# Dynamic Programming

## Solving Optimization Problems

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://about.softuni.bg

# Table of Contents

1. What is Dynamic Programming?
2. Fibonacci Sequence
3. Move Down/Right Sum
4. Longest Common Subsequence
5. Longest Increasing Subsequence

# What is Dynamic Programming?

- "**Controlled**" brute force / exhaustive search
- Key ideas:
  - **Subproblems**: like original problem, but smaller
    - Write solution to one **subproblem** in terms of solutions to smaller acyclic subproblems
  - **Memoization**: remember the **solution** to subproblems we've already solved, and **re-use**
    - **Avoid** exponentials
  - **Guessing**: if you don't know something, **guess it!** (try all possibilities)

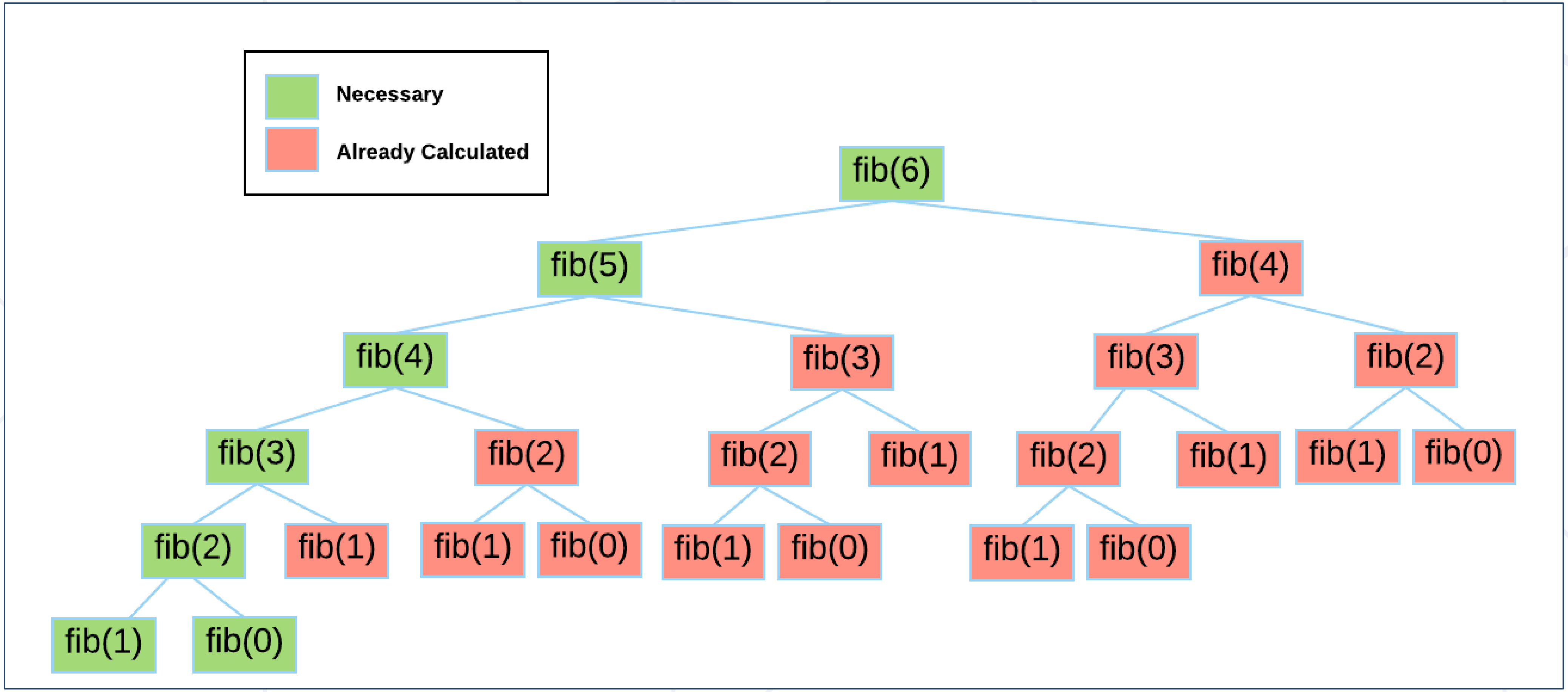
# Fibonacci Sequence

Recursive Approach

# Example: Fibonacci Sequence

- **The Fibonacci sequence** holds the following integers:
  - 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …
  - The **first two** numbers are **0** and **1**
  - Each subsequent number is the sum of the previous two numbers
- Recursive mathematical formula:
  - $F_0 = 0, F_1 = 1$
  - $F_n = F_{n-1} + F_{n-2}$

# Recursive Approach



# Memoization

- DP → sub-problems **overlap**
- In order to **avoid solving** problems **multiple times**, memorize
  - **Memoization** → **save/cache** sub-problem solutions **for later use**
- Typically using an **array**, **matrix** or a **hash table**

# Compare Fibonacci Solutions

- Recursive Fibonacci
  - **~ O($1.6^n$)**
- Recursive Fibonacci (with memorization)
  - **~ O(n)**
- If we want to find the 36th Fibonacci number:
  - Recursive solution takes **48 315 633** steps
  - Iterative or recursive (with memorization) takes ~**36** steps

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | **6** | 1 | 8 | 9 | 4 | 2 |
| 1 | **8** | 0 | 3 | 5 | 6 | 7 |
| 3 | **4** | **8** | **7** | 2 | 1 | 8 |
| 0 | 9 | 2 | **8** | 1 | 7 | 9 |
| 2 | 7 | 1 | **9** | **7** | **8** | 2 |

# Move Down/Right Sum

Largest Sum in Matrix of Numbers

# "Move Down / Right Sum" Problem

- You are given a matrix of numbers
  - Find the **path with largest sum**
  - Start → top left
  - End → bottom right
  - Move only right/down
  - There won't be negative numbers

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | **6** | 1 | 8 | 9 | 4 | 2 |
| 1 | **8** | 0 | 3 | 5 | 6 | 7 |
| 3 | **4** | **8** | **7** | 2 | 1 | 8 |
| 0 | 9 | 2 | **8** | 1 | 7 | 9 |
| 2 | 7 | 1 | **9** | **7** | **8** | 2 |
| 4 | 5 | 6 | 1 | 2 | **5** | **6** |
| 9 | 3 | 5 | 2 | 8 | 1 | **9** |
| 2 | 3 | 4 | 1 | 7 | 2 | **8** |

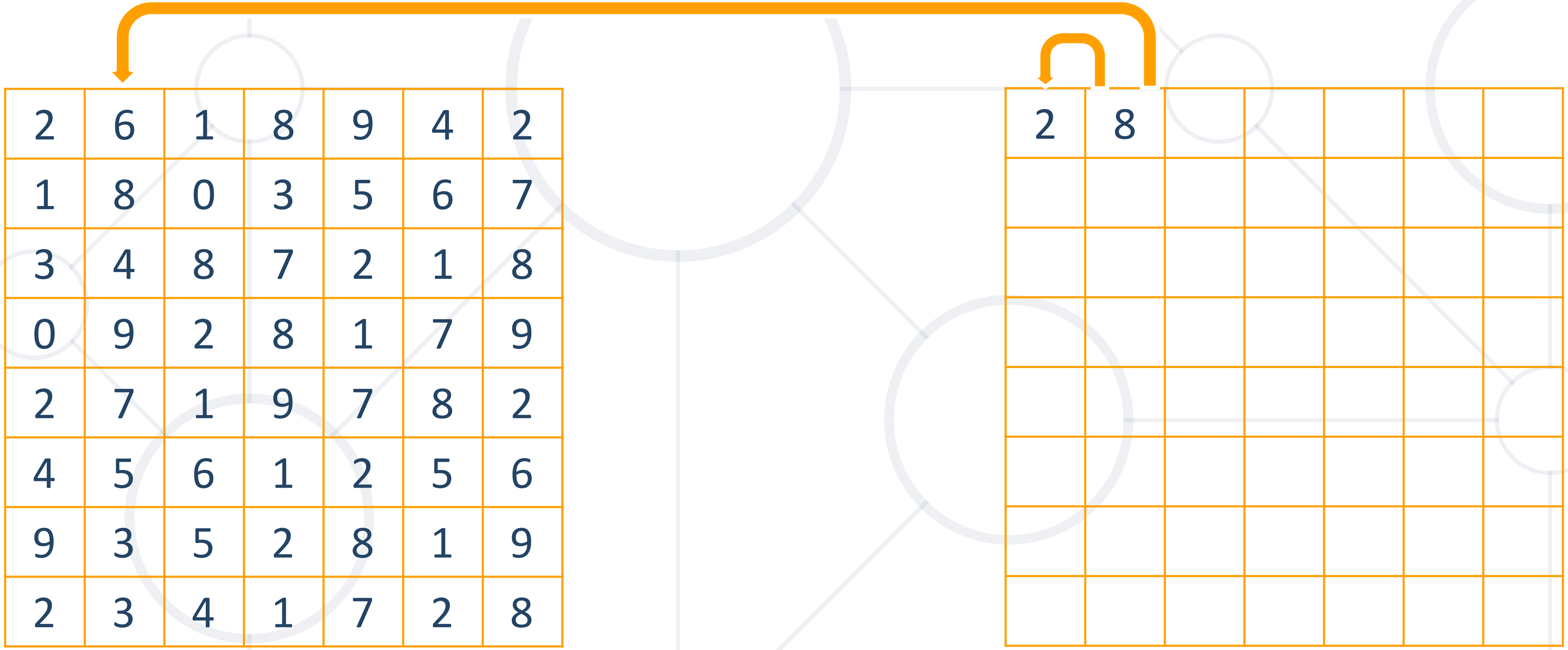
# Building the DP Matrix (1)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

