1. On an overnight camping trip in Sunnydale National Park, you are woken from a restless sleep by a scream. As you crawl out of your tent to investigate, a terrified park ranger runs out of the woods, covered in blood and clutching a crumpled piece of paper to his chest. As he reaches your tent, he gasps, "Get out... while... you...", thrusts the paper into your hands, and falls to the ground. Checking his pulse, you discover that the ranger is stone dead.

   You look down at the paper and recognize a map of the park, drawn as an undirected graph, where vertices represent landmarks in the park, and edges represent trails between those landmarks. (Trails start and end at landmarks and do not cross.) You recognize one of the vertices as your current location; several vertices on the boundary of the map are labeled EXIT.

   On closer examination, you notice that someone (perhaps the poor dead park ranger) has written a real number between 0 and 1 next to each vertex and each edge. A scrawled note on the back of the map indicates that a number next to an edge is the probability of encountering a vampire along the corresponding trail, and a number next to a vertex is the probability of encountering a vampire at the corresponding landmark. (Vampires can't stand each other's company, so you'll never see more than one vampire on the same trail or at the same landmark.) The note warns you that stepping off the marked trails will result in a slow and painful death.

   You glance down at the corpse at your feet. Yes, his death certainly looked painful. Wait, was that a twitch? Are his teeth getting longer? After driving a tent stake through the undead ranger's heart, you wisely decide to leave the park immediately.

   Describe and analyze an efficient algorithm to find a path from your current location to an arbitrary EXIT node, such that the total *expected number* of vampires encountered along the path is as small as possible. *Be sure to account for* ***both*** *the vertex probabilities* ***and*** *the edge probabilities!*

2. Consider the following solution for the union-find problem, called *union-by-weight*. Each set leader $\overline{x}$ stores the number of elements of its set in the field $weight(\overline{x})$. Whenever we UNION two sets, the leader of the *smaller* set becomes a new child of the leader of the *larger* set (breaking ties arbitrarily).

```
MAKESET(x):
  parent(x) ← x
  weight(x) ← 1
```

```
FIND(x):
  while x ≠ parent(x)
      x ← parent(x)
  return x
```

```
UNION(x, y)
  x̄ ← FIND(x)
  ȳ ← FIND(y)
  if weight(x̄) > weight(ȳ)
      parent(ȳ) ← x̄
      weight(x̄) ← weight(x̄) + weight(ȳ)
  else
      parent(x̄) ← ȳ
      weight(x̄) ← weight(x̄) + weight(ȳ)
```

Prove that if we use union-by-weight, the *worst-case* running time of FIND is $O(\log n)$.

3. ***Prove or disprove***[1] each of the following statements.

   (a) Let G be an arbitrary undirected graph with arbitrary distinct weights on the edges. The minimum spanning tree of G includes the lightest edge in every cycle in G.

   (b) Let G be an arbitrary undirected graph with arbitrary distinct weights on the edges. The minimum spanning tree of G excludes the heaviest edge in every cycle in G.

4. In Homework 2, you were asked to analyze the following algorithm to find the kth smallest element from an unsorted array. (The algorithm is presented here in iterative form, rather than the recursive form you saw in the homework, but it's exactly the same algorithm.)

```
QUICKSELECT(A[1..n], k):
  i ← 1; j ← n
  while i ≤ j
      r ← PARTITION(A[i..j], RANDOM(i, j))
      if r = k
          return A[r]
      else if r > k
          j ← r − 1
      else
          i ← r + 1
```

   The algorithm relies on two subroutines. $\text{RANDOM}(i, j)$ returns an integer chosen uniformly at random from the range $[i..j]$. $\text{PARTITION}(A[i..j], p)$ partitions the subarray $A[i..j]$ using the pivot value $A[p]$ and returns the new index of the pivot value in the partitioned array.

   What is the *exact* expected number of iterations of the main loop when $k = 1$? ***Prove*** your answer is correct. A correct $\Theta(\cdot)$ bound (with proof) is worth 7 points. You may assume that the input array $A[\,]$ contains n distinct integers.

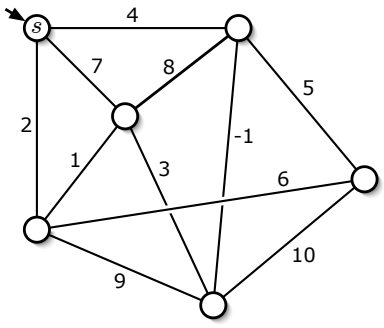
5. Find the following spanning trees for the weighted graph shown below.

   (a) A breadth-first spanning tree rooted at s.

   (b) A depth-first spanning tree rooted at s.

   (c) A shortest-path tree rooted at s.

   (d) A minimum spanning tree.



   You do *not* need to justify your answers; just clearly indicate the edges of each spanning tree. Yes, one of the edges has negative weight.

[1] But not both! If you give us *both* a proof *and* a disproof for the same statement, you will get no credit, even if one of your arguments is correct.