WIA2005 Algorithm Design & Analysis
Semester 2, 2016/17
Tutorial 1

---

1. Ilustrate the insertion sort operation on array A = $\langle 41, 51, 69, 36, 51, 68 \rangle$.

2. The following is the pseudocode for insertion sort procedure:

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

   Modify the algorithm to sort array into decreasing order.

3. Write a pseudocode for linear search for the following requirement:
   Input: A sequence of *n* numbers A = $\langle a_1, a_2, \ldots, a_n \rangle$ and a value *v*.
   Output: An index *i* such that *v* = A[*i*] or the special value NIL if *v* does not appear in A.

4. Express the function $n^3 / 1000 - 100n^2 - 100n + 3$ in terms of Θ-notation.

5. For the following pairs of functions, f(n) and g(n), detemine if they belong to Case 1: f(n) = O(g(n)) or Case 2: g(n) = O(f(n)). Formally justify your answer.

   a. $f(n) = 3n + 2$, $g(n) = n$
   b. $f(n) = (n^2 - n)/2$, $g(n) = 6n$
   c. $f(n) = n + 2\sqrt{n}$, $g(n) = n^2$
   d. $f(n) = n^2 + 3n + 4$, $g(n) = n^3$

6. For each of the following group of functions, sort the functions in increasing order of asymptotic (big-O) complexity:

   $f_1(n) = n^{0.999999 \log n}$
   $f_2(n) = 10000000n$
   $f_3(n) = 1.000001^n$
   $f_4(n) = n^2$

7. Given the iterative function below, calculate their time complexity analysis.
   a. `function1 (){`

```
        for (int i = 1; i <= n; i ++) {
            printf("Hello world");
        }
    }
```

b.
```
function2(){
        for (int i = 1; i <=n; i ++) {
                for (int j = 1; j <=n; j ++) {
                    printf("Hello world");
          }
      }
```

c.
```
function3 (){
        for (int i = 1; i^2 <= n; i ++) {
            printf("Hello world");
        }
    }
```

d.
```
function4 (){
        for (int i = 1; i <= n; i = i*2) {
            printf("Hello world");
        }
    }
```

e.
```
function3(){
        for (int i = n/2; i <=n; i ++) {
            for (int j <= n/2; j <=n; j = 2*j) {
                for (int k = 1; k <= n; k*2) {
                    printf("Hello world");
                    }
                }
            }
        }
    }
```

8. a) Using the substitution method, find the time complexity of a recursive program with the following recurrence relation:

$T(n) = n + T(n-1); n>1$
$= 1 ; n= 1$

b) Outline the time analysis of the following recursive programs using recursion tree method for

i) $T(n) = 2T(n/2) + n^2$; where n>1

ii) $T(n) = T(n/3) + T(2n/3) + n$; where n>1

9. Using the master methods, solve the following recurrences:
    a. $T(n) = 2T(n/4) + 1$
    b. $T(n) = 2T(n/4) + \sqrt{n}$
    c. $T(n) = 2T(n/4) + n$
    d. $T(n) = 2T(n/4) + n^2$