→

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

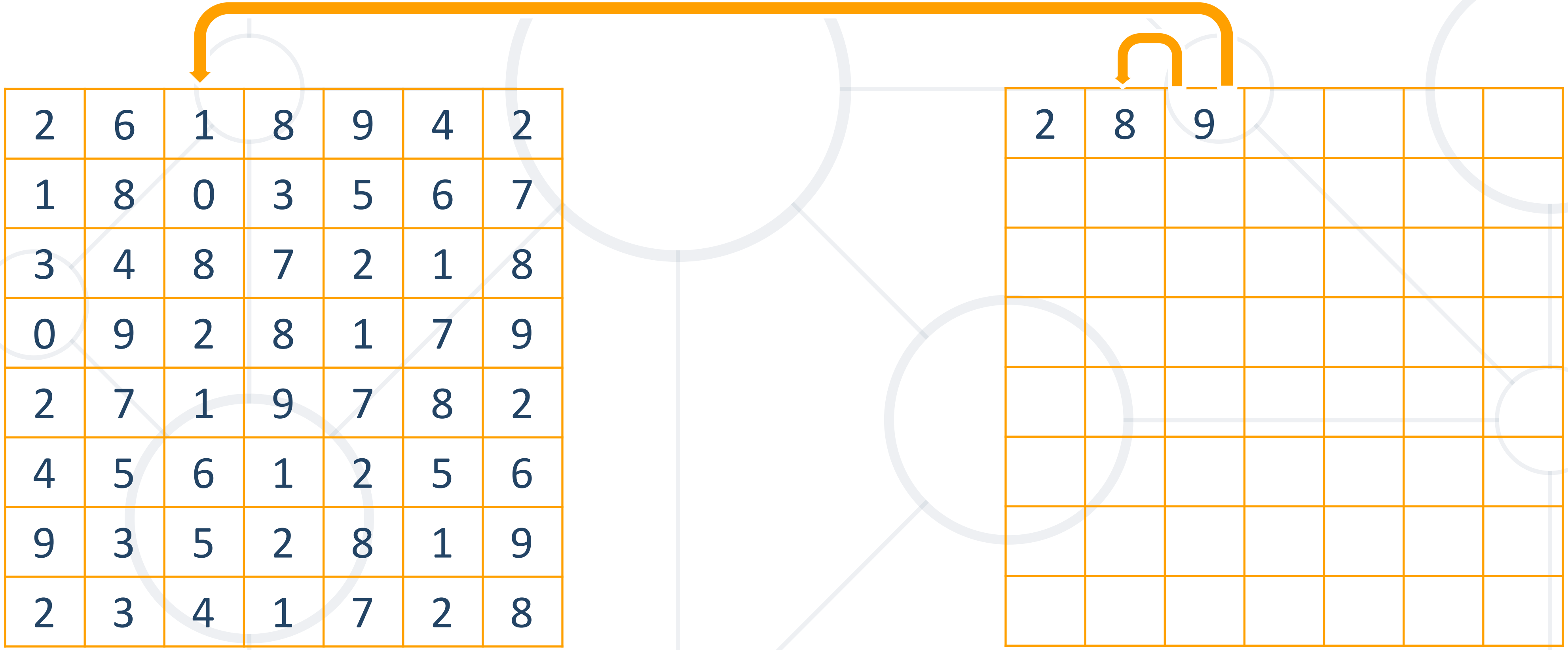
# Building the DP Matrix (2)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

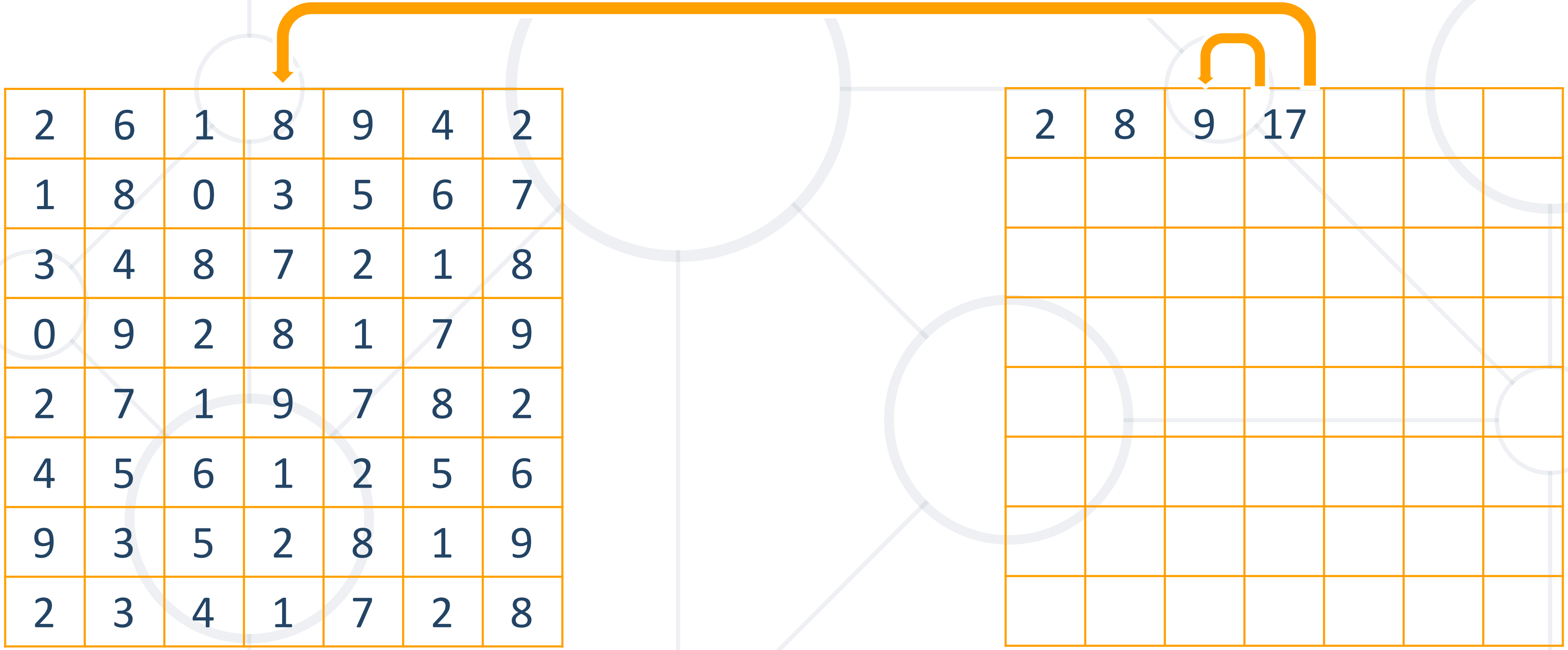
# Building the DP Matrix (3)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

