♠

Day 18: Searching Algorithm Assignment

Linear Search Tasks

1. **Find First Occurrence**

   Given an array and a target element, write a function that uses linear search to return the index of the first occurrence of the target element. If the element is not found, return `-1`.

2. **Count Occurrences**

   Write a function that finds how many times a target element appears in an array using linear search.

3. **Find Maximum and Minimum**

   Use linear search to find the maximum and minimum values in an unsorted array.

4. **Find the Missing Number**

   You have an array of numbers from `1` to `n`, but one number is missing. Write a function to find the missing number using linear search.

5. **Check for Duplicates**

   Given an array, write a function using linear search to check if there are any duplicate elements in the array.

6. **Find the First Even Number**

   Write a function that uses linear search to find the first even number in an array. Return the index of the first even number, or `-1` if there are no even numbers.

7. **Search in a Matrix (Row-wise)**

   Given a 2D matrix, write a function to search for a target element in each row individually using linear search. Return the row and column indices if

found, otherwise return `-1`.

8. **Find All Prime Numbers**

   Given an array, use linear search to find all the prime numbers in the array and return them in a list.

9. **Find the Most Frequent Element**

   Write a function that finds the most frequently occurring element in an array using linear search. If there's a tie, return any one of the most frequent elements.

10. **Find Elements Within a Range**

    Given an array and a range `[low, high]`, use linear search to return a list of all elements within that range.

Binary Search Tasks

1. **Basic Search in Sorted Array**

   Given a sorted array and a target element, write a function that uses binary search to find the index of the target element. If the element is not found, return `-1`.

2. **Find First and Last Occurrence in a Sorted Array**

   Given a sorted array with duplicate elements, use binary search to find the index of the first and last occurrence of a target element.

3. **Find the Smallest Element Greater than Target**

   Given a sorted array and a target value, find the smallest element that is strictly greater than the target. If there's no such element, return `-1`.

4. **Square Root of an Integer**

   Write a function to find the integer part of the square root of a given number using binary search. For example, if the input is `15`, the output should be `3` (since \(3^2 = 9\) and \(4^2 = 16\)).

5. **Search in a Rotated Sorted Array**

   A sorted array has been rotated at an unknown pivot. Write a function that uses binary search to find a target element in this rotated sorted array.

6. **Find Peak Element in a Mountain Array**

A "mountain array" is an array where elements strictly increase to a maximum value and then strictly decrease. Use binary search to find the peak element (the maximum value) in a mountain array.

7. **Find the Insert Position**

   Given a sorted array and a target value, use binary search to find the index at which the target should be inserted to keep the array sorted. If the target is already present, return its index.

8. **Find Missing Number in Sorted Array**

   Given a sorted array of unique numbers ranging from `0` to `n`, but with one number missing, find the missing number using binary search.

9. **Find the Number of Occurrences in a Sorted Array**

   Given a sorted array with duplicate elements, use binary search to find the number of occurrences of a target element.

10. **Binary Search in a Matrix**

    Given a 2D matrix where each row and column is sorted, write a function to search for a target element using binary search logic. Return the row and column indices if found, otherwise return `-1`.
