Part I: Multiple Choice Questions (單選題) 共15題，每題2分，共30分。答錯不倒扣。

1. Which of the following are not included in the mail communications of mobile standard of mobile IP communication:
   A. Indirect routing of datagrams
   B. Register COA to the foreign agent
   C. Agent discovery
   D. Registration with the home agent
   E. All of the above are correct.

2. Which of the following protocol is not a stateless protocol:
   A. TCP
   B. UDP
   C. HTTP
   D. SOAP
   E. None of the above is correct.

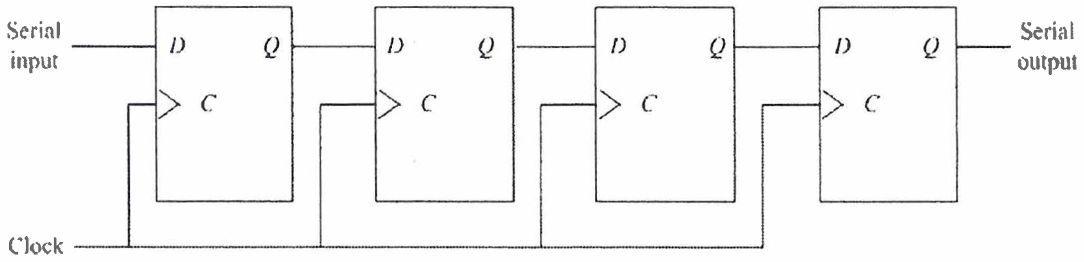
3. Which of the following addresses is more likely an IP address assigned by DHCP
   A. 59.124.164.10
   B. 140.114.131.1
   C. 192.168.1.120
   D. 129.89.38.174
   E. None of the above is correct.

4. Which of the following addresses in C++ is not true?
   A. When "++i * 7" is evaluated, i will be incremented before 7 is multiplied by its original value.
   B. An object defined in the ForInit section of a "for" loop is not visible outside that loop's body.
   C. The body of a "do-while" loop is always executed at least once.
   D. If a function only needs to access the value contained in an actual parameter, then it should be passed by reference to the function.
   E. All of the above are correct.

5. In the basic computer, there are four steps for computer instruction cycle: 1) Fetch an instruction from memory, 2) Execute the instruction, 3) Decode the instruction, and 4) Read the effective address from memory if the instruction has an indirect address. Which of the following sequences is the correct instruction cycle:
   A. 1234
   B. 4132
   C. 1342
   D. 4312
   E. None of the above is correct.

6. Which of the following is not true:
   A. ARQ (Automatic Repeat request) protocols are reliable data transfer protocols
   B. The Select-Repeat protocol receiver will acknowledge a correctly received packet whether of not it is in order.
   C. Because TCP only acknowledges bytes up to the first missing byte in the stream, TCP is said to provide cumulative acknowledgments.
   D. FTP uses a separate control connection and is an "in-band" protocol.
   E. All of the above are correct.

7. The content of the following 4-bit shift register is initially 1101. The register is shifted six times to the right with the serial input being 101101. What is the content of the register after each shift?



   A. 1011
   B. 1101
   C. 1110
   D. 0111
   E. None of the above is correct.

8. Which of the followings is **not** true?
   A. Using semaphores is one way to solve the critical-section problems in systems.
   B. Busy waiting may be one of the disadvantages for mutual-exclusion solutions.
   C. A deadlock may be caused by indefinite blocking, a situation where processes wait indefinitely with the semaphore. It can be solved by time-stamping method.
   D. With the assumption that only storage interlock is available, busy waiting may not be required for user-coded solutions to solve critical-section problems.
   E. All the above are correct

9. When a computer executes programs, if a process is spending more time paging than executing, this high paging activity is called
   A. Paging process
   B. Threshing
   C. Local replacement
   D. Segment exchange
   E. None of the above is correct.

