請用 2B 鉛筆作答於答案卡，並先詳閱答案卡上之「畫記說明」。

是非題（若你覺得該小題命題正確，請填（A），錯誤，請填（B）。每題答對得 4 分，答錯倒扣 4 分，未填答者不計分也不扣分，倒扣至是非題總分為 0 分為止。）

1. A binary heap may not be a complete binary tree.
2. A complete binary tree can be transformed into a heap in $\Theta(n)$ steps.
3. There is a min-heap with seven distinct elements so that the preorder traversal of it gives the elements in sorted order.
4. Arrays are inefficient for implementing dynamic sorted lists because the *insert* and *delete* operations require moving a quarter of the elements, on average.
5. Computers handle postfix notation more easily than infix notation because infix requires parentheses but postfix does not.
6. Hash tables that use separate chaining need rehashing to avoid performance degradation due to long chains.
7. Using *quatratic probing* to resolve hash collision, we add i^2 to the address upon the i-th collision.
8. If a binary tree has 1,000 nodes, it has a minimum height of 10. (The height of a tree with one node is 0.)
9. If T is a BST and both the left and right subtrees of T are AVL trees, then T must be an AVL tree.
10. The postfix expression "5 3 + 7 7 - 1 - /" evaluates to 4.

複選題（每題 10 分，題內每個選項單獨計分，答對得 2 分，答錯倒扣 2 分，倒扣至複選題總分為 0 分為止。）

11. Which of the following statements are true?
    (A). $f(n) \in O(g(n))$ if and only if $g(n) \in \Omega(f(n))$.
    (B). $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$
    (C). $o(g(n)) = O(g(n)) - \Theta(g(n))$
    (D). $\omega(g(n)) \cup \Theta(g(n)) = \Omega(g(n))$
    (E). $\omega(g(n)) \cap \Theta(g(n)) = \emptyset$
12. Which of the following statements are true?
    (A). Performing preorder traversal on BST produces a sorted list.
    (B). The k-th smallest element in a BST with n elements can be found with $O(n)$ in time.
    (C). Every subtree of a BST is another BST.
    (D). The minimal AVL tree with height 5 has 20 nodes. (The height of a tree with one node is 0.)
    (E). When the structure of an AVL tree changes (e.g., with insertion or deletion), one single rotation is sufficient to restore the AVL tree property.
13. A graph is a pair (V, E), where V is a set of nodes, called vertices, and E is a collection of pairs of vertices, called edges. There are several ways to represent a graph, among them Adjacency Matrix and Adjacency List.
    - Adjacency Matrix: Let A be a matrix of size $|V| \times |V|$. $A[u, v] = 1$ if there is an edge between vertex u and v, and 0, otherwise.
    - Adjacency List: All vertices connected to a vertex v are listed on an adjacency list for vertex v.

    Which of the following are correct?
    (A). Adjacency List is more suitable for dense graph.
    (B). Using Adjacency Matrix, checking edge between vertices u and v takes constant time.
    (C). The big advantage of BFS is that it has much lower memory requirement than DFS.
    (D). BFS is better in finding shortest paths than DFS.
    (E). Using Adjacency List, the time complexity of topological sort on a directed graph is $O(|E| + |V|)$.

見背面

14. An algorithm is said to be *insensitive to input* if its running time is independent of the initial state of its input. For sorting algorithms, that means that the run time depends only on the number of elements to be sorted, not on their initial arrangement. Which sorting algorithms are insensitive to input?

    (A). Bubble sort.
    (B). Selection sort.
    (C). Merge sort.
    (D). Quick sort.
    (E). Radix sort.

15. Which of the following statements are true?

    (A). Hash tables are generally used in cases where insertion and deletion operations are more often than searches.
    (B). Open-addressed hash tables cannot be used if the data does not have unique keys.
    (C). With linear probing, searches become faster as we remove data from the hash table.
    (D). Both linear and quadratic probing can probe all locations in the table.
    (E). Double hashing resolves the clustering issue by using key-dependent probe intervals.

16. Which of the following statements are true?

    (A). An algorithm is $O(\log n)$ if it takes a constant time to cut the problem size by a fraction.
    (B). The Horner's method of evaluating the polynominal

$$p(x) = a_0x^n + a_1x^{n-1} + a_2x^{n-2} + \cdots + a_{n-1}x + a_n$$

by

$$p(x) = (\cdots(((a_0)x + a_1)x + a_2)x + \cdots + a_{n-1})x + a_n$$

runs in $\Theta(n^2)$ time.

    (C). The running time of the following recursive pseudo-code as a function of n is $\Theta(n^3)$.

```
def function(n):
    if n < 2: return
    else: counter = 0
    for i in range(1,8):
        function(n/2)
    for i in range(1,n*n*n):
        counter += 1
```

    (D). The complexity of the following function is $O(n \log n)$.

```
def function(n):
    count = 0
    for i in range(n/2,n):
        j = 1
        while j + n/2 <= n:
            k = 1
            while k <= n:
                count += 1
                k *= 2
            j *= 2
    print(count)
```

    (E). The complexity of the following function is $O(\log n)$.

```
def Function(n):
i = s = 1
while s < n:
    i += 1
    s += i
    print('*')
```

試題隨卷繳回