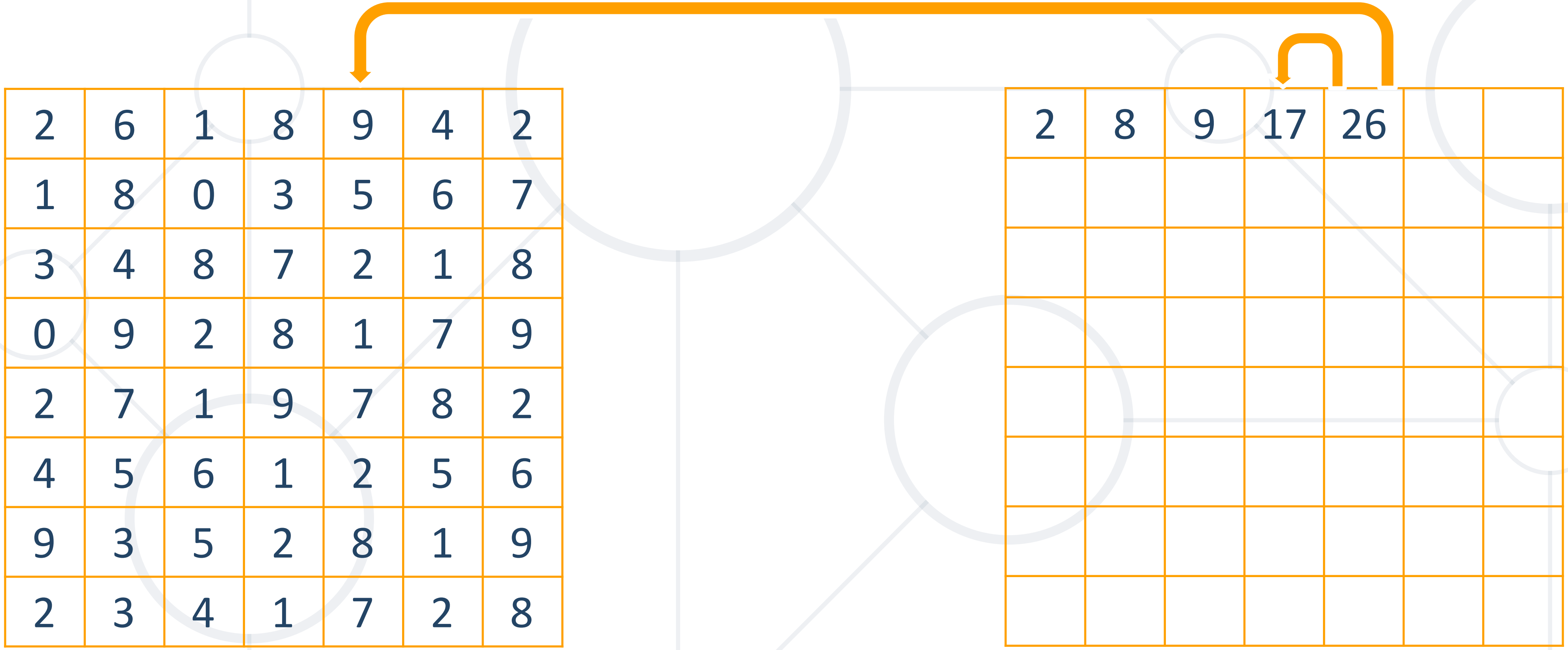
# Building the DP Matrix (4)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

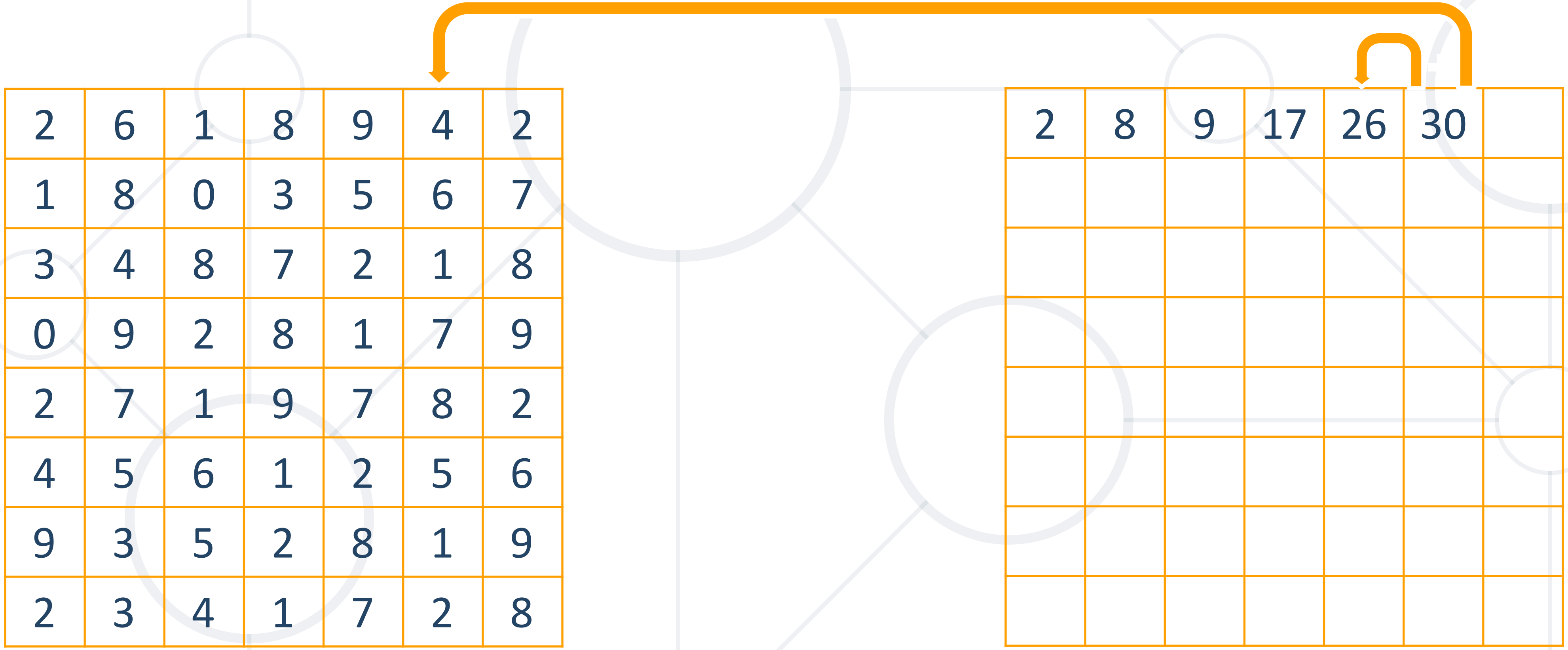
# Building the DP Matrix (5)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

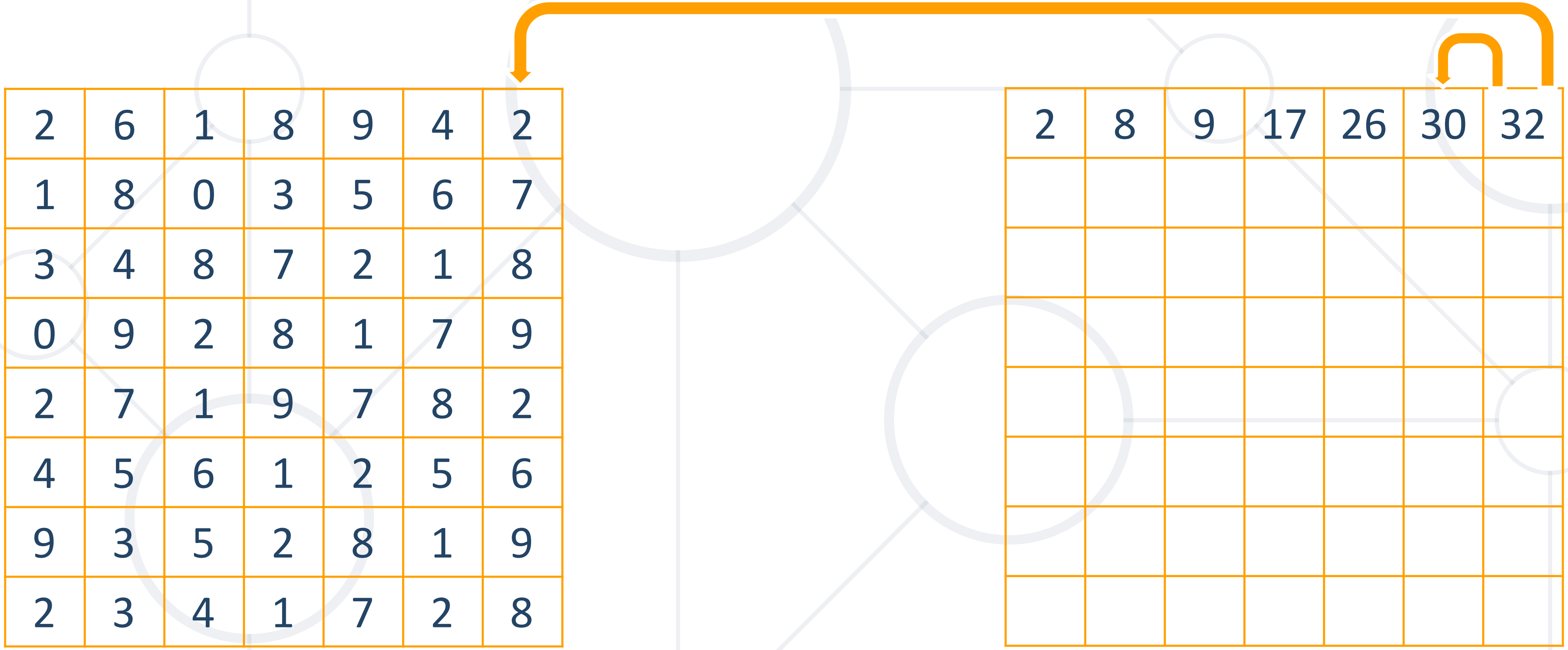
# Building the DP Matrix (6)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Building the DP Matrix (7)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Building the DP Matrix (8)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Building the DP Matrix (9)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | | | | | | |
| 6 | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Building the DP Matrix (10)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | | | | | | |
| 6 | | | | | | |
| 6 | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Building the DP Matrix (11)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | | | | | | |
| 6 | | | | | | |
| 6 | | | | | | |
| 8 | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Building the DP Matrix (12)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 |   |   |   |   |   |   |
| 6 |   |   |   |   |   |   |
| 6 |   |   |   |   |   |   |
| 8 |   |   |   |   |   |   |
| 12 |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |

