```java
package array;


/* 已知三个升序整数数组a[l]、b[m]、c[n]，在3个数组中各找一个元素，使得组成的三元组距离最小。
三元组的距离定义是：假设a[i]、b[j]、c[k]是一个三元组，那么距离为 max( abs(a[i]-b[j]), abs(a[i]-c[k]), abs(b[j]-c[k]) ) */

class FindMinDistanceOfThreeArrays {

    public static Integer findMinDistanceOfThreeArrays(int[] a, int[] b, int[] c) {

        if (a == null || b == null || c == null) {
            return null;
        }

        int minDistance = Integer.MAX_VALUE;

        int pointerOfA = 0, pointerOfB = 0, pointerOfC = 0;
        while (pointerOfA < a.length
                && pointerOfB < b.length
                && pointerOfC < c.length) {

            if (a[pointerOfA] <= b[pointerOfB] && a[pointerOfA] <= c[pointerOfC]) {
                minDistance = Math.min(minDistance, Math.max(b[pointerOfB], c[pointerOfC]) - a[pointerOfA]);
                pointerOfA++;

            } else if (b[pointerOfB] <= c[pointerOfC] && b[pointerOfB] <= a[pointerOfA]) {
                minDistance = Math.min(minDistance, Math.max(a[pointerOfA], c[pointerOfC]) - b[pointerOfB]);
                pointerOfB++;

            } else if (c[pointerOfC] <= b[pointerOfB] && c[pointerOfC] <= a[pointerOfA]) {
                minDistance = Math.min(minDistance, Math.max(a[pointerOfA], b[pointerOfB]) - c[pointerOfC]);
                pointerOfC++;

            }
        }

        return minDistance;
    }


    public static void main(String[] args) {

        int[] a = {3, 4, 5, 11};
        int[] b = {10, 12, 14, 16, 17};
        int[] c = {20, 21, 23, 24, 37, 40};

        System.out.println(findMinDistanceOfThreeArrays(a, b, c));
    }
}
```