**CPSC 2150: Week 12 HW Graph Algorithm Implementation using STL**

**Due: As indicated by submission link**

**Total Marks: 10**

**Instructions – PLEASE READ**

1. This work may be done in pairs or individually. Each pair of students must submit their own copies and clearly state names and ids of both students in their own respective submissions.
2. You should submit only one version via D2L. Check instructions from TA regarding what to submit (zip/just code). Code files must always be included.
3. Keep a copy of everything you submit in some online storage that is accessible by you only.

# EXERCISES

***2 points reserved for use of STL Containers and Algorithms: The more STL Containers and Algorithms you use in your code the higher your score will be! Scores receivable are:***

* ***0 (No use of STL)***
* ***1 (used STL Containers)***
* ***2 (used STL Containers and Algorithms).***

1. A delivery company is interested in efficient ways to deliver some goods from a source city to one or more destination cities in Canada using a road network.
   1. A file named distances.txt exists with 1 or more lines where each line has the following format.
      * sourceCity destinationCity distanceInKilometers
      * souceCity and destinationCity are single word strings and distanceInKilometers is an unsigned integer. All 3 values in each line are separated by a single space.
      * For example, X Y 10 means sourceCity is X, destinationCity is Y and distance between them is 10 kilometers.
   2. Use the Graph class provided with this assignment to solve the problem.
      * There is a constructor that builds the graph based off of a “distances.txt” file
      * Add methods to this class (and the Vertex class) as necessary
   3. In the main function, test your program as follows
      * User inputs starting location and 1 or more destination locations separated by commas
        1. Inputs locations must be checked to see if they are valid locations within the map
        2. If there are more than one destination locations then user wants to visit those locations in the order of input
        3. Here are three example cases,

**CASE 1:**

Enter the source location: A

Enter the destination location(s): D

Shortest Route from A to D:

Distance: 10 kilometers

Route: A -> C -> D

**CASE 2:**

Enter the source location: A

Enter the destination location(s): F,K,D

Shortest Route from A to F:

Distance: 50 kilometers

Route: A -> C -> F

Shortest Route from F to K:

Distance: 150 kilometers

Route: F -> T -> R -> K

Shortest Route from K to D:

Distance: 100 kilometers

Route: K -> D

**CASE 3:**

Enter the source location: K

Enter the destination location(s): M,Z

No route exists for the given locations.

*// suppose that it is not possible to go from M to Z*

* 1. Dijkstra’s algorithm must be used to find shortest paths (if paths exists) between two locations.