THE AUSTRALIAN NATIONAL UNIVERSITY

Second Semester 2015

COMP3600/COMP6466
(Algorithms)

Writing Period: 2.5 hours duration

Study Period: 15 minutes duration

Permitted Materials: None

Answer ALL Questions

All your answers must be written in the boxes provided in this booklet. You may be provided with scrap paper for working, but it must **not** *be used to write final answers.*

There is additional space at the end of the booklet in case the boxes provided are insufficient. Label such overflow boxes with the question number. Note that the full mark is 100.

Do not remove this booklet from the examination room.

Student Number:

Official use only:

Q1 (30)	Q2 (20)	Q3 (25)	Q4 (25)	Total (100)

QUESTION 1 [30 marks]

(a) [2+2+2=6 marks] In open addressing hashing, three probing techniques are introduced.

(i) List the names of these three probing techniques.

QUESTION 1(a)(i)	**[2 marks]**

(ii) Which probing techniques among the three listed probing techniques in Part (i) cause the primary and secondary clustering problems?

QUESTION 1(a)(ii)	**[2 marks]**

(iii) Design a hash function for the open addressing schema such that no primary and secondary clustering problems occur. In addition, the number of probing sequences derived from this function should be $O(m^3)$, assuming that there are m slots in the hash table.

QUESTION 1(a)(iii)	**[2 marks]**

(b) [4+1=5 marks]

(i) Draw the final binary search tree created by inserting elements 15, 17, 10, 3, 12, 5, 12, 7, 2 one by one in this order into a binary search tree that is initially empty.

QUESTION 1(b)(i) **[4 marks]**

(ii) List the key sequence obtained by in-order traversal on the binary search tree in Part (i).

QUESTION 1(b)(ii) **[1 mark]**

(c) The post-order and in-order traversal node key sequences of a binary search tree T are 19, 22, 21, 32, 29, 35, 25; and 19, 21, 22, 25, 29, 32, 35, respectively. Reconstruct the binary search tree T, give your answer as a diagram.

QUESTION 1(c)	**[4 marks]**

(d) [4+2=6 marks] Consider an array whose elements are 2, 11, 5, 17, 3, 15, 10 in that order.

(i) Draw the final MIN-HEAP after heapifying the array.

QUESTION 1(d)(i)	**[4 marks]**

(ii) If the element with key value 11 in the MIN-HEAP of Part (i) increases its key value to 13, draw the resulting MIN-HEAP after performing this updating operation.

QUESTION 1(d)(ii)	**[2 marks]**

(e) Assume that T is a red black tree that contains N internal nodes, show the height of T is no greater than $2\log(N+1)$. *Hint: use mathematical induction on the black height of each node in the tree.*

QUESTION 1(e)	**[3 marks]**

(f) [3+3=6 marks] To implement operations like `make_set, union, and find_set` on disjoint sets, two data structures: *linked lists and directed forests*, have been introduced for representing the disjoint sets.

(i) In the linked list representation, let $x_1, x_2, \ldots, x_n$ be n objects, assume that there are n `make_set`(x_i) operations with $1 \leq i \leq n$ and then $n-1$ `union`(x_i, x_{i+1}) operations with $1 \leq i < n$. Show that the total time complexity of implementing these operations is $O(n \log n)$ if the weighted-union heuristic is applied.

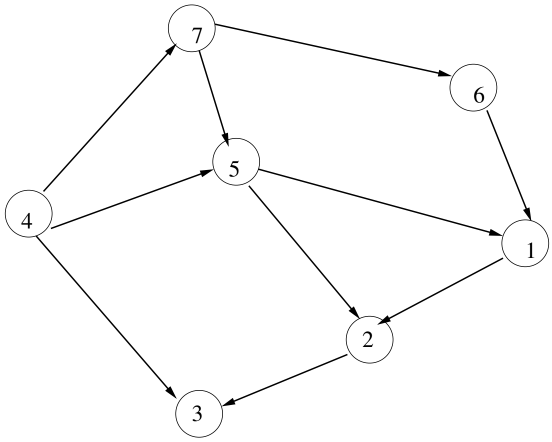
QUESTION 1(f)(i)	**[3 marks]**

(ii) Make use of the directed forest data structure to represent the disjoint sets, and use the two introduced heuristics: path compression and union by rank to find all connected components in a graph with 9 vertices, with edges 1-4, 2-5, 1-8, 5-7, 4-9 provided in this order. Use diagrams to illustrate the forest of directed trees and the ranks of tree roots at each major step.

