**ICE-4112**

**Artificial Intelligence and Neural Computing Lab**

**BSc (Engg.) Part-IV, ODD Semester 2020**

**Assignment-1**

The purpose of this assignment is to show the adjacency list of a given graph. The graph will be taken by a list of vertex from an input file. Then the next task is to **traverse that graph** and **find the shortest path from any given source vertex and destination vertex** using **Breadth First Search** algorithm. (\*The adjacency list representation of a graph consists of an array of list. Each list corresponds to a vertex in the graph and gives the neighbors of that graph). For example, the graph has the following adjacency list representation.

**1: 2 3**

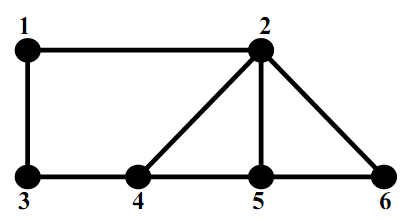
**2: 1 4 5 6**

**3: 1 4**

**4: 2 3 5**

**5: 2 4 6**

**6: 2 5**



**File Format:**

The input file will be in two parts. The first part begins with a line consisting of a single integer *n* which gives the number of vertices in the graph. Each subsequent line will represent a single edge in the graph by a pair of distinct numbers in the range *1* to *n*, separated by a space. These numbers are the end vertices of the corresponding edge. The first part of the input file defines the graph, and will be terminated by a dummy line containing “0 0”. After these lines are read your program will print the adjacency list representation of the graph to the output file. For instance, the lines below define the graph pictured above, and cause the above adjacency list representation to be printed.

6

1 2

1 3

2 4

2 5

2 6

3 4

4 5

5 6

0 0

The second part of the input file will consist of a number of lines, each consisting of a pair of integers in the range *1* to *n*, separated by a space. Each line specifies a pair of vertices in the graph; a starting point (or source) and a destination. The second part of the input file will also be terminated by the dummy line “0 0”. For each source-destination pair your program will do the following:

* Perform a **Breadth First Search (BFS)** on the given source vertex. This assigns a parent vertex (which may be nil) to every vertex in the graph.
* Use the results of BFS to print out the distance from the source vertex to the destination vertex, then use the parent pointers to print out a shortest path from source to destination.

Output File:

1: 2 3

2: 1 4 5 6

3: 1 4

4: 2 3 5

5: 2 4 6

6: 2 5

The distance from 1 to 5 is 2

A shortest path from 1 to 5 is: 1 2 5

The distance from 3 to 6 is 3

A shortest path from 3 to 6 is: 3 1 2 6

The distance from 2 to 3 is 2

A shortest path from 2 to 3 is: 2 1 3

The distance from 4 to 4 is 0

A shortest path from 4 to 4 is: 4

**Examples**

**Input File:**

**6**

**1 2**

**1 3**

**2 4**

**2 5**

**2 6**

**3 4**

**4 5**

**5 6**

**0 0**

**1 5**

**3 6**

**2 3**

**4 4**

**0 0**

Thus, your program’s operation can be broken down into two basic steps, corresponding to the two groups of input data.

1. Read and store the graph and print out its adjacency list representation.

2. Enter a loop which processes the second part of the input. Each iteration of the loop should read in one pair of vertices (source, destination), run BFS on the source vertex, print the distance to the destination vertex, then find and print the resulting shortest path, if it exists, or print a message if no path from source to destination exists.