ADS Assignment

Problem 1:

Given an array of integers, perform the following operations:

1. **Find the second largest element in the array.**
2. **Move all zeros to the end of the array while maintaining the order of non-zero elements.**

Input:

arr = [10, 0, 5, 20, 0, 8, 15]

Output:

Second largest element: 15

Array after moving zeros: [10, 5, 20, 8, 15, 0, 0]

Constraints:

- Do not use built-in sort functions.
- The array may contain duplicate elements or zeros at any position.
- Array length ≥ 2.

Problem 2:

Write a program that performs the following operations on strings:

1. **Check whether two given strings are anagrams of each other.**
2. **Identify the longest word in a given sentence.**
3. **Count the number of vowels and consonants in the same sentence.**

Input:

String 1: listen

String 2: silent

Sentence: Practice makes a man perfect

Output:

Are 'listen' and 'silent' anagrams? true

Longest word: Practice

Vowels: 9, Consonants: 17

Problem 3:

Given a **sorted array of integers** (which may include duplicates), perform the following operations:

1. **Search for a given key and return its index (if found) with Binary Search.**
2. **Find the first and last occurrence of the key in the array.**
3. **Count the total number of times the key appears.**
4. **Find any peak element in the array (an element greater than its neighbors).**

Input:

arr = [1, 3, 3, 3, 5, 6, 8], key = 3

Input for Peak Element:

arr =[1, 2, 18, 4, 5, 0]

Output:

Key found at index: 2

First occurrence: 1

Last occurrence: 3

Total count of key: 3

Peak element: 18

Problem 4:

Write a recursive program that performs the following operations:

1. **Check if a number is prime using recursion**.
2. **Check whether a given string is a palindrome.**
3. **Find the sum of digits of a given number.**
4. **Calculate the nth Fibonacci number.**
5. **Calculate a raised to the power b**

Input:

num = 7
str = "racecar"
num = 1234
fibIndex = 6
a = 2, b = 5

Output:

Is prime: true
Is 'racecar' a palindrome? true
Sum of digits of 1234: 10
Fibonacci(6): 8
2^5 = 32

Constraints:

- Do not use loops or built-in reverse methods.
- Use charAt() for string access.
- You can assume valid positive integer inputs.

Problem 5:

Dry Run & Analyze: Time and Space Complexity

1. Dry run the code for n = 4. How many times is * printed? What is the time complexity?

```
 void printTriangle(int n) {
    for (int i = 0; i < n; i++)
        for (int j = 0; j <= i; j++)
            System.out.print("*");
}
```

2. Dry run for n = 8. What's the number of iterations? Time complexity?

```
void printPattern(int n) {
    for (int i = 1; i <= n; i *= 2)
        for (int j = 0; j < n; j++)
            System.out.println(i + "," + j);
}
```

3. Dry run for n = 20. How many recursive calls? What values are printed?

```
void recHalf(int n) {
    if (n <= 0) return;
    System.out.print(n + " ");
    recHalf(n / 2);
}
```

4. Dry run for n = 3. How many total calls are made? What's the time complexity?

```
void fun(int n) {
   if (n == 0) return;
   fun(n - 1);
   fun(n - 1);
}
```

5. Dry run for n = 3. How many total iterations? Time complexity?

```
void tripleNested(int n) {
   for (int i = 0; i < n; i++)
      for (int j = 0; j < n; j++)
         for (int k = 0; k < n; k++)
            System.out.println(i + j + k);
}
```