# Building the DP Matrix (13)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | | | | | | |
| 6 | | | | | | |
| 6 | | | | | | |
| 8 | | | | | | |
| 12 | | | | | | |
| 21 | | | | | | |
| | | | | | | |

# Building the DP Matrix (14)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | | | | | | |
| 6 | | | | | | |
| 6 | | | | | | |
| 8 | | | | | | |
| 12 | | | | | | |
| 21 | | | | | | |
| 23 | | | | | | |

# Building the DP Matrix (15)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

MAX()

+

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |

# Building the DP Matrix (16)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

MAX()

+

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |

# Building the DP Matrix (17)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

MAX()

+

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 |  |  |  |
| 6 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |

# Building the DP Matrix (18)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

MAX()

+

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 |  |  |
| 6 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |

# Building the DP Matrix (19)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

MAX()

+

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 |  |
| 6 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |

# Building the DP Matrix (20)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

MAX()

+

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |

# Building the DP Matrix (21)

Start

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 8 | 9 | 4 | 2 |
| 1 | 8 | 0 | 3 | 5 | 6 | 7 |
| 3 | 4 | 8 | 7 | 2 | 1 | 8 |
| 0 | 9 | 2 | 8 | 1 | 7 | 9 |
| 2 | 7 | 1 | 9 | 7 | 8 | 2 |
| 4 | 5 | 6 | 1 | 2 | 5 | 6 |
| 9 | 3 | 5 | 2 | 8 | 1 | 9 |
| 2 | 3 | 4 | 1 | 7 | 2 | 8 |

