國立臺北科技大學

九十二學年度資訊工程系碩士班入學考試

軟體設計試題

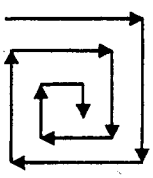
填准考證號碼

注意事項：

1. 本試題共[redacted]題，配分共 100 分。
2. 請按順序標明題號作答，不必抄題。
3. 全部答案均須答在答案卷之答案欄內，否則不予計分。

Problem 1. [20%]

A level-*N spiral* is a two-dimensional array of characters shown below.

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level *N*
A	AB DC	ABC HID GFE	ABCD LMNE KPOF JIHG	ABCDE PQRSF OXYTG NWVUH MLKJI	ABCDEF TUVWXG SFGHYH REJIZI QDCBAJ PONMLK	

Please write a function **void generate_spiral(char a[MAX_N][MAX_N], int n)** to generate a level-*n* spiral. Your program should store the resulting spiral in array **a**, which is assumed to be large enough to hold the target spiral. Note that you are encouraged to provide a recursive implementation. Full credits will be given to correct recursive implementations; only partial credits (at most 15%) will be given to correct iterative implementations.

Problem 2. [20%]

The following program is written in C++. Please answer the results that will be printed by the seven **cout** commands in the main program.

```
#include <iostream>
using namespace std;
class Base
{
    public:
        Base(int nx=5) { x = nx; }
        int f1() { return 0; }
        virtual int f2() { return 1;}
        int x;
};
class Derived : public Base
{
    public:
        Derived(int nx=7) { x = nx; }
        int f1() { return 2;}
        int f2() { return 3;}
        int operator+(Derived &rhs) {return x * rhs.x;}
        int x;
};
int main()
{
    Base b(6);
    cout << b.x << endl;            // (1) _____
    Derived d(4);
    cout << d.x << endl;            // (2) _____
    cout << d.Base::f2() << endl; // (3) _____
    b = d;
    cout << b.x << endl;            // (4) _____
    Base *b_ptr;
    b_ptr = &d;
    cout << b_ptr->f1() << endl;  // (5) _____
    cout << b_ptr->f2() << endl;  // (6) _____
    cout << d + d << endl;          // (7) _____
    return 0;
}
```

注意：背面尚有試題

Problem 3. [20%]

The *size* of a node x in a binary search tree is the number of keys (nodes) stored in the subtree rooted at x. A node x is *balanced*, if the size of the right child and the size of the left child of x differs by at most one. A *completely balanced* binary search tree is a binary search tree that every node in the tree is balanced. Given an arbitrary binary search tree, please describe how to rebuild the tree into an equivalent completely balanced binary search tree. Your algorithm should run in time $O(n)$ and it is allowed to use $O(n)$ auxiliary storage, where n is the size of the tree.

Problem 4. [15%]

In all of the five recurrences shown below, it is assumed that $T(1)=d$ for some constant d. State, using the "big oh" notation, the solution to each of the five recurrences shown below. Just state the answer – you do not need to justify them.

(1) $T(n)=2T(n/2)+c$

(2) $T(n)=2T(n/2)+cn^2$

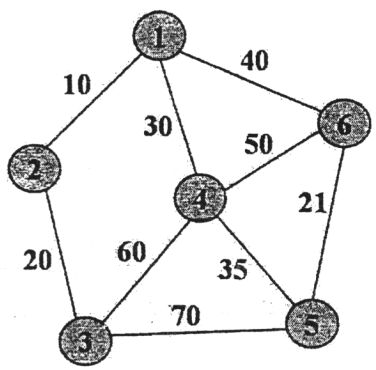
(3) $T(n)=T(n/2)+cn$

(4) $T(n)=T(n/2)+c\log n$

(5) $T(n)=T(n/4)+T(3n/4)+cn$

Problem 5. [25%]

Consider the edge-weighted connected graph $G = (V, E)$ shown below, where V is the vertex set and E is the edge set of G respectively.



(1) Please find a minimum-cost spanning tree of G by Kruskal's algorithm. Please show your work step by step. [5%]

(2) Write down the pseudo-code of the Kruskal's algorithm and show that the time complexity of the Kruskal's algorithm is $O(|E| \log |E|)$. [10%]

(3) Please prove that Kruskal's algorithm generates a minimum-cost spanning tree for every connected undirected graph. [10%]