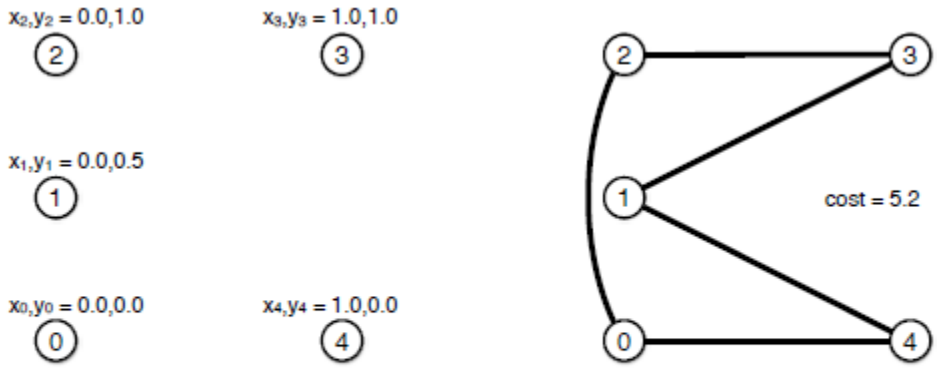
CS490
Assignment 3

**Traveling Salesman**

In this assignment you will design an algorithm to solve a fundamental problem faced by every traveling salesperson, aptly named The Traveling Salesman Problem (TSP). All traveling salespeople start from their home, travel to several cities to sell their goods, and complete the day by returning home. To minimize their costs, traveling salespeople strive to visit every city exactly once using the shortest total travel distance. This amounts to finding a visitation order of each city that minimizes the sum of distances traveled when moving from one city to another. Figure 1 illustrates a small TSP and a feasible solution to that problem. The cities are labeled from 0..4.




(a) The TSP input data. (b) A solution to the TSP.

Figure 1: A Traveling Salesman Example

A solution to the traveling salesman problem is an ordering in which to visit all vertices. The cost of a solution is the distances between each consecutive vertex in the tour. (Using Euclidean distance: the distance between two points p and q is $\sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$. This is actually a variant on the "regular" Traveling Salesman problem called the Euclidean Traveling Salesman Problem. It has some nice qualities- like all distances following the Triangle Inequality- that you can take advantage of).

The input consists of |N| + 1 lines. The first line contains one number |N| (the number of vertices). It is followed by |N| lines, each line represents a point ($x_i$, $y_i$), where $x_i$ and $y_i$ are double values.

The output has two lines. The first line contains the length of your tour. If you want to round it to 6 signifigant digits (what C++ does by default when outputting doubles) that's fine.
The next line is order of cities in the tour. You should not output the starting city twice, but you should add the cost of going from the last city back to the start in your tour cost.

(example on back)

Input Example (based on the figure)
5
0 0
0 0.5
0 1
1 1
1 0
Output Example
5.2 0
0 4 1 3 2

This output represents the following tour, 0->4, 4_>1, 1->3, 3->2, 2->0