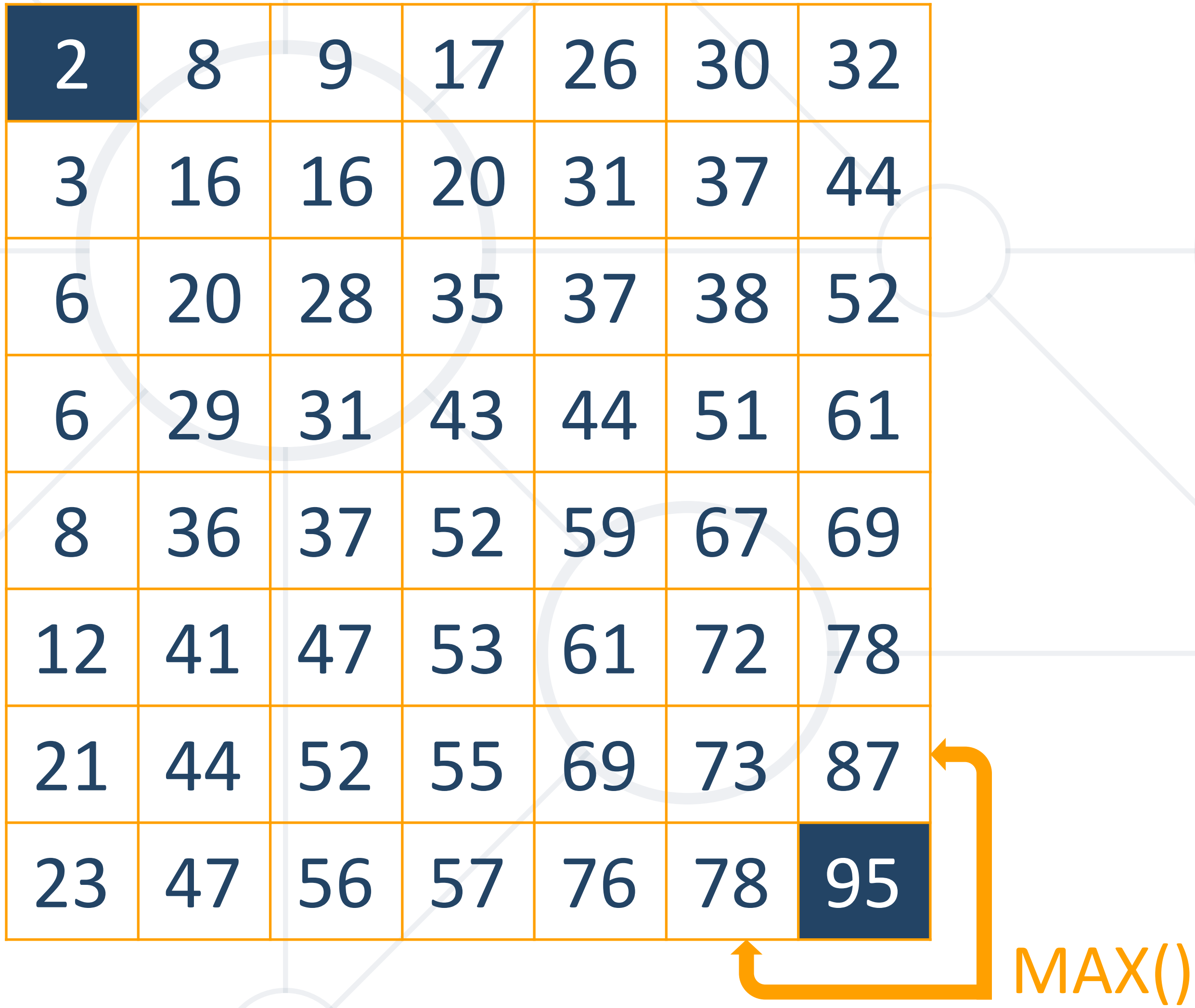
End

Start

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

End

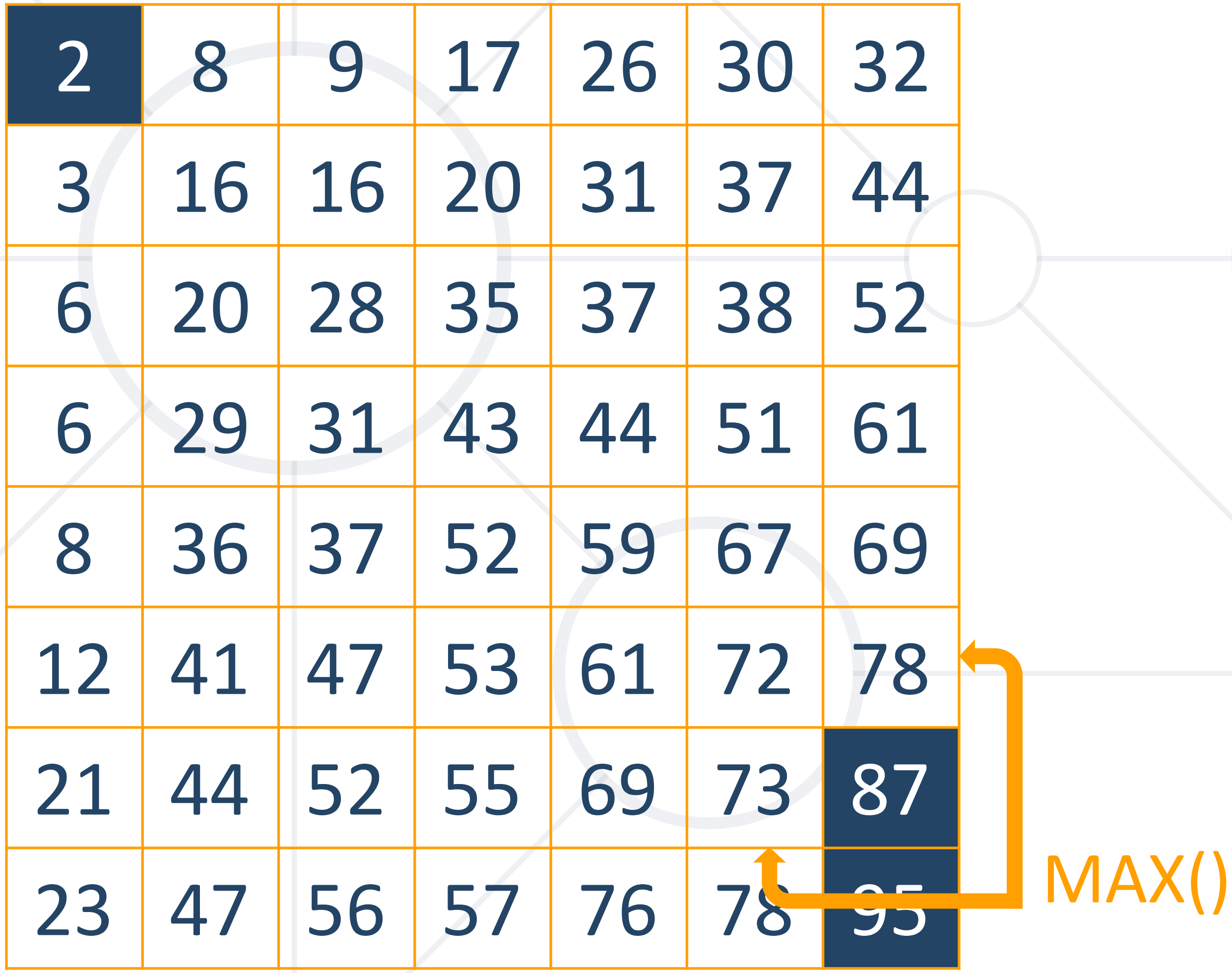
# Finding the Path (1)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

MAX()

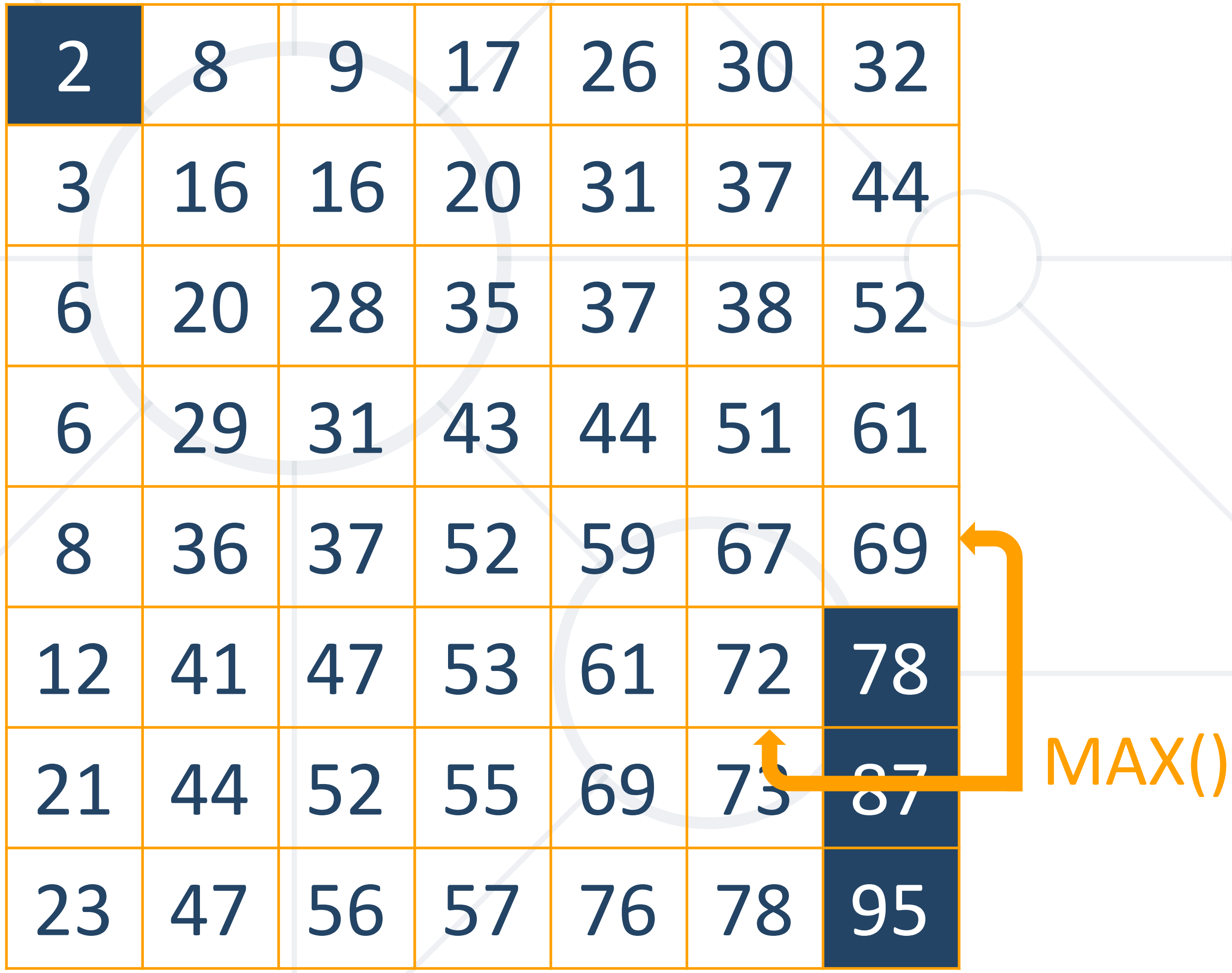
# Finding the Path (2)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

MAX()

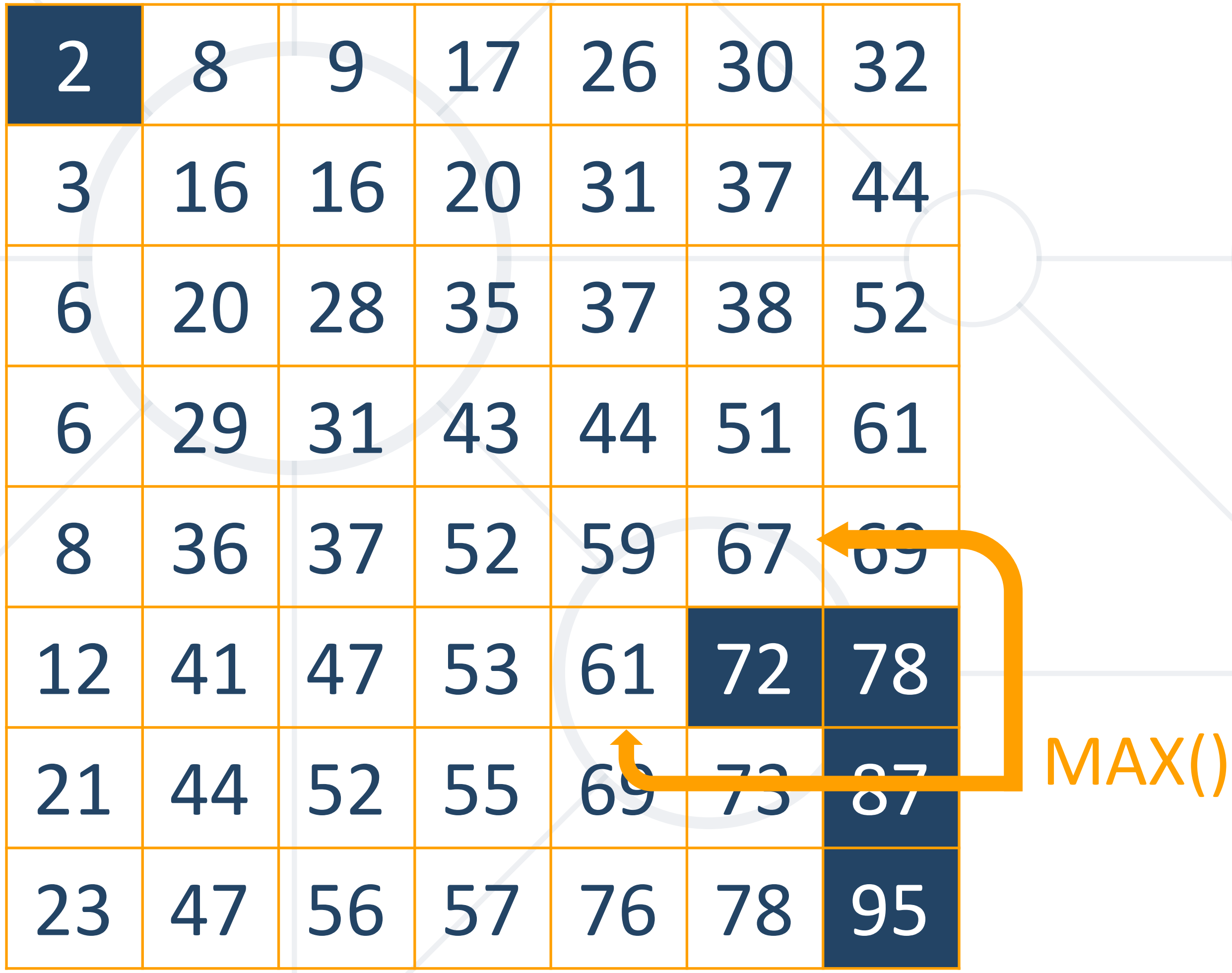
# Finding the Path (3)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

