AddEmUP

You will implement three methods in the `AddEmUP` class. The `AddEmUP` class has a single constructor with a 2D array `(int[][])` as its single parameter. The three methods you will implement are the `rowSum(int row)`, the `getState(int row)` and the `commonSum()` methods.

The `rowSum(int row)` returns a list of all possible sums using two entries from a given row. The list should have no duplicate values.

The following code shows the results of the `rowSum` method.

The following code	Returns

`int[][] nums = { { 3, 6, 8}, {2, 12, 7}, {8, 6, 4} };`
`AddEmUp addEm = new AddEmUp(nums);`
 	
`List<Integer> ans = addEm.rowSum(0);`	

`ans.size();`	`3`
`ans.contains( new Integer(9) );`	`true`
`ans.contains( new Integer(11) );`	`true`
`ans.contains( new Integer(14) );`	`true`

Remember, the list should have no duplicate values. One more example, the following code shows the results of the `rowSum` method.

The following code	Returns

`int[][] nums1 = { { 3, -1, 2, 0}, {2, 2, 1, 2} };`
`AddEmUp addEm = new AddEmUp(nums1);`
 	
`List<Integer> ans = addEm.rowSum(0);`	

`ans.size();`	`5`
`ans.contains( new Integer(2) );`	`true`
`ans.contains( new Integer(5) );`	`True`
`ans.contains( new Integer(3) );`	
`ans.contains( new Integer(1) );`	
`ans.contains( new Integer(-1) );`	

The `getState(int row)` determines the state of a row in the 2D array. (For this method, repeated sums count multiple times.) Remember, a number, x, is even if `x % 2 == 0`. This method returns:

- `"EVEN"` if there exist more even numbers in the List of all possible sum of two entries in a given row.
- `"ODD"` if there exist more odd numbers in the List of all possible sum of two entries in a given row.
- `"NEITHER"` if there exist the same number of even and odd numbers in the List of all possible sum of two entries in a given row.

Note: `-1 % 2 == -1` which is `ODD`

The following code shows the results of the `getState` method.

The following code	Returns
`int[][] nums = { { 3, 6, 8}, {2, 12, 7}, {8, 6, 4} };` `AddEmUp addEm = new AddEmUp(nums);`	
`addEm.getState(0);`	`"ODD"`
`addEm.getState(1);`	`"ODD"`
`addEm.getState(2);`	`"EVEN"`

One more example, the following code shows the results of the `getState` method.

The following code	Returns
`int[][] nums1 = { { 3, -1, 2, 0}, {2, 2, 1, 2} }` `AddEmUp addEm = new AddEmUp(nums1);`	
`addEm.getState(0);`	`"ODD"`
`addEm.getState(1);`	`" NEITHER"`

The `commonSum()` returns a List of all values that are contained in every List returned by `rowSum(k)` method for all rows in the 2D array. That is, a List of all values that would be contain in the `rowSum(k)` method for all possible values of `k`.

The following code shows the results of the `commonSum` method.

The following code	Returns

`int[][] nums = { { 3, 6, 8}, {2, 12, 7}, {8, 6, 4} };`
`AddEmUp addEm = new AddEmUp(nums);`
 	
`ans = addEm.commonSum();`	

`ans.size();`	`1`
`ans.get(0);`	`new Integer(14)`

- All 2D arrays will be rectangular. That is, each row in the array will be the same length