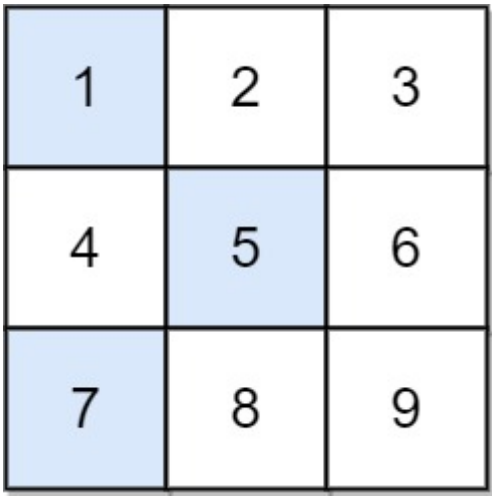
1289. Minimum Falling Path Sum II

Given an `n x n` integer matrix `grid`, return *the minimum sum of a falling path with non-zero shifts*.

A falling path with non-zero shifts is a choice of exactly one element from each row of `grid` such that no two elements chosen in adjacent rows are in the same column.

Example 1:




```
Input: arr = [[1,2,3],[4,5,6],[7,8,9]]
Output: 13
Explanation:
The possible falling paths are:
[1,5,9], [1,5,7], [1,6,7], [1,6,8],
[2,4,8], [2,4,9], [2,6,7], [2,6,8],
[3,4,8], [3,4,9], [3,5,7], [3,5,9]
The falling path with the smallest sum is [1,5,7], so the answer is 13.
```

Example 2:

```
Input: grid = [[7]]
Output: 7
``````Python
import sys
class Solution:
    def minFallingPathSum(self, matrix: List[List[int]]) -> int:
        n = len(matrix)
        dp = [[0]*n for i in range(n)]
        for i in range(n):
            if i!=0:
```
```

```
                min1,min2 = self.minInRow(dp[i-1])
            for j in range(n):
                if i==0:
                    dp[i][j]=matrix[i][j]
                else:
                    if dp[i-1][j]== min1:
                        dp[i][j] = min2+matrix[i][j]
                    else:
                        dp[i][j] = min1+matrix[i][j]
        return min(dp[n-1])
        # print(dp)



    def minInRow(self,row):
        row = sorted(row)
        return row[0],row[1]
```