【模拟】整数对最小和

题目描述与示例

题目描述

给定两个整数数组 `array1` 、 `array2` ，数组元素按升序排列。假设从 `array1` 、 `array2` 中分别取出一个元素可构成一对元素，现在需要取出 `k` 对元素，并对取出的所有元素求和计算和的最小值。

注意：两对元素如果对应于 `array1` 、 `array2` 中的两个下标均相同，则视为同一对元素。

输入描述

输入两行数组 `array1` 、 `array2` ，每行首个数字为数组大小 `size(0 < size <= 100)`

`0 < array1[i] <= 1000`

`0 < array2[i] <= 1000`

接下来一行为正整数 `k`

`0 < k <= array1.size()*array2.size()`

输出描述

满足要求的最小和

示例

输入

```
1 3 1 1 2
2 3 1 2 3
3 2
```

输出

```
4
```

说明

用例中，需要取 `2` 对元素

取第一个数组第 `0` 个元素与第二个数组第 `0` 个元素组成 `1` 对元素 `[1,1]`；

取第一个数组第 `1` 个元素与第二个数组第 `0` 个元素组成 `1` 对元素 `[1,1]`；

求和为 `1+1+1+1=4` ，为满足要求的最小和。

解题思路

读懂题目就能完成这题。

因为数据量不大，两个数组的长度均不超过 `100` ，所以整数对最多的数目仅为 `100*100 = 10000` `O(n1n2log(n1n2))` 的时间复杂度是可以通过这个题目的。

整体流程即为

1. 输入数据，初始化列表 `ans`
2. 双重遍历两个数组，获得所有的整数对的和 `nums1[i] + nums2[j]` ，加入 `ans` 中。该过程的时间复杂度为 `O(n1n2)` 。
3. 对 `ans` 进行从小到大的排序，该过程的时间复杂度为 `O(n1n2log(n1n2))` 。
4. 取 `ans` 中前 `k` 个元素进行求和并输出

取前 `k` 个元素的过程可以用优先队列进行优化，可以将排序的时间复杂度降到 `O(klogk)` 。

代码

Python

```python
# 题目：【模拟】2024E-整数对最小和
# 分值：100
```

```python
# 作者：许老师-闭着眼睛学数理化
# 算法：模拟
# 代码看不懂的地方，请直接在群上提问


# 输入数组长度n1，数组nums1
lst1 = list(map(int, input().split()))
n1 = lst1[0]
nums1 = lst1[1:]

# 输入数组长度n2，数组nums2
lst2 = list(map(int, input().split()))
n2 = lst2[0]
nums2 = lst2[1:]

# 输入k
k = int(input())

ans = list()

# 双重遍历所有整数对
for i in range(n1):
    for j in range(n2):
        # 将整数对的和加入ans中
        ans.append(nums1[i] + nums2[j])

# 对ans进行从小到大排序
ans.sort()
# 取ans中前k个元素进行求和并输出，即为答案
print(sum(ans[:k]))
```

Java

```java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // 输入数组长度n1，数组nums1
        String[] input1 = scanner.nextLine().split(" ");
        int n1 = Integer.parseInt(input1[0]);
        int[] nums1 = new int[n1];
        for (int i = 0; i < n1; i++) {
            nums1[i] = Integer.parseInt(input1[i + 1]);
```

```
        }

        // 输入数组长度n2，数组nums2
        String[] input2 = scanner.nextLine().split(" ");
        int n2 = Integer.parseInt(input2[0]);
        int[] nums2 = new int[n2];
        for (int i = 0; i < n2; i++) {
            nums2[i] = Integer.parseInt(input2[i + 1]);
        }

        // 输入k
        int k = scanner.nextInt();

        List<Integer> ans = new ArrayList<>();

        // 双重遍历所有整数对
        for (int i = 0; i < n1; i++) {
            for (int j = 0; j < n2; j++) {
                // 将整数对的和加入ans中
                ans.add(nums1[i] + nums2[j]);
            }
        }

        // 对ans进行从小到大排序
        Collections.sort(ans);
        // 取ans中前k个元素进行求和并输出，即为答案
        int sum = 0;
        for (int i = 0; i < k; i++) {
            sum += ans.get(i);
        }

        System.out.println(sum);
        scanner.close();
    }
}

```

C++

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

```

```cpp
int main() {
    // 输入数组长度n1，数组nums1
    int n1;
    cin >> n1;
    vector<int> nums1(n1);
    for (int i = 0; i < n1; ++i) {
        cin >> nums1[i];
    }

    // 输入数组长度n2，数组nums2
    int n2;
    cin >> n2;
    vector<int> nums2(n2);
    for (int i = 0; i < n2; ++i) {
        cin >> nums2[i];
    }

    // 输入k
    int k;
    cin >> k;

    vector<int> ans;

    // 双重遍历所有整数对
    for (int i = 0; i < n1; ++i) {
        for (int j = 0; j < n2; ++j) {
            // 将整数对的和加入ans中
            ans.push_back(nums1[i] + nums2[j]);
        }
    }

    // 对ans进行从小到大排序
    sort(ans.begin(), ans.end());
    // 取ans中前k个元素进行求和并输出，即为答案
    int sum = 0;
    for (int i = 0; i < k; ++i) {
        sum += ans[i];
    }

    cout << sum << endl;

    return 0;
}

```

时空复杂度

时间复杂度：`O(n1n2log(n1n2))`。排序所需的时间复杂度。

空间复杂度：`O(1)`。仅需若干常数变量