

# CSE331 – Project 5

## Weighted Graph

Due Date: 11:59 pm Dec.4, 2015

### 1. Project Description

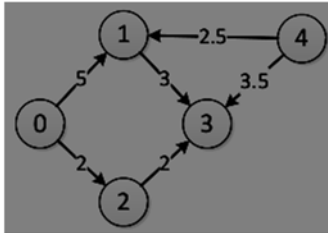
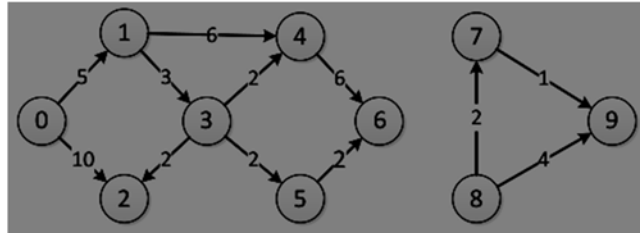
You will implement Dijkstra's algorithm for finding the shortest path from a source vertex in a graph to a destination vertex.

For a given source vertex in the graph, the algorithm finds the path with lowest cost between that vertex and every other vertex. It can also be used for finding costs of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined. For example, if the vertices of the graph represent cities and edge path costs represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between one city and all other cities.

### 2. Programming Notes

- For this project you have been given a great deal of latitude in the implementation. However, the general restrictions in the project guidelines still apply. You must provide a Makefile (feel free to modify one from a previous project), and it must produce an executable called 'prog'.
- Your program must accept three command line arguments, (1) a filename of the text file which contains a graph, (2) a source vertex and (3) a destination vertex.
- Each line in the text file contains a number indicating which vertex it is, followed by numbers separated by spaces, indicating which vertices this vertex has an edge to. The last number in a line is the distance between two vertexes.
- You may assume that all input to your program is in the correct format and the given source and destination vertexes exist in a graph.
- You may NOT assume that a given graph is a connected graph.
- If there is a path from a source vertex to a destination vertex, your program must output the shortest path and the cost it takes. The vertices must be listed in an order starting from a source vertex to a destination vertex, using a symbol “->” between each vertex. Display the cost with a floating point number with 2 digits after the point. Please see the examples below.
- If there is no path from a source vertex to a destination vertex, your program must output string “NO PATH FOUND”. Please see the examples below.
- Your program may not be graded if it doesn't follow the input or output format.

- Input and output examples:

		<pre> &gt;prog g1.txt 0 3 0-&gt;2-&gt;3 4.00  &gt;prog g1.txt 3 0 NO PATH FOUND  &gt;prog g2.txt 0 6 0-&gt;1-&gt;3-&gt;5-&gt;6 12.00  &gt;prog g2.txt 8 9 8-&gt;7-&gt;9 3.00  &gt;prog g2.txt 0 8 NO PATH FOUND </pre>
<p>g1.txt</p> <pre> 0 1 5.0 0 2 2.0 2 3 2.0 1 3 3.0 4 1 2.5 4 3 3.5 </pre>	<p>g2.txt</p> <pre> 0 1 5.0 0 2 10.0 1 3 3.0 1 4 6.0 3 2 2.0 3 4 2.0 3 5 2.0 4 6 6.0 5 6 2.0 7 9 1.0 8 7 2.0 8 9 4.0 </pre>	

### 3. Project Deliverables

The following files must be submitted via Handin no later than 11:59 pm Friday Dec.4, 2015:

- \*.cpp \*.h – your source code files must use these extensions
- Makefile – produces the executable “prog”