

Computer Science COMP1140 in 2009 – Assignment one

Due: 5pm Friday, Sep. 25
Late Penalty: 25% per day

No programming is needed for this assignment. Drop your assignment into the COMP1140 box on the ground floor of the CSIT Building. The total marks of this assignment is 100 points. Don't forget to add your name. Neat handwriting is acceptable.

Question 1 (10/100).

A sorting algorithm is *stable* if elements with equal keys are left in the same order as they occur in the input sequence, which of the following sorting algorithms are stable and which are not? Why? Justify your claims.

1. Shell sort
2. Quick sort
3. Merge sort
4. Bubble sort

Question 2 (15/100).

Given an integer sequence 15, 21, 6, 45, 17, 7, 52, 4, 78, 5, 9, 23, 11, which is stored by an array $A[1..13]$.

- (a) Construct a **min-heap** for the elements in the sequence (5 points).
- (b) If the element of value 7 changes to 10, the min-heap must be adjusted to become a min-heap again. Give the basic steps of the adjustment (5 points).
- (c) Prove that the time for building a min-heap of n elements is $O(n)$ (5 points).

Question 3 (15/100)

A *bipartite* graph is a graph whose vertices can be divided into two disjoint sets V_1 and V_2 such that every edge connects a vertex in V_1 and one in V_2 ; that is, there is no edge between two vertices in the same set.

Give an efficient algorithm using BFS to determine if an undirected graph is bipartite.

Question 4 (10/100).

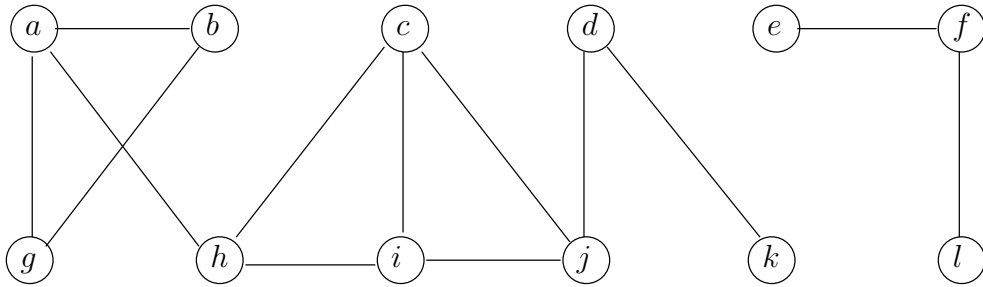


Figure 1: G for Questions 4 and 6

Use the `find_tree` and `merge_tree` procedures to find the connected components of the graph G in Figure 1. Initially each vertex of G forms a tree. You will add the edges in the following prescribed order:

(a, b) , (c, h) , (d, k) , (e, f) , (f, l) , (a, h) , (i, j) , (d, j) , (h, i) , (c, i) , (c, j) , (a, g) , (g, b) .

Show the trees as they appear after each edge addition. Whenever there is a choice to be made when forming a new tree from two trees of equal height, use lexicographic order.

Question 5 (10/100).

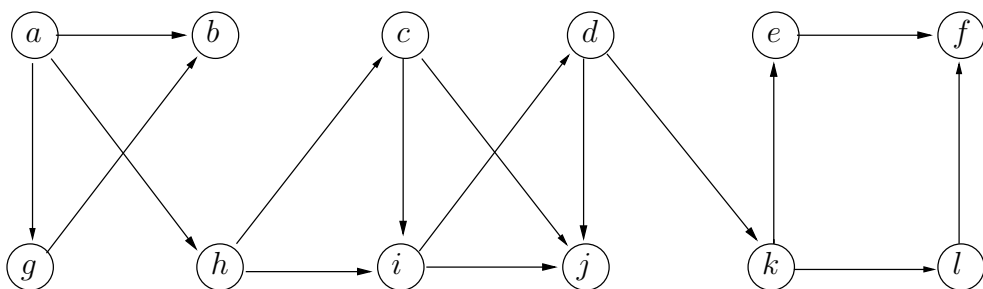


Figure 2: D for Question 5

Apply DFS to D Figure 2 starting at vertex a .

1. Show the DFS tree.
2. List the vertices in topological order, and justify your approach.

Question 6 (15/100).

Apply BFS to G in Figure 1 starting at vertex a .

1. Show the BFS tree.
2. For each vertex u , show the distance of u from a .
3. A DFS tree classifies the edges of a simple graph into tree, back, forward, and cross edges. A BFS tree can also be used to classify the edges into the same four categories. Prove that in BFS of an undirected graph, there are no back edges and no forward edges.
4. Classify the edges of G according to BFS.

Question 7 (15/100).

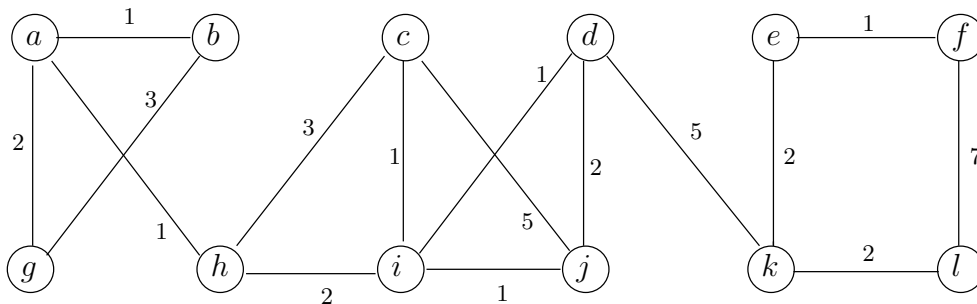


Figure 3: G for Question 7

1. Show the minimum spanning tree found by Prim's algorithm applied to the weighted graph G in Figure 3. Indicate the order in which the vertices and edges have been added to the tree; and at each step, show the key values for every vertex in G . Initialise the algorithm at vertex a .
2. Run Dijkstra's algorithm on the same graph G in Figure 3, starting at vertex c . Indicate the order in which the vertices are inserted into S ; at each step, show the key values for every vertex in G . Finally, show the shortest distance from c for every vertex in G .
3. Explain exactly how Prim's and Dijkstra's algorithm differ.

Question 8 (10/100).

1. Prove that a tree of size n has $n - 1$ edges.
2. In BFS as given on slide 29 of the lecture notes: Under what conditions will all vertices have been visited at the end of the algorithm?

For those of you keen for a challenge or extra marks . . .

Question 9 (10 marks)

Consider the following alternative algorithm that finds a minimum spanning tree in an undirected and weighted graph G :

While there exists a cycle C in G , remove the edge of largest weight in C .

Prove that the output of the algorithm is a minimum spanning tree for G .

Hint: Use induction with induction hypothesis: At each step of the algorithm, the edges of G contain a minimum spanning tree of G .

Note: With respect to all the graph theory questions, whenever there is a choice to be made between vertices, always use lexicographic order.