Programming Practice for Data Science

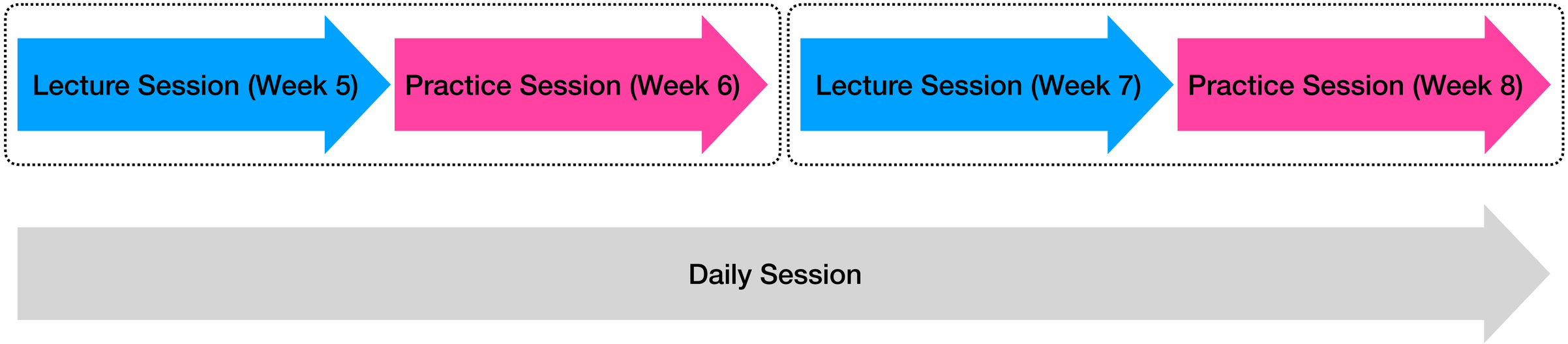
Lecture 5: Divide and Conquer (10/11/24)

Taesup Kim
Graduate School of Data Science
Seoul National University

Course Structure

- **Lecture Session (every other week)**
  - Data Structure and Algorithm with Examples (scoring X / attendance O)
- **Practice Session (every other week)**
  - Coding Test with GradeScope (scoring O / attendance O)
- **Daily Session**
  - Coding Test with GradeScope (scoring O)

Course Structure



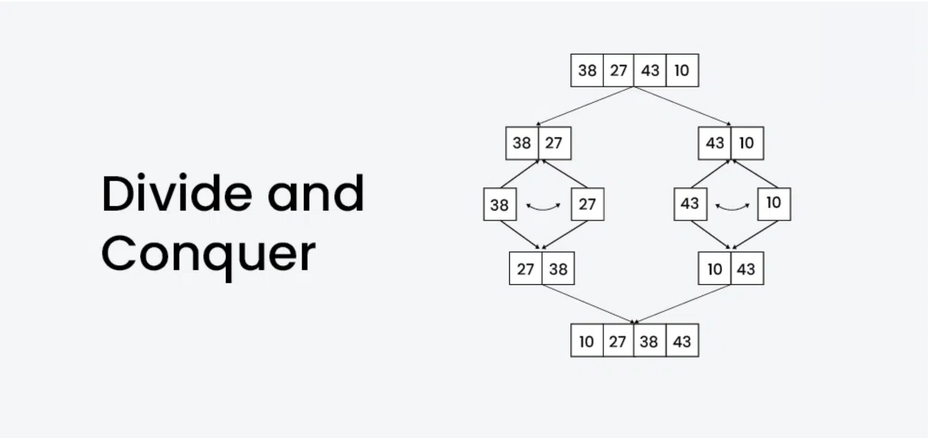
Course Structure

- Attendance 20%
- Practice Session 60%
- Daily Session 20%

Divide and Conquer

Definition

- **Divide ▶ Conquer ▶ Merge**
  - **Divide**: break down the original problem into smaller subproblems
  - **Conquer**: solve each of the smaller subproblems individually
  - **Merge**: combine the sub-problems to get the final solution of the whole problem



Divide and Conquer

Definition

- **Divide**
  - Break down the original problem into smaller subproblems
  - Each subproblem should represent a part of the overall problem
  - The goal is to divide the problem until no further division is possible (base case)

Divide and Conquer

Definition

- **Conquer**
  - Solve each of the smaller subproblems individually
  - If a subproblem is small enough (often referred to as the “base case”), we solve it directly without further recursion
  - The goal is to find solutions for these subproblems independently

