Problem 0: Fibonacci

https://github.com/vinaykumarvallepu/CSE_5311_Hands_On_4/blob/main/fibonacci.java

fib(5) -> fib(4) + fib(3)

fib(4) -> fib(3) + fib(2)

fib(3) -> fib(2) + fib(1)

fib(2) -> fib(1) + fib(0)

fib(2) -> fib(1) + fib(0)

fib(3) -> fib(2) + fib(1)

fib(2) -> fib(1) + fib(0)

fib(5) = 5

Problem 1:

1.Code:https://github.com/vinaykumarvallepu/CSE_5311_Hands_On_4/blob/main/kSorted.java

2.Time complexity: Time complexity of this program is O(N*K*log K) where 'N' is no of entries in each array and 'K' is the no of arrays. It divides into half and at each array is traversal. At each level it merges two arrays of size of N, time takes for the operation is O(n). Time Complexity is O(N*K*log(K)).

3.Improvement:

- **Using Min heap**: Using the priority Queue (Min heap) which reduces the time complexity as the operations takes O(log(K)) time.
- **Using divide and Conquer method**.

Problem 2:

1.Code: https://github.com/vinaykumarvallepu/CSE_5311_Hands_On_4/blob/main/removeDuplicate.java

2. Time Complexity: This algorithms traversal only once. So, the time complexity is O(n).

3.Improvement: This approach is already effective with time complexity of O(n) or we can use Binary Search to solve.