LeetCode
Explore
Problems
Interview
New
Contest
Discuss
Store
🎁 LeetCoding Challenge + GIVEAWAY! 🎁
Description
Solution
Discuss (76)
Submissions
Java
Autocomplete
Problems
Pick One
Prev
/99
Next
Console
Contribute
Run Code
Submit

631. Design Excel Sum Formula

Hard 139 162 Add to List Share

Design the basic function of **Excel** and implement the function of the sum formula.

Implement the `Excel` class:

- `Excel(int height, char width)` Initializes the object with the `height` and the `width` of the sheet. The sheet is an integer matrix `mat` of size `height x width` with the row index in the range `[1, height]` and the column index in the range `['A', width]`. All the values should be **zero** initially.
- `void set(int row, char column, int val)` Changes the value at `mat[row][column]` to be `val`.
- `int get(int row, char column)` Returns the value at `mat[row][column]`.
- `int sum(int row, char column, List<String> numbers)` Sets the value at `mat[row][column]` to be the sum of cells represented by `numbers` and returns the value at `mat[row][column]`. This sum formula **should exist** until this cell is overlapped by another value or another sum formula. `numbers[i]` could be on the format:
    - `"ColRow"` that represents a single cell.
        - For example, `"F7"` represents the cell `mat[7]['F']`.
    - `"ColRow1:ColRow2"` that represents a range of cells. The range will always be a rectangle where `"ColRow1"` represent the position of the top-left cell, and `"ColRow2"` represents the position of the bottom-right cell.
        - For example, `"B3:F7"` represents the cells `mat[i][j]` for `3 <= i <= 7` and `'B' <= j <= 'F'`.

Note: You could assume that there will not be any circular sum reference.

- For example, `mat[1]['A'] == sum(1, "B")` and `mat[1]['B'] == sum(1, "A")`.

Example 1:

```
Input
["Excel", "set", "sum", "set", "get"]
[[3, "C"], [1, "A", 2], [3, "C", ["A1", "A1:B2"]], [2, "B", 2], [3, "C"]]
Output
[null, null, 4, null, 6]

Explanation
Excel excel = new Excel(3, "C");
 // construct a 3*3 2D array with all zero.
 //   A B C
 // 1 0 0 0
 // 2 0 0 0
 // 3 0 0 0
excel.set(1, "A", 2);
 // set mat[1]["A"] to be 2.
 //   A B C
 // 1 2 0 0
 // 2 0 0 0
 // 3 0 0 0
excel.sum(3, "C", ["A1", "A1:B2"]); // return 4
 // set mat[3]["C"] to be the sum of value at mat[1]["A"] and the values sum of the rectangle
range whose top-left cell is mat[1]["A"] and bottom-right cell is mat[2]["B"].
 //   A B C
 // 1 2 0 0
 // 2 0 0 0
 // 3 0 0 4
excel.set(2, "B", 2);
 // set mat[2]["B"] to be 2. Note mat[3]["C"] should also be changed.
 //   A B C
 // 1 2 0 0
 // 2 0 2 0
 // 3 0 0 6
excel.get(3, "C"); // return 6
```

Constraints:

- `1 <= height <= 26`
- `'A' <= width <= 'Z'`
- `1 <= row <= height`
- `'A' <= column <= width`
- `-100 <= val <= 100`
- `1 <= numbers.length <= 5`
- `numbers[i]` has the format `"ColRow"` or `"ColRow1:ColRow2"`.
- At most `100` calls will be made to `set`, `get`, and `sum`.

Accepted 7,648 | Submissions 20,150

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Google | 5 Uber | 2

Related Topics

Graph Design Topological Sort

```java
class Excel {

    public Excel(int height, char width) {

    }

    public void set(int row, char column, int val) {

    }

    public int get(int row, char column) {

    }

    public int sum(int row, char column, String[] numbers) {

    }
}

/**
 * Your Excel object will be instantiated and called as such:
 * Excel obj = new Excel(height, width);
 * obj.set(row,column,val);
 * int param_2 = obj.get(row,column);
 * int param_3 = obj.sum(row,column,numbers);
 */
```