國立成功大學

114學年度碩士班招生考試試題

編　號：148

系　所：電機資訊學院-資訊聯招

科　目：程式設計

日　期：0210

節　次：第 2 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。

Part I. 資料結構 (50%)

注意: 資料結構共有 13 題。

- 第 1-11 題為選擇題，請將答案劃記於答案卡，否則，不予計分。
- 第 12-13 題為問答題，請將答案書寫於答案紙，否則，不予計分。

一、 單選題 (12 分，每題 3 分): 本大題之答案請劃記於答案卡

1. [3%] Consider the following two functions written in C language:

```
void enqueue(int queue[], int *front, int *rear, int size, int value) {
    if ((*rear + 1) % size == *front) {
        printf("Queue is full\n");
        return;
    }
    if (*front == -1) *front = 0;
    *rear = (*rear + 1) % size;
    queue[*rear] = value;
}
int dequeue(int queue[], int *front, int *rear, int size) {
    if (*front == -1) {
        printf("Queue is empty\n");
        return -1;
    }
    int value = queue[*front];
    if (*front == *rear) {
        *front = *rear = -1;
    } else {
        *front = (*front + 1) % size;
    }
    return value;
}
```

Assume the queue Q with size = 5, initially empty (front = -1, rear = -1). What will be the content of the array Q after executing the given sequence of operations?

```
enqueue(Q, &front, &rear, size, 10);
enqueue(Q, &front, &rear, size, 20);
dequeue(Q, &front, &rear, size);
enqueue(Q, &front, &rear, size, 30);
enqueue(Q, &front, &rear, size, 40);
enqueue(Q, &front, &rear, size, 50);
dequeue(Q, &front, &rear, size);
enqueue(Q, &front, &rear, size, 60);
enqueue(Q, &front, &rear, size, 80);
enqueue(Q, &front, &rear, size, 90);
dequeue(Q, &front, &rear, size);
dequeue(Q, &front, &rear, size);
enqueue(Q, &front, &rear, size, 100);
```

(A) {40, 50, 60, 80, 90}
(B) {80, 90, 100, 50, 60}
(C) {-1, 50, 60, 80, 100}
(D) {50, 60, 80, 100, 40}
(E) {60, 80, 100, -1, 50}

2. [3%] Given a node structure as shown below:

```
typedef struct Node {
    int data;
    struct Node* next;
} Node;
```

What will the following function do when used on a singly linked list?

```
void unknownFunction(Node* node) {
    if (node == NULL || node->next == NULL) {
        return;
    }
    node->data = node->next->data;
    Node* temp = node->next;
    node->next = node->next->next;
    free(temp);
}
```

(A) The function will delete the specified node, including the last node in the list.

(B) The function will delete the specified node only if it is a middle node or any node except the last one.

(C) The function will produce a segmentation fault when applied to the last node in the list because it attempts to access node->next->data.

(D) The function will remove the node but corrupt the list structure if the deleted node has duplicate values in the list.

(E) None of the above is correct.

3. [3%] The following C code is provided. Analyze the functionality of the code and answer the question.

```
#include <stdio.h>
int operation1(int parent[], int x) {
    if (parent[x] != -1) {
        int rootX = operation1(parent, parent[x]);
        parent[x] = rootX;
        return rootX;
    }
    return x;
}
void operation2(int parent[], int x, int y) {
    int rootX = operation1(parent, x);
    int rootY = operation1(parent, y);
    if (rootX != rootY) {
        parent[rootY] = rootX;
    }
}
int main() {
    int parent[6] = {-1, -1, -1, -1, -1, -1};
    parent[2] = 1;
    parent[3] = 2;
    parent[4] = 3;

    operation1(parent, 4);
    operation2(parent, 1, 3);
    operation2(parent, 5, 4);

    for(int i=0; i<=5;i++){
        printf("%d ", parent[i]);
    }
```

```
    return 0;
}
```

Which of the following statements is correct?