10. Consider the following C++ program subroutine. Find the return value if n = 12.

```
long test( long n )
{   if ( n == 0 || n == 1 )  return n;
    else return test( n - 1 ) + test( n - 2 ); }
```

A. 66
B. 78
C. 89
D. 144
E. None of the above is correct.

11. Consider the following function declarations in the same program. What is this kind of declarations called:

```
int square( int x) {return x * x;}
float square(float x) { return x * x; }
```

A. function overloading
B. inline functions
C. function templates
D. virtual functions
E. None of the above is correct

12. Which of the following is True

A. Host A is sending host B a large file over a TCP connection. Assume host B has no data to send host A. Host B will not send acknowledgements to host A because host B cannot piggyback the acknowledgements on data.
B. The size of the TCP sliding-window size never changes throughout the duration of the connection.
C. Suppose host A is sending host B a large file over a TCP connection. The number of unacknowledged bytes that host A sends cannot exceed the size of the receiver buffer.
D. Suppose host A is sending host B a large file over a TCP connection. If the sequence number for a segment of this connection is *m*, then the sequence number for the subsequent segment will necessarily be $m+1$.
E. None of the above is correct.

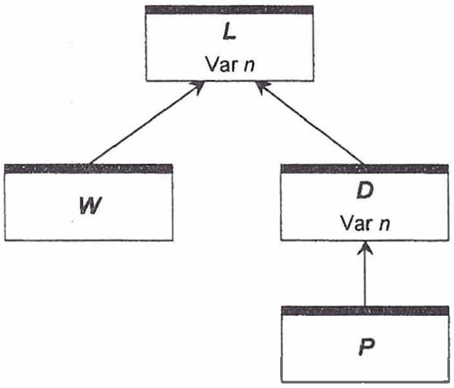
13. Suppose there are four routers between a source host and a destination host. Ignoring fragmentation, an IP segment sent from the source host to the destination host will travel over how many interfaces? How many forwarding tables will be indexed to move the datagram from the source to the destination?

A. 4 interfaces; 4 routing tables
B. 8 interfaces; 4 routing tables
C. 10 interfaces; 5 routing tables
D. 12 interfaces; 6 routing tables
E. None of the above is correct.

14. Which of the following descriptions about *inheritance* is **not** true

A. Every object of derived class is also an object of base class.
B. It creates new class from existing class.
C. Derived class can effect change to private base-class members through inherited non-private member functions.
D. Derived class typically represents larger set of objects than base classes
E. All of the above are correct.

15. Consider the following activation records in a program. The arrows between activation records represent control links. Which of the followings is true?

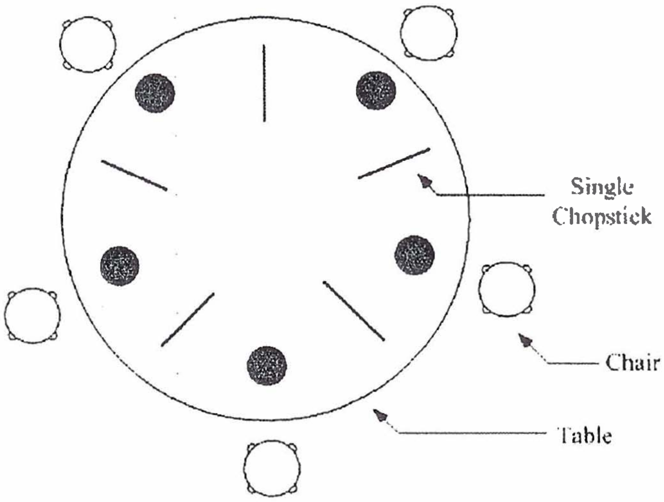


A. Changing the variable *n* in *D* also changes the variable n in *L*.
B. *D* calls *L*.
C. There are three procedures (functions) and one main program declaration in this program.
D. Variable *n* in *L* is a local variable to this program.
E. None of the above is correct.

Part II: Problem Solving/Essays Questions 5 大題，共 70 分。

1. Consider the dining-philosophers problem. There are 5 philosophers who spend their lives thinking and eating. The philosophers share a common circular table surrounded by five chairs, each belonging to one philosopher. In the center of the table there is a bowl of rice, and the table is laid with 5 single chopsticks. When a philosopher thinks, she does not interact with her colleagues. From time to time, a philosopher gets hungry and tries to pick up 2 chopsticks that are closest to her (the chopsticks that are between her and her left and her neighbors). A philosopher may pick up only one chopstick at a time.

