

# Problem 3588: Find Maximum Area of a Triangle

## Problem Information

Difficulty: **Medium**

Acceptance Rate: 0.00%

Paid Only: No

## Problem Description

You are given a 2D array

`coords`

of size

$n \times 2$

, representing the coordinates of

$n$

points in an infinite Cartesian plane.

Find

twice

the

maximum

area of a triangle with its corners at

any

three elements from

coords

, such that at least one side of this triangle is

parallel

to the x-axis or y-axis. Formally, if the maximum area of such a triangle is

A

, return

$2 * A$

.

If no such triangle exists, return -1.

Note

that a triangle

cannot

have zero area.

Example 1:

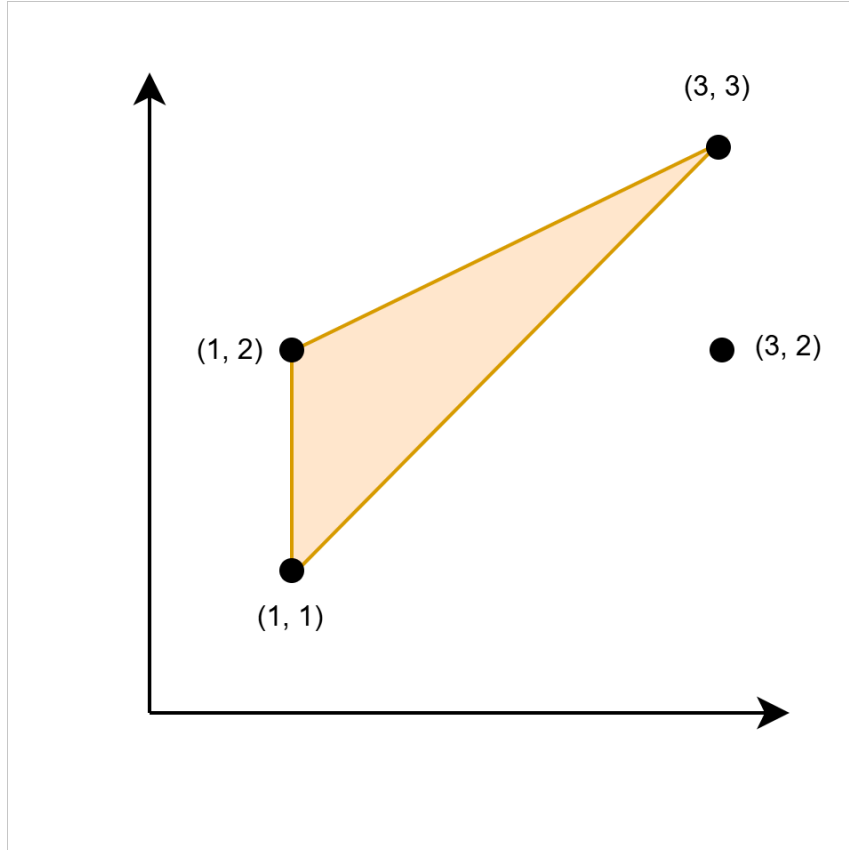
Input:

coords = [[1,1],[1,2],[3,2],[3,3]]

Output:

2

Explanation:



The triangle shown in the image has a base 1 and height 2. Hence its area is

$$\frac{1}{2} * \text{base} * \text{height} = 1$$

.

Example 2:

Input:

coords = [[1,1],[2,2],[3,3]]

Output:

-1

Explanation:

The only possible triangle has corners

(1, 1)

,

(2, 2)

, and

(3, 3)

. None of its sides are parallel to the x-axis or the y-axis.

Constraints:

$1 \leq n \leq \text{coords.length} \leq 10$

5

$1 \leq \text{coords}[i][0], \text{coords}[i][1] \leq 10$

6

All

`coords[i]`

are

unique

.

## Code Snippets

**C++:**

```

class Solution {
public:
    long long maxArea(vector<vector<int>>& coords) {

    }

};

```

### Java:

```

class Solution {
    public long maxArea(int[][] coords) {

    }

}

```

### Python3:

```

class Solution:
    def maxArea(self, coords: List[List[int]]) -> int:

```

### Python:

```

class Solution(object):
    def maxArea(self, coords):
        """
        :type coords: List[List[int]]
        :rtype: int
        """

```

### JavaScript:

```

/**
 * @param {number[][]} coords
 * @return {number}
 */
var maxArea = function(coords) {

};

```

### TypeScript:

```

function maxArea(coords: number[][]): number {

```

```
};
```

### C#:

```
public class Solution {  
    public long MaxArea(int[][] coords) {  
  
    }  
}
```

### C:

```
long long maxArea(int** coords, int coordsSize, int* coordsColSize) {  
  
}
```

### Go:

```
func maxArea(coords [][]int) int64 {  
  
}
```

### Kotlin:

```
class Solution {  
    fun maxArea(coords: Array<IntArray>): Long {  
  
    }  
}
```

### Swift:

```
class Solution {  
    func maxArea(_ coords: [[Int]]) -> Int {  
  
    }  
}
```

### Rust:

```
impl Solution {  
    pub fn max_area(coords: Vec<Vec<i32>>) -> i64 {
```

```
}  
}
```

### Ruby:

```
# @param {Integer[][]} coords  
# @return {Integer}  
def max_area(coords)  
  
end
```

### PHP:

```
class Solution {  
  
    /**  
     * @param Integer[][] $coords  
     * @return Integer  
     */  
    function maxArea($coords) {  
  
    }  
}
```