(A) In the function main(), operation1(parent, 4) updates all nodes in the chain starting from 4, and operation2(parent, 1, 3) merges two different groups.

(B) The output of the program is "-1 -1 1 1 1 1".

(C) The output of the program is "-1 5 1 2 3 -1".

(D) The function operation1() compresses paths, and the function operation2() merges groups if they have different representatives.

(E) The function operation1() creates a loop, and the function operation2() merges all elements into one group.

4. [3%] Consider the following two functions written in C language for the insertion into a hash table and the deletion from the table:

```
void insert(int table[], int size, int key) {
    int index;
    for (int i = 0; i < size; i++) {
        index = (key % size + i * i) % size;
        if (table[index] == -1 || table[index] == -2) {
            table[index] = key;
            return;
        }
    }
}
void delete(int table[], int size, int key) {
    int index;
    for (int i = 0; i < size; i++) {
        index = (key % size + i * i) % size;
        if (table[index] == key) {
            table[index] = -2;
            return;
        }
    }
}
```

Given that the keys are all non-negative integers and the keys to be deleted are always in the hash table. Which of the following implementations correctly searches for a specific key in the hash table, considering collision handling and the presence of deleted slots?

(A)

```
int search(int table[], int size, int key) {
    int index;
    for (int i = 0; i < size; i++) {
        index = (key % size + i) % size;
        if (table[index] == -1) {
            return -1;
        }
        if (table[index] == key) {
            return index;
        }
    }
    return -1;
}
```

(B)

```
int search(int table[], int size, int key) {
    int index;
    for (int i = 0; i < size; i++) {
        index = (key % size + i * i) % size;
        if (table[index] == key) {
            return index;
        }
        if (table[index] == -1 || table[index] == -2) {
            return -1;
        }
    }
    return -1;
}
```

(C)

```
int search(int table[], int size, int key) {
    int index;
    for (int i = 0; i < size; i++) {
        index = (key % size + i * i) % size;
        if (table[index] == -1) {
            return -1;
        }
        if (table[index] == key) {
            return index;
        }
        if (table[index] == -2) {
```

```
                continue;
            }
        }
        return -1;
    }
```

(D)

```
int search(int table[], int size, int key) {
    int index;
    for (int i = 0; i < size; i++) {
        index = (key % size + i * i) % size;
        if (table[index] == -2) {
            return -1;
        }
        if (table[index] == key) {
            return index;
        }
    }
    return -1;
}
```

(E) None of the above is correct.

二、 複選題 (28 分，每題 4 分)：本大題之答案請劃記於答案卡。

• 每題有五個選項，其中至少有二個是正確答案。

5. [4%] Please successively insert the data pairs containing the following keys into an empty height-based min leftist tree: 10, 20, 30, 40, 50, 60, 70, 5, 4, 15, 16. After each insertion, the tree should still be a min leftist tree. Which of the following descriptions are correct for the resultant tree?
   Note: In a tree, each step from top to bottom is called the level of a tree. The level count starts with 1 and increments by 1 at each level or step.
   (A) The node containing key=10 has left child with key=30 and right child with key=20.
   (B) The node containing key=10 and the node containing key=16 are siblings.
   (C) After repeatedly performing the delete-min operation twice on the leftist tree, the node containing 20 is in the left subtree of the root.
   (D) The difference of height between the left and right subtrees of the root is 2.
   (E) The 3rd level of the tree has four nodes.

6. [4%] Please successively insert the data pairs containing the following keys into an empty min binomial heap (B-heap): 10, 20, 30, 40, 50, 60, 70, 5, 4, 15, 16. For each insertion of a single key, min-tree

joining (pairwise combine) is performed. Which of the following descriptions are correct for the resultant B-heap?

Note: In a tree, each step from top to bottom is called the level of a tree. The level count starts with 1 and increments by 1 at each level or step.

