Project Description

It's time to show off what you've learned about technical interviewing!

For this project, you will be given five technical interviewing questions on a variety of topics discussed in the technical interviewing course. You should write up a clean and efficient answer in Python, as well as a text explanation of the efficiency of your code and your design choices. A qualified reviewer will look over your answer and give you feedback on anything that might be awesome or lacking—is your solution the most efficient one possible? Are you doing a good job of explaining your thoughts? Is your code elegant and easy to read?

Answer the following questions:

Question 1

Given two strings `s` and `t`, determine whether some anagram of `t` is a substring of `s`. For example: if `s = "udacity"` and `t = "ad"`, then the function returns `True`. Your function definition should look like: `question1(s, t)` and return a boolean `True` or `False`.

Question 2

Given a string `a`, find the longest palindromic substring contained in `a`. Your function definition should look like `question2(a)`, and return a string.

Question 3

Given an undirected graph G, find the minimum spanning tree within G. A minimum spanning tree connects all vertices in a graph with the smallest possible total weight of edges. Your function should take in and return an adjacency list structured like this:

```
{'A': [('B', 2)],
 'B': [('A', 2), ('C', 5)],
 'C': [('B', 5)]}
```

Vertices are represented as unique strings. The function definition should be `question3(G)`

Question 4

Find the least common ancestor between two nodes on a binary search tree. The least common ancestor is the farthest node from the root that is an ancestor of both nodes. For example, the root is a common ancestor of all nodes on the tree, but if both nodes are descendents of the root's left child, then that left child might be the lowest common ancestor. You can assume that both nodes are in the tree, and the tree itself adheres to all BST properties. The function definition should look like `question4(T, r, n1, n2)`, where `T` is the tree represented as a matrix, where the index of the list is

particular order. For example, one test case might be

```
question4([[0, 1, 0, 0, 0],
           [0, 0, 0, 0, 0],
           [0, 0, 0, 0, 0],
           [1, 0, 0, 0, 1],
           [0, 0, 0, 0, 0]],
          3,
          1,
          4)
```

and the answer would be `3`.

Question 5

Find the element in a singly linked list that's `m` elements from the end. For example, if a linked list has 5 elements, the 3rd element from the end is the 3rd element. The function definition should look like `question5(ll, m)`, where `ll` is the first node of a linked list and `m` is the "mth number from the end". You should copy/paste the `Node` class below to use as a representation of a node in the linked list. Return the value of the node at that position.

```
class Node(object):
  def __init__(self, data):
    self.data = data
    self.next = None
```

NEXT