题目描述：

给定一个随机的整数（可能存在正整数和负整数）数组 nums ，请你在该数组中找出两个数，其和的绝对值(|nums[x]+nums[y]|)为最小值，并返回这个两个数（按从小到大返回）以及绝对值。
每种输入只会对应一个答案。但是，数组中同一个元素不能使用两遍。

输入描述：

一个通过空格分割的有序整数序列字符串，最多 1000 个整数，且整数数值范围是 [-65535, 65535]。

输出描述：

两数之和绝对值最小值

示例 1

输入：

```
-1 -3 7 5 11 15
```

输出：

```
-3 5 2
```

说明：

因为 |nums[0] + nums[2]| = |-3 + 5| = 2 最小，所以返回 -3 5 2

```
#include <iostream>
#include <sstream>
#include <vector>
using namespace std;

class Solution {
public:
    // Time Complexity: O(max(nlogn, m*k), Space Complexity: O(1)  不算原数组 nums 的空间
    void minABS(vector<int>& nums, vector<int>& res) {
        // sort, 非递减序
        //sort(nums.begin(), nums.end());

        int len = nums.size();
        Qsort(nums, 0, len - 1);

        if (nums[0] >= 0) { //  全是正数，0
```

```
            res[0] = nums[0];
            res[1] = nums[1];
            res[2] = nums[0] + nums[1];
        }
        else if (nums[len - 1] <= 0) { // 全是负数，0
            res[0] = nums[len - 2];
            res[1] = nums[len - 1];
            res[2] = -(nums[len - 2] + nums[len - 1]);
        }
        else { // 负、正均有
            // 找到第一个正数的索引
            int idx = 0;
            while (idx < len && nums[idx] < 0) idx++;
            for (int i = 0; i < idx; i++) {
                for (int j = idx; j < len; j++) {
                    int abs_val = abs(nums[i] + nums[j]);
                    if (abs_val < res[2]) { // 找到绝对值更小的数，更新结果
                        res[0] = nums[i];
                        res[1] = nums[j];
                        res[2] = abs_val;
                    }
                }
            }
        }
}

void Qsort(vector<int>& nums, int start, int end) { // 对 nums[start, end]进行快速排序
        if (start < end) {
            int mid = partition(nums, start, end);
            Qsort(nums, start, mid - 1);
            Qsort(nums, mid + 1, end);
        }
}

int partition(vector<int>& nums, int start, int end) { // 一次分划，左侧<=，右侧>
        int i = start + 1;
        int j = end;
        while (i <= j) {
            while (i < nums.size() && nums[i] <= nums[start]) i++;
            while (nums[j] > nums[start]) j--;
            if(i < j) swap(nums[i], nums[j]);
        }
        swap(nums[start], nums[j]);
        return j;
```

```
    }
};


int main() {
    vector<int> nums; //  输入序列
    string input;
    getline(cin, input);
    stringstream ss(input);
    string split;
    while (getline(ss, split, ' ')) {
        nums.push_back(atoi(split.c_str()));
    }

    vector<int> res(3);
    res[2] = 65535*2; // (1 << 31) - 1;
    Solution sol;
    sol.minABS(nums, res);
    for (int i = 0; i < 3; i++) { //  输出结果
        cout << res[i] << " ";
    }
    return 0;
}
```