1. (20%) Please select the correct answer with EXPLANATION. Answers with wrong or without explanation will get at most 1 point.

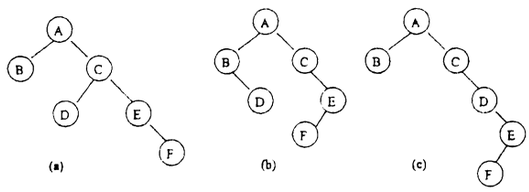
   (a) (5%) The maximum black height of the root in a Red Black tree of height n is (1) n (2) $\lfloor \frac{n}{2} \rfloor$ (3) $\lceil \frac{n}{2} \rceil$ (4) $n-1$ (5) None of above is correct.

   (b) (5%) Suppose a queue is implemented in a fixed array capable of holding 100 entries, using the method discussed in the textbook and in class. Suppose the queue is initially empty, and then objects are put into the queue at the rate of 10 per minute while meantime they are processed and removed from the queue at the rate of 5 per minute. After 120 elements have been added to the queue, which of the following is true? (1) 60 elements will be in the queue, 20 in the beginning, while 40 in the end of the array. (2) 60 elements will be in the queue, 40 in the beginning, while 20 in the end of the array.(3) 60 elements will be in the beginning of the queue, all in one contiguous segment of the array. (4) 60 elements will be in the end of the queue, all in one contiguous segment of the array. (5) None of above is correct, since you can't add 120 elements to an array holding 100 entries.

   (c) (5%) Let T be a full binary tree with root r where each node stores an integer key..Consider the algorithm below. Here $v.left$ and $v.right$ give the left and the right children of v, respectively.
   Algorithm TreeAlg(v)
   1. if v is external (i.e. a leaf) **then return** $2 \times v.key$;
   2. **else return** TreeAlg($v.left$)+TreeAlg($v.right$) - $v.key$;
   What does the algorithm return when called on the root r?
   (1) (the sum of all keys at internal nodes) - (twice the sum of all keys at external nodes)
   (2) (twice the sum of all keys at internal nodes)-(the sum of all keys at external nodes)
   (3) (twice the sum of all keys at internal nodes) - (twice the sum of all keys at external nodes)
   (4) (twice the sum of all keys at external nodes)-(the sum of all keys at internal nodes) (*)
   (5) None of above is correct

   (d) (5%) Let T be a binary search tree storing keys "A", "B", "C", "D", "F". Which of the following is a not a valid order for post-order traversal? (1) B,A,C,F,D (2) B,F,C,A,D (3) B,C,D,F,A (4) B,C,F,D,A (5) C,B,D,F,A

2. (9%) For each of the following three trees, either indicate how to color the nodes red and black (use R and B to label the nodes) to make the tree a red-black tree (in this case, You should explain the reasons for coloring **each** node), or explain why that is not possible. Note that the NIL leaf nodes are not shown in these pictures.



3. (21%) Given a connected undirected graph $G = (N, A)$ where N and A denote the set of the

(背面仍有題目,請繼續作答)

nodes and arcs in G, respectively. Let $|N| = n$ and $|A| = m$. Let c_{ij} be a positive integer that represents the length for arc $(i,j) \in A$, and $C_{\max} = \max_{(i,j)\in A}\{c_{ij}\}$. Let C be an $n \times n$ matrix where $C[i][j] = c_{ij}$ for each $(i,j) \in A$, $C[i][j] = M$ (a very large number) for each $(i,j) \notin A$. Each node $i \in N$ is associated with a distance label $d[i]$, which is set to be M in the beginning. Given an origin node s and destination node t, then $d[s] = 0$. The Dijkstra's algorithm iteratively conducts two operation: **NodeSelection**(Q), and **DistanceUpdate**(d, C). In particular, **NodeSelection**(Q) selects a node i in Q with the minimum distance label and moves i from Q to P; and **DistanceUpdate**(i, d, C) updates $d[j] = \min_{(i,j)\in A}\{d[j], d[i] + C[i][j]\}$.

