1. Definition:

Binary Search is a **divide-and-conquer** algorithm that repeatedly divides a sorted array or search space in half to find a target value or the optimal solution.

- **Time Complexity**: O(log n)
- **Space Complexity**: O(1) for iterative version, O(log n) for recursive version (due to call stack)

2. Where It Can Apply:

Binary search can be applied in **two major contexts**:

A. Classic Binary Search on Arrays

- When you are given a **sorted array**, and you need to find a particular element or condition.

B. Binary Search on Answer / Search Space

- When the **search space is numeric** or **monotonically increasing/decreasing**, and you are asked to find:
    - The minimum/maximum possible value
    - First/last position that satisfies a condition
    - Optimal value based on some constraint

3. How to Identify Binary Search Pattern:

Look for these signs:

Clue	Example
Array is **sorted**	Find an element in sorted array
Asked to find **first/last occurrence**	First bad version
Question asks for **"minimum days", "maximum capacity", "smallest/largest X such that..."**	Ship within D days, Split Array Largest Sum
You're allowed to **search within a range**	Between 1 to n, guess the number

4. How to Apply It:

Binary Search generally follows these steps:

Classic Binary Search Template:

```
int left = 0, right = n - 1;
```

```
while (left <= right) {
    int mid = left + (right - left) / 2;
    if (condition(mid)) {
        // move left or right accordingly
    }
}
```

Binary Search on Answer Template:

```
int left = minPossible, right = maxPossible;
while (left < right) {
    int mid = left + (right - left) / 2;
    if (can(mid)) {
        right = mid; // try smaller value
    } else {
        left = mid + 1; // try larger value
    }
}
return left;
```

5. Benefits:

Benefit	**Description**
Fast	O(log n) time for search
Smart brute-force	Try all values smartly in a small time
Works on custom condition checks	Doesn’t need exact target match
Saves memory	Works in-place in most cases

6. Common Problems Categories Using Binary Search:

Category	**Example Problem**
Search in Sorted Array	Binary Search, Search in Rotated Array
Find First/Last Position	First Bad Version, Find Peak Element

Min/Max Condition Value	Capacity to Ship Packages, Koko Eating Bananas
Search on Custom Function	Guess Number, H-index
Lower/Upper Bound	Insert Position, Floor/Ceil in BST

#	Problem	Difficulty	Link
✓ 1	Binary Search	Easy	🔗 Link
✓ 2	First Bad Version	Easy	🔗 Link
✓ 3	Search Insert Position	Easy	🔗 Link
✓ 4	Guess Number Higher or Lower	Easy	🔗 Link
✓ 5	Search a 2D Matrix	Medium	🔗 Link
✓ 6	Search in Rotated Sorted Array	Medium	🔗 Link
✓ 7	Find Peak Element	Medium	🔗 Link
8	Find Minimum in Rotated Sorted Array	Medium	🔗 Link
✓ 9	Koko Eating Bananas	Medium	🔗 Link
10	Minimum Absolute Sum Difference	Medium	🔗 Link
#	**Problem**	**Difficulty**	**Link**

✓ 11	Capacity To Ship Packages Within D Days	Medium	🔗 Link
12	Minimum Number of Days to Make m Bouquets	Medium	🔗 Link
13	Split Array Largest Sum	Hard	🔗 Link
14	Kth Smallest Element in a Sorted Matrix	Medium	🔗 Link
15	Median of Two Sorted Arrays	Hard	🔗 Link