Divide and Conquer

Definition

- **Merge**
  - Combine the sub-problems to get the final solution of the whole problem
  - Once the smaller subproblems are solved, we recursively combine their solutions to get the solution of larger problem
  - The goal is to formulate a solution for the original problem by merging the results from the subproblems

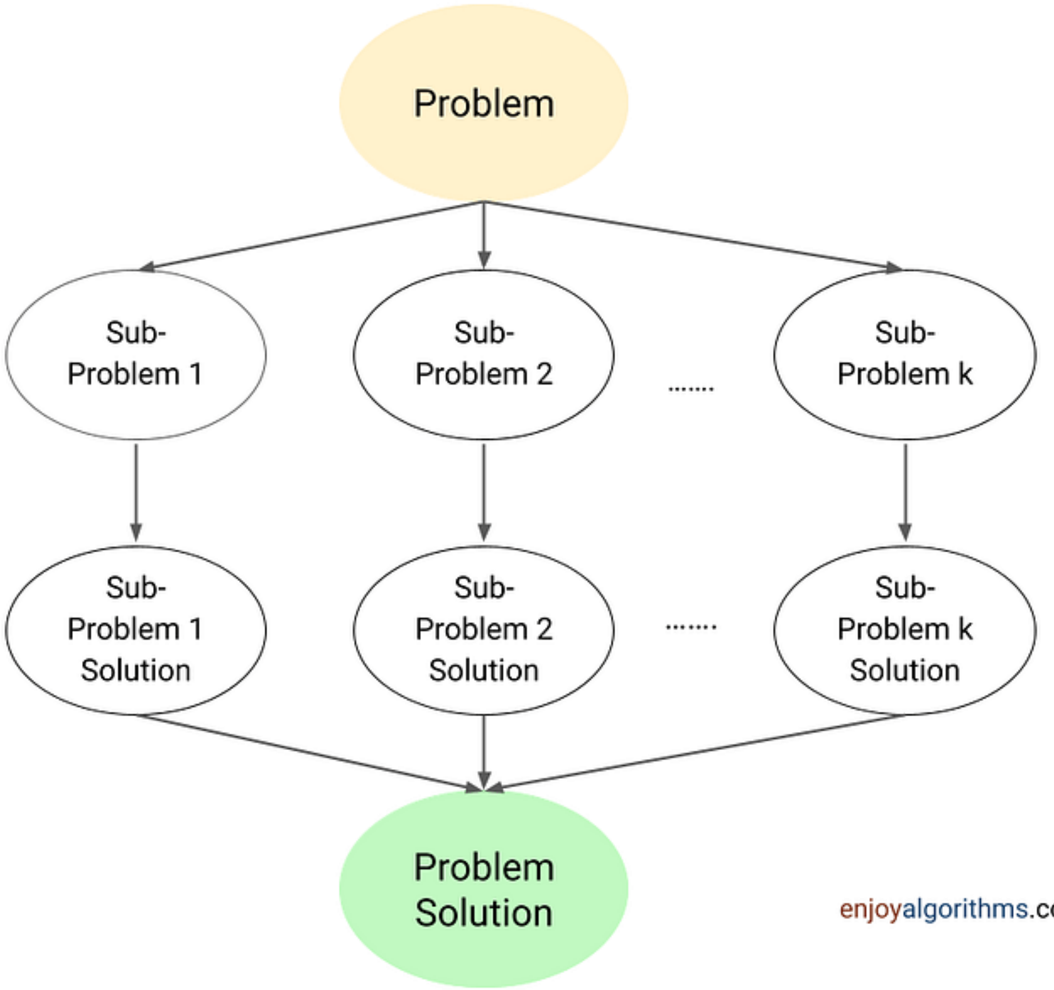
Divide and Conquer

Definition

Divide
Dividing the problem into smaller sub-problems

Conquer
Solving each sub-problems recursively

Combine
Combining sub-problem solutions to build the original problem solution



Divide and Conquer vs. Recursive

- **Divide and Conquer**
  - breaks the problem into smaller subproblems, solves them independently, and then combines the results
- **Recursive**
  - involve a function calling itself to solve a problem
  - not necessarily split the problem into subproblems, but repeatedly apply the same solution process.

