IIT Roorkee
CSN 261:Data Structures Laboratory

Assingment 5

Name:Ritesh Singh

Enrollment Number:18114067

Batch-O3

Email: rsingh1@cs.iitr.ac.in

Problem 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 ax2 + bx + c, where a, b and c are real constants. The inputs for 2x2 + 5x + 6 and 2x3 + 5x2 + 1x + 1 are shown below (real constants followed by their power of x).

Algorithms Used :

1) In this I have used three unordered map to store original polynomial, added polynomial and the multiplied polynomial.

2) The coffeicients are mapped to there corresponding powers.

3) For addition, added the coffeicients of same power.

4) For multiplication, mutiplied for each degree of power.

Data Structure Used :

1) Unordered Map

2) Arrays

Snapshots :

```
ritesh@kaneki:~/Desktop/Ds_lab_5$ time ./a
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 for addition , 2 for multiplication ,3 for exit
1
Coefficient     Power
2               3
7               2
6               1
7               0

Enter 1 for addition , 2 for multiplication ,3 for exit
2
Coefficient     Power
4               5
20              4
39              3
37              2
11              1
6               0

Enter 1 for addition , 2 for multiplication ,3 for exit
3

real    0m34.897s
user    0m0.001s
sys     0m0.006s
ritesh@kaneki:~/Desktop/Ds_lab_5$
```

Problem 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.

Algorithms Used :

1) The algorithm used is Kruskal Algorithm.This algorithm is a minimum-spanning-tree algorithm which finds an edge of the least possible weight that connects any two trees in the forest.It is a greedy algorithm in graph theory as it finds a minimum spanning tree for a connectedweighted graph adding increasing cost arcs at each step.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. If the graph is not connected, then it finds a minimum spanning forest (a minimum spanning tree for each connected component).

2) Sort all the edges in non-decreasing order of their weight.

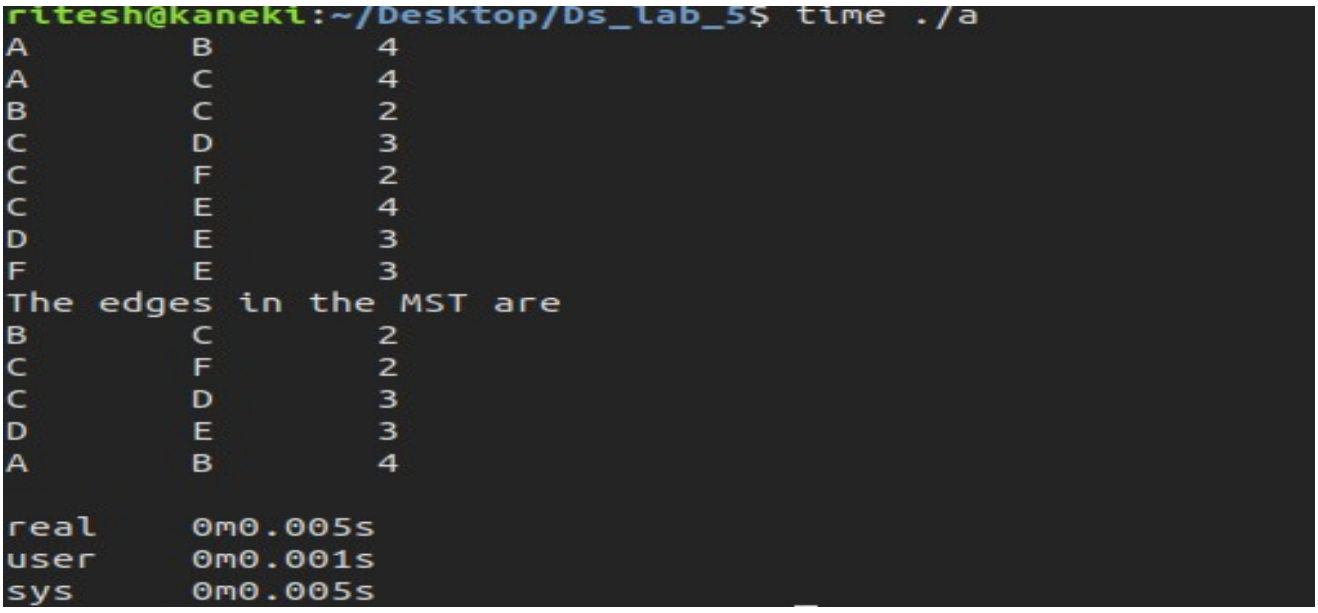
3) Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.

