國立中正大學
109 學年度碩士班招生考試
試 題

［第 2 節］

科目名稱	資料結構
系所組別	資訊管理學系-乙組

—作答注意事項—

※作答前請先核對「試題」、「試卷」與「准考證」之系所組別、科目名稱是否相符。

1. 預備鈴響時即可入場，但至考試開始鈴響前，不得翻閱試題，並不得書寫、畫記、作答。

2. 考試開始鈴響時，即可開始作答；考試結束鈴響畢，應即停止作答。

3.入場後於考試開始 40 分鐘內不得離場。

4.全部答題均須在試卷（答案卷）作答區內完成。

5.試卷作答限用藍色或黑色筆（含鉛筆）書寫。

6. 試題須隨試卷繳還。

國立中正大學 109 學年度碩士班招生考試試題

科目名稱：資料結構

系所組別：資訊管理學系-乙組

1. Figure 1 is a social network of persons. (a) Please draw the adjacent array for the network; (b) Suppose the hub of a social network is defined as the one with the most links upon him/her. In this graph, the hub is Bob. Please write a subroutine (in any programming code) to find the hub. (10 points for each sub-question)

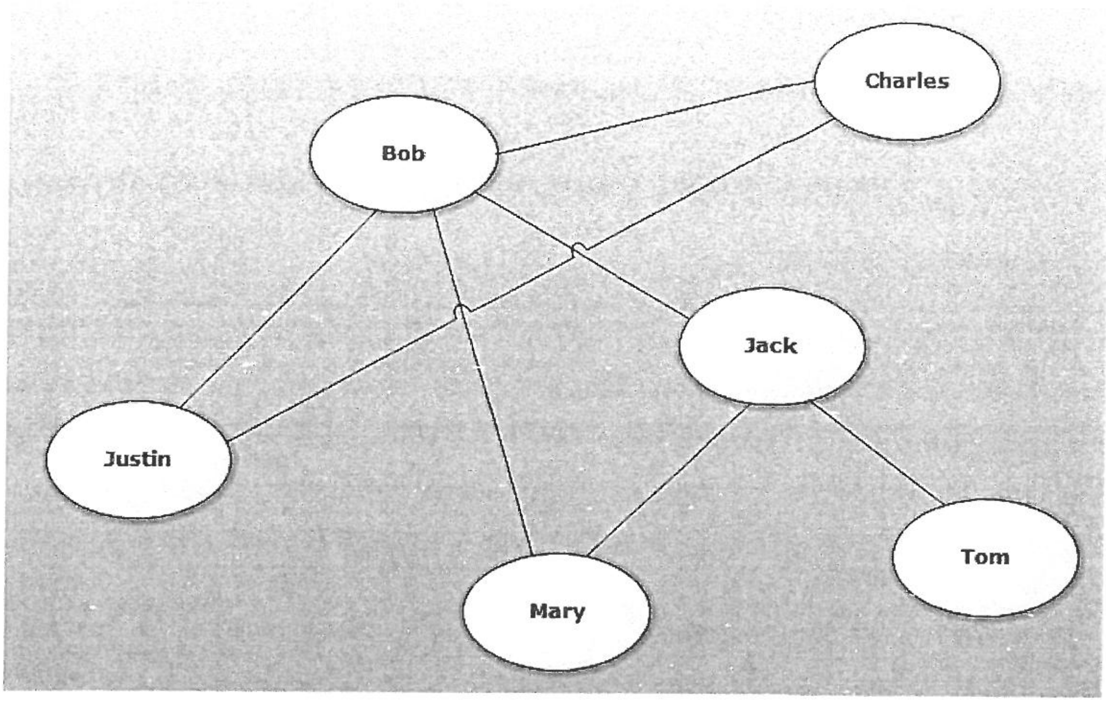
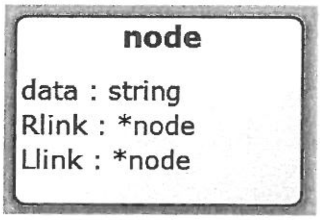


Figure 1.

2. Assume you use AVL tree to store the data and there come the six data 15, 4, 26, 1, 8 and 14 in the order and one by one. (a) Please draw the AVL tree from an empty AVL tree to insert the data into the tree (i.e., you need to draw six AVL trees to show the six insertions). (b) Suppose you delete the data 26, and 4 one by one from the tree. Please draw the AVL tree. Also, you need to draw two AVL tree to show the results of the two deletions. (10 points for each sub-question)

3. (a) Write a subroutine to count the number of nodes in a double circularly linked list. Suppose each node in the circular linked list has three fields, as follows:



node

data : string
Rlink : *node
Llink : *node

Figure 2.

In addition, there is a link variable, Top, pointing to a node in the circular linked list. (b) suppose you want to delete a node with data being 'CCU'. Please write the subroutine to do it. Note your program should consider the situation that the data with value being ‘CCU’ may or may not exist in the double circularly linked list. (10 points for each sub-question)

4. (a) Write a recursive function to add all the elements in a $n \times n$ (two dimensional) array of integers, n>= 1. For example, the result of $\begin{bmatrix} 3 & 8 \\ 9 & 13 \end{bmatrix}$ for the addition is 33. (b) Assume the addition of two matrixes is defined as the addition of the two elements in the same position of the two arrays. For example, $\begin{bmatrix} 3 & 8 \\ 9 & 13 \end{bmatrix}$ + $\begin{bmatrix} 10 & 3 \\ -2 & 1 \end{bmatrix}$ = $\begin{bmatrix} 13 & 11 \\ 7 & 14 \end{bmatrix}$. (Note: your answer must be written in a recursive function, otherwise, no points). (10 points for each sub-question)

5. Figures 3(a) and 3(b) are the depth-first and breath-first search algorithms, respectively. The two algorithms are annotated with numbers before each of their statements. Please write the execution sequence in these numbers step by step to travel the graph from vertex 3 in Figure 3(c). If more than one link can be traverse, the algorithm always chooses the link the smallest one in alphabetical order.

You only need to write the step numbers, like 1, 2, 3, 4, 3, 4, …., to answer the following two sub-questions. (a) Travel the graph in DFS manner; (b) Travel the graph in BFS manner. (10 points for each sub-question)

```
Algorithm DFS($G$, $v$):
  label $v$ as visited
  1 for all edge $e$ in $G$.incidentEdges($v$) do
    2 if edge $e$ is unvisited then
      3 $w \leftarrow G$.opposite($v$,$e$)
      4 if vertex $w$ is unexplored then
        5 label $e$ as a discovery edge
        6 recursively call DFS($G$,$w$)
      7 else
        8 label $e$ as a back edge
```

Figures 3(a)

```
Algorithm BFS($s$):
  1 initialize collection $L_0$ to contain vertex $s$
  2 $i \leftarrow 0$
  3 while $L_i$ is not empty do
    4 create collection $L_{i+1}$ to initially be empty
    5 for all vertex $v$ in $L_i$ do
      6 for all edge $e$ in $G$.incidentEdges($v$) do
        7 if edge $e$ is unexplored then
          8 $w \leftarrow G$.opposite($v$,$e$)
          9 if vertex $w$ is unexplored then
            10 label $e$ as a discovery edge
            11 insert $w$ into $L_{i+1}$
          12 else
            13 label $e$ as a cross edge
    14 $i \leftarrow i+1$
```

Figure 3(b)

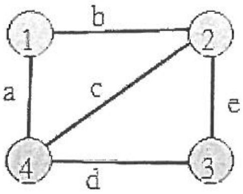


Figure 3(c)