MAX()

# Finding the Path (4)



| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

# Finding the Path (5)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

# Finding the Path (6)

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

# Finding the Path (7)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

# Finding the Path (8)

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | 35 | 37 | 38 | 52 |
| 6 | 29 | 31 | 43 | 44 | 51 | 61 |
| 8 | 36 | 37 | 52 | 59 | 67 | 69 |
| 12 | 41 | 47 | 53 | 61 | 72 | 78 |
| 21 | 44 | 52 | 55 | 69 | 73 | 87 |
| 23 | 47 | 56 | 57 | 76 | 78 | 95 |

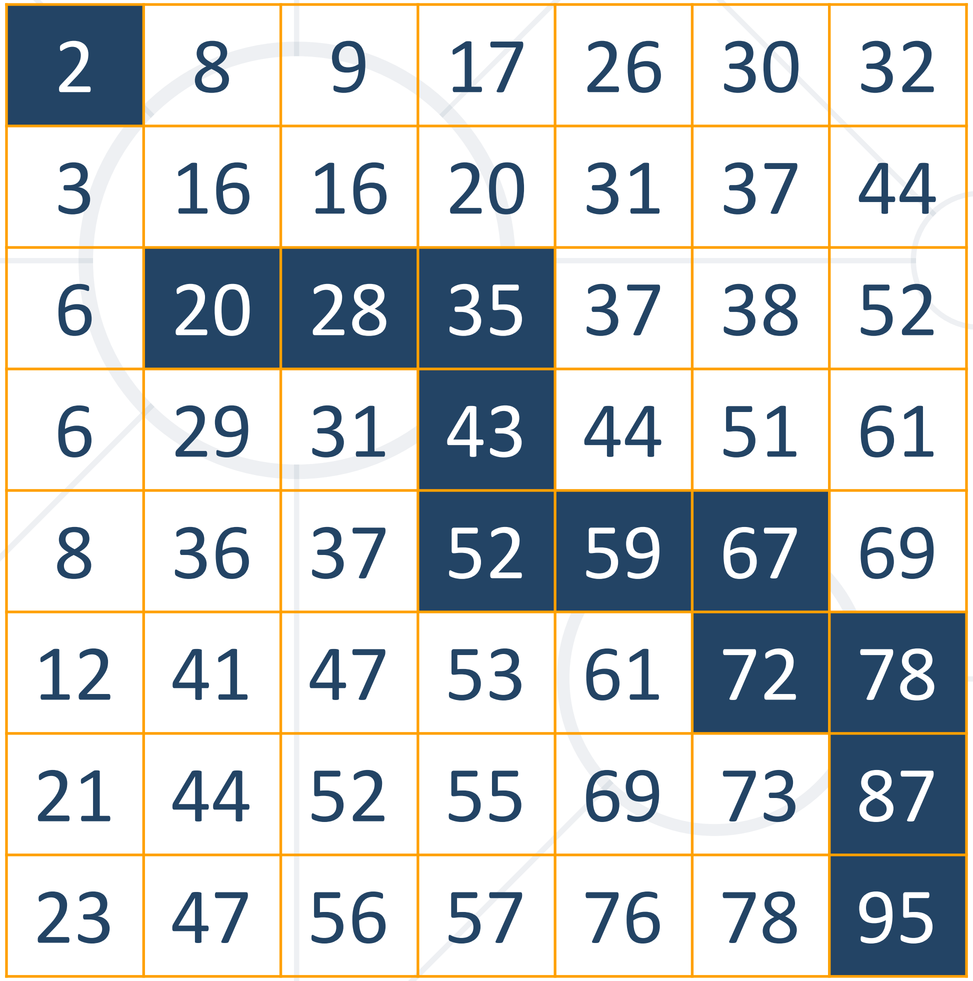
# Finding the Path (9)

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | 28 | **35** | 37 | 38 | 52 |
| 6 | 29 | 31 | **43** | 44 | 51 | 61 |
| 8 | 36 | 37 | **52** | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

# Finding the Path (10)

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | 20 | **28** | **35** | 37 | 38 | 52 |
| 6 | 29 | 31 | **43** | 44 | 51 | 61 |
| 8 | 36 | 37 | **52** | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

# Finding the Path (11)



| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | 16 | 16 | 20 | 31 | 37 | 44 |
| 6 | **20** | **28** | **35** | 37 | 38 | 52 |
| 6 | 29 | 31 | **43** | 44 | 51 | 61 |
| 8 | 36 | 37 | **52** | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

# Finding the Path (12)

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | 8 | 9 | 17 | 26 | 30 | 32 |
| 3 | **16** | 16 | 20 | 31 | 37 | 44 |
| 6 | **20** | **28** | **35** | 37 | 38 | 52 |
| 6 | 29 | 31 | **43** | 44 | 51 | 61 |
| 8 | 36 | 37 | **52** | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

# Finding the Path (13)

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | **8** | 9 | 17 | 26 | 30 | 32 |
| 3 | **16** | 16 | 20 | 31 | 37 | 44 |
| 6 | **20** | **28** | **35** | 37 | 38 | 52 |
| 6 | 29 | 31 | **43** | 44 | 51 | 61 |
| 8 | 36 | 37 | **52** | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

# Finding the Path (14)

| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | **8** | 9 | 17 | 26 | 30 | 32 |
| 3 | **16** | 16 | 20 | 31 | 37 | 44 |
| 6 | **20** | **28** | **35** | 37 | 38 | 52 |
| 6 | 29 | 31 | **43** | 44 | 51 | 61 |
| 8 | 36 | 37 | **52** | **59** | **67** | 69 |
| 12 | 41 | 47 | 53 | 61 | **72** | **78** |
| 21 | 44 | 52 | 55 | 69 | 73 | **87** |
| 23 | 47 | 56 | 57 | 76 | 78 | **95** |

→

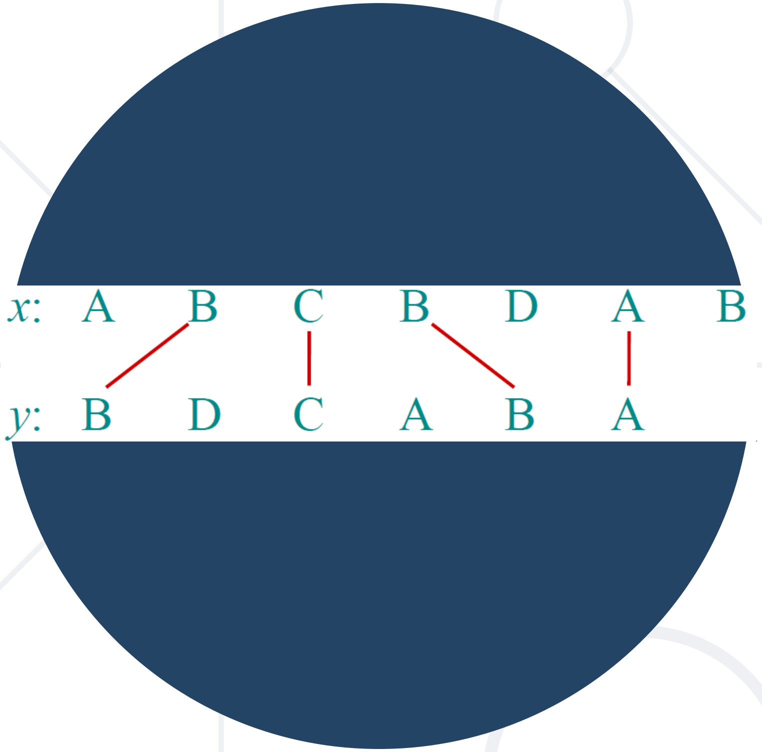
| | | | | | | |
|---|---|---|---|---|---|---|
| **2** | **6** | 1 | 8 | 9 | 4 | 2 |
| 1 | **8** | 0 | 3 | 5 | 6 | 7 |
| 3 | **4** | **8** | **7** | 2 | 1 | 8 |
| 0 | 9 | 2 | **8** | 1 | 7 | 9 |
| 2 | 7 | 1 | **9** | **7** | **8** | 2 |
| 4 | 5 | 6 | 1 | 2 | **5** | **6** |
| 9 | 3 | 5 | 2 | 8 | 1 | **9** |
| 2 | 3 | 4 | 1 | 7 | 2 | **8** |

