CS2010 PS3 - Hospital Tour v2 (with R-option)

Released: Thursday, 13 September 2012
Due: Tuesday, 24 September 2012, 8am (extension due to Quiz 1)

Collaboration Policy. You are encouraged to work with other students or teaching staffs (inside or outside this module) on solving this problem set. However, you **must** write the Java code **by yourself**. In addition, when you write your Java code, you **must** list the names of every collaborator, that is, every other person that you talked to about the problem (even if you only discussed it briefly). This list may include certain posts from fellow students in CS2010 IVLE discussion forum. Any deviation from this policy will be considered cheating, and will be punished severely, including referral to the NUS Board of Discipline. It is not worth it to cheat just to get 15% when you will lose out in the other 85%.

R-option. This PS has R-option at the back. This additional task usually requires understanding of data structure or algorithm *beyond* CS2010. A self research on those relevant additional topics will be needed but some pointers will be given. CS2010R students *have to* attempt this R-option. CS2010 students can choose to attempt this R-option too for higher points, or simply leave it.

Last Year's Story. In PS2, Steven already mentioned about "attending birth classes" last June 2011 to prepare for the arrival of his baby. One of the event conducted during one of the birth classes is the "hospital tour" where several parents-to-be[1] are gathered together and shown various facilities to help birth in that hospital. The rooms shown are typically the:

- *First class* single room for the new mother and her family to stay for few nights after delivery,
- Delivery suites (already mentioned in PS2),
- Nursery (a room full of cute newborn babies where the babies are monitored, fed using formula milk if the baby's mother choose not to breastfeed the baby, bathed, etc),
- Neo-natal Intensive Care Unit (a room that you hope your baby will *never* use),
- Etc

Steven observed that during the tour that he and Grace joined, we were only shown "the better rooms" of the hospital although the hospital itself is much bigger than that. For example, the double and four-bedded rooms, the surgery room (for Caesarean section), the cashier room (where you have to fork out thousand of dollars to pay your bills – especially if you are not a Singapore Citizen → no baby bonus), birth registration room, etc, are not shown for 'logical reasons'. "This is interesting.", Steven thought. As a Computer Scientist, his mind started wandering on what happened behind the scenes so that his hospital tour was impressive...

[1]99% of the attendees are young couples expecting their first baby – including Steven and his wife Grace. Couples do not normally go to birth classes again for their second (or third, etc) baby.

The Actual Problem. This is what happened behind the scenes, many years before Steven visited that hospital...

Assume that you are the manager of your hospital. You know the layout of your hospital (a connected weighted graph) and you have given rating score to each room (the weight of each vertex of your graph). You want to decorate certain room(s) in your hospital so that visitors will feel better when visiting your hospital. As your budget is limited, you want to decorate only the *important* room(s) with the lowest rating score. You define important room as a room that links different buildings in your hospital such that if that room (to be precise, the corridor beside that room) is blocked, the buildings in the hospital becomes 'disconnected'. You want to know the lowest rating score of an important room in your hospital.

For example, suppose your hospital is a connected weighted graph as shown below:

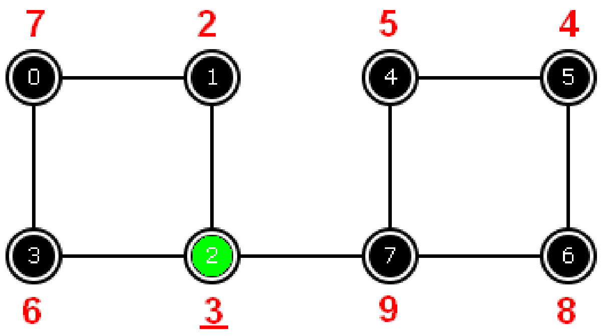

Figure 1: A Sample Hospital Layout

There are two major buildings (`0,1,2,3` and `4,5,6,7`) linked by a corridor `2-7` besides the two important rooms (`2` and `7`). For example, if room `2` is blocked, then people currently in room `0`, `1`, or `3` will not be able to visit people in rooms `4`, `5`, `6`, or `7`. Or in another word, the hospital becomes 'disconnected'. Similar situation if room `7` is blocked. The other rooms are not important rooms. For example, if room `1` is blocked, people from room `0` can still go to any other rooms in the hospital (other than room `1` of course) via path `0-3-2` and so on.

