

COMP3506/COMP7505—Algorithms and Data Structures

School of Information Technology and Electrical Engineering

Week 3 Tutorial

Question 1.

The logarithm of a number to a given base is the power or exponent to which the base must be raised in order to produce the number. (<http://en.wikipedia.org/wiki/Logarithm>)

$$\text{if } x = b^y, \text{ then } y = \log_b(x)$$

Determine the logarithms of the following numbers to the base 10.

- (a) $\log_{10}(1000)$
- (b) $\log_{10}(100)$
- (c) $\log_{10}(10)$
- (d) $\log_{10}(1)$

Determine the logarithms of the following numbers to the base 2

- (e) $\log_2(1)$
- (f) $\log_2(2)$
- (g) $\log_2(4)$
- (h) $\log_2(8)$
- (i) $\log_2(16)$

Question 2.

Analyse the running time of the following function for arbitrary values of the input parameter n .

Algorithm: PrintIntegers(n)

Input: An integer giving the number of integers to print

Output: Printed integers, no return value

```
i = 0
while i < n do
    i += 1
    println i
return
```

Question 3.

Analyse the running time of the Algorithm BinarySum (below) for arbitrary values of the input parameter n .

Algorithm: BinarySum(A, i, n)

Input: An array A and integers i and n .

Output: The sum of the n integers in A starting at index i .

if $n = 1$ **then**

return $A[i]$

return BinarySum($A, i, \lceil n/2 \rceil$) + BinarySum($A, i + \lceil n/2 \rceil, \lfloor n/2 \rfloor$)

Question 4.

Order the following functions by asymptotic growth rate:

$4n \log n + 2n$	2^{10}	$2^{\log n}$
$3n + 100 \log n$	$4n$	2^n
$n^2 + 10n$	n^3	$n \log n$

Question 5.

There is an algorithm, find2D, to find an element x in an $n \times n$ array A .

The algorithm find2D iterates over the rows of A , and calls the algorithm arrayFind (below) on each row, until x is found or it has searched all rows of A .

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

$i \leftarrow i + 1$

return -1

- What is the worst case running time of find2D in terms of n ?
- What is the worst case running time of find2D in terms of N , where N is the total size of A ?
- Would it be correct to say that find2D is a linear-time algorithm? Why or why not?