Algorithms Laboratory (CS29203)
Assignment 9: Hashing & Graph Algorithms
Department of CSE, IIT Kharagpur

14th November 2024

Question-1

(50 points)
Consider a string of English alphabets. We define the following two operations on the string as follows:

- *Right shift*: Replace every letter with the successive letter of the English alphabet, where *'z'* is replaced by *'a'*. For example, *"abc"* can be right-shifted to *"bcd"*, or *"xyz"* can be right-shifted to *"yza"*.
- *Left shift*: Replace every letter with the preceding letter of the English alphabet, where *'a'* is replaced by *'z'*. For example, *"bcd"* can be left-shifted to *"abc"*, or *"yza"* can be left-shifted to *"xyz"*.

We can keep shifting the string in both directions to form an endless shifting sequence. For example, shift *"abc"* to form the sequence: *"abc", "bcd", ... "xyz", "yza", ... , "zab", "abc", ...*

Consider an array `S[ ]` of strings. Your task is to group together all `S[i]` that belong to the same shifting sequence. You may return the answer in any order.

You **must** use the idea of **hashing** to solve the problem. The time complexity should not exceed $O(L)$, where L is the sum of the lengths of all strings.

Example 1:

```
(Input) S = ["abc","bcd","acef","xyz","az","ba","a","z"]
(Output) "acef"
         "a","z"
         "abc","bcd","xyz"
         "az","ba"
```

Example 2:

```
(Input) S = ["x"]
(Output) "x"
```

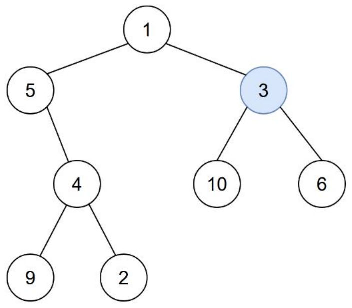
Question-2

(50 points)
Imagine a graph representation of a social network (e.g. Facebook/Twitter), where each node represents an user with an unique non-negative integer ID associated with it, and (undirected, unweighted) edges to connect nodes (or people) who are friends. For simplicity we assume the graph to be a binary tree. A global news is out-broken and is being spread across social media. We are trying to see how fast the news is spread among people with the help of graph algorithms.

Suppose the starting point of the news is known, i.e. we know which user has initiated the spreading process (hence the node ID is known). So the news starts spreading from that starting node, and spreads to another node if:

- the news has not yet spread into that node
- the node is adjacent to the node to which the news has been already spread

Consider that spreading of the news from one node to its neighbor node(s) take 1 unit of time. Your task is to find out how much time will it take to spread the news to the entire network.



For example, consider the above network graph where the starting node ID is 3. In this case, the unit of time needed to spread to the whole network is 4 because of the following reason:

- Time 0: node 3
- Time 1: node 1, 10, 6
- Time 2: node 5
- Time 3: node 4
- Time 4: node 9, 2

Since the input is a tree, you have to convert this to a graph first. You may use the following idea. Let your tree is the following:

```
    1
   / \
  2   3
     / \
    4   5
```

Then by traversing the tree you get the following information:

- Node 1 is connected to nodes 2 and 3.
- Node 2 is connected to node 1.
- Node 3 is connected to nodes 1, 4, and 5.
- Node 4 is connected to node 3.
- Node 5 is connected to node 3.

Now simply build a 2D adjacency matrix accordingly where the following neighbour information is stored:

- Node 1: [Node 2, Node 3]
- Node 2: [Node 1]
- Node 3: [Node 1, Node 4, Node 5]
- Node 4: [Node 3]
- Node 5: [Node 3]

Example:

```
(Input) Enter the tree nodes: [1  5  3  -1  4  10  6  9  2]
        Enter the ID of starting node: 3
(we use the same input format as previous lab, where a missing child is denoted as -1)

(Output) Total unit of time required is: 4
```