CS 3353 Spring 2024

Program 2

Building Internet on Misty Mountains

Due: 3/29 (Fri)

The infamous Misty Mountains in Middle Earth was a place of Goblins and Barlogs. But recently, some human is starting to settle in the mountains. However, because of the tough terrain, there has been no Internet services to its inhabitants until recently, where Elon Tusk has developed the Start-Link satellite system.

A group of human families has decided to work together to share Internet access. Each family has its own house, and each house can access the internet in one of the two ways:

- By installing a dish to connect directly to Start-Link. For each house, the cost of installing a dish will be different.
- By connecting the house (via cable) to another house that already has internet connection. Notice that the other house need not have a satellite, as long as it is connected to the Internet (via connection to another house) then it is fine. There is a cost of building a cable between the two houses, and it may be impossible to build a cable between two houses. Each cable is bi-directional (i.e. you can go from either end of the cable to the other end).

However, given the current limitation of Start-Link, only one house can install a dish to connect directly to Start-Link.

Your task is to write a program to determine the best way to connect all the houses to the Internet.

Problem specification

You will be given an input file that contain the following information:

- The first line contains 1 number (n), denoting the number of houses (the houses are labeled from house 1 to house n)
- The second line contains n (floating point) numbers, the k-th number denotes the cost of installing a satellite at house k)
- The third line contain one number, m, denoting the number of pairs of houses such that a cable can be built between them
- Then each of the subsequent m lines denote a pair of houses and the cost to connect the two houses via a cable. The first two numbers denote the two houses (notice that the first number need not be smaller than the second), and the third number denotes the cost (in floating point number).

You are to write a program to find the least cost needed to connect all the houses to the Internet and also return the cables that need to be built. Notice that if there is more than one solution that give the same minimum cost, you only need to return any one of them.

In addition, you should also show the path that connect from the satellite to house 1.

A sample input file is as below:

```
4
21.5 10.5 11.5 16
5
1 4  5
2 3 4.5
4 2  2.7
1 3 6.2
1 2  6.25
```

For this file:

- There are four houses
- For each house (1, 2, 3, 4), the cost of installing a satellite at that house is (21.5 10.5 11.5 16) respectively.
- There are five cables that can be build.
- Between house 1 and 4, building a cable cost 5
- Between house 2 and 3, building a cable cost 4,5
- Next three lines are similar.
  (Notice that one cannot build a cable between house 3 and 4)

Algorithm

You are to devise the algorithm to find a solution. Here are some hints to help you devise the algorithm:

- The algorithm requires you to find the minimum spanning tree of a certain graph G, where you need to convert the input data to.
- The key is the figure out how many vertices are there in the graph.
- A further hint for your graph: your graph should have weights on the edges, but no weights on the vertices.

Program Specification:

You are given a structure called Link, which is defined as follows:

```
struct Link {

  public:
  int v1, v2;
  float w;
};
```

This does not come with any methods. It will be used for parameter passing and return for the methods below. You are welcomed (but not required) to add any methods to it. (I will provide an overloaded

ostream << method). This structure is used mainly to store information about cables between two houses and its cost.

You are to implement following class called MyGraph. It represents an undirected graph with weights on each edge. You must implement the following methods:

Methods:

- Constructors:
  - MyGraph(int n): Create a graph with n vertices. The vertices are labelled **1..n**
  - MyGraph(const MyGraph& g): Construct a new graph that is a copy of g
- Methods
  - bool addEdge(int a, int b, float w): Add an edge between vertex a and b, with weight w. If the edge already exists or a vertex is not on the graph, do nothing and return false. Otherwise (addition is successful) return true.
  - void output(ostream& os): Output the graph to the ostream& specified. Your output should have the following format:
    - The first line print the number of vertices.
    - Each subsequent line prints an edge. It should print three numbers: the two vertices associated with the edge (print the vertex with the smaller number first), and the weight of the edge. You should have one space character between the numbers, and no space at the end. You can order the edges whatever way you want.
    - You must follow the format strictly. Otherwise points will be taken off and you are not eligible for extra credit
  - pair< vector<pair<int, int> > , float> MST() This function return the MST of the current tree. The output should be a pair:
    - The first object is a vector of pairs, each pair represent an edge, where the two numbers show the vertices of the edge. You should represent each edge such that the first number is smaller than the second one
    - The second floating point number is the total weight of the MST.

You are also to implement the following function (does not belong to any specific class):

- pair<vector<Link>, vector<int> >Prog2(vector<float> satcost, vector<Link> linkcost, int& sat_conn)
  - This is the function for the main task of program 2. The three inputs are
    - Vector<float> satcost: the cost of installing a satellite at each house. Notice that the house is labelled 1 to n, so the vector has n+1 entries and satcost[0] is ignored.
    - Vector<Link>: a vector of links such that each entry contains two houses and the cost of building a cable between them. Notice that if a pair of house is not in the linkcost vector, that means you cannot build a cable between them
  - The output should be the following:
    - You should update the sat_conn parameter to store the house that connect to Start-Link.

        - The function returns contain a pair. The first vector stores the list of the cables that should be built (stored as a vector). The cables in the list can be of any order. The second is a list that store the path from satellite to house 1. The first entry of that array should be 0 (denoting the satellite) and the last entry is 1 (corresponds to house 1).
    - For example, for the test case about the output should be a pair
        - Sat_conn should be updated with the value 2.
        - The first vector of the returning pair stores [(1, 4, 5), (2, 4, 2.7), (2, 3, 4.5)] (Notice that the edges can be in any order.)
        - The second vector should be [0 2 4 1]

What to hand in

You are given 3 files:

- MyGraph.h : declaration of all the types/classes/functions
- Prog2test.cpp: The driver program that read a file "test2_1.txt" and call the function and output the result.
- MyGraph.cpp: contains the code for implementation of all the classes and methods (right now there is no code in the methods (or are make to return empty vectors).

Your task is to modify MyGraph.cpp and MyGraph.h to implement the solution of the program.

You should well comment your code, and in your comments, you should describe your algorithm to solve the program.

You should zip MyGraph.h and MyGraph.cpp into a zip file and upload it to Canvas (you should upload ONLY that file)

Grading

For programs that does not compile, the best you can get is a 40

For programs that compile but does not run, the best you can get is a 70

Otherwise, your program will be judged by whether you get the MST correct (10 points), and whether you get the answer of the problem correct (15 points)

So the maximum basic score you can get is 95.

Bonus

For all programs that get full marks, their program will be timed, and the fastest of such program will get an extra bonus:

Rank	Bonus
1	60
2	50
3-4	40
5-8	35
9-15	30

16-24	20
25-34	15
35-48	10
49-60	5

If there are ties, the bonus will be averaged and split (e.g. if two students tied for first place, each will get a bonus of 55 points).