CSC 225 (Fall 2016): Algorithms and Data Structures I
Assignment 1—theoretical part

Question 1. In pseudo code, describe **Algorithm** ComputeAverage(A,n) that returns, for an input of an array A with n elements, the average value of these n elements.

Question 2 & 3. Consider the following algorithm described in pseudo code.

```
Algorithm Compute(n)
Input: positive integer n
Output: sum of all integers from 1 to n

value ← 0

for i ← 0 to n-1 do
   value ← value + (i + 1)
end
return value
```

a) Determine the number of primitive operations of Algorithm Compute(n), counting assignments, comparisons, and returns only.

b) In pseudo code, describe Algorithm RecursiveCompute(n), an algorithm that outputs the sum of all integers from 1 to n as *recursive* algorithm.

c) Determine the number of primitive operations of Algorithm RecursiveCompute(n), counting assignments, comparisons, and returns only.

d) In pseudo code, describe Algorithm ComputeFast(n), an algorithm that outputs the sum of all integers from 1 to n in constant time.

Question 4. For the following, $\lg n = \log_2 n$

a) Order the following functions by their growth rate (big-Oh).

$f_0(n) = \lg n$ $\quad f_1(n) = 17n + \sqrt{n}$ $\quad f_2(n) = n^3 + 882$ $\quad f_3(n) = 112 \cdot 3^n$

$f_4(n) = n \lg n^3$ $\quad f_5(n) = n^3 \lg n$ $\quad f_6(n) = 28n^2 + 3$ $\quad f_7(n) = 1^n$

b) Which of the following functions are big-theta of one another?

$g_0(n) = 17 \cdot 2^{\lg n}$ $\quad g_1(n) = 1032$ $\quad g_2(n) = n \lg n$ $\quad g_3(n) = n \lg n + \sqrt{n}$

$g_4(n) = n^{\frac{1}{2}} + 81$ $\quad g_5(n) = 881n$ $\quad g_6(n) = n^2 + 13$ $\quad g_7(n) = 1^n$

.

Question 5. Consider the following algorithm described in pseudo code.

.

Algorithm arrayFind(x, A)
Input: An element x and an n-element array A
Output: The index i such that $x = A[i]$ or -1 if no element of A is equal to x.
$i \leftarrow 0$
while $i < n$ **do**
  if $x = A[i]$ **then return** i
  else $i \leftarrow i+1$
end
return -1

Counting assignments, comparisons, and returns only, calculate the worst-case, $T(n)$, and best-case, $T_b(n)$, running times of Algorithm arrayFind.