Data Structures

Final exam, Fall 2004

作答在答案卷上 題目卷不必繳回

1. (18 pts) True or False? (No explanations needed). Score= max{0, right – 1/2 wrong};
(答對得一分; 答錯倒扣 0.5 分)

1. The worst-case running time of Quicksort is O(n log n).
2. To obtain good performance in open addressing, we should choose the table size so that the load factor is close to 1.
3. Dijkstra's Algorithm finds a shortest path even when there are edges with negative weights but no negative cycles.
4. Prim's algorithm works even when there is a negative edge.
5. Finding an arbitrary element in a treap with *n* elements takes O(log n) time in the worst case.
6. Given a connected undirected graph G, if the weights of all edges are distinct (i.e., different), there is at most one minimum spanning tree.
7. Given a connected undirected graph G, if the weights of all edges are distinct, there is exactly one shortest path from one node to another node.
8. The worst-case complexity of inserting one element into a skip list is O(log n).
9. Treap is a data structure for supporting the priority queue ADT.
10. Inserting an element into a pairing heap can be done in O(1) time in the worst-case.
11. Shell sort is a stable sorting algorithm.
12. Under the adjacency matrix representation of graphs, it takes O(n) worst-case time to determine if an undirected graph contains a vertex that is connected to no other vertices.
13. When we do a Delete-Min from a Fibonacci heap, we remove the node containing the minimum value, add the children of the deleted node to the list of roots, and then perform a consolidate operation which involves linking roots of equal degree until all roots remaining have distinct degrees. Suppose that before the consolidate operation there are 8 roots in the list, with degrees 1, 1, 1, 1, 1, 3, 3, 3. After the consolidate, there are 2 roots remaining in the heap.
14. The amortized time of delete-min for Fibonacci heaps is O(1).
15. The worst case running time of a find operation in disjoint sets (implemented by trees with union-by-height and path compression) may require O(n) time.
16. Depth-first-search can be used to decide whether an undirected graph has a cycle.
17. Skip lists provide an efficient randomized data structure for implementing the *list* ADT.
18. Regardless of the increment sequence, the shell sort algorithm cannot do better than Ω(n log n)

2. (10 pts) True or False? (Explanation needed).

Consider the quick-union (i.e., union by height with path compression for find) algorithm for disjoint sets. We know that a sequence of *n* operations (unions and finds) can take asymptotically slightly more than linear time in the worst case. Assume that we start with *n* singletons (i.e., *n* sets each containing one element).

1. If all the finds are done before all the unions, a sequence of *n* operations is guaranteed to take *O(n)* time. *True or False? Why?*
2. If all the unions are done before all the finds, a sequence of *n* operations is guaranteed to take *O(n)* time. *True or False? Why?*

3. (12 pts) Complete the following table. Fill in the *worst-case* time complexity for each operation with respect to the associated data structure.

	Find minimum	Delete minimum	Union	Decrease key
Min-Binomial heap				
Min-Fibonacci heap				
Min-max heap				

4. (16 pts)

(1) Figure 1 is a binomial heap on 10 keys. Note that it consists of two trees. If we apply the following sequence of 3 operations:

delete- minimum, insert 15, decrease key 72 to 13

What will be the resulting binomial heap? Show your work in sufficient detail.

(2) Suppose the heap in Figure 1 is a Fibonacci heap. Apply the same sequence of operations. Show the final Fibonacci heap and intermediate steps.

5. (15 pts)

Using Dijkstra's algorithm on the graph in Figure 2 to determine the Shortest Paths from node a to all the remaining nodes. Show your derivation in sufficient detail.

6. (15 pts)

With respect to *(1) selection sort, (2) insertion sort, (3) shell sort, (4) merge sort, and (5) quick sort*, decide whether the sorting algorithm is likely to produce the arrangement of values shown in Figure 3 at an intermediate iteration of the algorithm? The height of the bars indicates the value of the item at that position and the values are in the process of being sorted in ascending (遞增) order . Why? (對於五種排序演算法, 分別簡單解釋過程中是否有可能有出現Figure 3的排列。)

7. (14 pts)

Consider an insertion of the key x = 222 into the initial hash table shown in Figure 4. For each of the following probing methods, indicate the sequence of table entries that would be probed (i.e., examined), and the final location of insertion for the key. Assume that the hash function is $h(x) = 2$. (**In each case start with the same initial table.**)

(a) Linear probing.
(b) Quadratic probing.
(c) Double hashing, where $g(x) = 7$
(i.e., the secondary hash function).

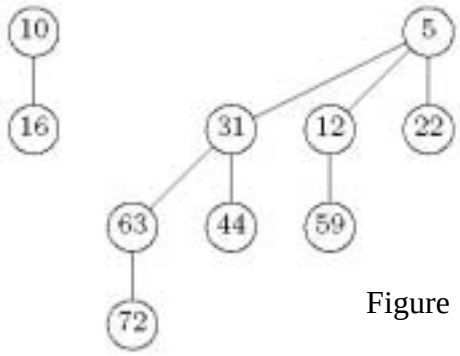
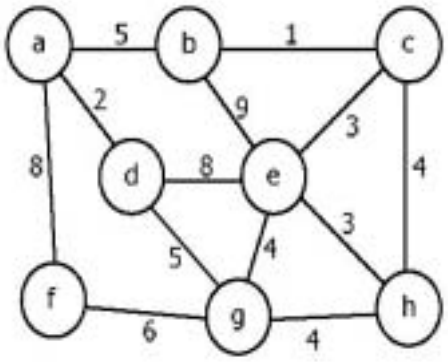


Figure 1

Figure 3

Figure 2



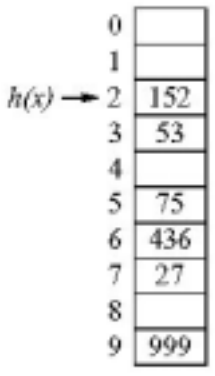


Figure 4