# 國立成功大學

# 115學年度碩士班招生考試試題

編　號：162

系　所：資訊管理研究所

科　目：資料結構

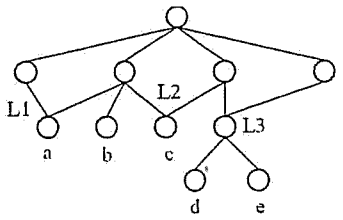
日　期：0203

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 [50%, 5% each]

1. In file organization and data storage, a secondary key is primarily used to:
   (a) Uniquely identify each record in a data file
   (b) Determine the physical storage location of records
   (c) Support efficient retrieval of records based on non-unique attributes
   (d) Guarantee consistency between different data files
2. An inverted list is best described as a data structure that:
   (a) Stores records sequentially according to their primary keys
   (b) Maps each attribute value to a list of records that contain that value
   (c) Maintains a balanced tree structure for fast insertion and deletion
   (d) Organizes records by reversing the order of their fields to enable backward traversal
3. A reentrant list is commonly used in data structure design to:
   (a) Allow multiple threads to safely access a shared list without synchronization
   (b) Enable traversal of a list in both forward and backward directions using a single pointer per node
   (c) Eliminate the need for auxiliary memory by storing list elements in contiguous locations
   (d) Support nested list structures by allowing a node to belong to more than one list simultaneously
4. Consider the DAG shown below, where all edges are directed downward, which of the following is the correct reentrant list representation of the DAG?




   (a) $\langle\langle a\rangle, \langle a, b, c\rangle, \langle c, \langle d, e\rangle\rangle, \langle\langle d, e\rangle\rangle\rangle$
   (b) $\langle\langle a\rangle, \langle\langle a\rangle, b, c\rangle, \langle c, d, e\rangle, \langle\langle d, e\rangle\rangle\rangle$
   (c) $\langle a, \langle\langle a\rangle, b, c\rangle, \langle c, \langle d, e\rangle\rangle, \langle d, e\rangle\rangle$
   (d) $\langle a, \langle a, b, c\rangle, \langle c, \langle d, e\rangle\rangle, \langle d, e\rangle\rangle$
5. In a dynamic memory allocator that supports block splitting and immediate coalescing, a doubly linked list is commonly used to manage free memory blocks. Which of the following statements best explains why a doubly linked list is preferred over a singly linked list for managing the free list?
   (a) A doubly linked list allows the allocator to locate a free block of suitable size in constant time during allocation.
   (b) A doubly linked list guarantees zero external fragmentation by preserving physical adjacency of memory blocks.
   (c) A doubly linked list makes boundary tags unnecessary for coalescing adjacent free blocks.
   (d) A doubly linked list enables constant-time removal and reinsertion of free blocks during allocation and deallocation without requiring traversal to find the predecessor block.
6. Suppose you are given $k$ sorted sequences $S_1, S_2, \cdots, S_k$, where the total number of elements across all sequences is $N$. You are required to merge them into a single sorted sequence. Which of the following statements regarding multi-way merging is correct?
   (a) A direct extension of two-way merging that repeatedly scans all $k$ sequences at each step yields a total time complexity of $O(N \cdot k)$, which is asymptotically optimal when $k = \Theta(N)$
   (b) If all sequences are of equal length, a divide-and-conquer multi-way merge can be performed in $O(N)$ time without auxiliary data structures.

(c) Using a binary min-heap of size $k$ to perform the merge yields a time complexity of $O(N \cdot \log(k))$, which is asymptotically optimal in the comparison model.

(d) Multi-way merging using a tournament tree achieves $O(k \cdot log(N))$ preprocessing time and $O(N)$ total merge time

7. Let $U = \{0,1,2,\cdots,n-1\}$ be a fixed universe of elements. A set $S \subseteq U$ is represented using a bit vector $B$ of length $n$, where $B[i] = 1$ if and only if $i \in S$. Which of the following statements is correct?

(a) Bit vectors require $O(|S|)$ space and therefore are inefficient when the universe size $n$ is large.

(b) Set union and intersection operations can be implemented in $O(n)$ time.

(c) Membership testing in a bit vector representation requires $O(\log(n))$ time.

(d) Bit-vector representations are asymptotically inferior to balanced search trees for all set operations when $n$ is large.

8. The Composite design pattern allows clients to treat individual objects and compositions of objects uniformly by organizing them into a tree structure. Which of the following statements about the Composite design pattern is correct?

