國立臺灣師範大學 114 學年度碩士班招生考試試題

科目：**軟體基礎**　　適用系所：**資訊工程學系**

注意：1.本試題共 5 頁，請依序在答案卷上作答，並標明題號，不必抄題。2.答案必須寫在指定作答區內，否則依規定扣分。

一、填充題 (共 20 分) 答案請標明清楚空格編號，填空格內容即可

(一) The following C code defines the data structure of stack implemented by an array. Please fill in the blanks to complete the function for isEmpty, push, and pop. It is assumed that the function iniStack() is called initially before using the stack. (共 4 個空格) (4 分)

```
// Define the maximum stack size
#define MAX 1000
typedef struct {
    char items[MAX];
    int top;
} Stack;

void initStack(Stack *s) {
    s->top = -1;
}

// Check if the stack is empty
bool isEmpty(Stack *s) {
    return_____(1)_____;
}
```

```
void push(Stack *s, char c) {    // Push an element c onto the stack
    if (s->top <_____(2)_____) {
        s->items[_____(3)_____] = c;
    }
}

char pop(Stack *s) {     // Pop an element from the stack
    if (!isEmpty(s)) {
        return s->items[_____(4)_____];
    }
    return '\0';              // Return null character if stack is empty
}
```

(二) The following function is designed to check if a string *s* has balanced parentheses (), {} by using a stack. Example: "({})" → True, "({)}" → False. Please fill in the blanks to complete the function. (共 3 個空格) (6 分)

```
bool isBalanced(const char *s) {
    Stack stack;
    initStack(&stack);

    for (int i = 0; s[i] != '\0'; i++) {
        char current = s[i];

        if (current == '(' || current == '{') {
            push(&stack, current);
        } else if (current == ')' || current == '}') {
            if (_____(5)_____) {
                return False;
            }
            char top = _____(6)_____;
            if ((current == ')' && top != '(') ||
                (current == '}' && top != '{') {
                return False;
            }
        }
    }

    Return _____(7)_____;
}
```

(三) The following C code is used to merge two sorted linked lists into a single sorted linked list. Example: list1: 1 → 3 → 5 and list2: 2 → 4 → 6

The returned sorted list is: 1 → 2 → 3 → 4 → 5 → 6

Please fill in the blanks to complete the function. Note that list1 and list2 are possible NULL and the resultant list has a dummy node as the head node. (共 5 個空格) (10 分)

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

struct Node* mergeSortedLists(struct Node* list1, struct Node* list2) {
    struct Node dummy;
    struct Node* tail = &dummy;
    dummy.next = NULL;

    while (list1 != NULL && list2 != NULL) {
        if (list1->data < list2->data) {
            tail->next = list1;
            list1 = ____(8)____;
        } else {
            tail->next = list2;
            list2 = ____(9)____;
        }
        tail = ____(10)____;
    }

    if (list1 != NULL) {
        tail->next = ____(11)____;
    } else {
        tail->next = ____(12)____;
    }