   Now, here is the problem. Obviously, a philosopher cannot pick up a chopstick that is already in the hand of a neighbor. When a hungry philosopher has both her chopsticks at the same time, she eats without releasing her chopsticks. When she is finished eating, she puts down both of her chopsticks and starts thinking again. Use *semaphores* to write an algorithm to solve the dining-philosopher problem. **(15 points)**



2. Answer the following questions. **(10 points)**
   a. How big is the LAN addresses space? How big is the IPv4 address space? How big is the IPv6 address space?
   b. Compare the frame structures for 10BaseT, 100BaseT, 802.11b, and Gigabit Ethernet. How do they differ?

3. Consider the following complete C++ program. Trace and find the outputs of it. **(10 points)**

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

void foo(unsigned int num, vector<bool> &vec);
```

```
void fun(const vector<bool> &vec);
void bar(vector<bool> &vec);
void morefun(const vector<bool> &vec);

int main()
{
    const unsigned int Number = 57;
    vector<bool> BitVec;
    foo(Number, BitVec);

    cout << "After foo BitVec = ";
    fun(BitVec);
    cout << endl << endl;
    bar(BitVec);
    cout << "After bar BitVec = ";
    fun(BitVec);
    cout << endl << endl;
    cout << "The number " << Number << " equals ";
    morefun(BitVec);
    cout << " in binary." << endl << endl;
    return 0;
}

void foo(unsigned int num, vector<bool> &vec)
{
    while (num != 0)
    {
        bool bit = ((num % 2) == 1);
        vec.push_back(bit);
        num = num / 2; }
}

void fun(const vector<bool> &vec)
{
    for (unsigned int ind = 0; ind < vec.size(); ++ind)
    if (vec[ind])
        cout << '1';
    else
        cout << '0';
}

void bar(vector<bool> &vec)
{
    unsigned int size = vec.size();
    for (unsigned int ind = 0; ind < size / 2; ++ind)
    {
        bool bit = vec[ind];
        vec[ind] = vec[(size - 1) - ind];
        vec[(size - 1) - ind] = bit; }
}
```

```
void morefun(const vector<bool> &vec)
{
    typedef vector<bool>::const_iterator VecIt;
    const VecIt end = vec.end();

    for (VecIt it = vec.begin(); it != end; ++it)
        if (*it)
            cout << '1';
        else
            cout << '0';
}
```

4. Assume a multi-branched tree is a tree structure, in which each node can have an arbitrary number of children (Figure 1 illustrates an example of a multi-branched tree).

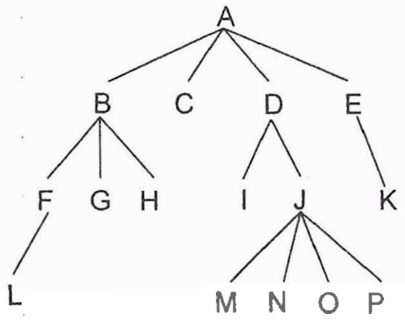


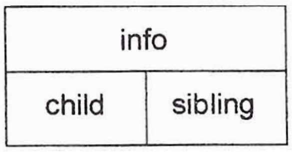
Figure 1: Example of A Multi-branched Tree

One way to represent multi-branched trees in C++ is with the following structure definition:

```
Struct mbNode {
    char info;
    mbNode *child, *sibling;
};
```

The child pointer from a node will point to the node's leftmost child, and the sibling pointer of a node will be used to point to the node's next sibling to the right.

If we diagram an **mbNode** like this:



the multi-branched tree shown in Figure 1 is shown in Figure 2.

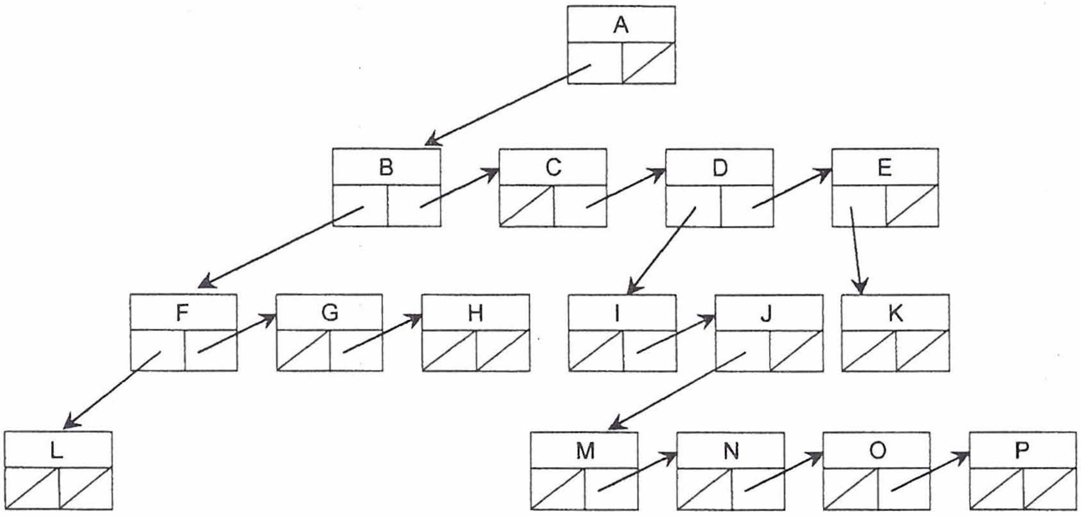


Figure 2: Multi-branched Tree for Figure 1

Here are some things to notice about this example. First, if the child pointer of a node is `null`, that means it has no children. Similarly, if the sibling pointer of a node is `null`, that means the node does not have any sibling next to its right. Second, because the root node has no siblings, its sibling pointer is always `null`.

Suppose we have a tree set up in this manner. It is easy to write a recursive traversal function (see the codes listed below) that visits each node in the tree and calls a client-supplied function on the `info` portion of that node.

