I. Adjacency Lists / Matrices

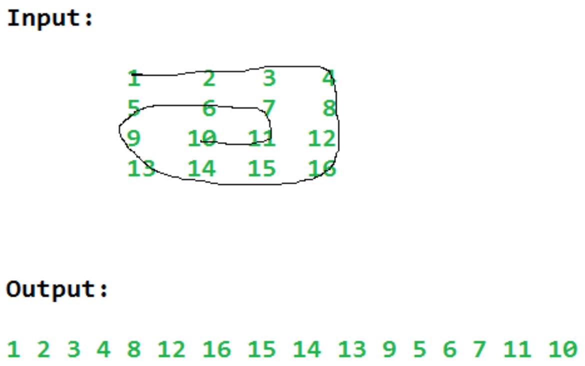
A) Write code to find whether or not that element in a matrix where the elements in each row and column are in a non-decreasing order

Example:

```
{  2, 14, 26, 37, 43, 51, }

{  4, 16, 28, 38, 44, 54, }

{  6, 18, 30, 39, 45, 57, }

{  8, 20, 32, 40, 46, 60, }

{ 10, 22, 34, 41, 47, 63, }

{ 12, 24, 36, 42, 48, 66, }
```

```
public static boolean contains(int val, int[][] a) {
        int len = a.length;
        int row = 0;
        for (int i = 0; i < len; i++) {
                if (val == a[0][i]) {
                        row = i;
                        return true;
                }
                if (val > a[0][i])
                        row = i;
        }
        boolean found = false;
        for (int j = 0; j < len; j++) {
                if (val == a[j][row]) {
                        return true;
                }
        }
        return found;
}
```

B) Spiral Matrix: Write code that traverses and prints out a matrix in a spiral form



```
Input:

1    2    3    4
5    6    7    8
9    10   11   12
13   14   15   16


Output:

1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10
```

See explanation here: https://www.geeksforgeeks.org/print-a-given-matrix-in-spiral-form/

II. Trie

A) Advantages of a Trie

- Can insert and find strings in ***O(L)*** time where *L* represent the length of a single word.
- Print words in alphabetical order
- Account for probable use of space

See more: https://www.geeksforgeeks.org/advantages-trie-data-structure/

B) Dis-advantages of a Trie

A lot of extra memory is required to faithfully implement a trie and can have slower retrieval depending where all the memory is stored.

See more: https://stackoverflow.com/questions/32835635/disadvantages-of-tries

C) Use cases for a Trie

1. Auto complete – see what words come after the few letters that have already been typed

2. *See some examples here:* https://stackoverflow.com/questions/29933907/what-are-some-other-possible-use-cases-of-a-trie-data-structure-other-than-t9-sp

D) Given a trie, and knowing that each word is denoted by an "isLeaf() == true," count the total number words present in a trie denoted by an alphabet of size 26, the children of each node are represented by a simple array.

```
final static alphabetLength = 26;

static class TrieNode
{
   TrieNode[] children =  new TrieNode[alphabetLength];
   boolean isLeaf;

    TrieNode(){
        isLeaf = false;
        for (int i = 0; i < alphabetLength; i++)
             children[i] = null;
    }
};
```

See the recursive solution here: https://www.geeksforgeeks.org/counting-number-words-trie/

III. B-Tree

A) List Properties of B-Tree

1) All leaves are at same level.

2) A B-Tree is defined by the term minimum degree 't'. The value of t depends upon disk block size.

3) Every node except root must contain at least t-1 keys. Root may contain minimum 1 key.

4) All nodes (including root) may contain at most 2t – 1 keys.

5) Number of children of a node is equal to the number of keys in it plus 1.

6) All keys of a node are sorted in increasing order. The child between two keys k1 and k2 contains all keys in range from k1 and k2.

7) B-Tree grows and shrinks from root which is unlike Binary Search Tree. Binary Search Trees grow downward and also shrink from downward.

8) Like other balanced Binary Search Trees, time complexity to search, insert and delete is O(Logn)

IV. Graph

A) Given a list of edges in a graph or "Forest," write code to find the distinct amount of "trees" or separate nodes (e.g. other metaphors, islands in an ocean, trees in a forest, disconected components)

Input : edges[] = {0, 1}, {0, 2}, {3, 4}

Output : 2

Explanation : There are 2 trees

```
   0    3

  / \    \

 1   2    4
```

See solution here: https://www.geeksforgeeks.org/count-number-trees-forest/

V. Dijkstra - Proofs

A) Does this algorithm work for negatives - why or why not?

See a good explanation here: https://stackoverflow.com/questions/13159337/why-doesnt-dijkstras-algorithm-work-for-negative-weight-edges

B) Does the shortest path change when weights of all edges are multiplied by 10?

See part 2 here: https://www.geeksforgeeks.org/interesting-shortest-path-questions-set-1/

C) Given a directed weighted graph and the shortest path from vertex 's' to 't' => D(s,y),

If the weight of every edge is increased by 10 units, does the shortest path remain same in the modified graph?

See part 23here: https://www.geeksforgeeks.org/interesting-shortest-path-questions-set-1/