Data Structures and Algorithms
Week 5 problem sheet

## A. Hash tables

1. What is a perfect hash function? Explain why it is almost always impossible to have a perfect hash function.

## B. Graph algorithms

Use the example weighted, undirected graph $G$ below to answer the following questions:



1. Suppose we start from vertex $b$ – in what order are the vertices of graph $G$ visited if we do a depth-first search (DFS)?
2. Suppose we start from vertex $b$ – in what order are the vertices of graph $G$ visited if we do a breadth-first search (BFS)?
3. Suppose we wish to find the shortest path from vertex $a$ to vertex $g$ in graph $G$. If we ignore the edge weights, what would your answers be (hint: there is more than one possible answer)?
4. Suppose we wish to find the shortest path from vertex $a$ to vertex $g$ in graph $G$. If we calculate a shortest path *using* the edge weights, what would your answer be?
5. On paper, show the steps followed by Prim’s method for calculating a **minimum spanning tree** for the graph shown below. Start your tree from vertex $v\_{0}$. You could write some code using PrimMST.java to check your answer.
* 
1. PrimMST can be implemented using either an **adjacency matrix** or an **adjacency list** to represent a weighted graph. What difference does the choice of implementation make to the big O complexity of the algorithm? Why?
2. Test your answer to the previous question by implementing an adjacency list version of the algorithm. You can use PrimMST.java as the basis of your second solution. Run some experiments to test the run times of these two algorithms. What do you notice?