QUESTION 1(f)(ii) **[3 marks]**

QUESTION 2 [20 marks]

(a) [3+5+2+2=12 marks] Perform a Depth-First Search on the following directed acyclic graph (DAG), starting at node 1 and exploring neighbouring nodes **in order of their labels**.



Give your answer as follows.

(i) The visiting order of the nodes:

QUESTION 2(a)(i)	**[3 marks]**

(ii) The discovery and finish times $d(v)$ and $f(v)$ of each node $v \in V$.

QUESTION 2(a)(ii)	**[5 marks]**

(iii) Classify the edges in the graph into tree edges, forward edges, back edges, and cross edges.

QUESTION 2(a)(iii)	**[2 marks]**

(iv) List the nodes in increasing order of their topological ranking.

QUESTION 2(a)(iv)	**[2 marks]**

(b) [3+2=5 marks] Let $G = (V, E)$ be a connected, simple, undirected graph with vertex set $V = \{1, 2, \ldots, 7\}$. A breadth-first tree T of G with root at vertex 1 is given by the following adjacency matrix A_T.

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

(i) Draw a diagram of T.

QUESTION 2(b)(i) **[3 marks]**

(ii) What is the maximum possible number of edges in G? Justify your answer.

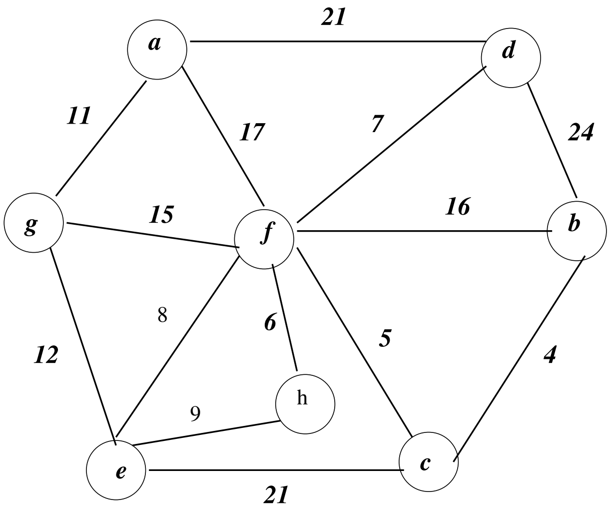
QUESTION 2(b)(ii) **[2 marks]**

(c) Let $G = (V, E)$ be an edge-weighted, connected, undirected graph with $|V|$ being odd. Suppose that the weight of each edge in E is an odd positive integer. Prove that the cost of every spanning tree in G is even, where the cost of a tree is the sum of weights of the edges in the tree.

QUESTION 2(c) **[3 marks]**

QUESTION 3 [25 marks]

(a) [2+15=17 marks] Given the graph $G = (V, E)$ below, there are two well-known algorithms for finding minimum spanning trees (MSTs) introduced in the course.



(i) What are the names of these two well-known algorithms? Nominate one of them for finding an MST in G.

QUESTION 3(a)(i)	**[2 marks]**

(ii) Show the order in which the tree edges are selected and the final tree.

QUESTION 3(a)(ii)	**[15 marks]**

QUESTION 3(a)(ii) **[15 marks]**

(b) Let $G = (V, E)$ be a connected, undirected graph and $e \in E$ an edge in a cycle of G with the maximum weight. Prove that if all the edge weights of G are distinct, then edge e will not be contained in any minimum spanning tree in G.

