COP 3503
Final
Practice

1) Give a Big O runtime for the following Algorithms. State what your Runtime is in terms of (e.g. N is number of Nodes, etc.)
    a. (5 pts) Primm's MST

    b. (5 pts) Tarjan's SCC

    c. (5 pts) Floyd's

    d. (5 pts) Merge Sort

2) (5 pts) What other algorithm discussed in class aside from Topological sort, can be used to check efficiently that a graph is Acyclic.

3) (10 pts) You oversee the movement of troops for the Empirical (not Imperial) forces. You are positioning troops in a war game; you need to assign each of your troops to exactly one enemy, and no enemy will have exactly one troop assigned to them. A troop will be successful in combat if their skill level is above their assigned opponent's skill. You have meticulously determined the skill of every combatant (both of enemies and your troops), and no soldiers have identical skills. Give an efficient high level solution for finding the maximum number of troops that can be successful.

4) Consider inserting the following elements into a skip list of height 4 with the given heights.

Value	4	2	1	1	3	1	1	2
Height	8	4	3	5	6	9	2	1

a. (5 pts) Draw the resulting skip list.

b. (10 pts) What are the number of comparisons needed to insert each values? Assume that you can compare the left and right sentinels if necessary, and comparisons happen for each level.

Value	4	2	1	1	3	1	1	2
Comparisons								

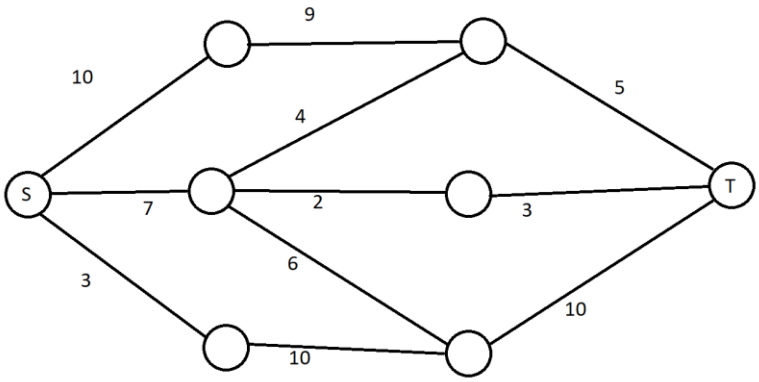
5) (10 pts) What is the problem with the following segment of MST code?

```
int MST(int n, int m, ArrayList<Edge>[] adj)
{
    if (m < n – 1)
        return IMPOSSIBLE;

    boolean[] vis = new boolean[n];
    vis[0]= true;
    PriorityQueue<Edge> pq = new PriorityQueue<Edge>();
    for (Edge e : adj[0])
        pq.add(e);

    int cost = 0;
    while (!pq.isEmpty() && tb > 0)
    {
        Edge cur = pq.poll();
        if (!vis[cur.en])
        {
            cost += cur.cost;
            vis[cur.en] = true;
            for (Edge e : adj[cur.en])
                pq.add(e);
        }
    }
    return cost;
}
public static class Edge implements Comparable<Edge>
{
    int en, cost;
    public int compareTo(Edge o)
    {
        return cost – o.cost;
    }
}
```

6) (5 pts) What is the worst cast runtime for a Disjoint Set's find method, if Disjoint Set uses path compression to improve performance?

7) (5 pts) Draw a R-B tree that is valid but would not be considered a valid AVL tree.

8) (5 pts) What is the maximum flow for the following graph?



9) (5 pts) Is there a graph that Bellman Ford works on, but Floyd's algorithm does not? If so, draw it; if not, explain why.

10) (15 pts) You are going camping. You want to fill your pack to capacity. However, you don't want to take to many items, so you wish to take as few items as possible. You can take multiple of the same item. Find a way to find the minimum number of items that fills your sack. Additionally, you are travelling with your S.O., and you do not want to take any positive multiple of two of the same item (even numbers can jinx a relationship). Make sure that the quantity of each item type you take is either 0 or a positive odd. Write the java code to solve this problem.

```
int hardKnapsck(int numTypes, int[] size, int capacity)
{

}
```

11) (5 pts) What algorithm paradigm/technique can be used to efficiently find the number of solutions to the following problem? Explain why and how.

You have some n students that want to help with their homework. Each student will meet with all n TAs over the course of n days. Every day a student will meet with exactly one TA, and each TA will meet with exactly one student. Some TAs have already scheduled time with a student during a particular day. How many ways can the remaining students and TAs have their schedules assigned to satisfy the constraint that all TAs will work with all students?