Indian Institute of Technology Roorkee

Department of Computer Science and Engineering

CSN-261: Data Structures Laboratory (Autumn 2019-2020)

Lab Assignment-5 (L5) **Date: September 4, 2019** **Duration: 4 Weeks**

General Instructions:

1. Every Lab Assignment will be performed by the students individually. No group formation is required and the evaluations will be done every week for the students individually.

Submission and Evaluation Instructions:

1. **Submit your** zipped folder (**<filename>.zip** or **<filename>.tar.gz**) through your account in Moodle through the submission link for this Lab Assignment in Moodle course site: https://moodle.iitr.ac.in/course/view.php?id=46.
2. **Hard deadline for Final submission in Moodle: October 2, 2019 (1:00 pm Indian Time).** For any submission after Final Deadline, 20% marks will be deducted (irrespective of it is delayed by a few seconds or a few days). The key to success is starting early. You can always take a break, if you finish early.
3. The submitted zipped folder (**<filename>.zip** or **<filename>.tar.gz**) must contain the following:
   (a) The source code files in a folder
   (b) A report file (**<filename>.DOC** or **<filename>.PDF**) should contain the details like:
      i. Title page with details of the student
      ii. Problem statements
      iii. Algorithms and data structures used in the implementation
      iv. Snapshots of running the codes for each problem
4. The submission by each student will be checked with others' submission to identify any copy case (using such detection software). If we detect that the code submitted by a student is a copy (partially or fully) of other's code, then the total marks obtained by one student will be divided by the total number of students sharing the same code.

Instructions for L5:

1. Objective of this Lab Assignment is to make the students familiar with different data structures while coding the programs in the C++ language to solve some real-life problems.
2. The students are expected to have a basic knowledge of data structures and the C++ programming language.
3. The student will have to demonstrate and explain the coding done for this Lab Assignment in the next laboratory class to be held on **October 9, 2019** for evaluation.

Problem Statement 1:
Write a C++ program to perform addition and multiplication of two polynomial expressions using any data structure chosen from STL. The polynomial expressions are of the form $ax^2 + bx + c$, *where a, b* and *c* are real constants. The inputs for $2x^2 + 5x + 6$ and $2x^3 + 5x^2 + 1x + 1$ are shown below (real constants followed by their power of *x*).

Input:
No. of terms in the expression: 3
Coefficient Power
2 2
5 1
6 0

No. of terms in the expression: 4
Coefficient Power
2 3
5 2
1 1
1 0

Enter 1 to add or 2 for multiply
1

Output:
2 3
7 2
6 1
7 0

Enter 1 to add or 2 for multiply
2

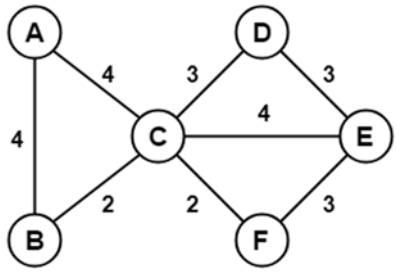
Output:
4 5
20 4
39 3
37 2
11 1
6 0

Problem Statement 2:
Given a set of nodes connected to each other in the form of a weighted undirected graph G, find the minimum spanning tree (*MST*). A spanning tree *T* of an undirected graph *G* is a subgraph that is a tree which includes all of the vertices of *G*, with minimum possible number of edges. *G* may have more than one spanning trees. The weight of a spanning tree is the sum of weights given to each edge of the spanning tree. A *minimum spanning tree (MST)* is a spanning tree whose weight is less than or equal to that of every other spanning tree.

For given input graph (given as a CSV file having the format as shown in the example below), implement Kruskal's algorithm in C++ program using **UNION FIND** data structures (**without using STL**) and show all the edges of the *MST* as output in both the command line and in the "dot file", where DOT is a graph description language. Also, print the total edge weight of the *MST*. For

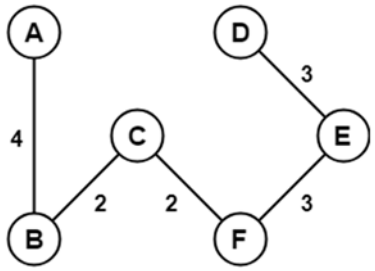
more details follow this link https://www.graphviz.org/doc/info/lang.html. Further use the “dot file” file to visualize the output graph in .pdf or .png file using **Graphviz.**

Input:



Node1 Node2 Weight
A B 4
A C 4
B C 2
C D 3
C F 2
C E 4
D E 3
F E 3

OUTPUT:
Node1 Node2 Weight
B C 2
C F 2
F E 3
D E 3
A B 4

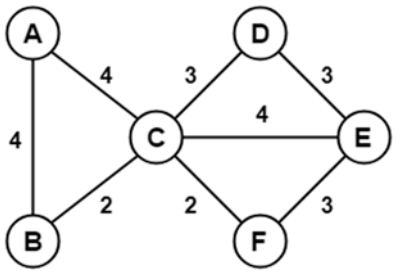


Problem Statement 3:

Write a C++ program to implement Prim's algorithm for a given input graph (given as a CSV file having the format as shown in the example below) using **Fibonacci heap** data structure to find the minimum spanning tree (*MST)*. **You can use STL** for the data structure used in this C++ program.
It is a greedy algorithm that finds a minimum spanning tree for a weighted undirected graph. This means it finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized. The algorithm generates the *MST* by adding one

vertex at a time, starting from an arbitrary vertex. At each step the cheapest possible edge weight is chosen from the already selected vertex. These algorithms find the minimum spanning forest in a possibly disconnected graph; in contrast, the most basic form of Prim's algorithm only finds minimum spanning tree in connected graphs.
Show all the edges of the *MST* as the output in command line. Also, print the total edge weight of the MST. Use *Newick* file format (https://en.wikipedia.org/wiki/Newick_format) for visualization of the *MST* in ETE Toolkit (http://etetoolkit.org/).

Input:



Node1 Node2 Weight
A B 4
A C 4
B C 2
C D 3
C F 2
C E 4
D E 3
F E 3

OUTPUT:
A B 4
B C 2
C F 2
F E 3
C D 3