# "Move Down / Right Sum" – Solution (1)

```
# First, find all base solutions
dp[0][0] = matrix[0][0]
for row in range(1, rows):
    dp[row][0] = dp[row - 1][0] + matrix[row][0]
for col in range(1, cols):
    dp[0][col] = dp[0][col - 1] + matrix[0][col]
# Fill rest of the cells
for row in range(1, rows):
    for col in range(1, cols):
        up = dp[row - 1][col]
        left = dp[row][col - 1]
        dp[row][col] = max(up, left) + matrix[row][col]
```

# "Move Down / Right Sum" – Solution (2)

```
path = deque()
while row > 0 and col > 0:
    path.appendleft([row, col])
    if dp[row - 1][col] > dp[row][col - 1]:
        row -= 1
    else:
        col -= 1
for idx in range(row, 0, -1):
    path.appendleft([idx, col])
for idx in range(col, 0, -1):
    path.appendleft([row, idx])
path.appendleft([0, 0])
```

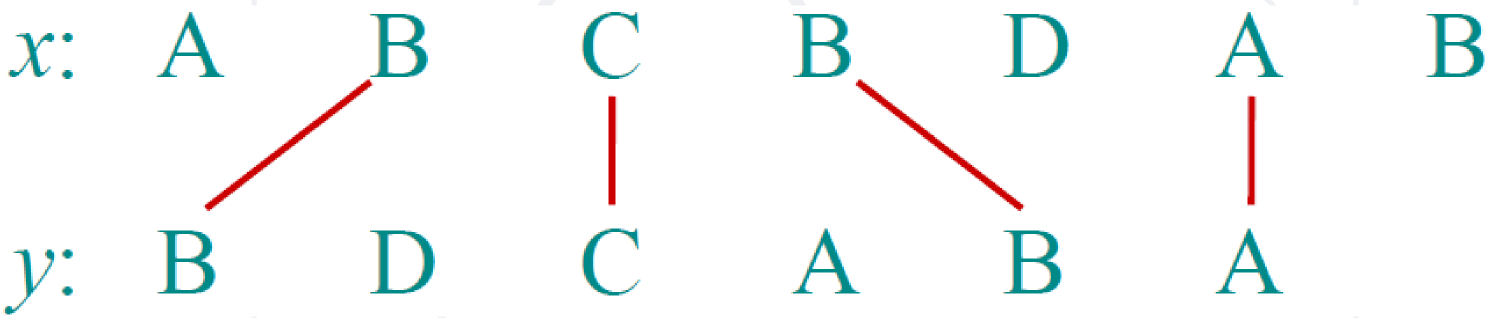


# Longest Common Subsequence (LCS)

## A Recursive DP Approach

# Longest Common Subsequence (LCS)

- Longest common subsequence (LCS) problem:
  - Given two sequences **x[1 … m]** and **y[1 … n]**
  - Find a longest common subsequence (LCS) to them both
- Example:
  - x = "A**BCB**D**A**B"
  - y = "**B**D**C**A**BA**"
  - LCS = "**BCBA**"



# LCS – Recursive Approach

- $S_1$ = **GCCCTAGCG**, $S_2$ = **GCGCAATG**
  - Let $C_1$ = the right-most character of $S_1$ ($C_1$ = G)
  - Let $C_2$ = the right-most character of $S_2$ ($C_2$ = G)
  - Let $S_1'$ = $S_1$ with $C_1$ "chopped-off" ($S_1'$ = GCCCTAGC)
  - Let $S_2'$ = $S_2$ with $C_2$ "chopped-off" ($S_2'$ = GCGCAAT)
- There are three recursive sub-problems:
  - $L_1$ = LCS($S_1'$, $S_2$)
  - $L_2$ = LCS($S_1$, $S_2'$)
  - $L_3$ = LCS($S_1'$, $S_2'$)

# LCS – Recursive Formula

- Let **lcs[x][y]** be the longest common subsequence of **S1[0 … x]** and **$S_2$[0 … y]**
- LCS has the following recursive properties:

```
lcs[-1][y] = 0
lcs[x][-1] = 0
lcs[x][y] = max(
  lcs[x-1][y],
  lcs[x][y-1]) or lcs[x-1][y-1]+1 when S1[x] == S2[y]
```

# Calculating the LCS Table

```
rows = len(first) + 1
cols = len(second) + 1
lcs = []
[lcs.append([0] * cols) for _ in range(rows)]
for row in range(1, rows):
    for col in range(1, cols):
        if first[row - 1] == second[col - 1]:
            prev = lcs[row - 1][col - 1]
            lcs[row][col] = prev + 1
        else:
            up = lcs[row - 1][col]
            left = lcs[row][col - 1]
            lcs[row][col] = max(up, left)
```

# Reconstructing the LCS Sequence

```
lcs_letters = deque()
row = rows - 1
col = cols - 1
while row >= 0 and col >= 0:
    if first[row - 1] == second[col - 1]:
        lcs_letters.appendleft(first[row - 1])
        row -= 1
        col -= 1
    elif lcs[row - 1][col] > lcs[row][col - 1]:
        row -= 1
    else:
        col -= 1
print(''.join(lcs_letters))
```

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 3 | 14 | 5 | 12 | 15 |

# Longest Increasing Subsequence

## Finding and Reconstructing LIS

# Longest Increasing Subsequence (LIS)

- Goal: find the largest subsequence of increasing numbers within a given sequence
- This subsequence is not necessarily contiguous, or unique
- Example:
  - {**3**, **5**, 8, **6**, **7**} → {3, 5, 6, 7}

# Longest Increasing Subsequence (1)

| LIS | |
|---|---|

↓

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

↓

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | | | | | | | | | | | |

# Longest Increasing Subsequence (2)

| | |
|---|---|
| LIS | 3 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | | | | | | | | | | |

# Longest Increasing Subsequence (3)

| LIS | 3, 14 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | | | | | | | | | |

# Longest Increasing Subsequence (4)

| | |
|---|---|
| LIS | 3, 5 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | | | | | | | | |

# Longest Increasing Subsequence (5)

| LIS | 3, 5, 12 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 |  |  |  |  |  |  |  |

# Longest Increasing Subsequence (6)

| | |
|---|---|
| **LIS** | 3, 5, 12, 15 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | | | | | | |

# Longest Increasing Subsequence (7)

| LIS | 3, 5, 7 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 |  |  |  |  |  |

# Longest Increasing Subsequence (8)

| | |
|---|---|
| LIS | 3, 5, 7, 8 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | | | | |

# Longest Increasing Subsequence (9)

| | |
|---|---|
| **LIS** | 3, 5, 7, 8, 9 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | | | |

# Longest Increasing Subsequence (10)

| LIS | 3, 5, 7, 8, 9, 11 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | | |

# Longest Increasing Subsequence (11)

| LIS | 3, 5, 7, 8, 9, 10 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | 6 | |

# Longest Increasing Subsequence (12)

| | |
|---|---|
| **LIS** | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | 6 | 1 |

# Longest Increasing Subsequence (13)

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 ✓ | 6 ✓ | 1 |

# Longest Increasing Subsequence (14)

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | 6 | 1 |

```
Subsequence sets:
     {3}, {3, 14}, {3, 5}, {3, 5, 12}, {3, 5, 12, 15}, {3, 5, 7},
     {3, 5, 7, 8}, {3, 5, 7, 8, 9}, {3, 5, 7, 8, 9, 11},
     {3, 5, 7, 8, 9, 10}, {1}
```

