B505/I500: Applied Algorithms

HW1 (Due: **Jan. 25 Friday 5pm**)

https://iu.instructure.com/courses/1771436

1. (10 pts) Illustrate the operation of Insertion-sort algorithm on array A=<31,41,59,26,41,58>.

2. (10 pts) The input to the algorithm *Unknown* illustrated below is an array A of N numbers. (1) what is the output of the algorithm? (2) using big-O notation to show the running time of the algorithm.

```
Input: Array A of N numbers;
Unknown(A)
for j = 1 to N-1
        if A[N] < A[j]
                exchange A[j] and A[N]
Output A[N];
```

3. (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.

4. (20 pts) An array A[1..n-1] contains all integers from 0 to n except two numbers. It would be easy to determine to the two missing numbers 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 two missing integers 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.)

5. (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.

6. (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.

7. (10 pts) Given an array of numbers as the input, devise an algorithm to generate a random permutation of the array, such that each number has equal probability to be placed in each position in the output array.