Storage Optimization

3. Storage Optimization

Amazon is experimenting with a flexible storage system for their warehouses. The storage unit consists of a shelving system which is one meter deep with removable vertical and horizontal separators. When all separators are installed, each storage space is one cubic meter (1' x 1' x 1'). Determine the volume of the largest space when a series of horizontal and vertical separators are removed.

Example

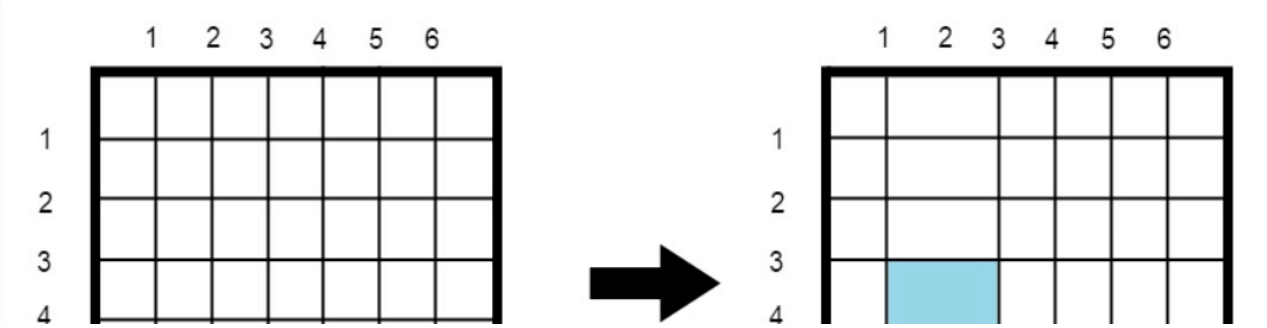
n = 6

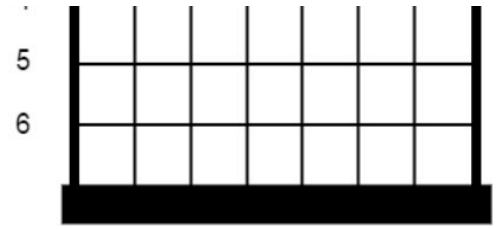
m = 6

h = [4]

v = [2]

Consider the diagram below. The left image depicts the initial storage unit with *n = 6* horizontal and *m = 6* vertical separators, where the volume of the largest storage space is *1 × 1 x 1*. The right image depicts that unit after the fourth horizontal and *second vertical separators are removed. The maximum storage volume for that unit is then 2 × 2 x 1 = 4 cubic meters:*





Function Description

Complete the function storage in the editor below.

storage has the following parameter(s):

int n: integer, the number of horizontal separators initially

int m: integer, the number of vertical separators initially

int h[x]: an array of integers, the horizontal separators to remove

int v[y]: an array of integers, the vertical separators to remove

Returns:

int: a long integer denoting the volume of the largest item that can be stored in the unit.

Constraints

- $1 \le n, m \le 10^5$
- $0 < x \le$ n
- $0 < y \le$ m
- $1 \le h[i] \le n$, where $1 \le i \le n$.
- $1 \le v[j] \le m$, where $1 \le j \le m$.
- The values in array *h* are distinct.
- The values in array *v* are distinct.

▶ Input Format for Custom Testing

▼ Sample Case 0

Sample Input 0

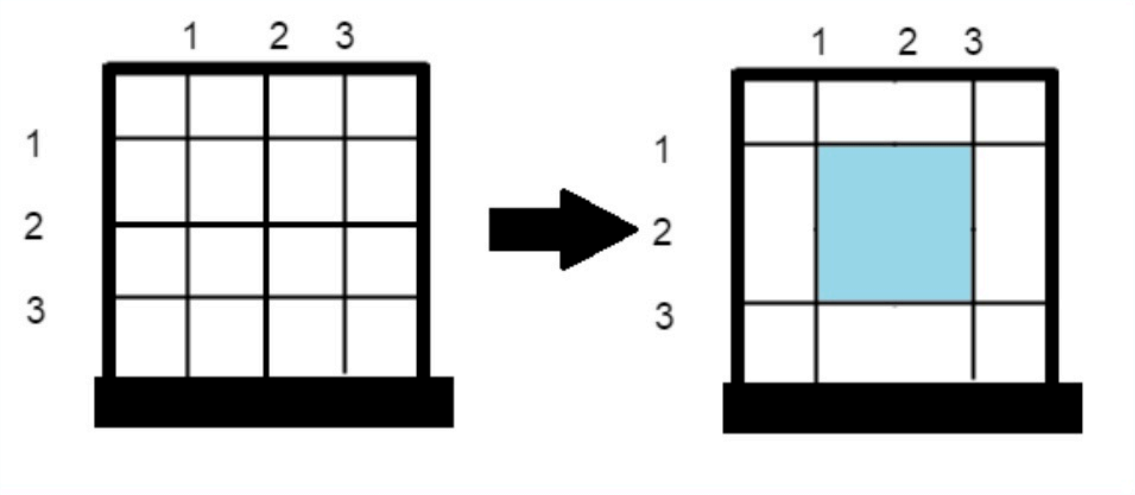
```
STDIN    Function
-----    --------
3    →   n = 3
3    →   m = 3
1    →   h[] size x = 1
2    →   h = [2]
1    →   v[] size y = 1
2    →   v = [2]
```

