Week 1. Problem set (solutions by Elgin Danil)

1. Compute asymptotic worst case time complexity of the following algorithm (see pseudocode conventions in [CLRS, §2.1]). You **must** use Θ-notation. For justification, provide execution cost and frequency count for each line in the body of the `secret` procedure. Optionally, you may provide the details for the computation of the running time $T(n)$ for worst case scenario. Proof for the asymptotic bound is not required for this exercise.

```
1  /* A is a 0-indexed array,
2    * n is the number of items in A */
3  secret(A, n):
4    for i = 1 to n
5      s := 1
6      j := i - 1
7      while j < n and A[j] ≥ A[i-1]
8        j := j + s
9        s := s + 2
10     exchange A[i-1] with A[min(j, n - 1)]
```

Line	Execution Cost	Frequency Count
1	—	—
2	—	—
3	—	—
4		
5		
6		
7		
8		
9		
10		

Answer: $\Theta(n^2)$.
Justification: *cost and frequency count is presented in the following table:*

Line	Execution Cost	Frequency Count
1	—	—
2	—	—
3	—	—
4	c1	n + 1
5	c2	n
6	c3	n
7	c4	$\sum_{i=1}^{n} t_i$
8	c5	$\sum_{i=1}^{n} t_i - 1$
9	c6	$\sum_{i=1}^{n} t_i - 1$
10	c7	n

(optional part) From this table, we conclude

$$T(n) = \ldots = \Theta(\ldots)$$

2. Indicate, for each pair of expressions (A, B) in the table below whether $A = O(B)$, $A = o(B)$, $A = \Omega(B)$, $A = \omega(B)$, or $A = \Theta(B)$. Write your answer in the form of the table with *YES* or *NO* written in each cell:

A	B	$A = O(B)$	$A = o(B)$	$A = \Omega(B)$	$A = \omega(B)$	$A = \Theta(B)$
$1 + \sin(n)$	$\left(1 + \frac{1}{n}\right)^{\frac{1}{n}}$	*Yes*	*No*	*No*	*No*	*No*
$\sqrt[5]{n}$	$\log_3^2 n$	*No*	*No*	*Yes*	*Yes*	*No*
n^2	$\log_2^n n$	*Yes*	*Yes*	*No*	*No*	*No*
$\left(1 + \frac{1}{10^{100}}\right)^n$	$n^{10^{100}}$	*No*	*No*	*Yes*	*Yes*	*No*
$\log_2((n+1)!)$	$n \log_2 n$	*Yes*	*No*	*Yes*	*No*	*Yes*

3. Let f and g be functions from positive integers to positive reals. Assume $f(n) > n$ for $n > 2025$. Using the formal definition of asymptotic notation [CLRS, §3.2], prove *formally* that

$$\min(f(n) - \sqrt{n}, g(n) + n) = O(f(n) + g(n))$$

Proof. By definition of big-O notation, we need to show that there exist constants c and n0 such that for all $n \geq n0$

$$\min(f(n) - \sqrt{n}, g(n) + n) \leq c * (f(n) + g(n))$$

Let $c = 1$ and $n0 = 2025$

By assumption, we have $f(n) > n$ for $n > 2025$

Consider the following cases:

1. First case ($(f(n) - \sqrt{n}) <= (g(n) + n)$)
   1) $f(n) - \sqrt{n} <= 1 * (f(n) + g(n))$
   2) $-\sqrt{n} <= g(n)$
   3) g is the function from positive integers to positive reals, it means that $g(n) > 0$. $\sqrt{n} >= 0$, it means that $-\sqrt{n} <= 0$. Hence, $-\sqrt{n} <= g(n)$is a true.

2. Second case ($(g(n) + n) <= (f(n) - \sqrt{n})$)
   1) $g(n) + n <= 1 * (f(n) + g(n)$
   2) $n <= f(n)$
   3) By assumption, $f(n) > n$ for $n > 2025$. We set $n0 = 2025$, it means that we consider $n > 2025$. Hence, $f(n) >= n$ is a true.

3. Conclusion: We considered all possible cases and proved that condition for big-O holds. Therefore, $\min(f(n) - \sqrt{n}, g(n) + n) \leq c * (f(n) + g(n))$ is a true. QED.

□

References

[CLRS] Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C., 2022. *Introduction to algorithms, Fourth Edition.* MIT press.