CSE 101 Homework 1
Winter 2015

This homework is due Friday January 16th at the start of class. Remember to justify your work even if the problem does not explicitly say so. Writing your solutions in \LaTeX is recommended though not required.

**Question 1** (Moving, 20 points). Alice is attempting to move a table from one side of her apartment to the other. The apartment is laid out on a hexagonal grid. The table is large enough that it takes up two adjacent hexagons at any given time. Alice can move the table by leaving one end fixed and rotating the other end either clockwise or counterclockwise one unit as shown below: Unfortunately, Alice’s apartment has several immovable obstacles, each taking up a full hexagon so that the table cannot be moved to overlap any obstacle. Prove a reasonable algorithm that given the layout of Alice’s apartment, the location of obstacles and the desired starting and ending configurations of the table, determines whether or not it is possible for Alice to move the table from the starting configuration to the ending one.

**Question 2** (DAG Detection, 20 points). Give a linear time algorithm that given a directed graph determines whether or not it is a DAG.

**Question 3** (Longest Path, 20 points). Find an efficient algorithm that given a DAG, \(G\), finds the length of the longest path in \(G\). Hint: topologically sort and compute the lengths of the longest path starting from \(v\) for each \(v\) in some order.

**Question 4** (Cheapest Reachability, 20 points). Bob is planning a trip to Digraphia. Digraphia has many cities, but for unfathomable reasons travel between them is restricted, with certain pairs of cities connected by one-way roads. Bob has determined the cheapest available flights into and out of each city, and the available travel routes. Provide an efficient algorithm to determine the pair of cities \(s\) and \(t\) so that \(t\) is reachable from \(s\) and so that the cost of flying into \(s\) plus the cost of flying out of \(t\) is minimized. If there are \(n\) cities and \(m\) accessible one-way roads, you should aim for a runtime of \(O(n \log(n) + m)\) or better. Hint: Try running depth first search exploring vertices in increasing order of entry cost.

Can you do better if the roads are all bi-directional?

**Question 5** (Pre- and Post- Orderings, 20 points). For each of the following tables either demonstrate a graph so that the given values could be the pre- and post- numbers for the vertices of that graph under a depth first search, or show that no such graph exists.
<table>
<thead>
<tr>
<th>Vertex</th>
<th>Pre</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

(a) $\begin{array}{|c|c|c|}
\hline
\text{Vertex} & \text{Pre} & \text{Post} \\
\hline
A & 1 & 8 \\
B & 2 & 5 \\
C & 3 & 4 \\
D & 6 & 7 \\
E & 9 & 10 \\
\hline
\end{array}$

(b) $\begin{array}{|c|c|c|}
\hline
\text{Vertex} & \text{Pre} & \text{Post} \\
\hline
A & 1 & 10 \\
B & 2 & 3 \\
C & 4 & 8 \\
D & 5 & 7 \\
\hline
\end{array}$

(c) $\begin{array}{|c|c|c|}
\hline
\text{Vertex} & \text{Pre} & \text{Post} \\
\hline
A & 1 & 10 \\
B & 2 & 3 \\
C & 4 & 8 \\
D & 5 & 7 \\
\hline
\end{array}$

**Question 6** (Extra credit, 1 point). *Approximately how much time did you spend working on this homework?*