1. Describe and analyze an algorithm to compute the shortest path from vertex s to vertex t in a directed graph with weighted edges, where exactly *one* edge $u \to v$ has negative weight. First check whether G has a negative length cycle. Then, find the shortest path length from s to t. *[Hint: Modify the input graph and run Dijkstra's algorithm.]*

2. You just discovered your best friend from elementary school on Twitbook. You both want to meet as soon as possible, but you live in two different cites that are far apart. To minimize travel time, you agree to meet at an intermediate city, and then you simultaneously hop in your cars and start driving toward each other. But where *exactly* should you meet?

   You are given a weighted graph $G = (V, E)$, where the vertices V represent cities and the edges E represent roads that directly connect cities. Each edge e has a weight $w(e)$ equal to the time required to travel between the two cities. You are also given a vertex p, representing your starting location, and a vertex q, representing your friend's starting location.

   Describe and analyze an algorithm to find the target vertex t that allows you and your friend to meet as soon as possible, assuming both of you leave home *right now*.

To think about later:

3. Let $G = (V, E)$ be a directed graph with edge length $\ell : E \to \mathbb{R}^+$. A subset of the edges $E' \subseteq E$ are considered risky. Describe an algorithm that given $G = (V, E)$, the edge lengths ℓ, the risky subset E', a node s and an integer h finds for each node $v \in V$ the shortest path distance from s to v among all paths that contain at most h risky edges. *Hint:* Apply the dynamic programming idea behind Bellman-Ford algorithm or layering.