Purpose: Use STL containers and algorithms to represent the ADT's:  Graph, Adjacency Matrix, Hamiltonian Circuit.  
  
Data file:  Restaurants.XML  
  
Part 1: Use the Dijkstra algorithm to create an Adjacency Matrix and then find the shortest path between all points using the Dijkstra algorithm.  Time the runtime of the algorithm, from construction of the Adjacency Matrix to completion of the Dijkstra algorithm.  
  
Part 2: Use the STL permutations algorithm, program the Hamiltion Circuit.  The issue is if done the traditional way, the BigO(N!), in this case is 19! which is 121,645,100,408,832,000.  If one permutation is done every millisecond, it will take 3,857,340.82 years.  Consequently, the Hamiltonian Circuit needs to be broken into managable chunks.  Time the runtime of your algorithm.  
  
A graph ADT is a collection of vertices. There are public methods to initialize the graph, insert and delete edges and vertices, print the graph, and search the graph. The edges are stored in an adjacency list for each vertex in the vertices variable.  
  
Graph:   
1.         private:   
2.                 vertices   
3.         public:   
4.                 Init()   
5.                 insertVertex( value)   
6.                 insertEdge( startValue, endValue, weight)   
7.                 deleteVertex( value)   
8.                 deleteEdge( startValue, endValue)   
9.               printGraph()   
10.             search( value)  
  
An adjacency matrix is a structure for representing direct connections between entities in a graph, such as locations. In a 2D matrix, these entities are listed on both the horizontal and vertical axis. If there is a direct connection between two entities, it means they are adjacent and there is a 1 at that location in the matrix.   
  
Dijkstra's algorithm, named after its discoverer, Dutch computer scientist Edsger Dijkstra, is a greedy algorithm that solves the single-source shortest path problem for a directed graph with non negative edge weights. For example, if the vertices (nodes) of the graph represent cities and edge weights represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between two or more cities.  
  
Dijkstra(L[1..n, 1..n]) : array [2..n]  
1. array D[2..n]  
2. set C   
3. C <- {2, 3, 4, 5, 6, …, n}  
4. for i <- 2 to n  
5. D[i] <- L[1,i]  
6. repeat n - 2 times  
7. v <- C  // minimum D[v] extract to C  
8. v <- C - {v}   
9. for each w in C do  
10. D[w] <- min(D[w], D[v] + L[v,w])  
11. return D  
  
In the mathematical field of graph theory, a Hamiltonian path (or traceable path) is a path in an undirected or directed graph that visits each vertex exactly once. A Hamiltonian cycle (or Hamiltonian circuit) is a Hamiltonian path that is a cycle. Determining whether such paths and cycles exist in graphs is the Hamiltonian path problem, which is NP-complete.