國立臺灣海洋大學103學年度研究所碩士班招生考試試題

考試科目：基礎計算機科學（含資料結構、演算法）

系所名稱：資訊工程學系碩士班不分組

1.答案以橫式由左至右書寫。2.請依題號順序作答。

---

1. (15 %) A matrix with *m* rows and *n* columns is called an *m*×*n* matrix. An *mxn* matrix is stored in an array in row-major order. Fill the five blanks in the function *void matrix_multiplication(double *a,double *b,double*c,int m,int n,int k)* used to calculate the matrix product: **C=AxB**, where the sizes of matrices **A**, **B**, and **C** are *mxn*, *nxk*, and *mxk*, respectively, and matrices **A**, **B**, and **C** are stored in arrays *a*, *b*, and *c*, respectively. The element C_{ij} in the *i*th row and the *j*th column of matrix **C** can be obtained by the formula: $C_{ij} = \sum_u A_{iu}B_{uj}$.

```
void matrix_multiplication(double *a,double *b,double*c,int m,int n,int k)
{     int i,j;
      for(i = 0; i < m; ++i) {
            for(j = 0; j <k; ++j) {
                  double *aPtr = a + ___(1)___, *ePtr = aPtr+n;
                  double *bPtr = b + ___(2)___;
                  double *cPtr = c + ___(3)___;
                  *cPtr = 0;
                  for(; aPtr<ePtr; aPtr += ___(4)___, bPtr += ___(5)___) {
                        *cPtr +=*aPtr * *bPtr;
                  }
            }
      }
      return;
}
```

2. (15 %) Write a function *int range_count(struct binarytree*rootPtr,int a,int b)* which accepts a pointer *rootPtr* to the root of a binary search tree *T* and returns the number of nodes in *T* having key values between *a* and *b* ($a \le key \le b$). The node structure of the binary search tree is defined as

```
struct binarytree {
      int key;
      binarytree *left, *right;
};
```

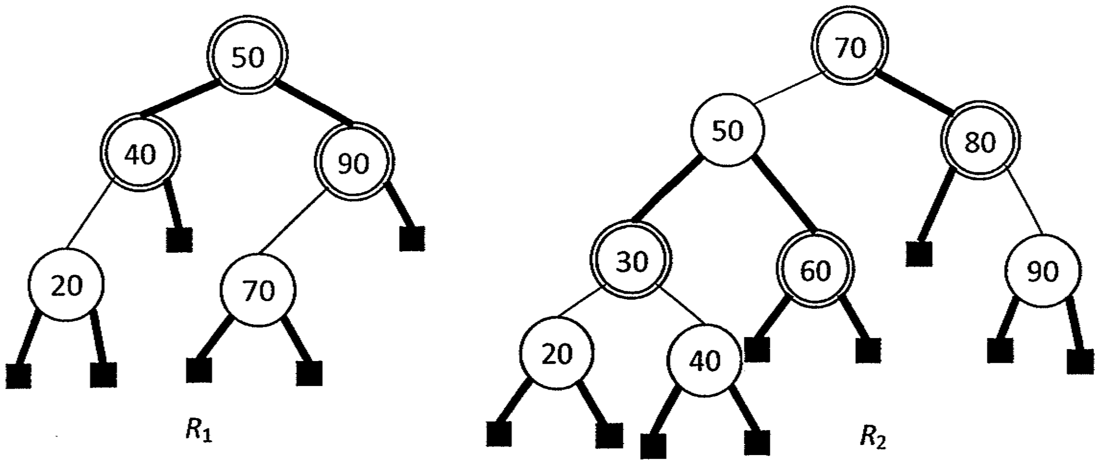
You may call other user-defined functions in the function *range_count*.

3. (10 %) Answer the following questions about the red-black trees R_1 and R_2.

(a) Show the red-black tree R_1 after insertion of 80.

(b) Show the red-black tree R_2 after deletion of 70.

Notice that the rebalancing rotation and color change are needed to keep the property of the red-black tree.



R_1 R_2

4. (10 %) There is a huge directed acyclic graph G. The in-degrees of the nodes in G are always less than three, and the out-degrees of the nodes in G are either zero or more than seven. If there is a directed path from node P to node Q, node P is defined as a predecessor of node Q. The adjacency lists and the inverse adjacency lists of G are both available. In order to determine if node P is a predecessor of node Q, we may use a graph search algorithm either to find a directed path from node P to node Q with the adjacency lists of G or to find a directed path from node Q to node P with the inverse adjacency lists of G. Give short answers to the following questions about determining if node P is a predecessor of node Q.

(a) Which of the following graph search algorithms is the better in terms of space complexity? Explain.

the depth-first search

versus

the breadth-first search

(b) If the breadth-first search is used, which of the following search directions is the better in terms of the worst-case time complexity? Explain.

searching from node P with the adjacency lists of G

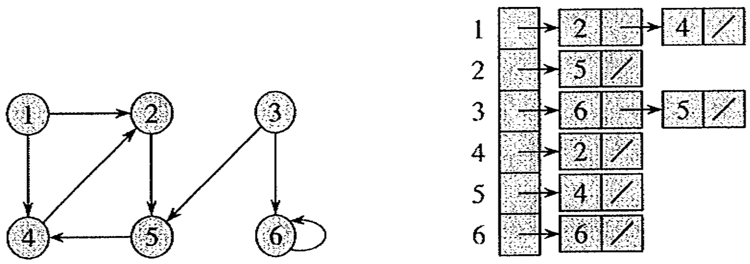
versus

searching from node Q with the inverse adjacency lists of G

5.(10%) Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n) = 2T(n/2) + \theta(n)$.

6.(15%) Describe briefly the Quicksort algorithm along with the time complexity. Show how Quicksort can be made to run in $O(n\lg n)$ time in the worst case.
Hint: Selection algorithm.

7.(10%) Illustrate the progresses of BFS and DFS, respectively, starting from vertex 3 on the following graph. Show the state of each phase.



(a) (b)

8.(15%) The single-source shortest paths problem can be solved by the Bellman-Ford algorithm.

(a) Find the shortest paths starting from vertex 1, going through all other vertices in the following graph by the Bellman-Ford algorithm and show the state of each phase.

(b) Describe briefly the Bellman-Ford algorithm along with the time complexity.