Now, among the two important rooms `2` versus `7`, room `2` has lower rating score (3 points) than room `7` (9 points). So, you will decorate room `2`. You will report `3` as the answer of your own query: "What is the lowest rating score of an important room in your hospital?".

The skeleton program `HospitalTour.java` is already written for you, you just need to implement one (or more) method(s)/function(s):

- `int Query()`
  Query your Adjacency Matrix data structure (already implemented in `HospitalTour.java` and returns the lowest weight (lowest rating score) of an important vertex (important room) of your graph (your hospital) (these rating scores are stored in another Vector of Integers). If your hospital has no important room. You will not decorate any room, i.e. you just return -1. Note that the answer for this query is still unique even though there can be more than one important rooms with similar lowest rating score.
- If needed, you can write additional helper methods/functions to simplify your code.

Subtask 1 (50 points). The given hospital layout map is a small weighted tree ($1 \leq V \leq 50$).

Example:

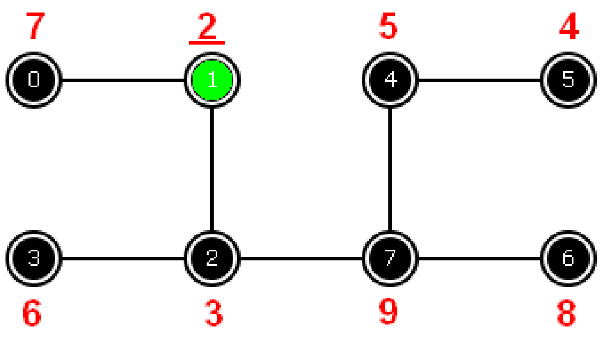

Figure 2: A Simplified Hospital Layout (Tree)

There are four important rooms in this hospital: `1`, `2`, `4`, `7`. Among these four, important room `1` has the lowest rating score: 2. Therefore, the answer for your own query is 2.

Subtask 2 (Additional 25 points). The given hospital layout map is a small, connected, and weighted general graph ($1 \leq V \leq 50$). You can use `Sample.txt` to check if your program can pass Subtask 2. PS: We have more rigorous test cases to judge your final submission.

Subtask 3 (Additional 25 points). Same as Subtask 2, but the graph is larger ($1 \leq V \leq 1000$).
PS 1: We have even more rigorous test cases than Subtask 2 to judge your final submission.
PS 2: For this Subtask 3, efficiency matters. The expected solution is $O(V^2 + VE)$ per test case.
PS 3: Our test data has $E = O(V)$, i.e. $E = c \times V$ for a small constant factor c.

Note: The test data to reach 50 points: `Subtask1.txt` as well as `Sample.txt` are given to you. You are allowed to check your program's output with your friend's. You are encouraged to generate and post additional test data in IVLE discussion forum.

R-option (Additional 10 bonus points). Your program must be able to solve Subtask 3 and also must be *extremely efficient* as you will be given a *large*, connected, and weighted general graph ($1 \leq V + E \leq 100000$). Hint: This requires a special algorithm beyond CS2010. If you can do this, you will get 10 bonus points so your total points can go up to 110. Beware that this part can take hours to complete... CS2010R students have to do this though :).

PS 1: A code that solves this R-option should be able to solve **Subtask 1, 2, and 3** easily.
PS 2: The expected time solution is $O(V + E)$ per test case.
PS 3: If you encounter stack overflow, try running your Java program with: '-Xss8m' flag.