Problem 1

Given set S of n real numbers and an integer I the 3-SUM problem is to decide (true/false) if S contains three elements that sum to I. It is well known that one can solve the problem in $O(n^2)$ time. The best known lower bound for the problem is $\Omega(n \log n)$, that is, there is no algorithm that can solve the problem in less time. Given the below algorithm, can you give any upper and lower bounds on the running time of the algorithm? Assume that you have a function DECIDE-3-SUM(S, I) that solves 3-SUM problem for a set S and an integer I.

Algorithm 1 PRINT-3-SUM VALUES

procedure PRINT-3-SUM VALUES(S,m)
▷ S is a list of real values and m is an integer
  for $I = 1, \ldots, m$ **do**
    if DECIDE-3-SUM(S, I) **then**
      Print(I)
    end if
  end for
end procedure

Problem 2

Give an $O(n)$ time algorithm to detect whether a given undirected graph contains a cycle. If the answer is yes, the algorithm should produce a cycle. (Assume adjacency list representation.)

Problem 3

Trace Prim's algorithm on the graph in Fig. 1 starting at node a. What's the output?

Problem 4

Trace Dijkstra's algorithm on the graph in Fig. 1 starting at node a and ending at g. What's the shortest path?

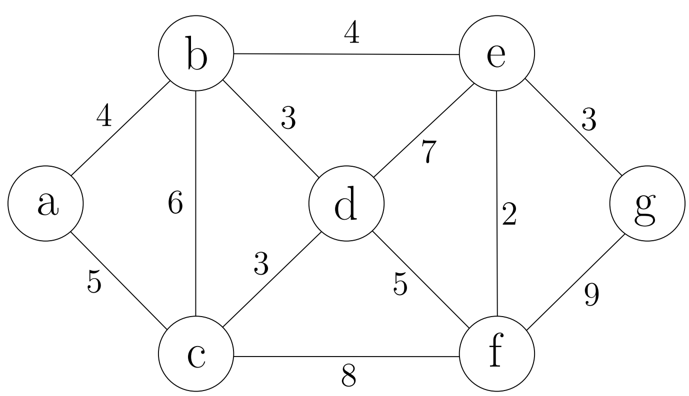

Figure 1: Input graph to Questions 3 and 4.

Problem 5

Let $G = (V, E)$ be an undirected graph with edges weights $w : E \to R^+$. For all $e \in E$, define $w_1(e) = \alpha w(e)$ for some $\alpha > 0$, $w_2(e) = w(e) + \beta$ for some $\beta > 0$, and $w_3(e) = w(e)^2$.

1. Suppose p is a shortest s-t path for the weights w. Is p still optimal under w_1? What about under w_2? What about under w_3?
2. Suppose T is minimum weight spanning tree for the weights w. Is T still optimal under w_1? What about under w_2? What about under w_3?

Problem 6

Consider the following generalization of the shortest path problem where in addition to edge lengths, each vertex has a cost. The objective is to find an s-t path that minimizes the total length of the edges in the path plus the cost of the vertices in the path. Design an efficient algorithm for this problem.

Problem 7

It is not uncommon that a given optimization problem has multiple optimal solutions. For example, in an instance of the shortest s-t path problem, there could be multiple shortest paths connecting s and t. In such situations, it may be desirable to break ties in favor of a path that uses the fewest edges.

Show how to reduce this problem to a standard shortest path questions. You can assume that the edge lengths ℓ are positive integers.

1. Let us define a new edge function $\ell'(e) = M\ell(e)$ for each edge e. Show that if P and Q are two s-t paths such that $\ell(P) < \ell(Q)$ then $\ell'(Q) - \ell'(P) \geq M$.
2. Let us define a new edge function $\ell''(e) = M\ell(e) + 1$ for each edge e. Show that if P and Q are two s-t paths such that $\ell(P) = \ell(Q)$ but P uses fewer edges than Q then $\ell''(P) < \ell''(Q)$.
3. Show how to set M in the second function so that the shortest s-t path under ℓ'' is also shortest under ℓ and uses the fewest edges among all such shortest paths.

Problem 8

Suppose we are to schedule print jobs on a printer. Each job j has associated a weight $w_j > 0$ (representing how important the job is) and a processing time t_j (representing how long the job takes). A schedule σ is an ordering of the jobs that tell the printer how to process the jobs. Let C_j^σ be the completion time of job j under the schedule σ.

Design a greedy algorithm that computes a schedule σ minimizing the sum of weighted completion times, that is, minimizing $\sum_j w_j C_j^\sigma$.