Computer Science CSC263H
St. George Campus

March 15, 2018
University of Toronto

Homework Assignment #6

Due: March 29, 2018, by 5:30 pm

- You must submit your assignment as a PDF file of a typed (**not** handwritten) document through the MarkUs system by logging in with your CDF account at:

  `https://markus.teach.cs.toronto.edu/csc263-2018-01`

  To work with one or two partners, you and your partner(s) must form a group on MarkUs.

- The PDF file that you submit must be clearly legible. To this end, we encourage you to learn and use the LaTeX typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You can use other typesetting systems if you prefer, but handwritten documents are not accepted.

- If this assignment is submitted by a group of two or three students, the PDF file that you submit should contain for each assignment question:

  1. The name of the student who *wrote* the solution to this question, and
  2. The name(s) of the student(s) who *read* this solution to verify its clarity and correctness.

- By virtue of submitting this assignment you (and your partners, if you have any) acknowledge that you are aware of the homework collaboration policy that is stated in the csc263 course web page: `http://www.cs.toronto.edu/~sam/teaching/263/#HomeworkCollaboration`.

- For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class, or is in the assigned sections of the official course textbook, by referring to it.

- Unless we explicitly state otherwise, you should justify your answers. Your paper will be marked based on the correctness and completeness of your answers, and the clarity, precision, and conciseness of your presentation.

- Your submission should be no more than 3.5 pages long in a 10pt font.

Question 1. (1 marks) Consider the *directed* graph G with nodes $\{1, 2, 3, 4, 5, 6, 7\}$ that is represented by its adjacency lists:

$A(1) = 6, 7, 4$
$A(2) = 6$
$A(3) = 1, 4$
$A(4) = 6$
$A(5) = 2, 6$
$A(6) = \text{NIL}$
$A(7) = 5, 2, 4$

a. Draw a Depth-First Search forest generated by the DFS of G under the following assumptions: *the DFS starts at node 1 and it explores the edges in the order of appearance in the above adjacency lists, and all the vertices are explored.* Do not draw forward, back, or cross edges. Show the discovery and finishing times $d[u]$ and $f[u]$ of every node u of G, as computed by this DFS of G.

b. How many back edges, forward edges, and cross-edges are found by the above DFS?

c. Suppose the graph G above represents seven courses and their prerequisites. For example, $A(1) = 6, 7, 4$ means that course 1 is a prerequisite for (i.e., it must be taken before) courses 6, 7 and 4; and $A(6) = \text{NIL}$ means that course 6 is not a prerequisite for any course.

Using Part (b) above and a theorem that we learned in class, prove that it is possible to take all the courses in a sequential order that satisfies all the prerequisite requirements. State the theorem that you use in your argument; do ***not*** give a specific course order here.

d. Now list the courses in an order they can be taken without violating any prerequisite. To do so you ***must*** use your DFS of Part (a) and an algorithm that we covered in a tutorial.

e. Draw a Breadth-First Search tree of G that ***starts at node 3*** and explores the edges in the order of appearance in the above adjacency lists.

Question 2. (1 marks) Suppose we have a road network between cities numbered from 1 to n. Each road connects a pair of cities, and can be traveled in both directions. The entire road network is represented by an undirected graph $G = (V, E)$, where $V = \{1, \ldots, n\}$ are the cities, and the edges E are the roads. Assume that the road network, i.e. the graph G, is connected. There are k roads which have been damaged, and your goal is to choose which roads to repair, so that the there is a path between each pair of cities, and the total cost of repairs is minimized.

The roads E are given to you in an array $R[1\,..\,m]$, where each array cell $R[i]$ contains two fields: $R[i].\mathit{edge}$ which contains the pair (u, v) of cities that the i-th road connects, and $R[i].\mathit{cost}$ which contains the cost of repairing the road, if it is damaged, or 0 if it is not. The cost of repairing a damaged road can be assumed to be positive. Design an algorithm to compute a set of damaged roads to be repaired, so that, after the repairs, it is possible to travel between every pair of cities by following roads that are either undamaged or repaired. Moreover, the *total cost* of repaired roads should be minimized. Your algorithm should have worst-case running time at most $O(m \log^* m + k \log k)$. Describe your algorithm in clear and concise English, prove it is correct, and analyze its running time.