

CSC331: Data Structures - BMCC Spring 2022
Professor Byron
Program #5 specifications: Spanning tree
Date Due: 1159pm Mon May 16, 2022; no credit for late submission

Program #5 specifications: Spanning tree

Your task for this assignment is to identify a spanning tree in one connected undirected weighted graph using C++.

1. Implement a spanning tree algorithm using C++. A spanning tree is an acyclic spanning subgraph of the of a connected undirected weighted graph. The program is interactive. Graph edges with respective weights (i.e., $v_1 v_2 w$) are entered at the command line and results are displayed on the console.
2. Each input transaction represents an undirected edge of a connected weighted graph. The edge consists of two unequal non-negative integers in the range 0 to 9 representing graph vertices that the edge connects. Each edge has an assigned weight. The edge weight is a positive integer in the range 1 to 99. The three integers on each input transaction are separated by space. An input transaction containing the string "end-of-file" signals the end of the graph edge input. After the edge information is read, the process begins. Use an adjacency matrix for recording input edges and show all code used to determine the spanning tree. The input data can be assumed to be valid, so is no need to perform data validation on the input data.
3. After the edges of the spanning tree are determined, the tree edges are displayed on the console, one edge per output line, following the message: "Spanning tree:". Each output line representing a tree edge contains two integers separated by space. These integers are the two vertices representing the edge.

Sample input transactions are as follows:

```
1 3 81
3 2 62
1 2 73
end-of-file
```

Sample output expected after processing the above input will be as follows:

```
Spanning tree:
1 3
2 3
```

4. The program will be run at the command prompt by navigating to the directory containing the executable version of the program after the program is compiled.
5. Your C++ program file should be named `csc331_prog5_lastname.cpp`. Your program should contain comments starting on line 1 of the program containing the following information:
 - a. course ID and section
 - b. your full name
 - c. the program file name
 - d. the program assignment number and due date
 - e. the program purpose

You are encouraged to add additional comments throughout the program that you feel might be helpful to the reader of your source code.

6. Submit your C++ program file as an attachment to an email message to `kbyron@bmcc.cuny.edu` using a subject in this form: `"csc331_prog5_lastname"`.
7. Do your own work. All students submitting copies of the same program will receive grades of zero for the assignment and may be referred to the Dean of Students, as described in the BMCC plagiarism policy.
8. Extra credit of 33/100 points will be awarded if the following features are implemented:
 - a. input graph edges are read from a text data file;
 - b. output graph edges are written to a text data file;
 - c. input and output file names are provided as command line parameters.
9. Additional extra credit of 33/100 points will be awarded if Prim's algorithm is used to find a minimum spanning tree (MST) in the input graph. Use an adjacency matrix for recording input edges and weights and show all code used to determine the MST.

After the edges of the MST are determined, the MST edges are displayed on the console, one edge per output line, following the message: "Minimum spanning tree:". Each output line representing an MST edge contains three integers separated by space. The first two integers are the two vertices representing the edge and the third integer represents the weight of the edge.

After displaying the edges of the MST, the program displays the message: "Edge weight total:" followed by the sum of the weights from the edges comprising the MST.

Sample input transactions are as follows:

1 3 81
2 3 62
1 2 73
end-of-file

Sample output expected after processing the above input will be as follows:

Minimum spanning tree:
1 2 73
2 3 62
Edge weight total: 135