國立臺北科技大學

九十四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. 本試題共五題，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 *full binary tree* is defined as a binary tree in which every non-leaf node has exactly two children. The nodes of a binary tree are constructed by the following declarations (in C language):

```
typedef struct node *tree_pointer;
typedef struct node {
        int data;
        tree_pointer left_child, right_child;
};
```

Please write a **recursive** function int IsFullBinaryTree(tree_pointer root), which determines whether the binary tree pointed by the root pointer is a full binary tree.

Problem 2 [20%]

Given the following C code, please show the output of statement (2-1) ~ (2-10) in the answer table.

```
#include <stdio.h>
#include <string.h>

int f1(int a[], int n)
{
        if (n >= 0) return (f1(a, n-1)+a[n]); else return 0;
}

int f2(char *s)
{
        int i;
        for (i=1; *s; i++, s++) ;
        return i;
}

void main()
{
     int i, j, a[5]={1,2,3,4,5};
     char s1[]="abcdefg", *s2="abcdefg";

     i = 65;
     printf("%d\n", i);                          // problem (2-1)
     printf("%c\n", i);                          // problem (2-2)

     for (i = 0, j = 0; i < 5; i++) j += i;
     printf("%d\n", i);                          // problem (2-3)
     printf("%d\n", j);                          // problem (2-4)

     for (i = 2, j = 10; i < j; j+=4, i*=2) ;
     printf("%d\n", i);                          // problem (2-5)
     printf("%d\n", j);                          // problem (2-6)

     printf("%s\n", s2+2);                       // problem (2-7)
     printf("%d\n", strcmp(s1, s2));             // problem (2-8)

     printf("%d\n", f1(a,3));                    // problem (2-9)
     printf("%d\n", f2(s1));                     // problem (2-10)
}
```

Please copy the following answer table onto your answer sheet.

Problem	Answer
2-1	
2-2	
2-3	
2-4	
2-5	
2-6	
2-7	
2-8	
2-9	
2-10	

注意：背面尚有試題

Problem 3　[20%]　（本題答錯有倒扣）

Given the following C++ code, please show the output of statement (3-1) ~ (3-10) in the answer table.

```
#include <iostream>
using namespace std;
class DataUnit{
public:
      int a;
      int b;
      int* c;
      int** d;
      int& e;
      DataUnit():a(2),b(3),c(&b),d(&c),e(a){ }
      void reset(){
            a = 2;
            b = 3;
      }
      void swap(int x , int y){
            int temp = x;
            x = y;
            y = temp;
      }
      void swap(int* x , int* y){
            int temp = *x;
            *x = *y;
            *y = temp;
      }
      void swap(int& x , int* y){
            int temp = x;
            x = *y;
            *y = temp;
      }
      void swap(int x , int** y){
            int temp = x;
            x = **y;
            **y = temp;
      }
      void swap(int* x , int& y){
            int temp = *x;
            *x = y;
            y = temp;
      }
      void swap(int* x , int** y){
            int temp = *x;
            *x = **y;
            **y = temp;
      }
      void swap(int** x , int y){
            int temp = **x;
            **x = y;
            y = temp;
      }
      void printData(){
            cout << "a = " << a << endl;
            cout << "b = " << b << endl;
            cout << "*c = " << *c << endl;
            cout << "**d = " << **d << endl;
            cout << "e = " << e << endl;
      }
};
```

```
void main(){

      DataUnit A;
      DataUnit* B = new DataUnit();

      B->swap(A.a, A.b);
      B->printData();          //problem (3-1)

      A.reset();
      B->swap(&(A.a), &(A.b));
      B->printData();          //problem (3-2)

      A.reset();
      B->swap(A.a, A.c);
      A.printData();           //problem (3-3)

      A.reset();
      B->swap(A.a, *(A.c));
      A.printData();           //problem (3-4)

      A.reset();
      B->swap(A.a, &(A.c));
      A.printData();           //problem (3-5)

      A.reset();
      B->swap(&(A.a), *(A.c));
      A.printData();           //problem (3-6)

      A.reset();
      B->swap(&(A.a), &(A.c));
      A.printData();           //problem (3-7)

      A.reset();
      B->swap(A.d, A.e);
      A.printData();           //problem (3-8)

      A.reset();
      B->swap(*(A.d), A.e);
      A.printData();           //problem (3-9)

      A.reset();
      B->swap(**(A.d), &(A.e));
      A.printData();           //problem (3-10)
}
```

Please copy the following answer tables onto your answer sheet.

Problem	3-1	3-2	3-3	3-4	3-5
a =					
b =					
*c =					
**d =					
e =					

Problem	3-6	3-7	3-8	3-9	3-10
a =					
b =					
*c =					
**d =					
e =					

註 1：每小題 2 分。

註 2：各小題 (a, b, *c, **d, e) 答案全對者得 2 分。

註 3：各小題答錯者，(a, b, *c, **d, e)答案中有任一個(以上)錯誤者，倒扣該題 1 分。

Problem 4　[20%]　（本題答錯有倒扣）

4-A. [6%]　Start with a base class Car and its derived class Porsche below in C++, please give output of the statement (4-1) ~ (4-3) in the answer table. You may note "N/A" (not applicable) if the statement is not meaningful.

```
#include <iostream>
using namespace std;

class Car{
protected:
      int price;
      int design;
      int mspeed;

public:
      Car(int p, int d): price(p), design(d){ }

      void maxSpeed(){
            mspeed = price - design;
            cout << mspeed << endl;
      }
};

class Porsche : public Car{
public:
      Porsche(int p, int d): Car(p, d){ }

      void maxSpeed(){
            mspeed = price + design;
            cout << mspeed << endl;
      }
};
```

```
void main(){

      Porsche* pPorsche = new Porsche(100, 50);
      pPorsche->maxSpeed();                          // problem (4-1)

      Car* pCar = static_cast <Car*> (pPorsche);
      pCar->maxSpeed();                              // problem (4-2)
      delete pPorsche;

      pCar = new Porsche(100, 50);
      pCar->maxSpeed();                              // problem (4-3)
      delete pCar;

}
```