```
void MapMBTreeRecursively(mbNode *tree, mbTreeMapFn fn)
{
     if (tree == NULL) return;
     fn(tree->info);
     if (tree->child != NULL) {
         MapMBTreeRecursively(tree->child, fn);
     }
     If (tree->sibling != NULL) {
         MapMBTreeRecursively(tree->sibling, fn);
     }
}
```

Please answer the following questions:

a. Trace the execution of a call to `MapMBTreeRecursively` on the tree shown in Figure 2 (i.e., the root node is A). Please write down the order in which the algorithm visits each node in Figure 2. For example, if you think that node A will be visited before node B, followed by node C, the traversal sequence will be {A, B, C, …}. **(8 points)**

b. Write a function `MapMBTreeIteratively(mbNode *tree, mbTreeMapFn fn)`, which does the same thing as `MapMBTreeRecursively` but does not use any recursion. In other words, `MapMBTreeIteratively` should visit the nodes in the same order that `MapMBTreeRecursively` visits the nodes. You can prepare your answer in C++ or in pseudocode. **(12 points)**

5. Assume that a data structure contains the `left` and `right` children pointers for a binary tree as well as contains a `next` pointer that points to the next cell in the linked list. The definition of such a structure is given below:

```
Struct TreeCell {
    char data;
    TreeCell *left, *right, *next;
};
```

In addition, assume that we are given a binary search tree comprised of `TreeCell` structures where the `left` and `right` children pointers are properly set to form a binary search tree, but the `next` field is uninitialized. An example of such a structure is illustrated in Figure 3.

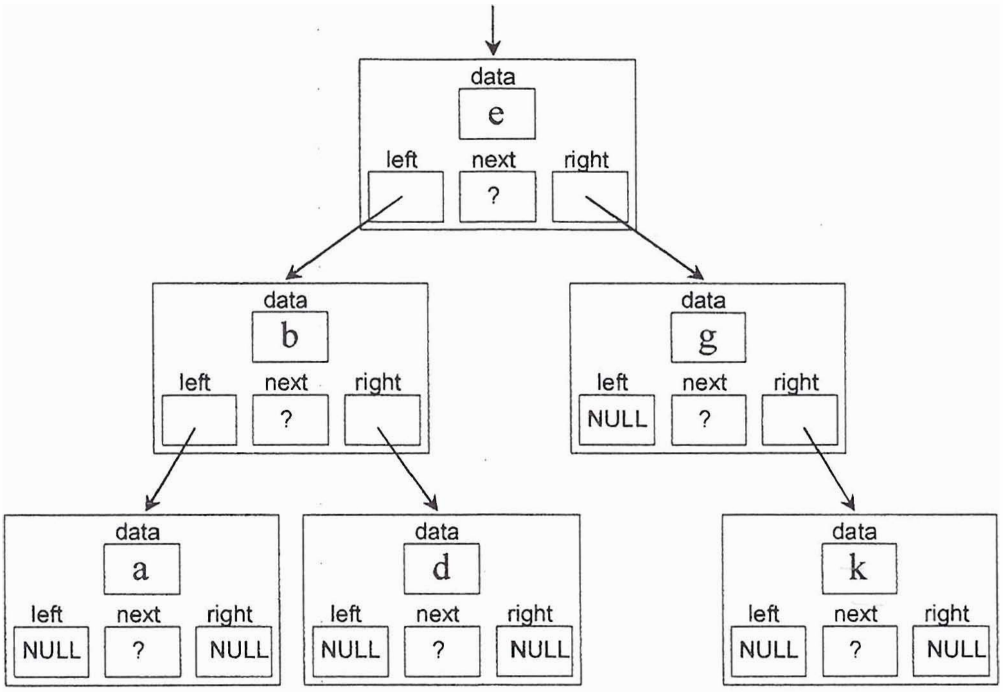


Figure 3: Example of Binary Search Tree with `next` Pointers Uninitialized

Now, we would like to have a function `SetNextLinks` that, given such a binary search tree, properly sets the `next` field in the cells of the tree so that if we follow these `next` links we would traverse the `TreeCells` in the sorted order. This function should return a pointer to the first cell in the "linked list" of `TreeCell` structures. For example, after calling the `SetNextLinks` function on the binary search tree shown in Figure 3, all next pointers are properly initialized and a pointer to the first cell (i.e., TreeCell structure with the data of "a") is returned (as shown in Figure 4).

Please write the function TreeCell *SetNextLinks(TreeCell *root). You can prepare your answer in C++ or in pseudocode. **(15 points)**

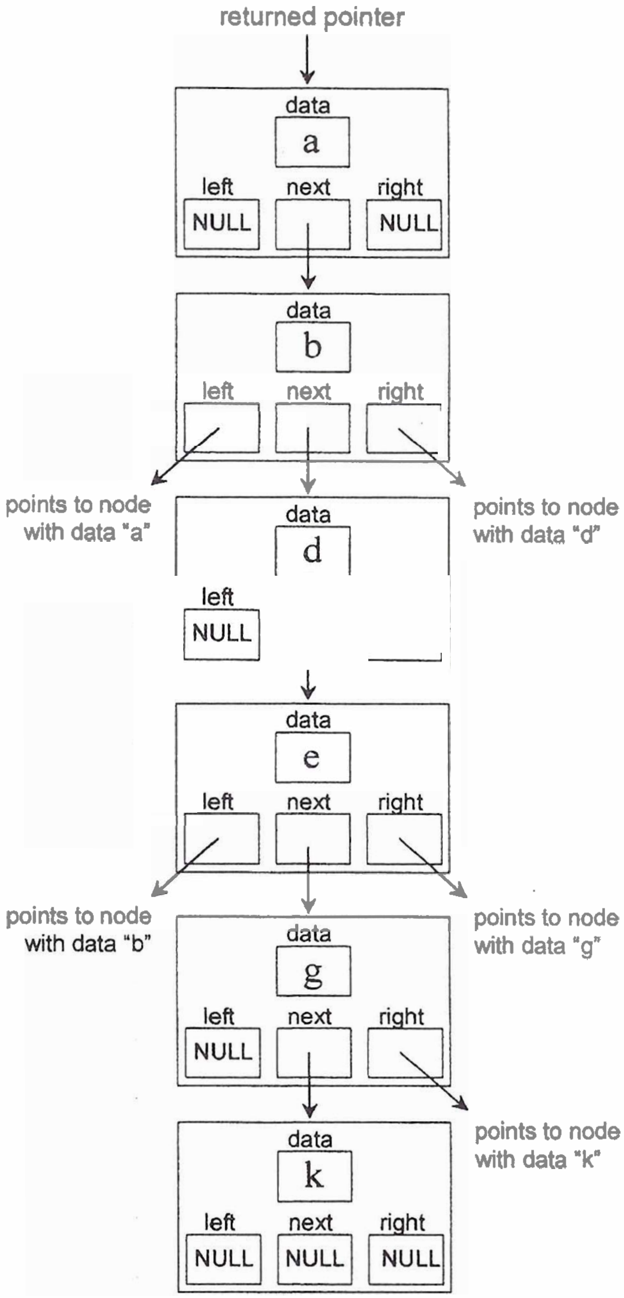


Figure 4: Result of SetNextLinks for Figure 3