    return dummy.next; // Return the head of the merged sorted list.
}
```

二、Sorting problem (共 8 分)

(一) Suppose there are 7 files which contains sorted data, whose sizes are as following:
(F1) = 1200, (F2) = 900, (F3) = 2500, (F4) = 3200, (F5) = 800, (F6) = 1800, (F7) = 1500;
We would like to perform 2-way merge sorting several times on these 7 files to get a file with total sorted data, where the result of each 2-way merge sorting is written back to a file. Please decide the order of performing 2-way merges such that the total number of data I/O can be minimized. (4 分)

(二) Given 9 numbers stored in the following array.

index	1	2	3	4	5	6	7	8	9
value	4	10	1	18	2	9	20	7	12

1. Which one of the following options (**A-D**) is the possible temporal result when performing **insertion sort**? (2 分)
2. Which one of the following options (**A-D**) is the possible temporal result when performing **iterative merge sort**? (2 分)

(A)

index	1	2	3	4	5	6	7	8	9
value	1	4	10	18	2	7	9	20	12

(B)

index	1	2	3	4	5	6	7	8	9
value	1	2	4	7	10	9	20	18	12

(C)

index	1	2	3	4	5	6	7	8	9
value	1	2	4	18	10	9	20	7	12

(D)

index	1	2	3	4	5	6	7	8	9
value	1	2	4	10	18	9	20	7	12

三、選擇題 (共 12 分)

(一) Suppose that a graph with m vertices and n edges are represented by an adjacency matrix. What is the time complexity to **get all the edges terminating at a particular vertex**? (單選) (3 分)

(A) $O(m \times n)$
(B) $O(m)$
(C) $O(n)$
(D) $O(m^2)$
(E) $O(n^2)$

(二) What is the worst-case time complexity for **finding the height** of a binary tree with n nodes? (單選) (3 分)

(A) $O(1)$
(B) $O(\log n)$
(C) $O(n)$
(D) $O(n \log n)$
(E) $O(n^2)$

(三) What is the time complexity for finding certain element in a **max heap** with n nodes? (單選)(3 分)

(A) O(1)

(B) O(log n)

(C) O(n)

(D) O(n log n)

(E) O(n^2)

(四) Which of the following traversal on a **binary search tree** can be used to **reconstruct the same tree** structure? (多選) (3 分)

(A) Preorder traversal.

(B) Inorder traversal.

(C) Postorder traversal.

(D) Level-order traversal.

四、 Given the following *adjacency multi-list* structure for a graph G. (共 10 分)

Head Nodes	
V0[0]	E1
V1[1]	E1
V2[2]	E2
V3[3]	E3
V4[4]	E5
V5[5]	E6
V6[6]	E7

E1	0	1	E2	Null
E2	0	2	E3	E4
E3	0	3	Null	E4
E4	2	3	E5	Null
E5	2	4	E6	E7
E6	2	5	Null	E8
E7	4	6	Null	E8
E8	5	6	Null	Null

(一) Please draw the **BFS spanning tree** of G **starting from vertex V3**. (3 分)

(二) In order to find the articulation points of the graph G1, traverse the graph starting from vertex 3 to get the dfs number (*dfn*) and the lowest dfn (*low*) reachable by a single back edge. Please write the *dfn* and *low* value for each vertex. If there is more than one adjacent vertex, **visit the vertex with smaller number first**. (4 分)

(三) Please draw the **bi-connected components** of G separated from the **articulation points**. (3 分)

五、 (7 分) Let $A[1 : N]$ represent an array containing N elements, and the index of A starts from 1. The first n, $n < N/2$, entries of A (i.e., $A[1 : n]$) store a sequence of n integers $a_1, a_2, \ldots, a_n$. Assume that these n integers are sorted in non-descending order, that is, $a_1 \le a_2 \le \ldots \le a_n$. Each of the remainder of array A stores a large number $M > a_n$.

On input an integer y, $y < M$, design an $O(\log n)$ algorithm to find the index k such that $A[k] = y$, or report that there is no such y in A. Note that the input data is only y, not including n and N.

六、(8 分) Finite groups are often used in cryptosystems. For example, in RSA, the multiplicative group used is the integers modulo n, represented by Z_n^*. In this group, the product of x and y is an ordinary integer product, but the result must be divided by n and then take the remainder. For example, in Z_{15}^*, $2 \times 4 = 8 \bmod 15 = 8$ and $4 \times 7 = 28 \bmod 15 = 13$. Given an integer a, the inverse of a in Z_n^* is the number b, such that $a \times b = 1$. For example, $2 \times 8 = 16 \bmod 15 = 1$, so the inverse of 2 in Z_{15}^* is 8.

On input two integers a and n design an efficient algorithm for computing the inverse of a in Z_n^*.

七、(共 15 分) Given a weighted undirected graph $G = (V, E, w)$, where V is the vertex set, E is the edge set, $w : E \rightarrow \mathbf{R}$ is the weight defined on the edges of G. Computing the minimum spanning tree and shortest path of G has many practical applications. Some algorithms require weights to be non-negative. Assume that some edges of G have negative weights.

(一) (7 分) Is it possible to add a fix number to the weights so that all weights are nonnegative and then run the algorithm to get a correct minimum spanning tree for the original graph?

(二) (8 分) Is it possible to do this while computing the shortest path?

For each case, if it is possible, provide a proof or a convincing reason; Otherwise, provide a counterexample showing that the output would be different.

八、(共 20 分) Given a weighted undirected graph $G = (V, E, w)$. Assume that $|V| > 2$ and the weights $w(e) > 0$ for each edge $e \in E$. A simple cycle of G that contains every vertex is called a Hamiltonian cycle. Computing the Hamiltonian cycle of a given graph is NP-hard. In this problem, we want to compute a Hamiltonian cycle whose weight is at most 2 times the weight of the shortest Hamiltonian cycle. The method is described as follows:

(a) Compute a minimum spanning tree T of G.

(b) Start from any vertex v_0, let $v_0, v_1, \ldots, v_{n-1}$ be the depth first listing of the vertices of the tree T.

(c) Output the cycle $C = v_0, v_1, \ldots, v_{n-1}, v_0$.

Assume that the weights on the edges of G satisfy the triangle inequality. Let S be a subgraph of G. Define $w(S)$ as the sum of all edge weights of subgraph S. Let C^* be the optimal Hamiltonian cycle of G. Prove $w(C) \leq 2w(C^*)$ through the following steps.

(一) (9 分) $w(C) \leq 2w(T)$.

(二) (8 分) $w(T) \leq w(C^* - \{e\})$ for any edge e in C^*.

(三) (3 分) $w(C) \leq 2w(C^*)$.

In the above notation, $C - \{e\}$ is the graph obtained from C by deleting edge e from C.