Sample Output 0

```
4
```

Explanation 0

There are *n = m = 3* separators in the vertical and horizontal directions. Separators to remove are *h = [2]* and *v = [2]* so the unit looks like this:



Return the volume of the biggest space, *4*, as the answer.

▼ Sample Case 1

Sample Input 1

```
STDIN    Function Parameters
-----    -------------------
2    →   n = 2
```

```
2    →   m = 2
1    →   h[] size x = 1
1    →   h = [1]
1    →   v[] size y = 1
2    →   v = [2]
```

Sample Output 1

```
4
```

Explanation 1

There are 2 vertical and two horizontal separators initially. After removing the two separators, *h = [1]* and *v = [2]*, the top-right cell will be the largest storage space at 4 cubic meters.

▼ Sample Case 2

Sample Input 2

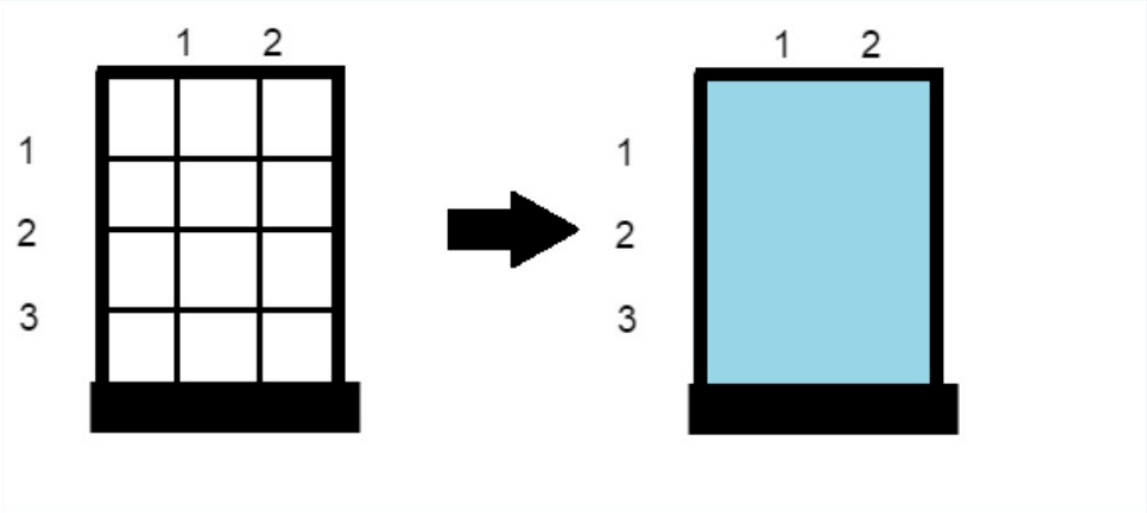
```
STDIN    Function
-----    --------
3    →   n = 3
2    →   m = 2
3    →   h[] size x = 3
1    →   h = [1, 2, 3]
2
3
2    →  v[] size y = 3
1    →  v = [1, 2]
2
```

Sample Output 2

```
12
```

Explanation 2

Initially there are *n = 3* horizontal and *m = 2* vertical separators. Remove separators *h = [1, 2, 3]* and *v = [1, 2]* so the unit looks like this:



The volume of the biggest storage space is *12 cubic meters*.

思路

横轴取最长的gap, 纵轴取最长的gap. 把两个乘到一起就是最大面积

力扣相似题目

https://leetcode-cn.com/problems/maximum-area-of-a-piece-of-cake-after-horizontal-and-vertical-cuts/

Code

```python
def storage(n, m, h, v):
    total_rows = n + 2
    total_column = m + 2
    row_set = set(i for i in range(total_rows))
    col_set = set(i for i in range(total_column))
    for removed_row in h:
        row_set.remove(removed_row)
    for removed_column in v:
```

```
            col_set.remove(removed_column)
        row_list = sorted(list(row_set))
        col_list = sorted(list(col_set))
        row_gap_max, col_gap_max = 0, 0
        for i in range(1, len(row_list)):
            row_gap_max = max(row_gap_max, row_list[i] - row_list[i - 1])
        for i in range(1, len(col_list)):
            col_gap_max = max(col_gap_max, col_list[i] - col_list[i - 1])
        return row_gap_max * col_gap_max
```