# GA3: Graphs

**Due date:** 04/20/2021 11:59PM

## Introduction

An urban-planning firm is working on a city development project. The firm looks to build the city in a way that can minimize the use of resources and maximize the revenue. As project team members, you are responsible for evaluating a set of computer-generated design blueprints of the city.

Your task is to write a program to help the firm decide whether a proposed design can hardly, partially, or fully fulfill the business objectives mentioned above. Based on the information generated by the program, a design can be rated as **Good/Fair/Bad**.

## Selection Criteria

Given a design blueprint *G,* you will evaluate its business potential based on three requirements:

**1**. For each location, is there a path to go from that location to every other location?
In the case that a path between two locations does not exist, the design will be rejected, immediately.

**2.** Are there locations with maximum number of neighboring locations of more than two? This characteristic will help the firm identify potential business spots. In case there are more than one location in the search result, the first location detected by the program will be used as the given location in the next criteria. If there is no such location, Location 0 will be chosen as the given location in the next criteria.

**3.** Suppose that we either want to use the least resource to build the roads connecting every location together or build every fastest route from a given location to everywhere else. Can we keep the length difference between the two alternatives down to 10 miles or less? This will help lower project expenditure and improve travel experience for future residents.

***Assumptions:***

- There will be no blank lines in the input file.
- You may use undirected graphs for the implementation.
- Each input file may contain a maximum of 10 vertices and 45 edges.

- The first line in the input file contains the total vertices in the graph (1-10) while the following lines contain an edge between two adjacent vertices with a specified weight. The weight represents the distance from one location to another in miles.
- You are allowed to use Priority Queue STL and Vector STL to implement the solutions.
- A design is considered to be “good” only when it satisfies *all* requirements and “bad” when it fails the first requirement. Otherwise, it is a ‘fair’ design.

# Examples

**Example 1:**

***Input31.txt***

```
5                 // The graph has 5 vertices
0 1 5             // An edge connecting Vertex 0 with Vertex 1, edge weight = 5
0 2 3             // ……. Vertex 0 with Vertex 2, edge weight = 3
1 2 7             // ……. Vertex 1 with Vertex 2, edge weight = 7
1 3 9             // ……. Vertex 1 with Vertex 3, edge weight = 9
2 4 3             // ……. Vertex 2 with Vertex 4, edge weight = 3
4 3 4             // ……. Vertex 4 with Vertex 3, edge weight = 4
```

***Output31.txt***

```
1. Yes
2. Yes (Location 1, Location 2)  // If there is no such location, print “2. No”
3. Yes (MST=15, SPT=24) // If the answer is no, print "3. No (MST=15, SPT=24)"
Good
```

***Command line:***

```
./graph input=input31.txt output=output31.txt
```

**Example 2:**

***Input22.txt***

5

0 1 5

0 2 3

1 2 6

3 4 7

***Output32.txt***

1. No

Bad

***Command line:***

./graph input=input32.txt output=output32.txt

**Example 3:**

***Input33.txt***

6

0 5 4

0 3 9

5 4 1

4 1 2

1 2 4

2 3 3

***Output33.txt***

1. Yes

2. No

3. Yes (MST=14, SPT=20) // If the answer is no, print "3. No (MST=14, SPT=20)"

Fair

***Command line:***

./graph input=input33.txt output=output33.txt

## ***Submission instructions for GA3:***

We expect every student of each group will participate to solve the problem and discuss with each other. ***We will count the lowest grade of a group for every group member***, so make sure everyone in your team is submitting the same copy.

***Every student of each group*** needs to submit ***the same copy*** of the solution. GA3 needs to be turned in to our Linux server. Make sure to create a folder under your root directory, name it ***"ga3"*** (name must be in lower case and exact), only copy your .cpp and .h files to this folder, no test case or other files needed. If you use ArgumentManager.h, don't forget to turn it in, too. ***GAs will be graded only once.*** You will not get the chance of resubmission after grading. So, make sure you are submitting the correct solution before the due date.