# DIT960 – Lab 1-Complexity

1. T(n) calculated for .

T(n) calculated for n.

T(n) calculated for .

T(n) calculated for *.*

T(n) calculated for *.*

2.

Ex1:

* Loop occurs n times
* For each i, sum is called 1 time
* Total is: T(n) = n  
   O(n)

Ex2: (ex2 is excluded du to being an example for the assignment)

* Outer loop runs n times.
* For each value of i, inner loop runs n-1 times (from 1 to n-1).
* Total is: n(n-1), O(n2).

Ex3:

* Loop [i] n times
* For each i, sum is called 1 time
* Loop [j] n times
* For each j, sum is called 1 time (total n-1)
* Total is: T(n) = 2n-1  
   O(n)

Ex4:

* Outer loop (n-1)/2 times
* Inner loop n\*2 times
* Sum is called n^2 times for each i
* Total is: T(n) = n^2(n-1) /2  
   O(n^2 log n)

Ex5:

* Outer loop n times
* Inner loop n times
* Sum is called n times for each i
* Outer loop [k] n times
* Sum called n times
* Total is: T(n) = n(n-1)/2 + n

O(n^2)

Ex6:

T(n)=1+T(n-1) will go n times until the base case occurs and add 1 each time, aka, 1+1+1… n times.  
Complexity O(n)

Ex7:

T(n)=n+T(n-1) will go n times until base case occurs and add n each time, aka, n+n+n… n times.  
Complexity O()

Ex8:

T(n)=1+T(n/2) will go n times until base case occurs and add n each time, aka, 1+1+1… (n/2) times.  
Complexity O(

Ex9:

T(n)=1+T(n/2) will go n times until base case occurs and add n each time, aka, n+n+n… (n/2) times.  
Complexity O(

Ex10:

T(n)=T(n-1)+T(n/2) will go n times until base case occurs and add (n-1) twice each time, this creates a tree that branches 2 times after each stage as such it has leafs. Complexity O()

PS. In Ex10, since the base case is T(0)=0 every n will result in a sum of n amounts of zeros, T(0)=1 is a more suitable base case because then we actually add something that does not result in 0.