(a) In the Composite pattern, leaf objects must implement all child-management operations, even if those operations are meaningless.

(b) The Composite pattern enables recursive composition of objects while allowing clients to interact with both primitive and composite objects through a common interface.

(c) The Composite pattern guarantees compile-time type safety by preventing clients from invoking child-related operations on leaf objects.

(d) The Composite pattern is primarily intended to eliminate inheritance by replacing it with object aggregation.

9. Consider a large sparse matrix $A$ of size $n \times n$, where most elements are zero. Let $nnz$ denote the number of non-zero elements in $A$. If $A$ is stored Compressed Sparse Row (CSR) format, what is the time complexity of multiplying $A$ with a dense vector of size $n$?

(a) $O(nnz)$ (b) $O(\log(n))$ (c) $O(n^2)$ (d) $O(n \cdot log(n))$

10. Consider a dictionary implemented as a hash table. The table has $m$ slots and stores $n$ keys. Collisions are resolved either by separate chaining or open addressing (linear probing). Assume uniform hashing. What is the expected time complexity for a successful search in a hash table using separate chaining?

(a) $O(1)$ (b) $O(n)$ (c) $O(\log(n))$ (d) $O\left(1+\frac{n}{m}\right)$

Part B. Multiple-Choice Questions. (50%)

Please note that there is exactly one correct answer for each question.

B-1. You are constructing a Max-Heap using the integers {1,2,3,4,5,6,7} stored in an array (index 1 to 7). Which of the following array representations represents a valid Max-Heap that contains the minimum possible number of inversions? (An inversion is a pair of indices $(i, j)$ such that $i < j$ and $A[i] > A[j]$) (4%)

(A) [–, 7, 6, 5, 4, 3, 2, 1] (B) [–, 7, 3, 6, 1, 2, 4, 5] (C) [–, 7, 5, 6, 4, 2, 1, 3]

(D) [–, 7, 6, 3, 5, 4, 1, 2] (E) [–, 7, 4, 6, 1, 2, 3, 5] (F) [–, 7, 6, 5, 1, 2, 3, 4]

B-2. Consider the following algorithm that takes an integer $n$ as input. What is the asymptotic time complexity of Algorithm-X as a function of $n$? (4%)

```
Algorithm-X(n)
c = 0
a = n
while a > 1
    b = 1
    while b <= a^2
        c = c + 1
        b = 2 * b
    a = a / 2
return c
```

(A) $O(\log n)$ (B) $O(n)$ (C) $O(n \log n)$

(D) $O(\log^2 n)$ (E) $O(\sqrt{n})$ (F) $O(n^2)$

B-3. We want to construct a binary search tree (BST) with the keys {1,2,3,4,5,6,7}. If we insert the keys in the order 4, 2, 6, 1, 3, 5, 7, we get a perfectly balanced tree of height 2. How many distinct insertion orders of these 7 keys result in a BST of height exactly 2? (Height is defined as the number of edges from the root to the deepest leaf) (4%)

(A) 16 (B) 48 (C) 64 (D) 80 (E) 24 (F) 32

B-4. Consider the graph $G$ with vertices $\{A, B, C, D\}$. Edge weights are: $(A, B) = 1, (B, C) = 2, (C, D) = 3, (D, A) = 3, (A, C) = 4$. If we run Prim's algorithm starting from $A$, and Kruskal's algorithm, which edge is added to the minimum spanning tree by Kruskal's but NOT necessarily by Prim's (depending on tie-breaking)? (4%)

(A) $(A, B)$ (B) $(B, C)$ (C) $(C, D)$ (D) $(D, A)$ (E) $(A, C)$

(F) None, they always produce the same set of edges for unique weights (or specified tie-breaking).

B-5. You have a list of items $[(3,A),(2,B),(3,C),(1,D)]$ where each item is $(Key, Value)$. You sort them based on $Key$. Which of the following result lists indicates that the sorting algorithm used was NOT stable? (4%)

(A) $[(1,D),(2,B),(3,A),(3,C)]$ (B) $[(1,D),(2,B),(3,C),(3,A)]$ (C) $[(3,A),(3,C),(2,B),(1,D)]$

(D) $[(3,C),(3,A),(2,B),(1,D)]$ (E) All of the above are stable.

