Back to Chapter
Previous Next

Background
The goal of this chapter i
Template I
This chapter shows a sni
Binary Search Template I
Sqrt(x)
Guess Number Higher or ...
Search in Rotated Sorted...
Template II
This chapter shows a sni
Binary Search Template II
First Bad Version
Find Peak Element
Find Minimum in Rotated ...
Template III
This chapter shows a sni
Binary Search Template III
Search for a Range
Find K Closest Elements
Find Peak Element
Template Analysis
This chapter sums up the
Conclusion
Binary Search is an imme
More Practices
In this chapter, we have p

Binary Search Template II

Report Issue

Template #2:

C++ Java Python Copy Run Playground

```java
int binarySearch(int[] nums, int target){
  if(nums == null || nums.length == 0)
    return -1;

  int left = 0, right = nums.length - 1;
  while(left < right){
    // Prevent (left + right) overflow
    int mid = left + (right - left) / 2;
    if(nums[mid] == target){ return mid; }
    else if(nums[mid] < target) { left = mid + 1; }
    else { right = mid; }
  }

  // Post-processing:
  // End Condition: left == right
  if(nums[left] == target) return left;
  return -1;
}
```

Template #2 is an advanced form of Binary Search.

Key Attributes:

- An advanced way to implement Binary Search.
- Use the element's right neighbor to determine if the condition is met and decide whether to go left or right
- Guarantees Search Space is at least 2 in size at each step
- Post-processing required. Loop/Recursion ends when you have 1 element left. Need to assess if the remaining element meets the condition.

Distinguishing Syntax:

- Initial Condition: `left = 0, right = length - 1`
- Termination: `left == right`
- Searching Left: `right = mid`
- Searching Right: `left = mid+1`