(A) In the resultant B-heap, the maximum degree of binomial tree is 0.

(B) The number of singly linked circular lists in the B-heap to maintain siblings and roots is 7.

(C) After performing the delete-min operation on the B-heap once, the minimum degree of binomial tree is 0.

(D) After performing the delete-min operation on the B-heap twice, the root list of min trees contains nodes with keys as 10 and 20.

(E) After performing the delete-min operation on the B-heap three times, the node containing key=50 is at the 2nd level of a min tree.

7. [4%] Given a tree containing n nodes, which of the following statements are correct?

(A) If the tree is a maximum cost spanning tree, the number of edges is n-1.

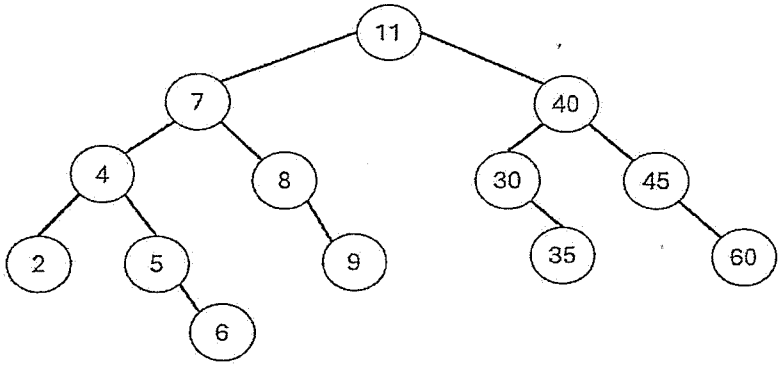
(B) If the tree is an AVL tree, the time complexity to delete an element is O(log n).

(C) If the tree is a compressed trie, the number of keys is n.

(D) If the tree is a red-black tree with n internal nodes, the rank of the root is at most $\log_2(n+1)$.

(E) If the tree is a binary search tree, its height is O(log n).

8. [4%] Consider the following AVL tree.



Which of the following statements are correct?

(A) The deletion of 35 requires rebalance operations.

(B) The insertion of 1 does not require rebalance operations.

(C) After deleting 9, the nodes containing keys 4 and 7 become siblings in the new tree.

(D) After successively deleting 35 and 60, the balance factor of the root in the new tree is 1.

(E) After successively inserting 27, 28, and 29, the balance factor of the root in the new tree is -1.

9. [4%] Considering the following two sequences, which are traversal results of a binary tree.

- Inorder-traversal result: H D I N B E J A O K P F L C G M

- Postorder-traversal result: H N I D J E B O P K L F M G C A

Please reconstruct the binary tree using the above two sequences. Which of the following descriptions are correct for the resultant binary tree?

Note: In a tree, each step from top to bottom is called the level of a tree. The level count starts with 1 and increments by 1 at each level or step.

(A) The balance factor of the root is 0.

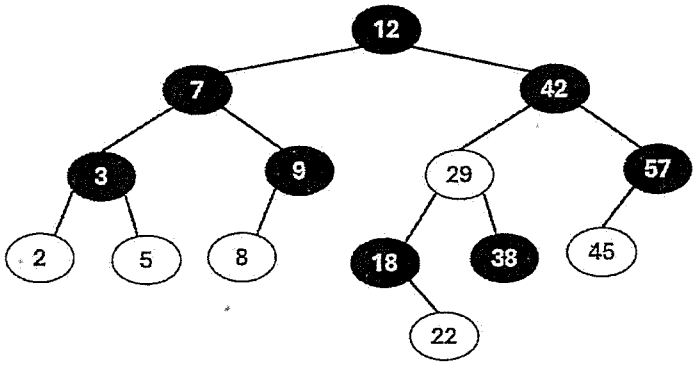
(B) The height of the tree is 4.

(C) The total number of leaf nodes is 7.

