331 Worksheet: Efficiency Analysis of Nonrecursive Algorithms

1. Given an integer n and a message, write the message n times.

```
Write(n, message) : n is a positive integer and message is a string.
        for i = 1 to n:
                print(message)
```

Θ(n)

2. Given n numbers, tell whether or not there is a duplicate in the set.

```
Duplicate(A[1..n]): A is an array of n numbers
        for i = 1 to n:
                for j = i+1 to n:
                        if A[i] = A[j] return false
        return true
```

$\Theta(n^2)$

3. Given a set of n student records, find the student with the most semester hours completed.

```
MostSenior(A[1..n]):  A is an array of records
        most <- A[1]
        for i = 2 to n:
                if A[i] has more hours than most:
                        most <- A[i]
        return most
```

Θ(n)

4. (I changed this a bit from the worksheet. This is what I MEANT.)
Given an integer n, print its digits separately in reverse. For example, if n = 742, I would print
2
4
7

```
BackPrint(n): n is a positive integer
        while n > 0:
                print( n mod 10)
                n = n / 10
```

Θ(logn)

5. Assume that FunctionX(n) takes time $\Theta(n)$ and FunctionY(n) takes time $\Theta(n^2)$, and FunctionZ(n) takes time $\Theta(\lg n)$.
What is the running time of these Functions?

a) FunctionA(n):

```
for i<- 1 to n:
        FunctionX(n)
```

$\Theta(n^2)$

b) FunctionB(n):

```
FunctionX(n)
for i<- 1 to n:
        FunctionZ(n)
```

$\Theta(n + n\lg n) = \Theta(n\lg n)$

c) FunctionC(n):

```
FunctionY(n)
for i<- 1 to n:
        FunctionX(n)
```

$\Theta(n^2 + n^2) = \Theta(n^2)$

d) FunctionD(n):

```
for i<- 1 to n:
        FunctionZ(n)
        FunctionY(n)
```

$\Theta(n(\lg n + n^2) = \Theta(n\lg n + n^3) = \Theta(n^3)$