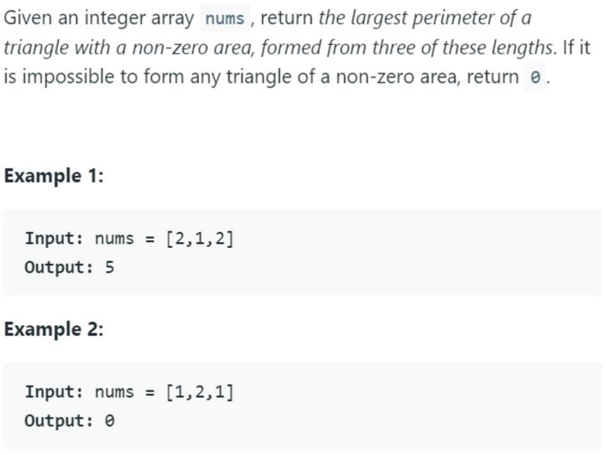
976. Largest Perimeter Triangle

20 June 2022 09:20 AM



Given an integer array `nums`, return *the largest perimeter of a triangle with a non-zero area, formed from three of these lengths*. If it is impossible to form any triangle of a non-zero area, return `0`.

Example 1:

```
Input: nums = [2,1,2]
Output: 5
```

Example 2:

```
Input: nums = [1,2,1]
Output: 0
```

25	6	9	11	8	12	10	3	2
0	1	2	3	4	5	6	7	8

Largest

S1 = 25

S2 = 11

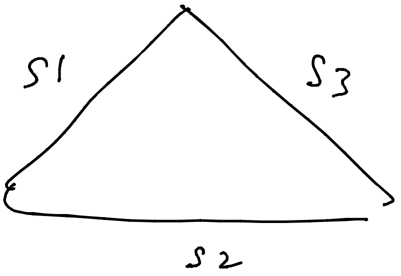
S3 = 12

X

$\boxed{S2 + S3 > S1}$

X

11 + 12 > 25



This doesn't work, so we sort the array

Sort

2	3	6	8	9	10	11	12	25
0	1	2	3	4	5	6	7	8

$\underbrace{6 \quad 7 \quad 8}$

↓

Cannot form traingle.

$$\left.\begin{array}{l} S1 + S2 > S3 \\ S2 + S3 > S1 \\ S1 + S3 > S2 \end{array}\right\} \rightarrow \boxed{S1 < S2 < S3}$$

for e.g

7 9 15

$S1 + S2 > S3$ ✓

$S2 + S3 > S1$ ✓

$S1 + S3 > S2$ ✓

2	3	6	8	9	10	11	12	25
0	1	2	3	4	5	6	7	8

i-2 i-1 i

peri = $\cancel{0}$ 33

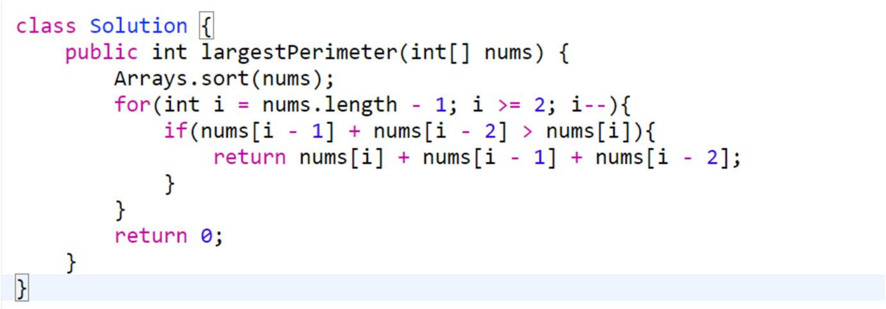
$$arr[i-1] + arr[i-2] > arr[i]$$

$11 + 10 > 12$

$\boxed{21 > 12}$ ✓

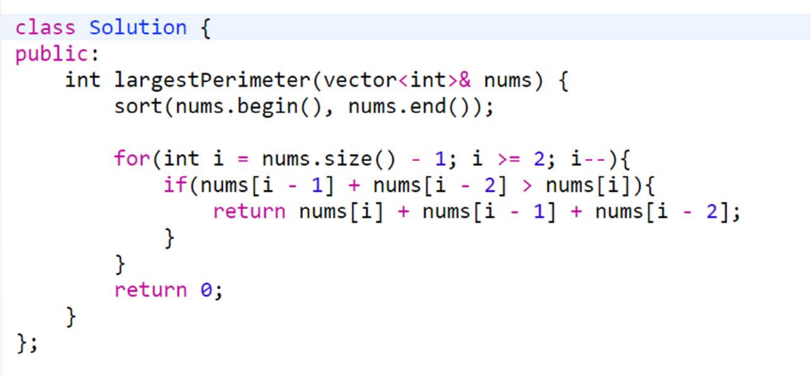
and the largest peri = 11 + 10 + 12 = 33.

Java



```
class Solution {
    public int largestPerimeter(int[] nums) {
        Arrays.sort(nums);
        for(int i = nums.length - 1; i >= 2; i--){
            if(nums[i - 1] + nums[i - 2] > nums[i]){
                return nums[i] + nums[i - 1] + nums[i - 2];
            }
        }
        return 0;
    }
}
```

C++



```
class Solution {
public:
    int largestPerimeter(vector<int>& nums) {
        sort(nums.begin(), nums.end());

        for(int i = nums.size() - 1; i >= 2; i--){
            if(nums[i - 1] + nums[i - 2] > nums[i]){
                return nums[i] + nums[i - 1] + nums[i - 2];
            }
        }
        return 0;
    }
};
```