**Logical Programs** :

**SET 1:**

Write a function **isPalindrome(str)** that takes a string as input and returns **true** if the string is a palindrome (reads the same forwards and backwards), and **false** otherwise.

Write a function **isPrime(num)** that takes a number as input and returns **true** if the number is prime (only divisible by 1 and itself), and **false** otherwise.

Write a function **getFibonacci(n)** that takes a number **n** as input and returns an array containing the first **n** Fibonacci numbers.

Write a function **factorial(n)** that takes a number **n** as input and returns the factorial of **n**.

Write a function **getDuplicates(arr)** that takes an array as input and returns an array containing the duplicate elements.

Write a function **capitalizeFirstLast(str)** that takes a string as input and capitalizes the first and last letter of each word.

Write a function **binarySearch(arr, target)** that takes a sorted array and a target value as input and returns the index of the target value in the array, or -1 if the target value is not found.

Write a function **throttle(fn, limit)** that takes a function **fn** and a limit in milliseconds as input and returns a throttled version of the function that will only execute once within the specified time limit, ignoring subsequent function invocations.

Write a function **shuffleArray(arr)** that takes an array as input and returns a shuffled version of the array.

Write a function **deepClone(obj)** that takes an object as input and returns a deep clone (a completely independent copy) of the object.

**SET 2:**

Write a function **reverseWords(str)** that takes a string as input and returns a new string with the words reversed.

Write a function **isAnagram(str1, str2)** that takes two strings as input and returns **true** if they are anagrams (contain the same characters in a different order), and **false** otherwise.

Write a function **truncateString(str, num)** that takes a string and a number as input and returns a truncated version of the string if it is longer than the specified number of characters, adding an ellipsis ("...") to the end.

Write a function **sumArray(arr)** that takes an array of numbers as input and returns the sum of all the numbers.

Write a function **capitalizeLongWords(str, num)** that takes a string and a number as input and returns a new string with all words longer than the specified number of characters capitalized.

Write a function **getDistinctValues(arr)** that takes an array as input and returns a new array with only the distinct values (removing any duplicates).

Write a function **isArmstrongNumber(num)** that takes a number as input and returns **true** if it is an Armstrong number, and **false** otherwise. An Armstrong number is a number that is equal to the sum of its own digits each raised to the power of the number of digits.

Write a function **countOccurrences(arr, target)** that takes an array and a target value as input and returns the number of occurrences of the target value in the array.

Write a function **flattenObject(obj)** that takes an object with nested objects as input and returns a flattened object with nested keys combined using dot notation.

Write a function **calculateAverage(arr)** that takes an array of numbers as input and returns the average (mean) value of the numbers.