QUESTION 3(b)	**[4 marks]**

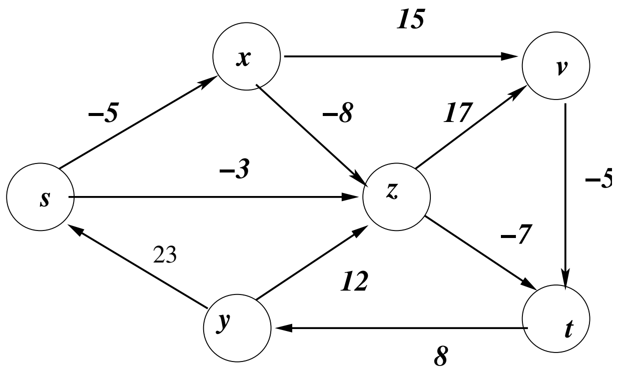
(c) Show with a counterexample that the following claim is not true:

Let C be a cycle in an edge-weighted, connected, undirected graph G, and let e be an edge of C whose weight is strictly less than the weight of every other edge of C. Then, G has a minimum spanning tree which includes e.

QUESTION 3(c)	**[4 marks]**

QUESTION 4 [25 marks]

(a) [3+12=15 marks] Two single-source shortest path algorithms, *the Dijkstra algorithm and the Bellman-Ford algorithm*, have been introduced in the course. In the following graph $G(V, E)$,



(i) To find a single-source shortest path from s to every other vertex in G, which of the two mentioned algorithms should be applied, and why?

QUESTION 4(a)(i)	**[3 marks]**

(ii) Apply the algorithm you selected in Part **(a)** (i). Give the major intermediate results (i.e., the result after each iteration that clearly indicates the d and π values of each vertex) and the final result.

QUESTION 4(a)(ii)	**[12 marks]**

QUESTION 4(a)(ii) **[12 marks]**

(b) This question is 10 marks for COMP3600 students and 6 marks for COMP6466/Honours.

A $d-$dimensional box with dimensions $(x_1, x_2, \ldots, x_d)$ *fits inside* another box with dimensions $(y_1, y_2, \ldots, y_d)$ if there exists a permutation π on $\{1, 2, \ldots, d\}$ such that $x_{\pi(1)} < y_1, x_{\pi(2)} < y_2, \ldots, x_{\pi(d)} < y_d$. (e.g. $< 2, 4, 1, 3, 5 >$ is a permutation of $< 1, 2, 3, 4, 5 >$).

(i) Describe an efficient method to determine whether or not one d-dimensional box fits inside another. (3 marks for COMP3600 and 1 mark for COMP6466/Honours)

QUESTION 4(b)(i)	**[3 marks]**

(ii) Suppose that you are given a set of n $d-$dimensional boxes $\{B_1, B_2, \ldots, B_n\}$. Describe an efficient algorithm to determine the longest sequence $\langle B_{i_1}, B_{i_2}, \ldots, B_{i_k}\rangle$ of boxes such that B_{i_j} fits inside $B_{i_{j+1}}$ for $j = 1, 2, \ldots, k-1$ and $1 \leq i_j \leq n$. Express the running time of your algorithm in terms of n and d. (7 marks for COMP3600 and 5 marks for COMP6466/Honours)

QUESTION 4(b)(ii) (more room on next page) **[7 marks]**

QUESTION 4(b)(ii) **[7 marks]**

(c) [4 marks] **This question is only for COMP6466/Honours students.**

Given a directed weighted graph $G = (V, E)$ with positive weights in all edges, assume that the shortest path between any two vertices in V contains no more than k edges. Devise an $O(k|E|)$ time algorithm to find a shortest path between a given pair of vertices $u \in V$ and $v \in V$.

QUESTION 4(c) (more room on next page)	**[4 marks]**

QUESTION 4(c) **[4 marks]**

Additional answers. Clearly indicate the corresponding question and part.

Additional answers. Clearly indicate the corresponding question and part.

Additional answers. Clearly indicate the corresponding question and part.