① To solve the given problem, we design a 2-tape Turing machine which in principle works according to the following pseudocode:

```
1. WHILE character on the 1st tape ≠ Δ:
2.   | copy character from the 1st tape to the 2nd tape
3.   | move both heads to the right                          } O(n)
4. write # to the 1st tape                                   } O(1)
5. move the head of the 2nd tape to the left
6. IF character on the 2nd tape ≠ Δ:
7.   | GOTO 5                                                } O(n)
8. move both heads to the right
9. IF character on the 2nd tape = Δ:
10.  | RETURN SUCCESS
11. copy character from the 2nd tape to the 1st tape         } O(n)
12. GOTO 8
```

TIME COMPLEXITY:

The proposed algorithm consists of 3 subsequent loops going over the whole input w and a single step that writes the # symbol to the output. In total the time complexity is $3n+1$ which belongs to $O(n)$ class.

SPACE COMPLEXITY:

The state of the 2 tapes at the end of the algorithm computation is always 1) $\Delta w \# w^\omega$ 2) $\Delta w \Delta^\omega$ which requires $3n+1$ memory cells and therefore belongs to the $O(n)$ class.

(2) The simplest division algorithm based on finding the greatest common divisor can be implemented into a RAM program as follows:

pseudocode that computes $N/D = (Q, R)$:

```
R := N
Q := 0
WHILE R ≥ D DO:
    R := R - D
    Q := Q + 1
RETURN (Q, R)
```

```
1. READ 1      } R1: R = I_1 = N
2. STORE 1     }
3. READ 2      } R2: D = I_2
4. STORE 2     }
5. SUB 0       } R3: Q = 0
6. STORE 3     }
7. LOAD 1      } R < D: jump to halt
8. SUB 2       }
9. JNEG =15    }
10. STORE 1    } R := R - D
11. LOAD 3     } Q := Q + 1
12. ADD =1     }
13. STORE 3    }
14. JUMP =7    } loop back to check condition
15. LOAD 3     } R0: Q accumulated
16. HALT       }
```

UNIFORM TIME COMPLEXITY:

- lines 1-6: $O(1)$
- lines 7-14: the loop can repeat up to N times in the worst case where $D = 1$, which is $O(n)$
- lines 15-16: $O(1)$
- the estimated time complexity of the whole algorithm therefore belongs to the linear class $O(n)$

UNIFORM SPACE COMPLEXITY:

- size of the input vector I is 2
- we need a total of 3 registers

$2 + 3 = 5 \in O(1)$

LOGARITHMIC TIME COMPLEXITY

- to estimate the value of $t_\pi^{log}(I) = \sum_{i=1}^{k} c_{(\pi,I)}^{log}(i)$, we need to estimate the value of the cost function for each line of the program
- for input $I = (N, D)$ we can do that in the following way:

1 : len(N) } executed once
2 : len(N) }
3 : len(D) }
4 : len(D) }
5 : len(D) }
6 : 0 }
...
15 : len(N) }
16 : 0 }

7 : len(N) } executed up to N times
8 : max{len(N), len(D)} }
9 : 0 }
10 : max{len(N), len(D)} }
11 : len(N) }
12 : len(N) }
13 : len(N) }
14 : 0 }

- then $t_\pi^{log}(I) = 3 \cdot len(N) + 3 \cdot len(D) + N \cdot (4 \cdot len(N) + 2 \cdot max\{len(N), len(D)\})$

LOGARITHMIC SPACE COMPLEXITY $I = (N, D)$

- to calculate the value of $s_\pi^{log}(I) = len(I) + \sum_{i=0}^{n} len(r_i)$, where n is the number of registers used by the RAM program, we need to analyze the maximum amount of memory needed for each register
- R0: needs $max\{len(N), len(D)\}$ bits to load the inputs in the beginning; later values are only smaller up to len(N) bits
- R1: needs len(N) bits and only gets smaller throughout the loop between lines 7-14
- R2: needs len(D) bits and does not change
- R3: starts at 0 and gets incremented in the loop and can contain a value of up to len(N) bits long
- then $s_\pi^{log}(I) = len(N) + len(D) + max\{len(N), len(D)\} + len(N) + len(D) + len(N) =$
$= 3\,len(N) + 2\,len(D) + max\{len(N), len(D)\}$

(3) Time complexity: $f(n) = O(n)$

- a finite automaton consists of finite states and transition functions
- for an input string of length n, the FA begins in the start state and transitions through states as it reads each symbol sequentially
- the number of steps is exactly n, therefore the time complexity is $O(n)$

Space complexity: $g(n) = O(1)$

- a finite automaton doesn't use any additional memory to process the input string apart from its internal finite set of states
- this means it doesn't depend on n and therefore the space complexity is $O(1)$.