國立成功大學
114學年度碩士班招生考試試題

編　號：172

系　所：資訊管理研究所

科　目：資料結構

日　期：0211

節　次：第 3 節

注　意：1.不可使用計算機
　　　　2.請於答案卷(卡)作答，於試題上作答，不予計分。

A-1 Multiple-Choice Questions [40%, 4% each]

1. If the time complexity of an algorithm is $T(n) = 3T\left(\frac{n}{9}\right) + n^{0.51}$, what is the Big-O complexity?

   (a) $O(\log(n^2))$ (b) $O(log(n/2))$ (c) $nlog(n)$ (d) $O(n^{0.51})$

2. Which statement is correct regarding NP-Hard and NP-Complete problems?
   (a) All NP-Hard problems are NP-Complete
   (b) NP-Hard problems are a subset of NP-Complete problems
   (c) All NP-Complete problems are NP-Hard
   (d) NP-Hard problems can only be solved using brute force.

3. If the size of the main memory is M and the size of the data is N, what is the number of passes approximately required for external merge sort?
   (a) $\log_M N$ (b) $\lceil \log_{M-1}(N/M) \rceil$ (c) $\lceil \log_M(N/M) \rceil$ (d) $\log_{M-1} N$

4. What is the time complexity of the enqueue and dequeue operations in a queue implemented using two stacks, considering the amortized analysis?
   (a) $O(1)$ for enqueue and $O(n)$ for dequeue
   (b) $O(1)$ for both operations
   (c) $O(n)$ for both operations
   (d) $O(n)$ for enqueue and $O(1)$ for dequeue

5. Which of the following is NOT true regarding closed hashing?
   (a) Collisions are resolved directly within the hash table itself by searching for the next available slot
   (b) Often requires the table to be resized and rehashed when the load factor is too high
   (c) Linear probing and quadratic probing are commonly used for resolving collisions in closed hashing
   (d) Also known as separate chaining, is a method of handling collisions in a hash table

6. Suppose n is the number of elements to be sorted and k is the number of digits (or characters) in the largest element (or string). What is the time complexity for Radix Sort?
   (a) $O(nlog(k))$ (b) $O(n+k)$ (c) $O(n \times k)$ (d) $O(klog(n))$

7. Which of the following best describes the typical use case of orthogonal lists?
   (a) Dense matrices with few zero elements
   (b) Network representations with few nodes having directed links
   (c) Small matrices where space complexity is not an issue
   (d) Graph representations with many edges

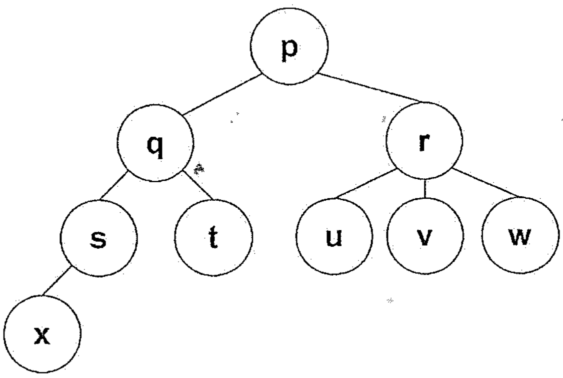
8. Which of the following is NOT a common application of priority queues?
   (a) Ticket booking management where some users are VIPs
   (b) Task scheduling in operating systems
   (c) Dijkstra's shortest path algorithm
   (d) Huffman coding for data compression

9. Which of the following is true regarding heap sort?
   (a) Building the initial heap takes $O(nlog(n))$ time.
   (b) Heap sort's complexity improves when the input is nearly sorted.
   (c) Heap sort is a divide-and-conquer algorithm
   (d) Heap sort is not a stable sorting algorithm

10. What is the time complexity for searching a string in a Trie with n total strings and m characters in the search string?
   (a) $O(nlog(n))$
   (b) $O(log(n))$
   (c) $O(m)$
   (d) $O(n \cdot m)$

A-2 [10%] You have a set of jobs, each with a specific load (or weight), and you want to partition them into two subsets such that the total load in each subset is as equal as possible. Show that this problem is NP-complete by reducing from the Subset Sum Problem.