(D) After performing a depth-first search starting at node B of the resultant tree, the shortest path from vertex B to vertex O in the spanning tree has 5 edges.

(E) The last element in the result of level-order traversal of the tree is P.

10. [4%] Considering the following red-black tree, where the node colored as white indicates a red node and the node colored as black indicates a black node.



Which of the following statements are correct?

(A) After inserting 1 into the tree, the number of black nodes becomes 9.

(B) After successively inserting 100 and 58 into the tree, the number of red nodes becomes 7.

(C) The deletion of 42 from the tree may reduce the height of the tree.

(D) After successively deleting 45 and 57 from the tree, the rank of the root is 3.

(E) The deletion of 12 can be done via moving 9 to the root.

11. [4%] Considering the following adjacency matrix.

	0	1	2	3	4	5	6	7
0					1		1	
1			1			1		
2		1				1		
3					1		1	
4	1			1				
5		1	1					1
6	1			1				
7						1		

Please reconstruct the undirected graph based on the above adjacency matrix. Which of the following statements related to the resultant graph are correct?

(A) The breath-first search starting at vertex 1 produces a spanning tree with 8 vertices.

(B) The degree of vertex 4 is 2.

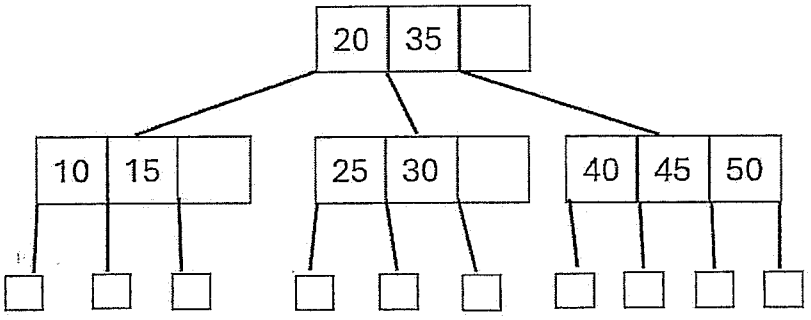
(C) The graph has one articulation point.

(D) The graph has one connected component.

(E) The graph has two cycles.

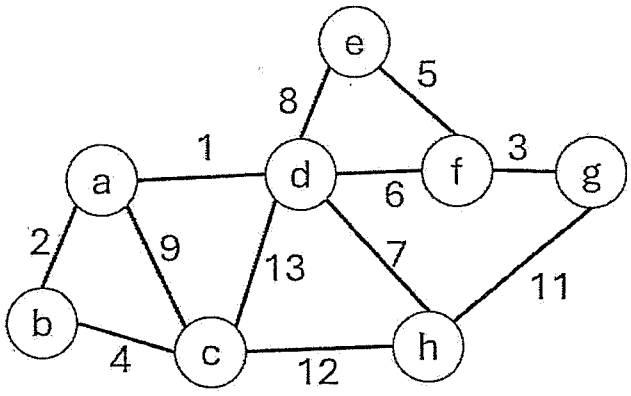
三、 問答題 (10 分): 本大題之答案請作答於答案紙

12. [5%] Considering the following B-tree of order 4.



Successively insert 62, 5, 85, and 12 one at a time into the above B-tree and then delete 5, 50, 62, 85, and 10 one at a time. Please show the new tree after completing all the insertion and deletion operations.

13. Considering the following graph G.



a) [1%] Write out the cost of the maximum cost spanning tree for graph G.

b) [2%] Write out the last edge selected into the maximum cost spanning tree for graph G when using Kruskal's method.

c) [2%] Write out the 5th edge selected into the maximum cost spanning tree for graph G when using Prim's method starting with vertex a.

