Programming Design, Spring 2013
Final project

Instructor: Ling-Chieh Kung
Department of Information Management
National Taiwan University

Note. As this is a project instead of homework, the expectation on your performance is higher than that for homework. As you will see, there is no standard answer for the tasks in this project and finding a feasible solution is very easy. So what we really care about is (1) the quality of your suggested solution, (2) whether you write a "good" program, and (3) how you present your ideas and design in your report. Try your best to make your program and report precise, concise, pretty, and easy to understand.

1 Teams, programs, and reports

Please form a group of **two or three** people for this project. One student cannot participate in two teams. Please indicate each team member's name and student ID on your report.

For this project, you need to write a C++ program and upload it to the online grading system PDOGS at http://stella.im.ntu.edu.tw/online-judgement/. You also need to write the documentation of your program as a report. In the report, please describe:

1. How does your algorithm work. Please describe it in words with no code.
2. The design of your algorithm. In particular, why do you design it in that way.
3. The design of your program. Describe the meaning of those classes or functions in words.
4. Anything else you want to say to the instructor.

Your report cannot be longer than five pages. It can be written in English or Chinese. Submit everything by **23:59:59** on **June 24, 2013**.

2 Tasks

In the final project, you will design your own algorithm to solve the well-known "traveling salesperson problem" (TSP). Imagine that you are a salesperson who needs to visits several customers in different places. You need to exist your office, visit each customer once and only once, and then go back to your office. Among all possible routes, you want to find the route whose total distance (or travel time) is the smallest. As an example, suppose your office is at location 1 and four customers locate at locations 2, 3, 4, and 5, respectively. Distances between all pairs of locations are given in the following table

From	To				
	1	2	3	4	5
1	0	3	4	10	3
2	3	0	9	4	8
3	4	9	0	6	2
4	10	4	6	0	8
5	3	8	2	8	0

According to this table, the distance between locations 1 and 2 is 3, that between locations 1 and 3 is 4, etc. A feasible route is $(1, 2, 3, 4, 5)$ with total distance $3+9+6+8+3 = 29$. However, it is not optimal: An alternative route $(1, 2, 4, 3, 5)$ with total distance $3 + 4 + 6 + 2 + 3 = 18$ is a better route.

In the testing data set, an instance of TSP contains a line of positive integers n, d_{12}, d_{13}, d_{14}, ..., $d_{n1,n}$. The first number n is the number of locations. d_{ij} is the distance between location i and j. It is assumed that $d_{ij} = d_{ji}$. As an example, the previous 5-location TSP instance will be presented in the testing data as

```
5 3 4 10 3 9 4 8 6 2 8
```

The only restriction on d_{ij} is that d_{ij}s are all positive integers. In particular, the triangular inequality does not apply. Your should output a sequence of n locations, such as `1 2 4 3 5`, as a route.

TSP has been shown to be NP-hard, which means no one knows how to design an algorithm that guarantees to find the shortest route in a polynomial time for any given instance. Therefore, you should to design your own algorithm that generates a "near-optimal" route for each given TSP instance. The route must be feasible; otherwise, it results in zero point. The shorter the route, the higher the scores. In the testing data, some TSP instances will contain more than 100 locations. Therefore, complete enumeration is impossible.

3 Suggestions on programming

As you have at least two persons in your team, this project requires you to do team work. For two or three of you to write a program together, you need to find a good way to divide the work. This is certainly not easy, but you need to start to try it.

For this project, one possible way of dividing the work is for one to write a class, say `TSP`, such that each object is a TSP instance. She should prepare the class with some basic functions, such as reading one line of input, calculating the total distance of a given route, returning a location satisfying a certain condition, etc. Then the other team member(s) may use this class to implement the algorithm designed by your team. When the algorithm implementer(s) find some operations of the class are missing, she (they) may ask the class developer to add that operation into the class.

In any case, please note that as the number of locations has no upper bound, you cannot use a static array to store distance information. You need a "pointer of an integer pointer", i.e., a variable of the type `int**`. With it you will be able to store distance information into a two-dimensional dynamic array.

4 Grading policy

We will grade your final project from the following four perspectives:

1. Readability of the report (30%): Your report will be graded according to those guidelines mentioned in Section 1 and its format. For some suggestions on formatting your report, please see "PDSp13_formatting.pdf".

2. Quality of the program (20%): Your program will be graded based on its logic, format, and overall quality. Please try to write a robust, consistent, extendable, and easy-to-read program.

3. Quality of the solution (50%): 20 TSP instances will be given to you (in 25 lines). For each TSP instance, PDOGS will process your suggested route to calculate its total distance. Suppose there are m teams generating feasible routes with total distances $D_1 \leq D_2 \leq \cdots \leq D_m$. In this case, team i gets

$$\frac{1}{2} + 2\left(\frac{D_m - D_i}{D_m - D_1}\right)$$

points for this instance. In particular, team 1 gets 2.5 point, team m gets 0.5 point, and teams generating the same distance always get the same point. A team's total points for this part is the sum of its points for all the instances.