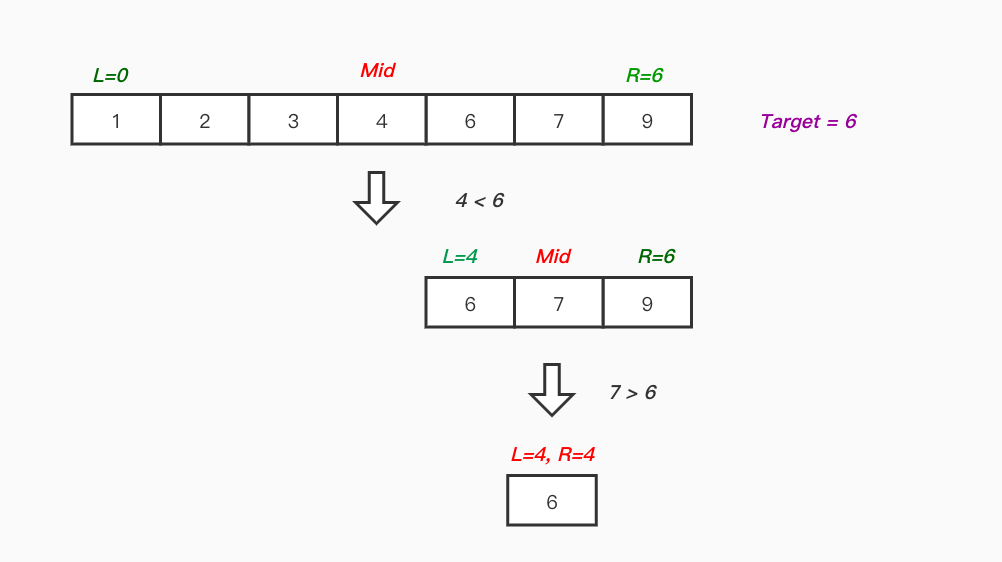
**Divide and Conquer**

Divide and Conquer is an algorithm that breaks a problem into smaller subproblems, solves them recursively, and combines their solutions to solve the original problem. It consists of three steps:

1. Divide: Split the problem into smaller, independent subproblems.

2. Conquer: Solve the subproblems recursively.

3. Combine: Merge the solutions of subproblems into the final solution.



**Binary Search (Divide & Conquer)**

using System;

class BinarySearchExample

{

static int BinarySearch(int[] arr, int left, int right, int target)

{

if (left <= right)

{

int mid = (left + right) / 2;

if (arr[mid] == target) return mid; // Found

if (arr[mid] < target)

return BinarySearch(arr, mid + 1, right, target); // Search right half

else

return BinarySearch(arr, left, mid - 1, target); // Search left half

}

return -1; // Not found

}

static void Main()

{

int[] sortedArr = { 3, 9, 10, 27, 38, 43, 82 };

int target = 27;

int result = BinarySearch(sortedArr, 0, sortedArr.Length - 1, target);

Console.WriteLine(result != -1 ? $"Found at index {result}" : "Not found");

}

}

**Output:**

Found at index 3

**When to Use Divide & Conquer?**

- Problems can be divided into similar subproblems(e.g., sorting, searching).

- Subproblems are independent (no overlapping dependencies).

- Efficiency (often reduces time complexity to O(n log n)).

**Key Advantages**

✔ Efficiency: Often faster than brute-force approaches.

✔ Parallelism: Subproblems can be solved independently (good for multi-threading).

✔Clarity: Breaks complex problems into manageable parts.