B505/I500: Applied Algorithms

HW2 (Due: **Sept. 25 Saturday, 11:59pm**)

1. (20 pts) Given the input of k sorted arrays, each containing N distinct integers in increasing order, devise an O(kN) algorithm to output the number of integers that occur in each of the k input arrays.

2. (20 pts) An array A[1..n] contains all integers from 0 to n except one number. It would be easy to determine to the missing number by using an auxiliary array B[0..n] to record which numbers appear in A. Here, we want to avoid the additional storage of B with the size O(n). Devise an algorithm to determine the missing integer in O(n) time under this constraint. (Note, you can still use additional constant memory as temporary storage; you should use only comparison operation, but not other operations such as additions and multiplications.)

3. (10 pts) Let A[1..n] be an array of *n* distinct numbers. If i < j and A[i] > A[j], then the pair (i, j) is called an inversion of A. a) List the five inversions of the array <2, 3, 8, 6, 1>. b) What array with elements from the set {1, 2, …, n} has the most inversions? How many does it have? c) What is the relationship between the running time of insertion sort and the number inversions in the input array? Justify your answer.

4. (20 pts) You are given two sorted lists of size m and n. Devise an O(log m + log n) time algorithm for computing the *k*th smallest element in the union of the two list.

5. (20 pts) Solve the following recursions with Big-O notation, assuming that $T(1)$ = constant:
   (a) $T(n) = T(n/2) + n$
   (b) $T(n) = T(n/5) + n^2$
   (c) $T(n) = T(n/3)$ + constant

6. (10 pts) Describe the sorting algorithm you would choose to sort an array *A* of *n* numbers in the following cases:
   a. *A* is nearly sorted;
   b. *A* consists of random numbers.

Explain the reasons for your choice.