

```

1 package array;
2
3
4 /*
5 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
6 小。
7 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(
8 abs(a[i]-b[j]), abs(a[i]-c[k]), abs(b[j]-c[k]) ) */
9
10 class FindMinDistanceOfThreeArrays {
11
12     public static Integer findMinDistanceOfThreeArrays(int[] a, int[] b, int[] c) {
13
14         if (a == null || b == null || c == null) {
15             return null;
16         }
17
18         int minDistance = Integer.MAX_VALUE;
19
20         int pointerOfA = 0, pointerOfB = 0, pointerOfC = 0;
21         while (pointerOfA < a.length
22             && pointerOfB < b.length
23             && pointerOfC < c.length) {
24
25             if (a[pointerOfA] <= b[pointerOfB] && a[pointerOfA] <= c[pointerOfC]) {
26                 minDistance = Math.min(minDistance, Math.max(b[pointerOfB],
27                     c[pointerOfC]) - a[pointerOfA]);
28                 pointerOfA++;
29
30             } else if (b[pointerOfB] <= c[pointerOfC] && b[pointerOfB] <=
31                 a[pointerOfA]) {
32                 minDistance = Math.min(minDistance, Math.max(a[pointerOfA],
33                     c[pointerOfC]) - b[pointerOfB]);
34                 pointerOfB++;
35
36             } else if (c[pointerOfC] <= b[pointerOfB] && c[pointerOfC] <=
37                 a[pointerOfA]) {
38                 minDistance = Math.min(minDistance, Math.max(a[pointerOfA],
39                     b[pointerOfB]) - c[pointerOfC]);
40                 pointerOfC++;
41             }
42         }
43
44         return minDistance;
45     }
46
47     public static void main(String[] args) {
48
49         int[] a = {3, 4, 5, 11};
50         int[] b = {10, 12, 14, 16, 17};
51         int[] c = {20, 21, 23, 24, 37, 40};
52
53         System.out.println(findMinDistanceOfThreeArrays(a, b, c));
54     }
55 }

```