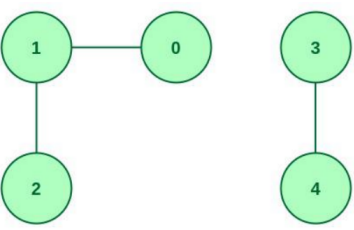


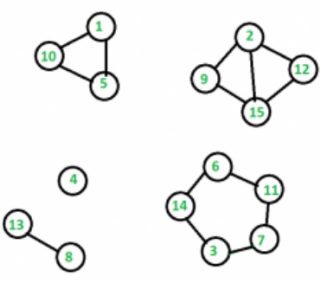
COL106: Assignment 5

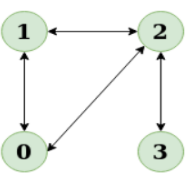
Question 1 - Use BFS/DFS

Find the number of connected components in a given graph. The graph input can be taken by first taking an integer input n , and a representation of the graph as either an adjacency matrix or an adjacency list.

Sample Test Cases

1.  In this graph, the number of connected components is 2. According to the above input format, you can write code such that the above graph can be inputted as:
 $n = 5$
Adjacency list: `[[1], [0, 2], [1], [4], [3]]`

2.  In this graph, the number of connected components is 5. According to the above input format, you can write code such that the above graph can be inputted as: (above graph does not have a 0 node, so we can just consider all node values decremented by 1 (node 1 becomes node 0, etc))
 $n = 15$
Adjacency list: `[[4, 9], [8, 11, 14], [6, 13], [], [0, 9], [10, 13], [2, 10], [12], [1, 14], [0, 4], [5, 6], [1, 14], [7], [2, 5], [1, 8, 11]]`

3.  In this graph, the number of connected components is 1 (consider all edges as undirected). According to the above input format, you can write code such that the above graph can be inputted as:
 $n = 4$
Adjacency list: `[[1, 2], [0, 2], [0, 1, 3], [2]]`

Question 2 - Use Trie

Build a search suggestion system: Given an array of product names build a trie containing all product names. Then, for every query string return a list of all products which could be possible autocomplete options after the given query string, that is, the query string should be the prefix of those product names.

Sample Test Cases

Product Names: products = ["mobile", "mouse", "moneypot", "monitor", "mousepad"]

Queries = ["m", "mon", "t", "mouse"]

Once you build the trie, it should be easy to return the answers to each of the above queries.

Sample outputs:

1. For "m": ["mobile", "mouse", "moneypot", "monitor", "mousepad"]
2. For "mon": ["moneypot", "monitor"]
3. For "t": []
4. For "mouse": ["mouse", "mousepad"]

Question 3 - Use KMP

Given an array of strings, find the longest common prefix string.

Sample Test Cases

1. Input strings = ["abcdefgh", "abcefg"]
Output = "abc"
2. Input strings = ["abcdefgh", "aefghijk", "abcefg"]
Output = "a"
3. Input strings = ["sasadadaghjt", "sadadadaghjt", "sadadaghjt", "sadadadadadadaghjt"]
Output = "sadada"

Question 4 - Stable Bucket Sort

Given a Vector of Vectors of length n , where each internal vector is of length 2 and contains integers, write a stable Bucket sort algorithm which sorts the outer vector according to the first element in the inner vectors only, in $O(n)$ time. Consider m is the maximum integer value present in the vectors. You are guaranteed that m is at most $\max(1000, 3n)$. (Therefore an $O(m)$ solution will also be $O(n)$)

Sample Test Cases

1. Input = [[2, 4], [2, 6], [1, 10], [1, 5], [1, 7]],
Output = [[1, 10], [1, 5], [1, 7], [2, 4], [2, 6]];
2. Input = [[50, 20], [10, 30], [20, 40], [50, 10], [10, 20], [15, 25]]
Output = [[10, 30], [10, 20], [15, 25], [20, 40], [50, 20], [50, 10]]
3. Input = [[35, 45], [50, 10], [35, 30], [50, 20], [10, 40], [35, 25], [10, 30], [50, 40], [35, 35], [10, 20]],
Output = [[10, 40], [10, 30], [10, 20], [50, 10], [50, 20], [50, 40], [35, 45], [35, 30], [35, 25], [35, 35]];