Graph Algorithms

Robin Whitty at www.theoremoftheday.org

Question set for Chapter 1

1. A graph is called **3-regular** (or ‘cubic’) if every vertex has degree 3. Prove that a 3-regular graph has an even number of vertices.

   [Hint: use the Handshaking Lemma]

2. In Chapter 1, Section 3, the following pseudocode was given to calculate the value of d(v) for a vertex v:

   d(v) := 0
   for $e \in E$ **do**
       if $v \in e$ **then** d(v) := d(v) + 1

   Amend this pseudocode to make it correct when the input graph can include self-loops.

3. Prove the Handshaking Lemma by induction on the number of vertices.

4. The following pseudocode specifies an algorithm which checks if a graph G has an isolated vertex (that is, having zero degree).

   Algorithm **SeekIsolated**
   Input: graph $G = (V, E)$
   Output: True if there is a vertex $v \in V$ for which d(v) = 0; False otherwise
       foundzero:=False   # initialise to a False answer, then look for evidence to the contrary
   1.   **for** $v \in V$ **do**
           foundincidence:=False # if this stays False in the following loop then
                                   # we’ve found a zero degree
   2.   **for** $e \in E$ **do**
   3.   **if** $v \in e$ **then** foundincidence:=True
           if not foundincidence **then** foundzero:=True
       return(foundzero)
   end

   Given an analysis of the complexity of this algorithm.

5. (a) In Chapter 1, Section 3 the following Lemma was proved:

   Lemma If G is a simple graph on two or more vertices then G has two vertices with equal degree.

   Is this Lemma true for non-simple graphs? What about if self-loops are allowed but not multiple edges? What if multiple edges are allowed but not self-loops? Give counter examples as appropriate.

   (b) We could speed up the algorithm Stupid, given Chapter 1, Section 3, by only checking all $\binom{|V|}{2}$ pairs of vertices. The **for** loop at 1. in the algorithm specification would still check every vertex but the **for** loop at 2. would only check vertices *further down the list*. What is the complexity of this new version of Stupid? Is it different from the original version when we use the big-Oh notation?