國立台灣科技大學九十七學年度碩士班招生試題

系所組別： 資訊工程系碩士班

科　　目： 資料結構

總分100分

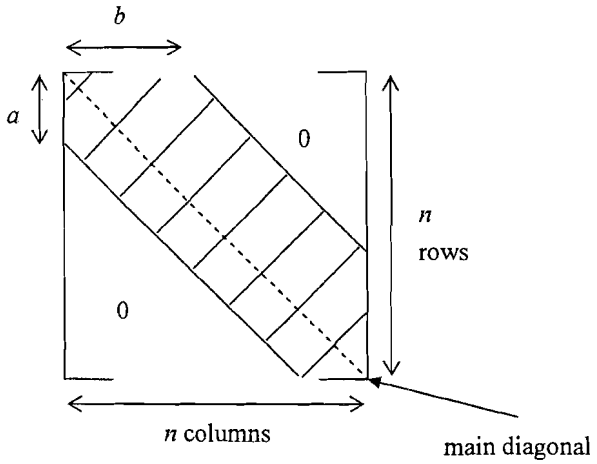
1. **[10%]** Consider the following expression:

   $$(A + B \times C)/D - E \times (F - G) + H$$

   (a) **[5%]** Please write the prefix form of the expression.

   (b) **[5%]** Please write the postfix form of the expression.

2. **[20%]** A generalized band matrix $A_{n,a,b}$ is an $n \times n$ matrix A in which all the nonzero terms lie in a band made up of $(a\text{-}1)$ diagonals below the main diagonal, the main diagonal, and $(b\text{-}1)$ diagonals above the main diagonal.



   (a) **[5%]** Consider the band of matrix $A_{100,40,60}$, how many elements are there in row 25?

   (b) **[5%]** How many elements are there in the band of $A_{100,40,60}$?

   (c) **[10%]** Assume that the band of $A_{n,a,b}$ is stored sequentially in an array B by diagonals starting with the lowermost diagonal. Thus, $A_{4,3,2}$ below would have the following representation:

$$\begin{bmatrix} 6 & 7 & 0 & 0 \\ 4 & 6 & 1 & 0 \\ 5 & 3 & 8 & 9 \\ 0 & 2 & 5 & 9 \end{bmatrix}$$

B[0]	B[1]	B[2]	B[3]	B[4]	B[5]	B[6]	B[7]	B[8]	B[9]	B[10]	B[11]
5	2	4	3	5	6	6	8	9	7	1	9
$a_{2,0}$	$a_{3,1}$	$a_{1,0}$	$a_{2,1}$	$a_{3,2}$	$a_{0,0}$	$a_{1,1}$	$a_{2,2}$	$a_{3,3}$	$a_{0,1}$	$a_{1,2}$	$a_{2,3}$

   Consider the matrix $A_{100,40,60}$ and suppose that $a_{60,50}$ is stored in $B[x]$, and $a_{20,40}$ is stored in $B[y]$. Please calculate x and y.

國立台灣科技大學九十七學年度碩士班招生試題

系所組別： 資訊工程系碩士班

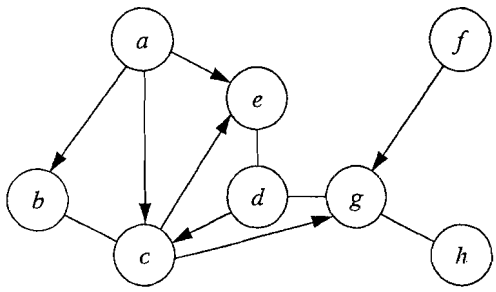
科　　目： 資料結構

3. [15%] Please answer the following questions related to binary search trees (BST). All keys in the BST are distinct.
   (a) [5%] Someone wants to search for an integer number x on a binary search tree, with the sequence of nodes 3, 253, 402, 399, 331, 345, 398 examined already, but still can not find it. Can you estimate the range of the number x?

   (b) [5%] How can we find the smallest element from a BST? Please simply describe your idea without giving the code. Note that from each node of the BST, we can find its left or right child (if there is one) through the `leftChild` or `rightChild` link respectively.

   (c) [5%] Similar to (b), but we want to find the second smallest element from the BST.

4. [25%] Given a hybrid graph G with a vertex a on the graph, please answer the following questions. Note that a hybrid graph G is a graph including both of directed and undirected edges. Also, having an undirected edge between vertices a and b is equivalent to having two directed edges from a to b and from b to a.
   (a) [5%] Suppose the graph is given as below



   starting from vertex a, please write down the visiting sequences by Depth-First Search and Breadth-First Search procedures respectively. During each step of visit, if two or more than two vertices are the possible candidates, you should always try the alphabetic order. For instance, from a, the vertex b instead of c or e should be visited first. The codes of the procedures are just included here for your reference. In the procedure, `V[G]` denotes all the