B-6. Suppose a problem $Q$ is known to be NP-complete. Which of the following statements must be true? (Assume $P \neq NP$). (2%)

(A) There exists a polynomial-time algorithm to solve $Q$.

(B) $Q$ cannot be verified in polynomial time.

(C) Any problem in $NP$ can be polynomially reduced to $Q$.

(D) $Q$ can be polynomially reduced to the shortest path problem.

(E) $Q$ is not in $NP$.

(F) Solving $Q$ takes $O(n!)$ time.

B-7. Which of the following is a strict advantage of a balanced binary search tree (like a red-black tree) over a hash table (with separate chaining)? (4%)

(A) Better average-case time complexity for insertion.

(B) Uses less memory for storing integers.

(C) Guaranteed constant time for search in the worst case.

(D) Efficiently supports range queries (e.g., keys between $10$ and $50$) and ordered iterations.

(E) Easier to implement.

(F) Better cache locality.

B-8. Dijkstra's algorithm is run on a graph with start node $S$. The graph contains a directed edge $A \to B$ with weight $-5$. Under what condition will Dijkstra's algorithm definitely fail to find the correct shortest path to some node? (4%)

(A) If the shortest path to $B$ goes through $A$.

(B) If node $B$ is visited (popped from priority queue) before node $A$.

(C) If the shortest path to $A$ is strictly shorter than the shortest path to $B$.

(D) If the negative edge is part of a negative cycle reachable from $S$.

(E) If node $A$ is visited (popped from priority queue) before node $B$, and the edge $A \to B$ offers a shorter path to $B$ than previously known.

(F) Dijkstra's algorithm never fails on negative edges as long as there are no negative cycles.

B-9. We are sorting the array [@, #, $, %, &, *] using Insertion Sort. At an intermediate step, the array state is [&, #, $, @, %, *]. Which of the following statements is definitely TRUE based on this state? (4%)

(A) & is the smallest element in the entire array.

(B) * is the largest element in the entire array.

(C) & < # is true.

(D) % < @ is false.

(E) The element % is guaranteed to be larger than @.

(F) Insertion sort has completed exactly 4 iterations (outer loop).

B-10. We have a binary tree (not necessarily a binary search tree) with distinct values. Surprisingly, its preorder traversal is identical to its postorder traversal. If the tree has height $H$ (single node has height 0), what can we say about the number of nodes $N$ in this tree? (4%)

(A) $N = 1$, regardless of $H$. (B) $N = H + 1$. (C) $N = 2^{H+1} - 1$.

(D) $N$ can be any value. (E) Such a tree cannot exist. (F) $N$ must be even.

B-11. Which of the following statements regarding the stability and properties of sorting algorithms is correct? (4%)

(A) Quicksort is a stable sorting algorithm when implemented using the Hoare partitioning scheme.

(B) Heapsort is classified as a stable sorting algorithm.

(C) Least significant digit (LSD) radix sort requires a stable underlying sub-sorting algorithm (such as counting sort) to function correctly.

(D) Merge sort fails to achieve $O(N \log N)$ time complexity when implemented on a linked list data structure.

(E) Insertion sort exhibits maximum efficiency (best-case performance) when the input array is in reverse sorted order.

(F) The number of swap operations performed by selection sort is $O(N^2)$ in the worst-case scenario.

B-12. Which of the following statements regarding topological sort in graph theory is correct? (4%)

(A) Topological sorting is exclusively applicable to undirected graphs.

(B) The computation of a topological ordering is impossible if the graph contains a cycle.

(C) The resulting topological ordering for a given graph is inherently unique.

(D) When utilizing depth-first search (DFS) for topological sorting, the vertex order is determined by their pre-order traversal indices.

(E) The computational time complexity of topological sorting is $O(V^2)$ in the worst case.

(F) Every directed acyclic graph (DAG) possesses exactly one source node and one sink node.

B-13. Consider the execution of Kruskal's algorithm on a sparse graph containing $V$ vertices and $E$ edges, where the graph density implies $E \approx V$. If the union-find data structure is implemented using weighted quick union (WQU) with path compression (PC) for cycle detection, which of the following components primarily dominates the overall asymptotic time complexity? (4%)

(A) The initialization of the union-find data structure.

(B) The sorting of all edges based on their weights.

(C) The execution of $E$ find operations.

(D) The execution of $V$ union operations.

(E) The construction of the adjacency list representation.

(F) The random selection of the initial vertex.