所別：資訊管理學系碩士班 丙組 科目：資料結構

1. Using the **module-division** method and **linear probing**, store the keys shown below in an array with 17 elements. How many collisions occurred? What is the density of the list after all keys have been inserted? (10 points)

   426, 183, 902, 348, 724, 274, 198, 853, 603

2. (a) Construct a 2-3 tree for the list **C,O,M,P,U,T,E,R** (use the alphabetical order of the letters and insert them successively) (5 points)

   (b) Assuming that the probabilities of searching for each of the keys (i.e., the letters) are the same, find the largest number and the average number of key comparisons for successful searches in this tree. (10 points)

3. Construct an AVL tree by inserting **8, 9, 10, 2, 1, 5, 3, 6, 4, 7, 11,** and **12** successively. You should note the balance factor of each node and show all necessary rotations. (10 points)

4. (a) Find the transitive closure matrix A^+ and the reflexive transitive closure matrix A* of the graph G. in Fig. 1. (10 points)

   (b) Explain the meaning of the difference between A^+ and A*. (5 points)

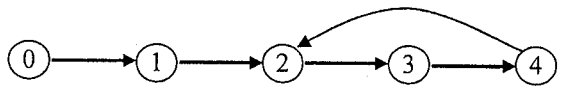


Fig. 1. Digraph G

注意：背面有試題

所別：資訊管理學系碩士班 丙組 科目：資料結構

5 (10 points) Write the postfix and prefix expression for the following infix expression. Assume these are the expressions in either Java or C. Please specify the language if you think it matters.

(1) *a/b-c+d*e-a*c* *(5 points)*

(2) *a*(b+c)/d-g* *(5 points)*

6 (25 points) Assume that we have information on the employees of a firm as the following table. For each employee, in addition to the employee's name, we have an occupational title, an ID number, and a location. We would like to be able to access quickly the information for any of the categories. For example, we might want to quickly retrieve the list of all employees who work in New York, or the list of all programmers. What is the data structure meet this requirement, as well as all of the following:

(1) allowing programmers to easily insert and retrieve any employee

(2) any employee data is not stored twice

(3) able to scale up to huge number of records

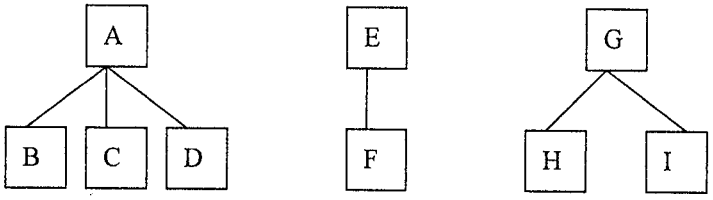
Please show your data structure in as example and explain how the *insert*, *delete*, and *search* operations should be done.

Node	ID Number	Name	Occupation	Location
A	20	John	Programmer	New York
B	30	Tom	Analyst	New York
C	50	Kevin	Architect	Taipei

7 (15 points) If $T_1, \ldots, Tn$ is a forest of trees, then the binary tree corresponding to this forest, denoted by $B(T_1, \ldots, Tn)$ is defined as follow:

- *is empty, if n=0*
- *has root equal to root (T_1); has left subtree equal to $B(T_{11}, T_{12}, \ldots, T_{1m})$, where $T_{11}, T_{12}, \ldots, T_{1m}$ are the subtrees of root (T_1); and has right subtree $B(T_2, \ldots, Tn)$.*

(1) Draw binary tree representation of the following forest (5 points)



(2) Define the inverse transformation of the one that creates the associated binary tree from a forest. Are these transformations unique? If not, use the example in (1) to show it.(10 points)