國立台灣科技大學九十七學年度碩士班招生試題

系所組別： 資訊工程系碩士班

科　　目： 資料結構

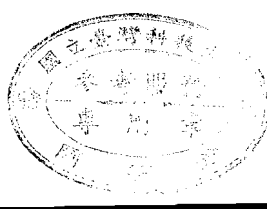
vertices in the graph G and Adj [u] records all the adjacent neighbors of u.

```
DFS(G)
1   for each vertex u∈V[G]
2       do tag[u] ← UNVISITED;
3   time ← 0;
4   for each vertex u∈V[G]
5       do if tag[u] = UNVISITED
6              then DFS-VISIT(u);

DFS-VISIT(u)
1   tag[u] = VISITED; output u;
2   time ← time+1; d[u] ← time
3   for each v ∈ Adj[u]
4       do if tag[v] = UNVISITED
5              then DFS-VISIT(v);
6   time ← time+1; f[u] ← time;

BFS(G)
1   for each vertex u∈V(G)
2       do tag[u] ← UNVISITED;
3   Q = ∅
4   for each vertex t∈V[G]
5       do if tag[t] = UNVISITED
6              then BFS-VISIT(t);

BFS-VISIT(t)
1   tag[t] = VISITED; output t;
2   ENQUEUE(Q,t);
3   while Q ≠ ∅
4       do u ← DEQUEUE(Q);
5          for each v ← Adj[u]
6              do if tag[v] = UNVISITED
7                     then tag[v] ← VISITED; output v;
8                          ENQUEUE(Q, v);
```

(b) [10%] Continue from (a) and starting from vertex *a* again, if now a visiting sequence S_d from DFS and a sequence S_b from BFS are given, can we uniquely define the graph? Please prove it if your answer is "yes" or disprove it by a counterexample if your answer is "no". Note that two graphs are called identical if they have the same adjacency matrix. Also, in the DFS or BFS, an alphabetic order should always be operated if there is more than one choice in each step of visit.

(c) [10%] Continue to previous questions, please write down the output of the following procedure SUB-A, which should include several vertex sets. (By outputting in the set form, the order of elements in each set is not important.)

```
SUB-A(G)
1    call DFS(G) to compute f[u] for each vertex u
2    compute H, where H is the graph with all directed
     edges in G reversed, and nothing will be changed
     for undirected edges
3    call DFS(H), but in the main loop of DFS (line 4),
     consider the vertices in the order of decreasing
     f[u], output the set of vertices collected in each loop of
     the DFS
```



國立台灣科技大學九十七學年度碩士班招生試題

系所組別： 資訊工程系碩士班

科　　目： 資料結構

5. [20%] The program below is developed for inserting a key into the hash table and finding a key from the hash table. Here, assume the hash table is already cleared, and the two modules, *hash_transA*(x, S) and *hash_transB*(x, S), are already implemented with different methods and can be invoked. The purpose of these two modules is for transforming a character string, x, into an integer N that satisfies $0 \le N \le S$-1.
   (a) [6%] For the hashing function, *hash_func*( ), give a double hashing based implementation, i.e. write out program statements.
   (b) [5%] When the load factor is 0.5, what is the expected value of the variable, j, just before returning from the module, *hash_insert*( )?
   (c) [9%] Write out the lost statements in the module, *hash_find*( ).

```
typedef struct {
    char Symbol[32];    char DeleteFlag;
} HashType;
int hash_transA(char *Key, int Size); //transform string into integer
int hash_transB(char *Key, int Size); //another transform
int hash_func(int Size, int count, int loc, int lod) //hashing function

int hash_insert( HashType *HashTab, int Size, char *Key )
{
    int j, k, na, nb;                //Size: hash table's size
    na = hash_transA(Key, Size);     //assume Size is a prime number
    nb = hash_transB(Key, Size-2); //assume Size-2 is also a prime
    for(j=0; j<Size; j++) {
        k = hash_func(Size, j, na, nb);
        if ( HashTab[k].Symbol[0]!=0 && HashTab[k].DeleteFlag==0) continue;
        strncpy(HashTab[k].Symbol, Key, 32); //string copy
        break;
    }
    if(j >= Size) return -1;    else return k;
}

int hash_find( HashType *HashTab, int Size, char *Key )
{
    int j, k, na, nb;
    na = hash_transA(Key, Size);    nb = hash_transB(Key, Size-2);
    for(j=0; j<Size; j++) {
        k = hash_func(Size, j, na, nb);
        ...
        return k; //found at location k
    }
    return -1;    //not found
}
```

6. [10%] The program module below is for quick sorting the elements of the array A between indices m and n. Suppose the partitioning module, *Partition*( ), always selects $A[m]$ as its pivot.
   (a) [5%] To prevent worst-case time complexity from occurring, give the statements that should be inserted to the position just before invoking *Partition*( ).
   (b) [5%] In the module, *QuickSort*( ), why insertion sort is invoked when k minus m or n minus k is less than 16?

```
void QuickSort(int *A, int m, int n)
{
    if(m >= n) return;
    int k;
    k = Partition(A, m, n);
    if(k-m >= 16) QuickSort(A, m, k-1);
    else          InsertionSort(A, m, k-1);
    if(n-k>=16)   QuickSort(A, k+1, n);
    else          InsertionSort(A, k+1, n);
}
```