# Algorithms in C++: Final Exam Assignment
# Power Grid

1. **Objective**

   The government is building a new town, named Utopia, on top of some abandoned farm land in NJ. The town will be represented as a graph. The edges represent streets in the town, and they have an official name as well as a length in meters. The vertices represent the intersection of streets. The government officials want to ensure safety of the workers building the town, so electricity must be supplied to the town immediately and every intersection must have a traffic light installed (which, of course, requires power). All intersections must be linked by the power grid that is being installed. (Upon completion of installing the town's power grid, the engineers will connect an intersection on the outskirts of the town to the state's power grid. You do not have to account for this aspect of the project.)

   Your goal is to write a program that reads the graph representation from a file and computes the minimum set of roads on which to install power lines. The results of your program will tell the engineers how many meters of wire to purchase and on what streets the power lines should be run.

2. **Problem**

   Create a C++ project with a single source file called *powergrid.cpp*. The program takes in a single command line argument <input file>. The file is guaranteed to be non-empty, as at least the first line will be present. The first line should be an integer between 1 and 1000, which represents the number of vertices in the graph. The rest of the file should contain a variable number of lines with four components: the starting vertex, the ending vertex, the weight of the edge connecting the two vertices, and the name of the edge. Any number of problems can occur on these lines, and your solution must be equipped to handle them. The output of the program for various errors is listed below:

   - No command line argument or too many arguments:

   ```
   $ ./powergrid
   Usage: ./powergrid <input file>
   ```

   - Input file not found in the same folder as the executable:

   ```
   $ ./powergrid notfound.txt
   Error: Cannot open file 'notfound.txt'.
   ```

   The rest of the input errors are in the content of the input file itself. Throughout the remainder of this document, the content of each sample input file is displayed in a gray box.

   - The number of vertices is not an integer between 1 and 1000.

   ```
   -3
   1,2,110,Maple Ave.
   1,3,90,Summit Ave.
   2,240,Main St.
   ```

   ```
   Error: Invalid number of vertices '-3' on line 1.
   ```

- Each line should have four components. Check the number of components before verifying the content of each. Line 4 has only 3 components, so it's considered to be invalid edge data.

```
3
1,2,110,Maple Ave.
1,3,90,Summit Ave.
2,240,Main St.
```

```
Error: Invalid edge data '2,240,Main St.' on line 4.
```

- The starting vertex must be in range. For instance, if there are 3 vertices, as in the file below, the only valid vertices and 1, 2, and 3.

```
3
1,2,110,Maple Ave.
4,3,90,Summit Ave.
2,3,240,Main St.
```

```
Error: Starting vertex '4' on line 3 is not among valid values 1-3.
```

- The same holds for the ending vertex. It must be a valid integer in the range 1..(number of vertices). In the example below, there are 3 vertices permitted, namely 1 through 3.

```
3
1,2,110,Maple Ave.
1,x,90,Summit Ave.
2,3,240,Main St.
```

```
Error: Ending vertex 'x' on line 3 is not among valid values 1-3.
```

- Finally, make sure the edge weight is a positive integer.

```
3
1,2,110,Maple Ave.
1,3,90,Summit Ave.
2,3,-240,Main St.
```

```
Error: Invalid edge weight '-240' on line 4.
```

The input is guaranteed to have no self-loops and no duplicate edges connecting the same pair of vertices. Streets are considered bidirectional.

After your program parses the input data, it will run the algorithm to produce the results needed by the engineers. Consider the following valid input:

```
3
1,2,110,Maple Ave.
1,3,90,Summit Ave.
2,3,240,Main St.
```

There are 3 vertices and 3 edges in this graph. Note that the top line indicates the number of vertices only and has nothing to do with the total number of edges. Vertex 1 is adjacent to vertices 2 and 3, and vertex 2 is adjacent to vertex 3.

The least amount of wire required to connect all vertices is 200 meters. Maple Ave. and Summit Ave. should have power lines installed.

The output of the program will thus be:

```
Total wire length (meters): 200
Maple Ave. [110]
Summit Ave. [90]
```

The streets must be alphabetized and the length of each street should be included in square brackets after the name and a single space.

Note that in Utopia, all street names are unique as well as all street lengths. Therefore, in Utopia, each configuration has at most one correct answer.

It is possible for no solution to be found, however. If the data management specialist gets sleepy, he might omit some of the street names in the data file he gives you. Under those circumstances your algorithm must simply output "No solution." without the quotes, as seen below.

```
3
1,2,110,Maple Ave.
```

```
No solution.
```

A test script is supplied for you to check the correctness of your program on some of the error conditions as well as small inputs. Once you are confident that it works, you should create your own test cases to ensure the correctness of your algorithm. The final test script used to determine your grade will have a total of 25 test cases, each worth 4 points. In addition, the script will contain the usual deductions for missing and/or incorrect name and pledge as well as deductions for memory leaks discovered with valgrind. Make sure your code compiles and runs correctly in the lubuntu virtual machine supplied at the start of the semester.

## 3. Submission

Create a zip file called *powergrid.zip* that contains *powergrid.cpp* and the *makefile*. Do not submit any extraneous files.

Your submission must be received by the deadline posted in Canvas. Since this project is in lieu of the final exam, the typical late policy does not apply. **If your submission is not received by the deadline, a zero will be recorded for the grade on this assignment.**

Your work must be completed individually. Moss will be used to check for code similarity. **If Moss finds your code similar to that of others in the class, all offending parties will be reported to the Honor Board.** Remember, with a fully open-ended project, the likelihood of your code overlapping with someone else's is minimal.