Programming Design, Spring 2014
Final Project

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

Teams and tasks

Please form a group of ***at most four*** people for this project. One student cannot participate in two teams. One team can include at most one tutor. Each team needs to do the following tasks:

1. Sign up at http://goo.gl/me9wt8 by indicating your members' names and student IDs and your preferred date for demonstration by ***2:00 pm, May 26, 2014***.
2. Write a program, upload it to PDOGS, complete with other teams, and earn points based on the relative performance. PDOGS will be closed at ***8 am*** on ***June 9, 2014***. You will be provided the ***complete testing data*** when PDOGS opens. Your program must terminate in ***thirty seconds***; otherwise it will be terminated by PDOGS, and you will earn no point.
3. Demonstrate your program and explain your design in class. Each team will have around 15 minutes for the demonstration. For any feature or functionality not required in the problem, you may demonstrate them in class. Demonstrations will be held on ***June 9*** and ***June 16, 2014***.
4. Write a written report (I mean, type it) to summarize all your works. Your report is due at ***8 am*** on ***June 9, 2014***. Please submit a PDF file to PDOGS. In the report, please describe:
   (a) How your algorithms work. Please describe it in words or pseudocodes.
   (b) When your algorithms work well or badly.
   (c) The structure of your program, including the meanings of those classes or functions.
   (d) Anything else you want to say to the instructor.

   Your report cannot be longer than ***five A4 pages***. It can be written in English or Chinese. Please try your best to write a precise, concise, well-organized, and easy-to-read report. Some suggestions for formatting your report are provided on the course website.

Problem description

In this project, your will really complete a system that can help you determine the routes for delivering apples. Of course you want to be as profitable as possible, so you will implement a system with some "smart" algorithm and efficient data structures to complete the task. As your system is to help you make decision, it is called a *decision support system* (DSS). Building decision support systems is exactly what a student majoring in Information Management should be able to do: Using Computer Science and Information Technology to support decision making.[1]

You will be given two kinds of networks: a general network introduced in Homework 7 and a directed star-ring network introduced in Homework 11. For each network, you will be given three store-customer relationships: one store and one customer, one store and multiple customers, and multiple stores and multiple customers. Therefore, in total you will be given six problem instances (see Table 1). Note that for each network structure, the three different relationships will be built on different networks.

For each problem instance, your task is to determine one or multiple routes for delivering apples. Each route must start from a store and then ends at that store. For each route, a deliverer carries several apples to deliver to customers. You need to tell the deliverer how many apples to carry and

[1] DSS is just one kind of *management information systems*. More types of management information systems will be introduced in the course "Management Information Systems".

Network	Store-customer relationship		
	One store One customer	One store Multiple customers	Multiple stores Multiple customers
General	1	2	3
Star-ring	4	5	6

Table 1: Instance numbering

which customers to deliver to. Please note that a customer's demand must be satisfied by only one delivery. That is why you do not need to tell the deliverer the number of apples to deliver to each customer: once you deliver to one customer, you satisfy all its demands. It is quite possible that your delivery plan includes multiple routes. The quality of your delivery plan is determined based on the total distances of all your routes. Of course, you must satisfy all the demands.

Let's use the network in Figure 1 to explain more. In this network, you have two stores at nodes 1 and 3 and three customers at nodes 4, 5, and 6. For this simple network, it is not hard to find a feasible plan. For example, the two colored routes in Figure 2 form a plan with the red route delivering to the customers at nodes 4 and 5 and the blue rout delivering to the customer at node 6 (instead, you may assign customer at node 5 to the store at node 3). The total distance of the two routes is $10 + 9 = 19$.