Part II. 演算法 (50%)

14. (10%) Given a set S of size $n = 2^{2^m}$, where $m \in Z^+$, we use the following algorithm to perform clustering and labeling:

```
F(S)
1   If |S| == 2 then return;
2   Apply a clustering algorithm with a time complexity of Θ(lg n) to
    divide the set S into three clusters, S1, S2, and S3, and label them
    where |S| = n, |S1| = |S2| = √n, and |S3| = n − 2√n.
3   Call F(S1)
4   Call F(S2)
```

Let $T(n)$ represent the time required to use this algorithm to cluster and label a set of size n. Provide an asymptotic tight bound (Θ) for $T(n)$, assuming that $T(n)$ is a constant for sufficiently small n.

15. (10%) We offer n courses where each course i has a start time s_i, a finish time f_i, and a weight w_i. Two courses are considered **compatible** if their time intervals do not overlap. For simplicity, we can assume that the courses are sorted in non-decreasing order of their finish times, denoted as $f_1 \leq f_2 \leq \cdots \leq f_n$. Additionally, let $p(j)$ for a course j represent the largest index $i < j$ such that courses i and j are disjoint. We define $p(j) = 0$ if no course $i < j$ is disjoint from j. Our goal is to find a set $S \subseteq \{1, \ldots, n\}$ of mutually compatible courses such that the total weight of the courses in S is maximized. We have designed a recursive algorithm for this purpose, but we have omitted an essential part. Please assist us in completing the missing portion. **Hint:** This algorithm returns the value of the optimal solution to the problem consisting of courses $\{1, \ldots, i\}$.

```
F(i)
    If i = 0 return 0
    return_____(10%)_____
```

16. (10%) We aim to select courses from a pool of n courses. Since this is still in the planning phase, each course has no specified start or finish times. Each course i is characterized by its duration p_i and weight w_i. The total available classroom time is C. We want to identify a subset of courses $S \subseteq \{1, \ldots, n\}$ such that the total duration of courses in S does not exceed C and the total weight of S is maximized. To achieve this, we have developed a dynamic programming algorithm that uses a $(n+1) \times (C+1)$ table $T[0{:}n, 0{:}C]$ to solve the problem. Here, $T[i,j]$ represents the maximum total weight achievable when considering only the subset of courses $\{1, \ldots, i\}$ and a classroom available for j units of time. Write the recursive formula to compute $T[i,j]$.

17. (10%) Assuming that the available time for the classroom is unlimited, we instead focus on the course completion time. We aim to schedule n courses, where each course i is characterized by its duration p_i and weight w_i. In this schedule, the finish time of each course is the total duration of all

courses scheduled before it, plus its duration. A schedule can be defined as a function $d(i)$, representing the order of course i in the schedule. That is, if $d(i) < d(j)$, course i is scheduled before course j. Therefore, the finish time of course i is denoted as $f_i = \sum_{j\in\{1,\dots,n\}:d(j)<d(i)} p_j$. This scheduling problem aims to minimize the total weighted finish time $\sum_{i\in\{1,\dots,n\}} w_i f_i$. Suppose we have 15 courses with corresponding durations [6, 6, 9, 83, 34, 44, 164, 38, 82, 180, 19, 128, 394, 512, 15] and weights [2, 4, 4, 6, 7, 7, 7, 10, 10, 10, 11, 12, 13, 13, 15]. What is the value of the optimal schedule, that is, the minimum total weighted finish time?

18. (10%) The Floyd-Warshall algorithm can solve the all-pairs shortest-paths problem on a directed graph $G = (V, E)$. Let $d_{ij}^{(k)}$ be the weight of a shortest path from vertex i to vertex j for which all intermediate vertices are in the set $\{1,2,\dots,k\}$ and $D^{(k)} = \left(d_{ij}^{(k)}\right)$ be a $n \times n$ matrix. Floyd-Warshall algorithm computes $D^{(k)}$ from $D^{(k-1)}$ as the following formula.

$d_{ij}^{(k)} =$_____

Please complete the above formula.