Divide and Conquer vs. Recursive

- **Divide and Conquer**
  - solving a large problem by solving smaller, independent subproblems
- **Recursive**
  - while divide and conquer often uses recursion, not all recursion is divide and conquer

“recursion simply refers to a function invoking itself to solve a problem, while divide and conquer involves splitting, solving, and combining”

Divide and Conquer vs. Recursive

- **Divide and Conquer**
  - solving a large problem by solving smaller, independent subproblems
- **Recursive**
  - while divide and conquer often uses recursion, not all recursion is divide and conquer

“Divide and Conquer involves dividing, solving, and combining. It frequently uses recursion, but is a broader concept.”

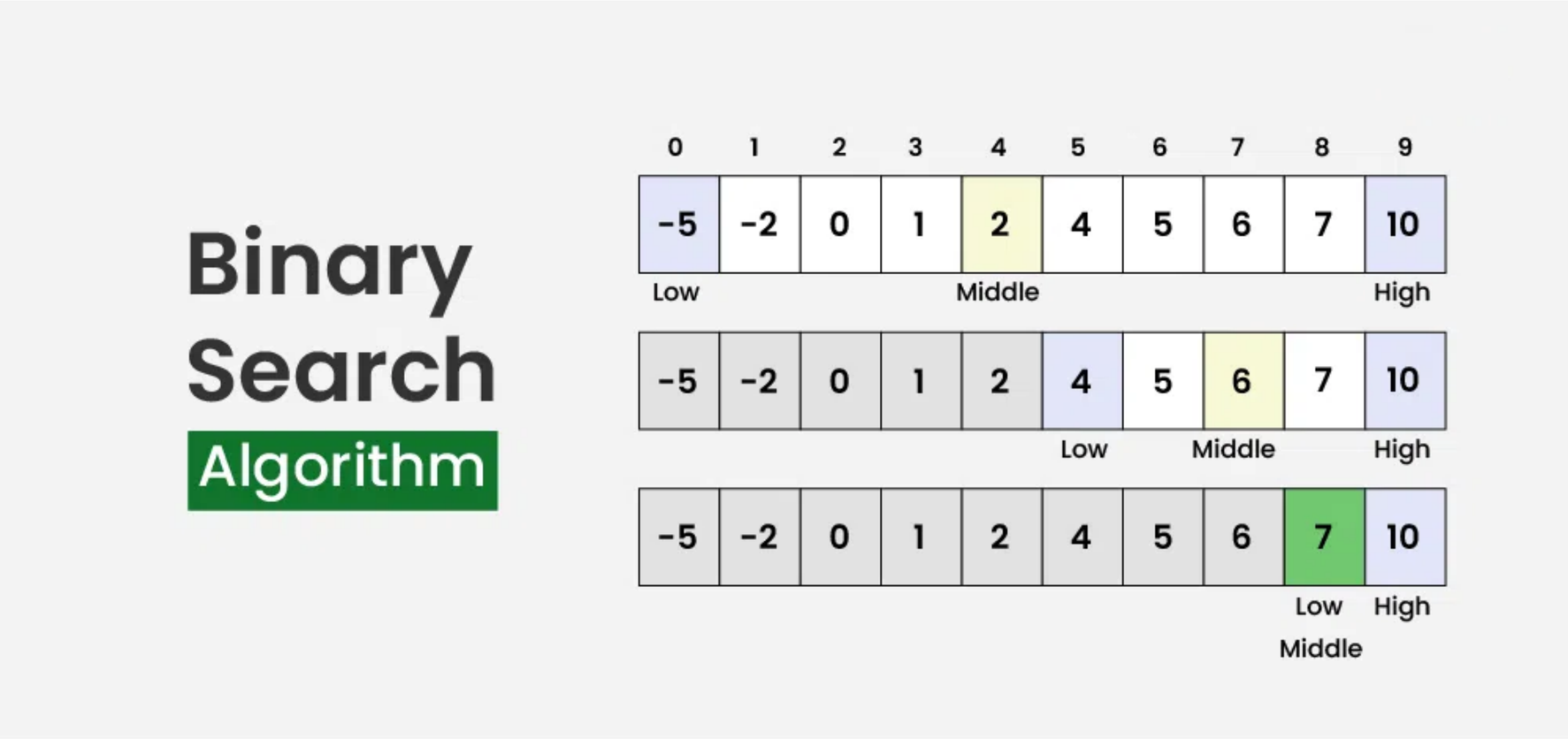
Divide and Conquer vs. DP

- **Dynamic Programming**
  - This technique is used **when subproblems are not independent** and tend to **overlap**.
  - It solves each subproblem only once and stores the result (memoization) to avoid redundant calculations.

Divide and Conquer

Examples

- **Binary Search**



Divide and Conquer

Examples

- **Merge Sort**