Algorithm Dijkstra(P, Q, d, C)
1. **initialize**: read s, t; set $P = \emptyset$, $Q = N$ and $d[\cdot] = M$ except $d[s] = 0$;
2. **while** $t \in Q$ **do**
3. $\quad i =$**NodeSelection**(Q);
4. $\quad Q = Q - \{i\}$; $P = P \cup \{i\}$;
4. $\quad$ **DistanceUpdate**(i, d, C);
5. **end while**

The efficiency of the Dijkstra's algorithm thus depends on **NodeSelection**(Q) and **DistanceUpdate**(d, C). Initializing Q as an array of $|B|$ buckets, denoted by B, where $B[u]$ points to an array that stores the indices for those nodes with distance labels equal to u, a naive implementation of **NodeSelection**(Q) is as follows:

Procedure NodeSelection(Q),
1. **for** $u = 0$ to $|B|$ **do**
2. $\quad$ **if** $B[u] \neq NIL$ **then**
3. $\quad\quad$ remove the first node index i from $B[u]$;
4. $\quad\quad$ **return** i;
5. **end for**

In other words, **NodeSelection**(Q) will search for the first non-empty bucket $B[u^*]$, remove the first node v from $B[u^*]$ (i.e. from. Q) and return its index v. Answer the following questions. (Answers without explanation get at most 1 point.)

(a) (5%) In order to avoid the problem of memory overflow when we dynamically allocate memory for array B, please give a suitable estimate (i.e. upper bound) on $|B|$, the size of B, using the big-O notation with parameters n, m, or $C_{\max}$.

(b) (6%) Using the big-O notation with parameters n, m, or $|B|$ to estimate the OVERALL complexity on the number of operations executed by the procedure **NodeSelection**(Q) during the entire process of the Dijkstra's algorithm. (hint: how many iterations will **NodeSelection**(Q) be executed? Why?)

(c) (10%) Prof. Chen claims that he can improve the efficiency for the procedure **NodeSelection**(Q) by adding one more input i (initially set to be 0) that corresponds to the index of the node selected in previous call of NodeSelection:

Procedure NodeSelection(Q, i),
1. **for** $u = i$ to $|B|$ **do**
2. $\quad$ **if** $B[u] \neq NIL$ **then**
3. $\quad\quad$ remove the first node index i from $B[u]$;
4. $\quad\quad$ **return** i;
5. **end for**

Do you agree with Prof. Chen? If yes, explain why and estimate the OVERALL complexity on the number of operations executed by the procedure **NodeSelection**(Q, i) during the entire process of the Dijkstra's algorithm; Otherwise, explain why it will not work.

4. Assume we are going to store data in an N*N **symmetric** square matrix. An example is shown as Figure1(a). This matrix can be represented by $\left\lceil \frac{N*N}{2} \right\rceil$ nodes of the square (shown as the shadow part) due to they are symmetric. We therefore can store such a matrix to a self-defined complete binary tree (SDCBT) structure. In SDCBT, we **assume** that **tree nodes can be overlapped (i.e. an overlapped node is (x,y,I)=(1,1,6) in Figure 1(b))**and **stored at the same memory space** which can save spaces. In this tree, each node contains item I (a integer value), X value, Y value, a left link pointer L, and a right link pointer R (see Figure 1(b)). Suppose that the tree root is at matrix (x,y) = (0,0). According to the definition, the Figure 1(a) can be transfer to Figrue 1(b). Please write a pseudo code to describe how you read in the $\left\lceil \frac{N*N}{2} \right\rceil$ nodes data by **column major** and build a complete tree with the **minimum memory space**. (10%)

Y \ X					
	1	2	3	4	11
	5	6	7	12	10
	8	9	13	9	8
	10	12	7	6	5
	11	4	3	2	1

Figure 1(a)

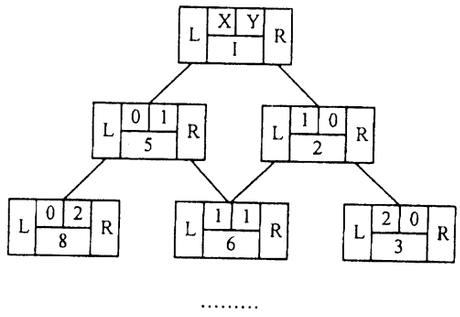


Figure 1(b)

5. Continue question 1. If we need to traversal the tree in-order and **each node can only be read once**, please write down this in-order traversal pseudo code for this self-defined binary tree. (10%)
6. Please discuss the pros and cons between array and linked list used in storing data. (9%)
7. In each machine, the meaning of an expression is to decide in what order the operations are carried out. For fixing the order of evaluation, it will first assign each operator a priority. If two adjacent operators have the same priority, operators of the same priority will be proceeded from left to right. If we define the operator ※ priorities is higher than #. The expression is : X=A※B#C#D※E#A※C. Please discuss which notation (infix, postfix, prefix) this expression is used? (2%) and how you transfer it into the other two notations. (6%) Which notation you will recommend to use for machines and **why**? Please explain by giving examples of this expression. (7%)
8. Please describe : 1)what is max heap and min heap, and 2) when is suitable to use this structure? (6%)