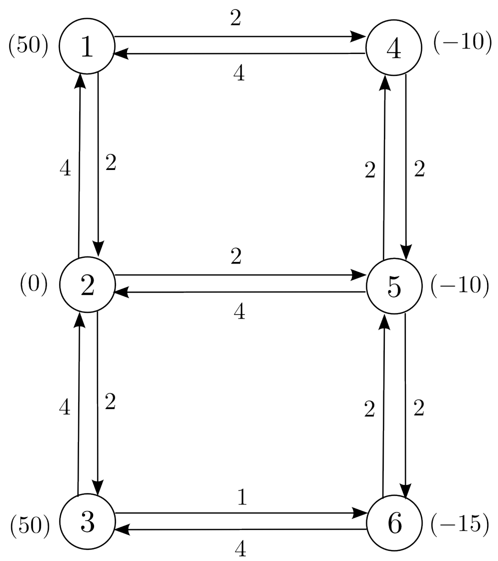

Figure 1: The network

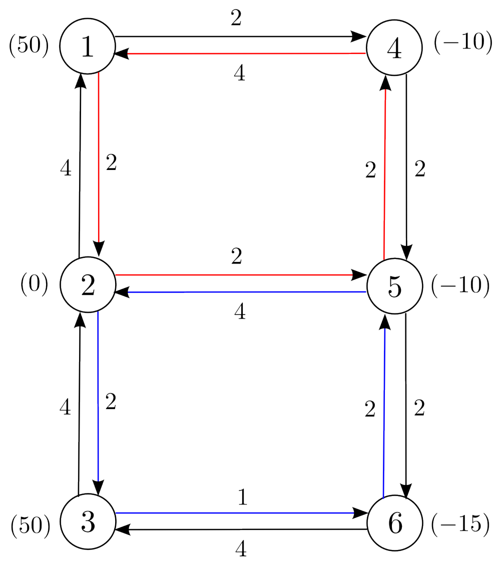

Figure 2: Delivery plan 1

Then immediately you see that this plan can be improved: The blue route can be shortened to the one in Figure 3. The total distance reduces to $10 + 5 = 15$. How about using just one store with the green route in Figure 4? The total distance is 13, which is the smallest among the three plans. However, without a more careful analysis, we cannot say that it is an optimal plan. Moreover, if the supply quantity at each store is 30 rather than 50, the green route becomes infeasible. Finding a feasible plan better than that in Figure 3 thus becomes unsolved.

Given a network with stores, store supply quantities, customers, and customer demand quantities, your are going to try your best to find a good delivery plan. You do not need to find an optimal plan; to get full credits, all you need to do is to find a plan that is better than all your classmates.

Input/output formats

The input format for this problem is based on that for Homework 7 and 11. For each problem instance, you will be given `AN` and `AA` tasks in the input file to construct the network. If a node has a positive