B-1. Use the following graph for questions B-1-1 through B-1-3. (12%)



B-1-1. Which of the following must be done to ensure that this tree is a min-heap? (4%)

a) Remove x

b) Remove one of u, v, or w

c) Correct the heap order property

d) None of the above

B-1-2. Assume that all necessary changes have been made to ensure this tree is a min-heap. After one call to removeMin(), which node will be at the root? (4%)

a) p

b) q

c) x

d) r

B-1-3. Consider an algorithm that sorts n numbers using a heap. The algorithm begins with an empty heap, sequentially inserts each number into the heap, and then repeatedly removes and outputs the smallest number from the heap until it is empty. What is the Big-O running time of this algorithm? (4%)

(a) O($\log^2 n$)

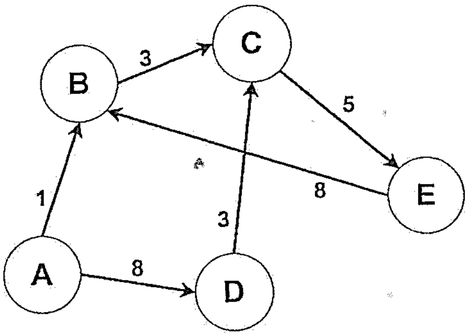
(b) O(n^2)

(c) O(n)

(d) O($n\, log\, n$)

(e) O($n^2\, log\, n$)

B-2. Use the following graph for questions B-2-1 through B-2-4. (12%)



B-2-1. If possible, provide two (2) valid topological orderings of the nodes in the graph above. If only one valid topological ordering exists, list that single ordering. If no valid topological ordering is possible, explain why. (4%)

B-2-2. What is the worst-case Big-O running time of performing a topological sort on a graph represented using an adjacency list? Express your answer in terms of V (the number of nodes) and E (the number of arcs). No explanation is required. (4%)

B-2-3. This graph is (Choose all that are true): (4%)

a) Directed
b) Weakly connected
c) Undirected
d) Complete
e) Acyclic
f) Strongly connected

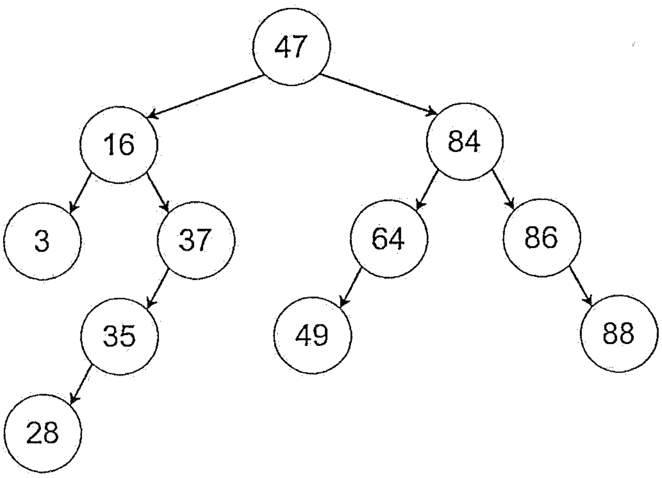
B-3. Provide exact answers (not in Big-O notation) to questions B-3-1 through B-3-3 in terms of V (the number of nodes). Do not include E (the number of arcs) in your answers. Explanations are not required. (12%)

B-3-1. The maximum number of arcs in a directed graph if self-loops are not allowed. (4%)
B-3-2. The minimum number of arcs in a connected undirected graph without self-loops. (4%)
B-3-3. The minimum degree of a node in a complete undirected graph with self-loops. (4%)

B-4. Provide the Big-O running time for each of the following operations. (6%)
B-4-1. Inserting an element into a binary search tree. (3%)
B-4-2. Performing a pre-order traversal of all elements in a binary search tree. (3%)

B-5. You are given an unknown AVL tree, which is known to be a complete tree. Additionally, the post-order traversal of this tree is provided as: 11, 33, 22, 55, 77, 66, 44. Construct and draw the tree based on this information. (2%)

B-6. Perform the following insertions and deletions sequentially on the binary search tree T, ensuring that each operation is completed by adding or removing a leaf while preserving the binary search tree property (swapping keys down into a leaf may be necessary). Do not use rotations to balance the tree. Draw the resulting tree after completing all operations. (6%)



1. T.insert(2)
2. T.delete(49)
3. T.delete(35)
4. T.insert(85)
5. T.delete(84)