National Institute of Technology Calicut
Department of Computer Science and Engineering
Fourth Semester B. Tech.(CSE)-Winter 2023-24
CS2094D Data Structures Laboratory
Assignment Cycle#3
Part B

Submission deadline (on or before): 4.04.2024, 11:00 PM

Policies for Submission and Evaluation:

- You must submit your assignment in the Eduserver course page, on or before the submission deadline.
- Ensure that your programs will compile and execute without errors using gcc compiler.
- During the evaluation, failure to execute programs without compilation errors may lead to zero marks for that evaluation.
- Your submission will also be tested for plagiarism, by automated tools. In case your code fails to pass the test, you will be straightaway awarded zero marks for this assignment and considered by the examiner for awarding F grade in the course. Detection of ANY malpractice related to the lab course can lead to awarding an F grade in the course.

Naming Conventions for Submission

- Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar, .tar, .gz). The name of this file must be

  `ASSGC<NUMBER>_<PART>_<ROLLNO>_<BATCHNO>_<FIRST-NAME>.zip`

  (Example: $ASSGC1\_A\_BxxyyyyCS\_CS01\_LAXMAN.zip$). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive.

- The source codes must be named as

  `ASSGC<NUMBER>_<PART>_<ROLLNO>_<BATCHNO>_<FIRST-NAME>_<PROGRAM-NUMBER>.c`

  (For example: $ASSGC1\_A\_BxxyyyyCS\_CS01\_LAXMAN\_1.c$). If you do not conform to the above naming conventions, your submission might not be recognized by our automated tools, and hence will lead to a score of 0 marks for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

- Violation of academic integrity will be severely penalized. Each student is expected to adhere to high standards of ethical conduct, especially those related to cheating and plagiarism. Any submitted work MUST BE an individual effort. Any academic dishonesty will result in zero marks in the corresponding exam or evaluation and will be reported to the department council for record keeping and for permission to assign F grade in the course. The department policy on academic integrity can be found at: https://minerva.nitc.ac.in/?q=node/650.

QUESTIONS

1. Write a Program to compute the Minimum spanning tree of a connected undirected graph G using Prim's algorithm . Print the sequence with which edges are picked by algorithm and total edge weight of spanning tree. The goal is to implement the following functions:

   (a) Sequence(x) : Print sequence in which edges will be picked if x is starting node.

   (b) Total_weight() : Print total cost of the minimum spanning tree.

   Input format:

   - First line contains an integer n ∈ [1, 1000], that denotes the number of vertices in the graph.
   - The subsequent n lines contain an Adjacency matrix of size (n * n) . If two nodes are directly connected by an edge, it is denoted by a positive integer in the matrix; else it is denoted by 0. The positive integer corresponds to the weight on the edge. Assume that all edge weights will be distinct.
   - Each line of input is a character from the menu list ['s','t','x'].
   - Input 's' calls the function Sequence(x).
   - Input 't' calls the function Total_weight().
   - Input 'x' terminates the execution of the program.
   - All the inputs in a line are separated by space.

   Output format:

   - Line contains an integer representation of the total edge weight of the spanning tree or , Line contains sequence of integer representing sequence of edges.

   Sample Input 1:

   ```
   6
   0 2 0 3 7 0
   2 0 1 0 8 0
   0 1 0 4 0 5
   3 0 4 0 0 6
   7 8 0 0 0 0
   0 0 5 6 0 0
   t
   s(1)
   x
   ```

   Sample Output 1:

   ```
   18
   1 2 1 0 0 3 2 5 0 4
   ```

   Sample Input 2:

   ```
   5
   0 1 5 0 0
   1 0 2 0 0
   5 2 0 3 6
   0 0 3 0 4
   0 0 6 4 0
   t
   s(1)
   x
   ```

   Sample Output 2:

   ```
   10
   ```

1 0 1 0 2 3 3 4

2. Write a Program to compute the Minimum spanning tree of a connected undirected graph G using Kruskal's Algorithm

Input format:

- First line contains an integer n ∈ [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:

```
5
0 1 2 3 4
1 0 2
2 0 1 3
3 0 2 4
4 0 3
0 1 7 10 5
1 1 3
2 7 3 4
3 10 4 2
4 5 2
```

Sample Output 1:

```
10
```

Sample Input 2:

```
6
0 1 2
1 0 2 4
2 0 1 3 4
3 2 4 5
4 1 2 3 5
5 3 4
0 7 8
1 7 3 6
2 8 3 3 4
3 3 2 2
4 6 4 2 5
5 2 5
```

SAMPLE INPUT 2:

```
6
0 1 2
1 0 2 4
2 0 1 3 4
3 2 4 5
4 1 2 3 5
5 3 4
0 7 8
1 7 3 6
2 8 3 9 4
3 9 2 1
4 6 4 2 5
5 1 5
```

Sample Output 2:

```
17
```

SAMPLE OUTPUT 2:

```
17
```

3. You are given a weighted undirected graph represented as an adjacency list. Your task is to implement Dijkstra's algorithm to find the shortest distances from a given source node to all other nodes in the graph.

Input format:

- The first line contains an integer (n), specifying 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 in ascending order of their labels.
- The subsequent n lines contain the label of the respective node, followed by the weights of the edges ($0 <$ weight < 100) corresponding to the adjacency list.
- The next line contains an integer (s), specifying the source node.
- All the inputs are separated by a space.

Output format:

- Single line containing the shortest distance from the source node to each node in the graph in ascending order of their labels (separated by space).

Sample Input 1:

```
5
1 2
2 1 3 4 5
3 2 4
4 2 3 5
5 2 4
1 5
2 5 10 2 3
3 10 8
4 2 8 5
5 3 5
2
```

Sample Output 1 :

```
5 0 10 2 3
```

Sample Input 2:

```
8
1 2
2 1 3
3 2 4 5
4 3 5 6
5 3 4 8
6 4 7
7 6 8
8 5 7
1 2
2 2 3
3 3 4 1
4 4 2 5
5 1 2 5
6 5 2
7 2 1
8 5 1
1
```

Sample Output 2 :

```
0 2 5 8 6 13 12 11
```

4. You are given a weighted undirected graph represented as a weighted adjacency matrix. Your task is to implement the Floyd-Warshall algorithm to find the shortest paths between all pairs of nodes in the graph.

Input format:

- The first line contains an integer (n), specifying the number of nodes in the graph.
- The next n lines contain n space-separated integers, representing the weighted adjacency matrix of the graph. Each integer represents the weight of the edge ($0 <$ weight < 100) between the corresponding pair of nodes. If there is no direct edge between two nodes, the weight is represented as -1.
- The diagonal elements of the adjacency matrix are always 0 (indicating no self-loops).

Output format:

- Print n lines, each containing n space-separated integers. The j-th integer in the i-th line represents the shortest distance from node i to node j.

Sample Input 1:

```
4
0 5 -1 7
5 0 3 -1
-1 3 0 2
7 -1 2 0
```

Sample Output 1 :

```
0 5 8 7
5 0 3 5
8 3 0 2
7 5 2 0
```

Sample Input 2:

```
5
0 3 -1 7 -1
3 0 2 -1 -1
-1 2 0 5 1
7 -1 5 0 6
-1 -1 1 6 0
```

Sample Output 2 :

```
0 3 5 7 6
3 0 2 7 3
5 2 0 5 1
7 7 5 0 6
6 3 1 6 0
```