4-B. [8%] Define a base class Car with a virtual method and its derived class Porsche below in C++, please give output of the statement (4-4) ~ (4-7) in the answer table. You may note "N/A" (not applicable) if the statement is not meaningful.

```
#include <iostream>
using namespace std;

class Car{
protected:
        int price;
        int design;
        int mspeed;

public:
        Car(int p, int d): price(p), design(d){ }

        virtual void maxSpeed() = 0;
};

class Porsche : public Car{
public:
        Porsche(int p, int d): Car(p, d){ }

        void maxSpeed(){
                mspeed = price + design;
                cout << mspeed << endl;
        }
};
```

```
void main(){

        Car* pCar;
        Porsche* pPorsche;

        pCar = new Car(80, 20);
        pCar->maxSpeed();                       // problem (4-4)
        delete pCar;

        pPorsche = new Porsche(100, 50);
        pPorsche->maxSpeed();                   // problem (4-5)

        pCar = dynamic_cast <Car*> (pPorsche);
        pCar->maxSpeed();                       // problem (4-6)
        delete pPorsche;

        pCar = new Porsche(100, 50);
        pCar->maxSpeed();                       // problem (4-7)
        delete pCar;

}
```

注意：背面尚有試題

4-C. [6%] Suppose a base class Car, an abstract base class Engine, and their derived class Porsche are defined below in C++, please give output of the statement (4-8) ~ (4-10) in the answer table. You may note "N/A" (not applicable) if the statement is not meaningful.

```
#include <iostream>
using namespace std;

class Car{
protected:
      int price;
      int design;
      int mspeed;

public:
      Car(int p, int d): price(p), design(d){ }
};

class Engine{
public:
      virtual void maxSpeed() = 0;
};

class Porsche : public Car, public Engine{
public:
      Porsche(int p, int d): Car(p, d){}

      void maxSpeed(){
            mspeed = price + design;
            cout << mspeed << endl;
      }
};
```

```
void main(){

      Car* pCar = new Porsche(100, 50);
      pCar->maxSpeed();                          // problem (4-8)
      delete pCar;

      Engine* pEngine = new Porsche(100, 50);
      pEngine->maxSpeed();                       // problem (4-9)
      delete pEngine;

      Porsche* pPorsche = new Porsche(100, 50);
      pPorsche->maxSpeed();                      // problem (4-10)
      delete pPorsche;
}
```

Please copy the following answer tables onto your answer sheet.

Problem 4-A	4-1	4-2	4-3
Answer			

Problem 4-B	4-4	4-5	4-6	4-7
Answer				

Problem 4-C	4-8	4-9	4-10
Answer			

註：每小題 2 分。答對者得 2 分；答錯者倒扣 1 分。

Problem 5 [20%]

A 2-d tree is used to store two-dimensional point data. In a 2-d tree, each node has a certain record structure.

```
Nodetype = record
    INFO: infotype;   // It can be any kind of int, char or string data type.
    XVAL: integer;
    YVAL: integer;
    LLINK: ↑nodetype;
    RLINK: ↑nodetype;
End
```

The INFO field of a node in a 2-d tree can be any user-defined type whatsoever. The fields XVAL and YVAL denote the coordinates of a point associated with the node. The LLINK and RLINK fields point to two children. If *N* is a node in this tree then the *level* of node *N* is defined inductively as

$$level(N) = \begin{cases} 0 & \text{If N is the root of the tree} \\ \text{level(P)} + 1 & \text{If N's parent is P} \end{cases}$$

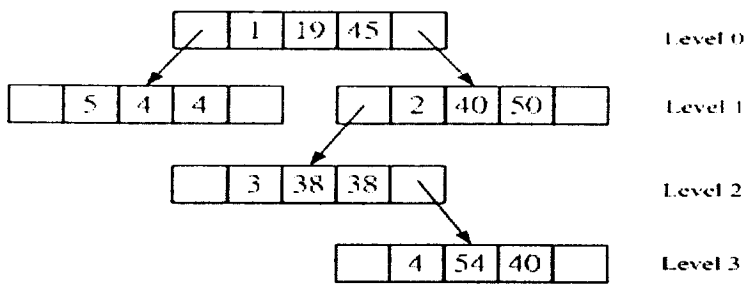
A 2-d tree is any binary tree satisfying the following conditions:

(1) If *N* is a node in the tree such that *level(N)* is **even**, then every node *M* in the subtree rooted at *N*.LLINK has the property that *M*.XVAL<*N*.XVAL, and every node *P* in the subtree rooted at *N*.RLINK has the property that *P*.XVAL≥*N*.XVAL.

(2) If *N* is a node in the tree such that *level(N)* is **odd**, then every node *M* in the subtree rooted at *N*.LLINK has the property that *M*.YVAL<*N*.YVAL, and every node *P* in the subtree rooted at *N*.RLINK has the property that *P*.YVAL≥*N*.YVAL.

For example, if we have a set of point data and related information, we should get a 2-d tree representation as following.

INFO,	(XVAL, YVAL)
1	(19, 45)
2	(40, 50)
3	(38, 38)
4	(54, 40)
5	(4, 4)



(5-A) Suppose *T* is a pointer to the root of a 2-d tree. Try to define an algorithm for inserting a node *N* into the 2-d tree pointed to by node *T* (10%).

(5-B) Try to implement this algorithm into a program by C or C++ code as detail as you can (10%).