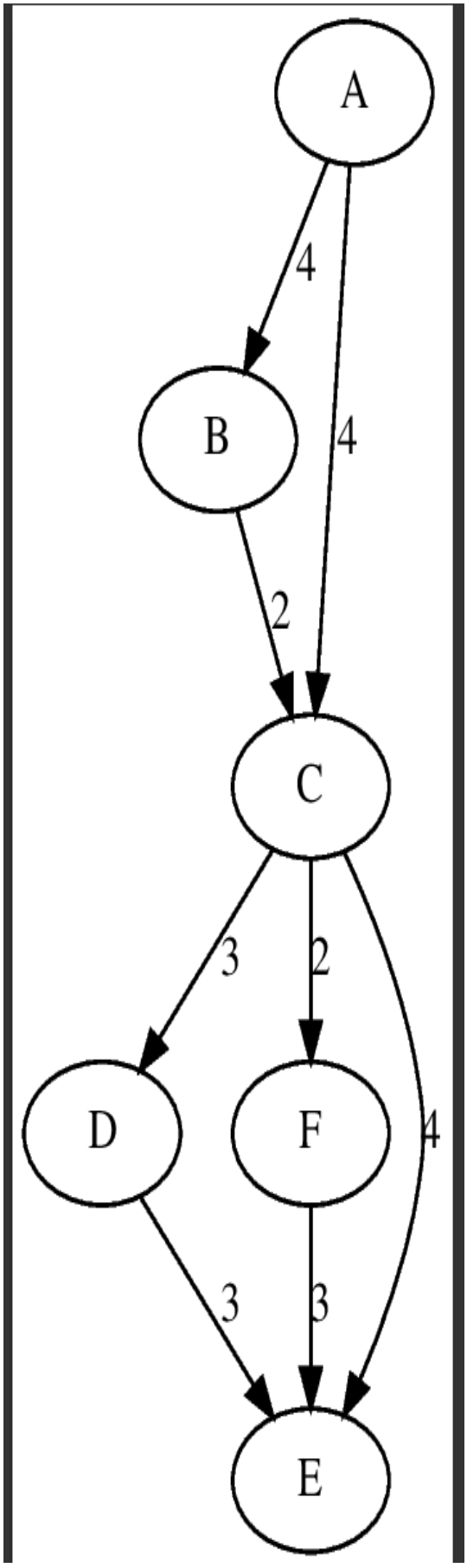
4) We use the Union Find Algorithm to check if it the current edge forms a cycle if it is added in the current MST. If yes discard it, else include it (union).

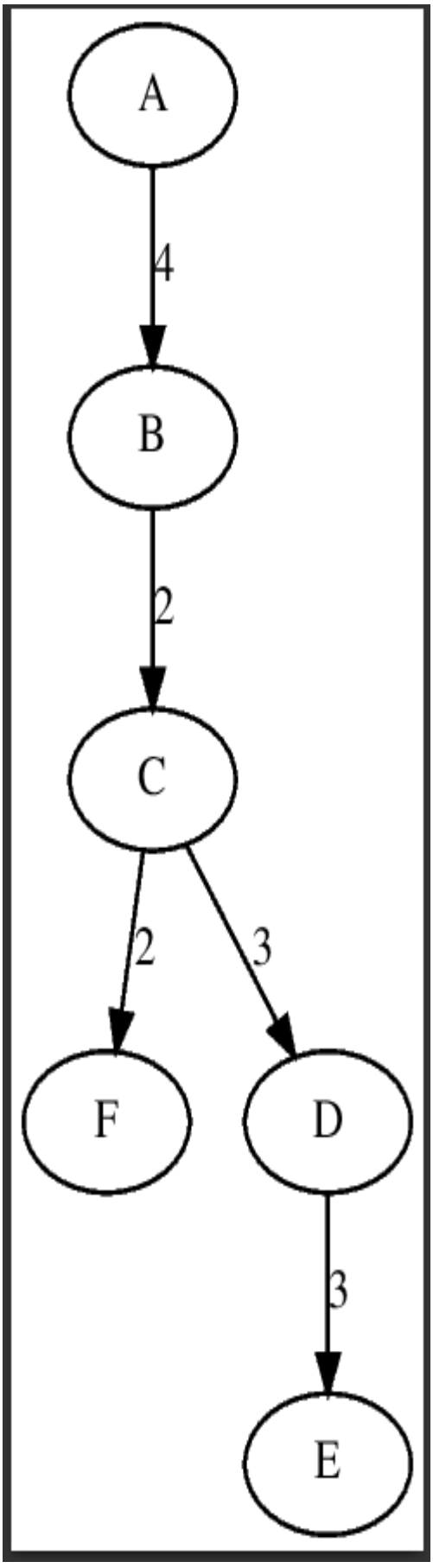
Data Structure Used :

1) Arrays
2) Structures

Snapshots :




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


A
4
B
2
C
2
3
F
D
3
E

Problem 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/).

Algorithms Used :

1) Prims Algorithm is a greedy algorithm that finds a minimum spanning treefor 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 operates by building this tree one vertex at a time, from an arbitrary starting vertex, at each step adding the cheapest possible connection from the tree to another vertex.

