QUESTIONS

1. Write a program to find the adjacency list of a given directed graph G which is represented as adjacency matrix.

   Input Format:

   - The first line of the input contains a positive integer n, the number of vertices in the graph, in the range 1 to 1000.
   - The next lines represents the Adjacency matrix representation of the given graph.

   Output Format:

   - The n lines contain the adjacency list of each node in ascending order. Each line contains the label of the respective node followed by the nodes adjacent to it sorted in ascending order from left to right separated by single space. If a node has no adjacent nodes, then the line corresponding to its adjacency list will contain the label of that node only.
     Note: In a graph with n vertices, the vertices are labeled from 0 to $n-1$.

   Sample Input:

   5
   0 1 0 0 1
   0 0 1 0 0
   0 0 0 0 0
   0 0 1 0 0
   0 0 0 1 0

   Sample Output:

   0 1 4
   1 2
   2
   3 2
   4 3

2. You are given a connected undirected graph G. Write a program for Breadth First Traversal(BFS) of the graph. Find the BFS traversal of the graph starting from the 0th vertex from left to right according to the graph.

   Input Format:

   - First line consists of V, the number of vertices in the graph, in the range 1 to 1000.
   - The subsequent n lines contains the label of the respective node followed by the nodes adjacent to it in ascending order.

   Output Format:

   - The corresponding Depth First traversal.

   Sample Input 1:

   5
   0 1 2 3
   1 0
   2 0 4
   3 0
   4 2

   Sample Output 1:

   0 1 2 3 4

Sample Input 2:

4
0 1 3
1 0 2
2 1
3 0

Sample Output 2:

0 1 3 2

3. You are given a connected undirected graph G. Write a program for Depth First Traversal(DFS) of the graph. You can use a recursive approach to find the DFS traversal of the graph starting from the 0th vertex from left to right according to the graph.

Input Format:

- First line consists of V, the number of vertices in the graph, in the range 1 to 1000.
- The subsequent n lines contains the label of the respective node followed by the nodes adjacent to it in ascending order.

Output Format:

- The corresponding Depth First traversal.

Sample Input 1:

5
0 1 2 3
1 0
2 0 4
3 0
4 2

Sample Output 1:

0 1 2 4 3

Sample Input 2:

4
0 1 3
1 0 2
2 1
3 0

Sample Output 2:

0 1 2 3

4. Write a program to compute the minimum spanning tree of a connected undirected graph G using the following algorithms:

   (a) Kruskal's algorithm

   (b) Prim's algorithm

Input Format:

- First line contains a character from { 'a', 'b'}:
  - If the input character is 'a' then compute the minimum spanning tree using Kruskal's algorithm

  - Else if the character is 'b' compute the minimum spanning tree using Prim's algorithm
- Second line contains an integer $n \in [1, 1000]$, that denotes the number of vertices in the graph.
- The subsequent n lines contain the label of the respective node followed by the nodes adjacent to it sorted in ascending order from left to right separated by single space.
- The subsequent n lines contain label of the respective node followed by the weights of the edges corresponding to the adjacency list separated by single space. The edge weights are real numbers in the range [-10000, 10000]. Further, no two edges have the same weight.

Output Format:

- Single line containing the sum of the edge weights of the minimum spanning tree.

Sample Input 1:

```
a
7
0 1 5
1 0 2 6
2 1 3
3 2 4 6
4 3 5 6
5 0 4
6 1 3 4
0 28 10
1 28 16 14
2 16 12
3 12 22 18
4 22 25 24
5 10 25
6 14 18 24
```

Sample Output 1:

```
99
```

Sample Input 2:

```
b
7
0 1 5
1 0 2 6
2 1 3
3 2 4 6
4 3 5 6
5 0 4
6 1 3 4
0 28 10
1 28 16 14
2 16 12
3 12 22 18
4 22 25 24
5 10 25
6 14 18 24
```

Sample Output 2:

```
99
```

5. Write a program that implements Dijkstra's algorithm for computing shortest paths in a directed graph with positive edge weights. Assume that the nodes are labeled from 0 to $n - 1$.

Input Format:

- The first line of the input contains a positive integer $n \in [1, 1000]$, the number of nodes in the graph.
- The subsequent n lines contain the label of the respective node followed by the nodes adjacent to it, sorted in ascending order from left to right separated by single space. If a node has no adjacent nodes, then the line corresponding to its adjacency list will contain the label of that node only.
- The subsequent n lines contain label of the respective node followed by the weights of the edges corresponding to the adjacency list separated by single space. The edge weights are positive real numbers in the range (0, 10000]. If a node has no adjacent nodes, then the line corresponding to its adjacent edge weights will contain the label of that node only.
- The rest of the input consists of multiple lines, each one containing a four-letter string followed by zero, one or two integers. The integers, if given, will be in the range 0 to n-1.
  - The string "apsp" is followed by a single integer, the label of the source vertex. Print the shortest path distance from the source vertex to all the n vertices in the graph, sorted in the order of their labels, in single space separated format. Print "INF" for nodes that are unreachable from the source vertex.
  - The string "sssp" is followed by two integers, respectively, labels of the source and destination nodes. Print the shortest path from the source node to the destination node, if such a path exists. Print "UNREACHABLE", otherwise.
  - The string "stop" means terminate the program.

Output Format:

- The output, if any, of each command should be printed on a separate line.

Sample Input:

```
9
0 1 4
1 5
2 3
3 6
4
5 2 7 8
6 2
7 4
8 5 7
0 2 20
1 3
2 7
3 5
4
5 1 6 4
6 0
7 2
8 2 1
apsp 0
sssp 0 6
sssp 0 7
sssp 5 6
sssp 8 7
sssp 4 0
stop
```

Sample Output:

```
0 2 6 13 12 5 18 10 9
18
10
13
1
UNREACHABLE
```

-