Chairiel Azizi bin Khyrudin
WIF190007 @ 17110481/2

WIA2005 Algorithm Design &Analysis
Semester 2
Tutorial 1

---

1. The following is an insertion sort algorithm.

```
  def InsertionSort(A):
1   for j in range(1,len(A),1):
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
```

   Illustrate the insertion sort operation on array A = 41, 51, 69, 36, 51, 68.

2. Modify the insertion sort algorithm to sort array into decreasing order.

```
def insertionSort(a):
    for j in range(1, len(a), 1):
        key = a[j]
        i = j - 1
        while (i >= 0 and a[i] < key):
            a[i + 1] = a[i]
            i = i - 1
        a[i + 1] = key
    return a

arr = [41, 51, 69, 36, 51, 68]
data = insertionSort(arr)
print(data)
```

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.

```
Get input for array A
Get input for value v

for i=0 in range(0,length of A):
        if v equal to A[I]:
                print(v)
        else if(v not equal to A[i]:
                print("NIL"
```

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

$= 1<n<n^2<n^3$
$= f(n) = \Theta(n^3)$

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

   a. $f(n) = 3n + 2$, $g(n) = n$

Case 1: True,
Case 2: True
Because both are O(n),the order of term of f(n) and g(n) are the same.

   b. $f(n) = (n^2 - n)/2$, $g(n) = 6n$

Case 1: False
Case 2: True
Because f(n) has a higher order term than g(n)

   c. $f(n) = n+2\sqrt{n}$, $g(n) = n^2$

$1 < \log n < \sqrt{n} < n < n \log n < n^2 < n^3 < \ldots. 2^n < 3^n < n^n$
Case 1: True
Case 2: False
Because g(n) has a higher order term than f(n)

   d. $f(n) = n^2 + 3n + 4$, $g(n) = n^3$

Case 1: True
Case 2: False
Because g(n) has a higher order term than f(n)

6. Given the iterative function below (in Java), calculate their time complexity.

```
a. function1 (){
        for (int i = 1; i <= n; i ++) {
             printf("Hello world");
```

```
        }
    }
```

O(n)

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

O(n^2)

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

O($n^{1/2}$)

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

O(log n)

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

first loop = n
second loop = log n
third loop = log n

O(n $\log^2$ n)

References:

Analysis of Algorithms | Set 4 (Analysis of Loops) - GeeksforGeeks

Time Complexity of a Loop when Loop variable “Expands or Shrinks” exponentially - GeeksforGeeks

Analysis of Algorithms | Set 5 (Practice Problems) - GeeksforGeeks