CS 361: Algorithms and Data Structures
Graph Algorithms

1. Suppose you are given a start word x and an ending word y that both have length n. You also have access to a dictionary D that contains a list of words (again, all with length n). The goal is to find a sequence of words, starting with x and ending at y, such that

(a) Each word in the sequence differs by only a single letter from the word that precedes it

(b) Each word in the sequence occurs in the dictionary

For example, if $x =$ `clash` and $y =$`clown`, we might have:

```
clash, flash, flask, flack, flock, clock, crock, crook, croon, crown, clown
```

Write a method that takes in both words x, and y, and the dictionary D and returns a valid sequence if one exists or null if no such sequence is possible.

2. Suppose you are given a 2-dimensional array M. Each cell in M contains a character: either 'U', 'D', 'L', or 'R' representing up, down, left, or right respectively. The goal is to find a lowest-cost path from the upper-left corner of the array to the bottom-right corner of the array. The rules for traversing the array are as follows:

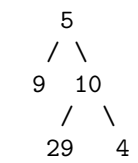
- If you move according to the direction in your current cell, you incur a cost of 0
- If you move in any other direction, you incur a cost of 1

For example, suppose M contains:

```
[D, D, U]
[L, R, D]
[R, R, L]
```

Then the lowest-cost path from $M[0][0]$ to $M[2][2]$ would have cost 1 if we follow the path {D, D, R, R} or {D, R, R, D}. Note in this case there are multiple lowest-cost paths.

3. Write a method that takes as input the root node of a binary tree and returns a 2-dimensional array that stores a level order traversal of the nodes. (Recall that a level order traversal is when we traverse the tree from left to right, top to bottom.) For example, if we have the following binary tree:



then your method would return the 2-dimensional array:

```
[[5],
[9, 10]
[29, 4]]
```

Note that each row of the 2-dimensional array potentially contains a different number of nodes.