### Dart:

```
class Solution {  
    int maxArea(List<List<int>> coords) {  
  
    }  
}
```

### Scala:

```
object Solution {  
    def maxArea(coords: Array[Array[Int]]): Long = {  
  
    }  
}
```

### Elixir:

```

defmodule Solution do
  @spec max_area(coords :: [[integer]]) :: integer
  def max_area(coords) do

  end

  end

```

## Erlang:

```

-spec max_area(Coords :: [[integer()]]) -> integer().
max_area(Coords) ->
.

```

## Racket:

```

(define/contract (max-area coords)
  (-> (listof (listof exact-integer?)) exact-integer?)
)

```

# Solutions

## C++ Solution:

```

/*
 * Problem: Find Maximum Area of a Triangle
 * Difficulty: Medium
 * Tags: array, greedy, math, hash
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

class Solution {
public:
    long long maxArea(vector<vector<int>>& coords) {

    }

};

```

## Java Solution:

```

/**
 * Problem: Find Maximum Area of a Triangle
 * Difficulty: Medium
 * Tags: array, greedy, math, hash
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

class Solution {
public long maxArea(int[][] coords) {

}

}

```

### Python3 Solution:

```

"""
Problem: Find Maximum Area of a Triangle
Difficulty: Medium
Tags: array, greedy, math, hash

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) for hash map
"""

class Solution:
def maxArea(self, coords: List[List[int]]) -> int:
# TODO: Implement optimized solution
pass

```

### Python Solution:

```

class Solution(object):
def maxArea(self, coords):
"""
:type coords: List[List[int]]
:rtype: int
"""

```

## JavaScript Solution:

```
/**
 * Problem: Find Maximum Area of a Triangle
 * Difficulty: Medium
 * Tags: array, greedy, math, hash
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

/**
 * @param {number[][]} coords
 * @return {number}
 */
var maxArea = function(coords) {

};
```

## TypeScript Solution:

```
/**
 * Problem: Find Maximum Area of a Triangle
 * Difficulty: Medium
 * Tags: array, greedy, math, hash
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) for hash map
 */

function maxArea(coords: number[][]): number {

};
```

## C# Solution:

```
/*
 * Problem: Find Maximum Area of a Triangle
 * Difficulty: Medium
 * Tags: array, greedy, math, hash
 */
```

```

* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

public class Solution {
public long MaxArea(int[][] coords) {

}

}

```

### C Solution:

```

/*
* Problem: Find Maximum Area of a Triangle
* Difficulty: Medium
* Tags: array, greedy, math, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

long long maxArea(int** coords, int coordsSize, int* coordsColSize) {

}

```

### Go Solution:

```

// Problem: Find Maximum Area of a Triangle
// Difficulty: Medium
// Tags: array, greedy, math, hash
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

func maxArea(coords [][]int) int64 {

}

```

### Kotlin Solution:

```
class Solution {  
    fun maxArea(coords: Array<IntArray>): Long {  
  
    }  
}
```

### Swift Solution:

```
class Solution {  
    func maxArea(_ coords: [[Int]]) -> Int {  
  
    }  
}
```

### Rust Solution:

```
// Problem: Find Maximum Area of a Triangle  
// Difficulty: Medium  
// Tags: array, greedy, math, hash  
//  
// Approach: Use two pointers or sliding window technique  
// Time Complexity: O(n) or O(n log n)  
// Space Complexity: O(n) for hash map  
  
impl Solution {  
    pub fn max_area(coords: Vec<Vec<i32>>) -> i64 {  
  
    }  
}
```

### Ruby Solution:

```
# @param {Integer[][]} coords  
# @return {Integer}  
def max_area(coords)  
  
end
```

### PHP Solution:

```

class Solution {

    /**
     * @param Integer[][] $coords
     * @return Integer
     */
    function maxArea($coords) {

    }

}

```

### Dart Solution:

```

class Solution {
  int maxArea(List<List<int>> coords) {

  }

}

```

### Scala Solution:

```

object Solution {
  def maxArea(coords: Array[Array[Int]]): Long = {

  }

}

```

### Elixir Solution:

```

defmodule Solution do
  @spec max_area(coords :: [[integer]]) :: integer
  def max_area(coords) do

  end

end

```

### Erlang Solution:

```

-spec max_area(Coords :: [[integer()]]) -> integer().
max_area(Coords) ->
.

```

### Racket Solution:

```
(define/contract (max-area coords)
  (-> (listof (listof exact-integer?)) exact-integer?)
)
```