國立臺灣師範大學 104 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，否則不予計分。

1. Given several documents where the title of each document consists of a sequence of words as follows:

   d1: "Set up your Apple ID for iCloud and iTunes"

   d2: "If your Apple ID has been locked"

   d3: "Apple Watch: Release date, price and specs"

   d4: "How to view iCloud Drive files on iPhone and iPad"

   You would like to write a program to support the full text retrieval function: i.e. by giving one or more words as a query, your program has to return the IDs of the documents whose titles contain all the words in the query.

   For example, if the query is "Apple", the program should output documents d1, d2, and d3.

   If the query is "iCloud" and "iPad", the program should output document d4.

   (a) (5 分) Please design a data structure to be the index structure such that the full text retrieval function can be performed efficiently. Please also show the constructed index structure for the given 4 documents.

   (b) (5 分) Please show the pseudo codes and explain how to perform the full text retrieval function according to the designed data structure efficiently.

2. Suppose you would like to construct a data structure for a dictionary of the following words to store the words and their page numbers:

   "able" (p12), "cat"(p10), "cook"(p1), "book"(p5), "look"(p8), "apple"(p3), "eat"(p12), "fruit"(p30), "it"(p23), "bike"(p15)

   Then your goal is to write a program to find the page number of a given word. If the given word is not in the dictionary, return "not found".

   (a) (6 分) Please show how to store the words in an array such that you can perform the binary search algorithm to find the page number of a given word. Please also show the pseudo codes of the binary search algorithm.

   (b) (4 分) Please describe the advantage and disadvantage of using a linked-list to maintain the words and their page numbers compared with using an array.

   (c) (5 分) Please design a data structure for the words such that all the words beginning with a given prefix can be found efficiently. For example, when you give a prefix "a", the system will find "able" and "apple". When you give "ab", the system will find "able".

3. Continued from the previous question.
   (a) (4 分) Please describe how to store the above words in a hash table with 5 buckets, where each bucket has 2 slots.
   (b) (6 分) Please explain the problems of "collision" and "overflow" when constructing a hash table. How to handle these problems?

4. Suppose that there are 6 users on a social networking site as follows.

   John, Mary, Tony, Andy, Paul, and Kelly

   The friendships between the users are as follows:

   (John, Mary), (John, Tony), (Paul, Kelly), (Tony, Paul), and (Mary, Andy).

   Note that the friendships are bi-directional. For example, according to the given friendship (John, Mary), John is a friend of Mary and Mary is a friend of John.

   (a) (3 分) Please design a data structure to maintain the friendships of the 6 users.
   (b) (4 分) Suppose there are N users in the social networking site. What is the time complexity to find the users who do not have any friend in the site? (i) If you use an adjacency matrix; (ii) if you use an adjacency list. Please explain your answer.
   (c) (5 分) Suppose a function DFS(v) is provided which can traversal the friendship graph in depth first search starting at node v and mark each visited node as "visited". Please give the pseudo codes to count the number of connected components in the friendship graph.
   (d) (3 分) In the DFS(v) function, please explain how to detect that the edges of the visited nodes starting at node v form a cycle?

5. Consider the following algorithm expressed in pseudocode:

```
void Comp (int n, const number A[][], const number B[][], number C[][])
{
    index i, j, k;
    for (i=1; j<= n; j++)
        for (j=1; j<= n; j++){
            C[i][j] = 0;
            for (k=1; k<= n; k++)
                C[i][j] = C[i][j] + A[i][k] * B[k][j];
        }
}
```

(a) (3 分) What does this algorithm compute?

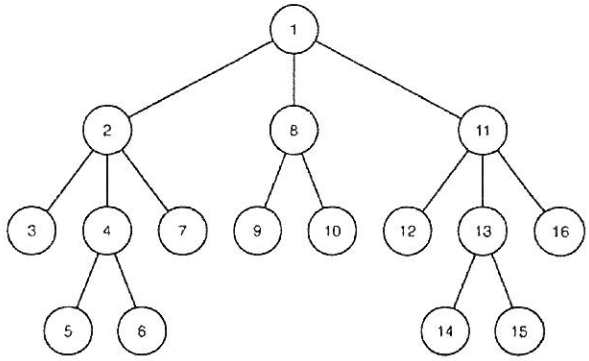
(b) (3 分) If n=2, $A = \begin{bmatrix} 3 & 1 \\ 2 & 5 \end{bmatrix}$, and $B = \begin{bmatrix} 4 & 7 \\ 8 & 6 \end{bmatrix}$, what is the value of C after running this algorithm?

(c) (3 分) What is the time complexity T(n) of this algorithm in terms of big-O notation?

(d) (3 分) Is the above algorithm a "polynomial-time algorithm"? Why?

(e) (3 分) In 1969, Strassen published an algorithm whose time complexity is better than the above algorithm. Strassen algorithm is categorized as "divide-and-conquer algorithm". Why?

6. The following figure shows a depth-first search of a tree. The nodes are numbered in the order in which they are visited. Notice that in a depth-first search a path is followed as deeply as possible until a dead end is reached. At a dead end we back up until we reach a node with an unvisited child, and then we again proceed to go as deep as possible.



The following psuedocode is a simple algorithm for doing a depth-first search. The procedure is called by passing the root at the top level.

```
void depth_first_tree_search (node v)
    {
        node u;
        visit v;
        for (each child u of v)
            depth_first-tree_search( _____ );
    }
```

(a) (3 分) There is a blank in the above algorithm. Please fill the blank.

(b) (3 分) If the number of nodes in the tree is n, what is the time complexity T(n) of this algorithm in terms of big-O notation?

(c) (3 分) Is the above algorithm an "inorder tree traversal algorithm"? Why?

(d) (3 分) Is the above algorithm a "recursive algorithm"? Why?

(e) (3 分) If we do a breadth-first search on the above tree, please draw a figure to number the nodes in the order in which they are visited.

7. Explain the following terms.

(a) (3 分) The 0-1 knapsack problem.

(b) (3 分) NP-complete.

(c) (3 分) Branch-and-bound algorithm

(d) (3 分) Euclid's algorithm

8. (8 分) Suppose you are given an array A of n elements containing only four distinct values 1, 2, 3, and 4. You want to sort the array in ascending order. Please design a linear-time algorithm to sort the array. Please also show the space complexity of your algorithm.