MarxisSolution — LIS CHEAT SHEET
Longest Increasing Subsequence - Theory of Computation

WHAT IS LIS?

Given a sequence of numbers, find the length of the longest subsequence

where elements are in strictly increasing order.

Example: [10, 9, 2, 5, 3, 7, 101, 18]

LIS: [2, 3, 7, 101] → Length = 4

RECURRENCE RELATION

For $O(n^2)$ Dynamic Programming:

dp[i] = max(dp[j] + 1) for all j < i where nums[j] < nums[i]

dp[i] = length of LIS ending at position i

Base case: dp[i] = 1 (each element alone is an increasing subsequence)

Answer: max(dp[0], dp[1], ..., dp[n-1])

$O(n^2)$ DYNAMIC PROGRAMMING SOLUTION

ALGORITHM:

1. Create dp array of size n, initialized to 1

2. For i = 0 to n-1:

```
For j = 0 to i-1:
    if nums[j] < nums[i]:
        dp[i] = max(dp[i], dp[j] + 1)
```

3. Return max value in dp array

EXAMPLE: Sequence [10, 9, 2, 5, 3, 7, 101, 18]

Index:	0	1	2	3	4	5	6	7
Value:	10	9	2	5	3	7	101	18
dp[i]:	1	1	1	2	2	3	4	3

LIS length = max(dp) = 4

LIS: [2, 3, 7, 101] or [2, 5, 7, 101] or [2, 3, 7, 18]

Time Complexity: $O(n^2)$

Space Complexity: $O(n)$

MarxisSolution — LIS CHEAT SHEET
Longest Increasing Subsequence - Theory of Computation

O(n log n) PATIENCE SORTING SOLUTION

ALGORITHM (Patience Sorting):

1. Initialize empty piles list
2. For each number x in sequence:

   Find leftmost pile where top card ≥ x using binary search

   If found, replace that pile's top with x

   Else, create a new pile with x

3. Number of piles = LIS length

EXAMPLE: Sequence [10, 9, 2, 5, 3, 7, 101, 18]

Step-by-step piles after each number:

10 → [10]

9 → [9] (replaces 10)

2 → [2] (replaces 9)

5 → [2, 5] (new pile)

3 → [2, 3] (replaces 5)

7 → [2, 3, 7] (new pile)

101 → [2, 3, 7, 101] (new pile)

18 → [2, 3, 7, 18] (replaces 101)

Number of piles = 4 → LIS length = 4

Time Complexity: O(n log n) — binary search for each element

Space Complexity: O(n)

MarxisSolution — LIS CHEAT SHEET
Longest Increasing Subsequence - Theory of Computation

COMPLEXITY COMPARISON

Approach	Time	Space	Best For
$O(n^2)$ DP	$O(n^2)$	O(n)	Small n ($\le 10^4$)
O(n log n)	O(n log n)	O(n)	Large n ($\le 10^5$)

COMMON MISTAKES

❌ Forgetting that subsequence ≠ subarray

Subsequence does NOT need to be contiguous

❌ Using <= instead of < for strictly increasing

When numbers are equal, they cannot be increasing

❌ Only returning max(dp) without reconstructing sequence

For interviews, often need actual subsequence, not just length

❌ Using $O(n^2)$ for large n (should use O(n log n) for $>10^4$ elements)

PRACTICE PROBLEMS

1. LIS: [1, 2, 3, 4, 5] → Answer: 5
2. LIS: [5, 4, 3, 2, 1] → Answer: 1
3. LIS: [3, 1, 4, 2, 5, 2, 6] → Answer: 4 ([1,2,5,6] or [1,4,5,6])
4. LIS: [2, 6, 3, 4, 1, 2, 9, 5, 8] → Answer: 4
5. LIS with equal numbers: [2, 2, 2, 2] → Answer: 1 (strictly increasing)

MarxisSolution — LIS CHEAT SHEET
Longest Increasing Subsequence - Theory of Computation

EXAM TIPS

- Always check for empty input (return 0)
- For $O(n^2)$ DP, start with dp[i] = 1 for all i
- For O(n log n), binary search on piles (not on original array)
- LIS has many applications: adaptive coding, patience card game, etc.

MORE RESOURCES

Visit our CS Student Hub for free interactive tools:

→ https://marxissolution.com/cs-student-hub/

Other resources:

→ Knapsack Visualizer

→ Coin Change Guide

→ Pumping Lemma Solver

 Free for CS students. Share with classmates.