```java
import java.util.SortedSet;
import java.util.TreeSet;

public class LongestIncreasingSubsequence {

    // Function to find the length of the longest increasing subsequence
    static int findLISLength(int[] arr, int n) {
        // Create a sorted set to hold the unique elements
        SortedSet<Integer> sortedUniqueElements = new TreeSet<>();
        for (int i = 0; i < n; i++) {
            sortedUniqueElements.add(arr[i]);
        }

        // Convert the sorted set to an array
        int[] sortedArray = new int[sortedUniqueElements.size()];
        int index = 0;
        for (int val : sortedUniqueElements) {
            sortedArray[index++] = val;
        }

        // Dynamic programming table
        int[][] dp = new int[index + 1][n + 1];

        // Fill the DP table
        for (int i = 0; i <= index; i++) {
            for (int j = 0; j <= n; j++) {
                if (i == 0 || j == 0) {
                    dp[i][j] = 0;
                } else if (arr[j - 1] == sortedArray[i - 1]) {
                    dp[i][j] = dp[i - 1][j - 1] + 1;
                } else {
                    dp[i][j] = Math.max(dp[i - 1][j], dp[i][j - 1]);
                }
            }
        }

        // The bottom-right corner of dp array holds the length of LIS
        return dp[index][n];
    }

    public static void main(String[] args) {
        int[] arr = {10, 22, 9, 33, 21, 50, 41, 60};
        int n = arr.length;
        System.out.println("Length of LIS is " + findLISLength(arr, n) +
"\n");
    }
}
```