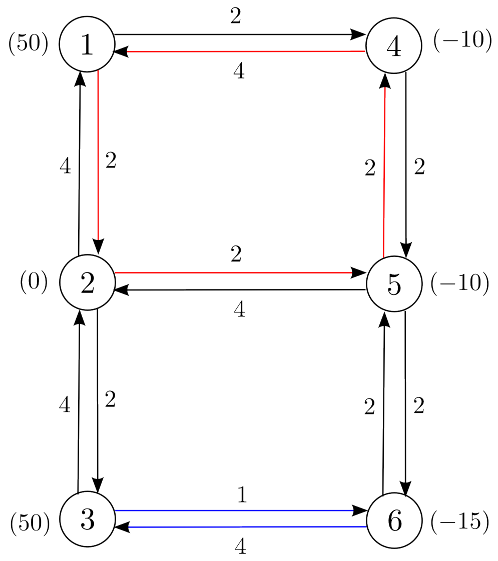

Figure 3: Delivery plan 2

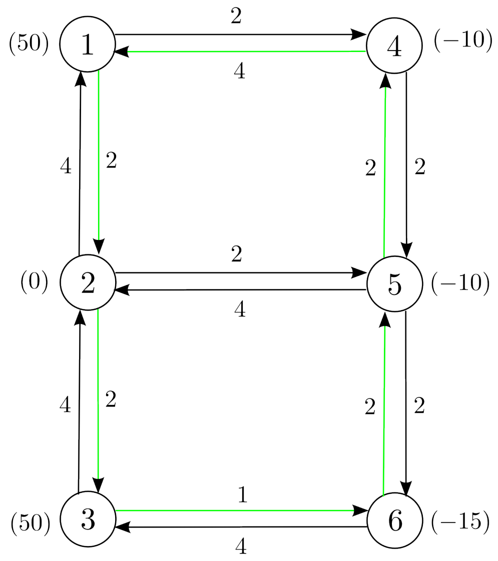

Figure 4: Delivery plan 3

weight, there is a store at it and the weight is the supply quantity. Similarly, if a node has a negative weight, there is a customer at it and the weight is the demand quantity. All node IDs are positive integers. For a star-ring network, the ID of the central node is 1.

In total you will be given six sets of `AN` and `AA` lines. Between each set, there is a line of `***`. The ith set creates the ith instance as labeled in Table 1. Therefore, you may design six different algorithms, one for each scenario, if you want.

For each instance, you need to output your delivery plan in a line. In this line, you may give one or multiple routes. For each route, list the node IDs of the nodes on the route in the order that they are passed. The first and last node IDs must be identical as the ID of a node with a store. After each route, list the node IDs for those nodes at which the customer gets delivery through this route. If there are multiple customers served by a route, list the corresponding node IDs from small to large. Between the node IDs for the route and the node IDs for customers, add a semicolon; between two routes, add a # symbol. Separate any two letters or symbols you output by a white space.

As an example, if we want to output the delivery plan in Figure 2, we should output

```
1 2 5 4 1 ; 4 5 # 3 6 5 2 3 ; 6
```

with a new line character appended at the end. Before #, we can see that the customers at nodes 4 and 5 are served by the red route $(1, 2, 5, 4, 1)$. Similarly, based on the output after #, the customer at node 6 is served by the blue route $(3, 6, 5, 2, 3)$. There is no restriction on the order for outputting routes. For example, output the blue route before the red route is completely fine.

The grading system will check feasibility for your output. For each route, it will check whether the starting node has a store, whether the route really exist, whether those customers are really on the route, and whether the store supply quantity is sufficient. If multiple routes originate from a single store, the system will check whether the store supply quantity is sufficient for all customers assigned to those routes. Finally, it will check whether all customers demands are fulfilled. The total distance of your delivery plan is meaningful (and will be graded) if and only if the plan is feasible.

As another example, suppose we want to output the delivery plan in Figure 4, the output should be

```
1 2 3 6 5 4 1 ; 4 5 6
```

with a new line character appended at the end. For this instance, it is also fine to output the route starting at node 3.

Grading criteria

- PDOGS (60%):

  For each instance, your feasible plan will give you a total distance z. Among all the feasible plans generated by all the teams, let $z_{\max}$ and $z_{\min}$ be the largest and smallest total distances, respectively. Then your grades for this instance is

  $$4 + 6\left(\frac{z_{\max} - z}{z_{\max} - z_{\min}}\right).$$

  Your total grades in this part is the sum of the six grades.

- In-class demonstrations (20%):

  - Designing a good algorithm and implementing a good program are both very good. However, to effectively deliver your ideas to other people is also important. Because of that, we will also judge how clearly you introduce your design to the class.
  - During the demonstration, you are allowed to input the testing data on-site and display the output to the class. Of course we do not want to see numbers and symbols put in a line. Therefore, you may prepare another program that visualize your output in an easier-to-understand way. There is no restriction on this task and you may do anything you want (or simply skip it). More points will be given to you if you can display your delivery plans in a more comprehensive way.

- Report (20%):

  Writing a clear report that explains everything in details is also challenging. Therefore, a team should be rewarded if their report is good. Please prepare your report that conveys your design and the program structure to the readers. The more easily one may understand your program after reading your report, the higher grades you earn. Finally, if you get anything from doing this project, you may leave your thoughts in the report. All feelings, comments, suggestions, or criticisms are welcome.