Lab 7

Exercise 1: Modify the code for the Bellman-Ford single-source shortest path algorithm so that it takes as input the origin x and destination y and returns the actual shortest *path* from x to y and not just the length of the shortest path. You can assume that y is reachable from x and that there are no negative weight cycles.

Example solution: We add a list `from` of length n, where there are n vertices. `from`[u] is the vertex right before u on the shortest path from x to u (if u is x, then we set `from`[u] to -1).

```
def bellmanFord(A, x, y):
    # E is a list of edges with weights
    E = []
    for i in xrange(len(A)):
        for p in A[i]:
            E += [[i] + p]
    # dist[i] is the length of the shortest path to i
    dist = [float('infinity')]*len(A)
    from = [-1]*len(A)
    dist[x] = 0
    for i in xrange(len(A) - 1):
        for e in E:
            u = e[0]
            v = e[1]
            weight = e[2]
            if dist[u] + weight < dist[v]:
                from[v] = u
                dist[v] = weight
    # look for negative weight cycles
    for e in E:
        u = e[0]
        v = e[1]
        weight = e[2]
        if dist[u] + weight < dist[v]:
            return -1

    # build the shortest path, from the end to the beginning
    L = []
    at = y
    while at != -1:
        L += [at]
        at = from[at]
    L.reverse()
    return L
```