DESIGN & ANALYSIS OF ALGORITHM

(CS 617-01) (SP18)

Homework 1

Submitted by

Ashok Kumar Shrestha

Due Date: 30 Jan 2018

Problem 1: page 41, 2-4 (d)

2-4 Inversions

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.
d. Give an algorithm that determines the number of inversions in any permutation on n elements in Θ(n lg n) worst-case time. (Hint: Modify merge sort.)

```
Merge(A,B,C)
    n=0,i=0,j=0,k=0
    While i<B.length && j<C.length
        if(B[i]<C[j])
            A[k] = B[i]
            i = i+1
        else
            A[k] = C[j]
            j= j+1
            n = n + B.length - i
        k = k+1

    while i<B.length:
        A[k] = B[i]
        i = i+1

    while j<C.length:
        A[k] = B[j]
        j = j+1
    return n

Inversion(A)
    if (A.length == 1)
        return 0;
    else // A.length > 1
        B = A[1...n/2]
        C = A[n/2+1...n]
        num1 = Inversion(B)
        num2 = Inversion(C)
        num3 = merge(A,B,C)
        return num1+num2+num3
```

Problem 2: Rewrite Insertion Sort in recursive pseudo-code.

Call: Insertion_Sort(A,2,n)

```
//Pseudocode: Insertion Sort

Insertion_Sort(A, n)
        if (n ==1)
                return;
        else    // i<=n
                Insertion_Sort(A,n-1)
                Key = A[ n ]
                j = n - 1
                while ( j > 0 & A[ j ] > key)
                        A[ j+1 ] = A[ j ]
                        j = j - 1
                A[ j+1 ] = key
                return;
```