0	2	7	3	6	5	9

(C,G),(F,G), (B,E), (E,C), (C,F), (B,C), (D,F), (A,D), (A,C), (A,B)



#4) LongestPath(G):

```
//Input: DAG G=(V,E) in adjacency list format
//output: the length of a longest path in G
    let n = |V|
    initialize an array lenpath[] of length n
    find a topological ordering of G
    for every v in V in reverse topological order:
        if v has no outgoing edges:
            lenpath[v] = 0
        else:
            for every u in V such that (v,u) in E:
                if lenpath[v] < lenpath[u] + 1:
                    lenpath[v] = lenpath[u] + 1
    return max(lenpath)
```

worst case running time: $O(|V|+|E|)$

#5) Worst Case Run Time:

$$N + (\log N \times N \times (1 + deg(i) + deg(i))$$
$$= N + (\log N \times (N + M + M)$$
$$= N + ((2M + N)\log N)$$
$$= 2M\log N \quad \text{Therefore } \boxed{O(M\log N)}$$

Best Case RunTime:

$$N + (1 \times N \times (1+1+1)$$
$$= N + (3N)$$
$$= 4N \quad \text{Therefore } \boxed{O(N)}$$