2) Initialize a tree with a single vertex, chosen arbitrarily from the graph.

3) Grow the tree by one edge: of the edges that connect the tree to vertices not yet in the tree, find the minimum-weight edge, and transfer it to the tree.

4) Repeat step 3 (until all vertices are in the tree).

Data Structure Used :

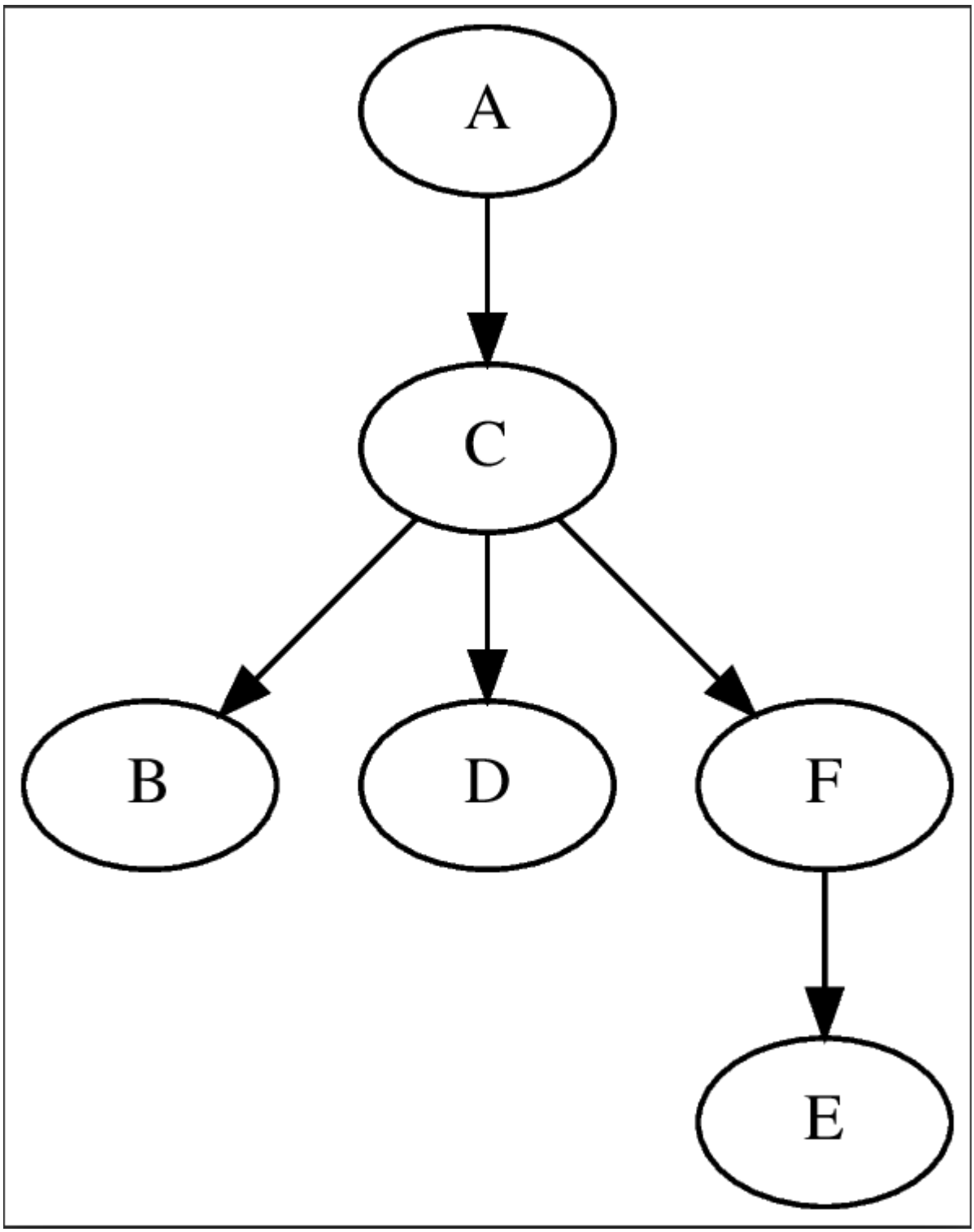
1) Fibbonaci Heap

2)Arrays

Snapshots :

```
ritesh@kaneki:~/Desktop/Ds_lab_5$ g++ problem3.cpp -o a
ritesh@kaneki:~/Desktop/Ds_lab_5$ time ./a
The input file is :
A       B       4
A       C       4
B       C       2
C       D       3
C       F       2
C       E       4
D       E       3
F       E       3
The Mst Graph is:
C - B
A - C
C - D
F - E
C - F

real    0m0.005s
user    0m0.001s
sys     0m0.005s
```


A
C
B
D
F
E