# Reconstructing LIS (1)

| | |
|---|---|
| LIS | |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | | | | | | | | | | | |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | | | | | | | | | | | |

# Reconstructing LIS (2)

| | |
|---|---|
| **LIS** | 3 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | | | | | | | | | | |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | | | | | | | | | | |

# Reconstructing LIS (3)

| | |
|---|---|
| LIS | 3, 14 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | | | | | | | | | |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | | | | | | | | | |

# Reconstructing LIS (4)

| | |
|---|---|
| **LIS** | 3, 5 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | | | | | | | | |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | | | | | | | | |

# Reconstructing LIS (5)

| | |
|---|---|
| **LIS** | 3, 5, 12 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | | | | | | | |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | | | | | | | |

# Reconstructing LIS (6)

| | |
|---|---|
| **LIS** | 3, 5, 12, 15 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | | | | | | |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | | | | | | |

# Reconstructing LIS (7)

|  |  |
|---|---|
| **LIS** | 3, 5, 7 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | 3 |  |  |  |  |  |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 |  |  |  |  |  |

# Reconstructing LIS (8)

| | |
|---|---|
| **LIS** | 3, 5, 7, 8 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | 3 | 4 | | | | |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 | 5 | | | | |

# Reconstructing LIS (9)

| LIS | 3, 5, 7, 8, 9 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | | | |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | | | |

# Reconstructing LIS (10)

| LIS | 3, 5, 7, 8, 9, 11 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | | |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | | |

# Reconstructing LIS (11)

| | |
|---|---|
| **LIS** | 3, 5, 7, 8, 9, 10 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | 6 | |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | |

# Reconstructing LIS (12)

| | |
|---|---|
| **LIS** | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **len[]** | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | 6 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (1)

| LIS | 1 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 ✓ | 6 ✓ | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (2)

| | |
|---|---|
| **LIS** | 10 |

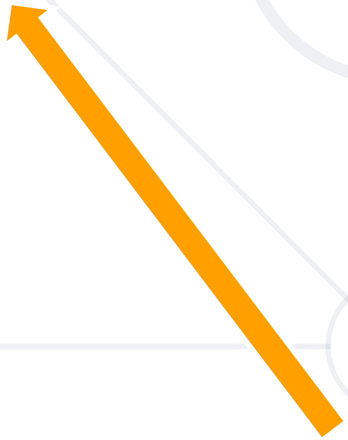
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (3)

| LIS | 10, 9 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |



| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (4)

| LIS | 10, 9 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (5)

| | |
|---|---|
| **LIS** | 10, 9, 8 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (6)

| LIS | 10, 9, 8 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (7)

| LIS | 10, 9, 8, 7 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (8)

| | |
|---|---|
| **LIS** | 10, 9, 8, 7 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (9)

| LIS | 10, 9, 8, 7, 5 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (10)

| LIS | 10, 9, 8, 7, 5 |
|---|---|

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (11)

| | |
|---|---|
| **LIS** | 10, 9, 8, 7, 5, 3 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **seq[]** | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| **index** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **prev[]** | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (12)

| | |
|---|---|
| LIS | 10, 9, 8, 7, 5, 3 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Reconstructing LIS - Right-Most Solution (13)

| LIS | 10, 9, 8, 7, 5, 3 |
|---|---|

Reverse →

| LIS | 3, 5, 7, 8, 9, 10 |
|---|---|

✓

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| seq[] | 3 | 14 | 5 | 12 | 15 | 7 | 8 | 9 | 11 | 10 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| len[] | 1 | 2 | 2 | 3 | 4 | 3 | 4 | 5 | 6 | 6 | 1 |

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| prev[] | -1 | 0 | 0 | 2 | 3 | 2 | 5 | 6 | 7 | 7 | -1 |

# Calculating LIS with Previous – Source Code

```
length = [0] * len(nums)
parent = [0] * len(nums)
best_len, best_idx = 0, 0
for curr_idx in range(len(nums)):
    curr_num, curr_len, curr_parent = nums[curr_idx], 1, -1
    for prev_idx in range(curr_idx - 1, -1, -1):
        prev_number = nums[prev_idx]
        prev_len = length[prev_idx]
        if curr_num > prev_number and prev_len + 1 >= curr_len:
            curr_len = prev_len + 1
            curr_parent = prev_idx
    length[curr_idx] = curr_len
    parent[curr_idx] = curr_parent
```

# Restoring LIS Elements – Source Code

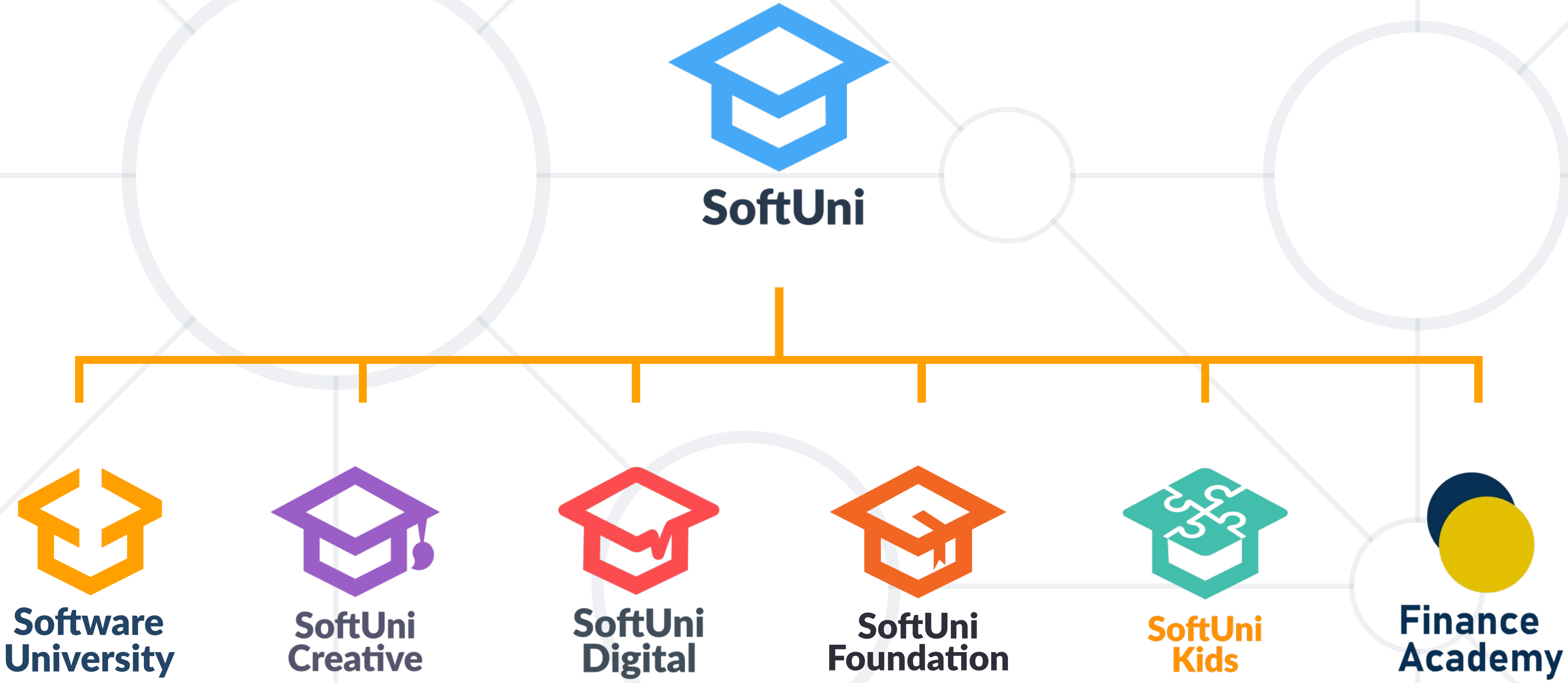
```python
lis = deque()
idx = best_idx

while idx != -1:
    lis.appendleft(nums[idx])
    idx = parent[idx]

print(*lis, sep=' ')
```

# Summary

Software University

- **DP** → Solve a problem by **solving overlapping subproblems**
- **Memoization** → **Save** subproblem **solutions** for later use
- **Optimal Substructure**
  - **Subproblems** should have **optimal solutions**
  - Combine optimal solutions for subproblems
  - Get optimal solution for original problem

# Questions?



SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

Finance Academy

© SoftUni – https://about.softuni.bg. Copyrighted document. Unauthorized copy, reproduction or use is not permitted.

# SoftUni Diamond Partners

Software University

# Educational Partners

# License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg

# Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

Software University

FORUM