MAT2440, Classwork34, Spring2025

ID:______________________________ Name:______________________________

1. Input (size n) ⟶ Find the number of operations $f(n)$ ⟶ $f(n) = $ O(g(n))

measurement of time complexity

2. Commonly Used Terminology for the Complexity of Algorithms.

Complexity	Terminology
$O(1)$	constant complexity
$O(\log n)$	Logarithmic complexity
$O(n)$	Linear complexity
$O(n \cdot \log n)$	Linearithmic complexity
$O(n^p), p > 0$	Polynomial complexity
$O(b^n), b > 1$	Exponential complexity
$O(n!)$	Factorial complexity

3.

TABLE 2 The Computer Time Used by Algorithms.

Problem Size	*Bit Operations Used*					
n	$\log n$	n	$n \log n$	n^2	2^n	$n!$
10	3×10^{-11} s	10^{-10} s	3×10^{-10} s	10^{-9} s	10^{-8} s	3×10^{-7} s
10^2	7×10^{-11} s	10^{-9} s	7×10^{-9} s	10^{-7} s	4×10^{11} yr	*
10^3	1.0×10^{-10} s	10^{-8} s	1×10^{-7} s	10^{-5} s	*	*
10^4	1.3×10^{-10} s	10^{-7} s	1×10^{-6} s	10^{-3} s	*	*
10^5	1.7×10^{-10} s	10^{-6} s	2×10^{-5} s	0.1 s	*	*
10^6	2×10^{-10} s	10^{-5} s	2×10^{-4} s	0.17 min	*	*

Time needed less ⟶ more

* means time of more than 10^{100} years

4. The Worst-Case Complexity.

The worst-case performance of an algorithm means the largest number of operations needed to solve the given problem using this algorithm on input of specified size. This worst-case analysis tells us how many operations an algorithm requires to guarantee that it will produce a solution.

5. The time complexity analysis of the **linear search** algorithm:

procedure *linear_search*(x: integer, $a_1, a_2, \cdots, a_n$: distinct integers)
$n :=$ the length of $\{a_i\}$
$i := 1$

while ($i \leq n$ and $x \neq a_i$)
    $i := i + 1$
if $i \leq n$ **then** *location* $:= i$
    else *location* $:= 0$
return *location* { *location* is the subscript of the term that equals x, or 0 if x is not found.}

(1) The number of operations considering the **worst-case**: x is not in the list

while ($i \leq n$ and $x \neq a_i$)	← 2	comparison	$\Big\}$ n times ($i = 1$ to n)) $3n$
$i := i + 1$	← 1	addition	
if $i \leq n$ **then** *location* $:= i$ **else** *location* $:= 0$	← 1	comparison	) $+1$

(2) The time complexity is $f(n) =$ $3n+1$ $= O($ n $)$ which is linear complexity.

(3) What is the best-case complexity of this algorithm?

Sol: best case: "$x = a_1$"

then the operations are: while loop $1 \leq n$ and $x = a_1$ ② comp.
if loop $1 \leq n$ ① comp.

3 operations

$f(n) = 3 = O(1)$ ← constant complexity.