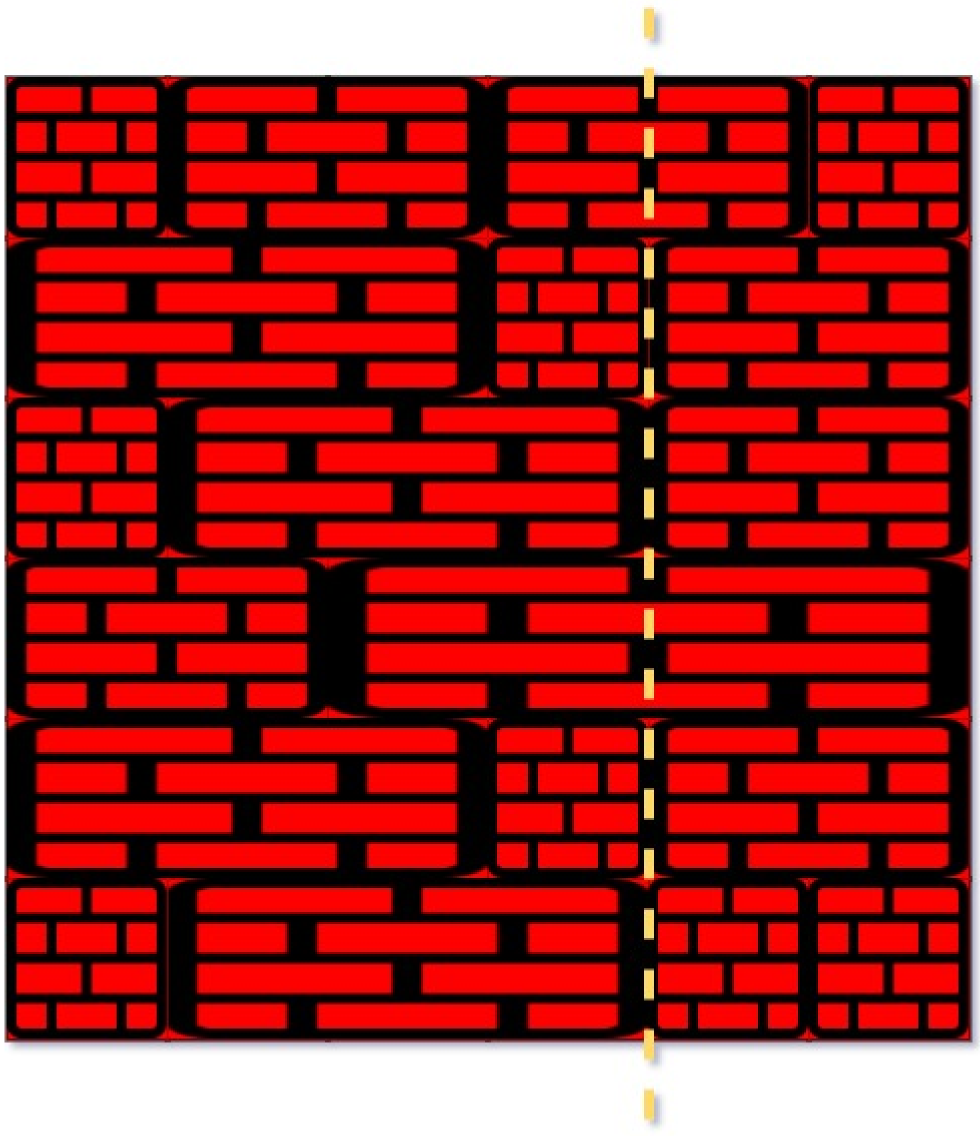
554. Brick Wall

There is a rectangular brick wall in front of you with `n` rows of bricks. The `i<sup>th</sup>` row has some number of bricks each of the same height (i.e., one unit) but they can be of different widths. The total width of each row is the same.

Draw a vertical line from the top to the bottom and cross the least bricks. If your line goes through the edge of a brick, then the brick is not considered as crossed. You cannot draw a line just along one of the two vertical edges of the wall, in which case the line will obviously cross no bricks.

Given the 2D array `wall` that contains the information about the wall, return *the minimum number of crossed bricks after drawing such a vertical line*.

Example 1:



```
Input: wall = [[1,2,2,1],[3,1,2],[1,3,2],[2,4],[3,1,2],[1,3,1,1]]
Output: 2
```

Example 2:

```
Input: wall = [[1],[1],[1]]
Output: 3
```

Constraints:

- `n == wall.length`
- `1 <= n <= 10<sup>4</sup>`
- `1 <= wall[i].length <= 10<sup>4</sup>`
- `1 <= sum(wall[i].length) <= 2 * 10<sup>4</sup>`
- `sum(wall[i])` is the same for each row `i`.
- `1 <= wall[i][j] <= 2<sup>31</sup> - 1`

-
  ```
  class Solution:
      def leastBricks(self, wall: List[List[int]]) -> int:
          freq = {}
          for i in range(len(wall)):
              row = wall[i]
              prefix = 0
              for j in range(len(row)):
                  prefix = prefix+row[j]
                  row[j] = prefix
                  freq[row[j]] = freq.get(row[j],0)+1
          # print(freq)
          if len(freq)==1:
              return len(wall)
          else:
              length = wall[0][-1]
              maxLength = 0
              for key in freq.keys():
                  if key!=length:
                      maxLength = max(maxLength,freq[key])
          return len(wall)-maxLength
  ```