Toeplitz Matrix

A Toeplitz matrix is a matrix where every left-to-right-descending diagonal has the same element. Given a non-empty matrix `arr`, write a function that returns true if and only if it is a Toeplitz matrix. The matrix can be any dimensions, not necessarily square.

For example,

```
[[1,2,3,4],
 [5,1,2,3],
 [6,5,1,2]]
```

is a Toeplitz matrix, so we should return `true`, while

```
[[1,2,3,4],
 [5,1,9,3],
 [6,5,1,2]]
```

isn't a Toeplitz matrix, so we should return `false`.

Constraints:

[time limit] 5000ms

[input] array.array.integer `arr`

$0 \le arr.length \le 20$

$0 \le arr[i].length \le 20$

$0 \le arr[i][j] \le 20$

[output] boolean

Hints & Tips

`arr[r][c]` in the array?

How many different left-to-right-descending diagonals are there, in terms of `R` and `C`?

If the user is stuck on checking each individual diagonal, try solving for the case of a generic 1-dimensional array. If we wanted to know whether all the elements in a single 1-dimensional array are the same, how might we solve that problem?

Solution

There are R+C-1 different diagonals, because each diagonal starts with an element in the top left (marked “S” as pictured below):

[[S,S,S,S], [S,,,], [S,,,]]

The top left diagonal is A[0][0], A[1][1], A[2][2], etc., and the diagonal to the right of that is A[0][1], A[1][2], A[2][3], etc. In general, the diagonal starting at A[r][c] is going to be enumerated by A[r+k][c+k].

How big is k? Say there are R rows and C columns. Then because (r+k, c+k) must be in bounds, 0 <= r+k < R and 0 <= c+k < C, so k < min(R-r, C-c) after some algebra.

Now, for each enumerated diagonal, let’s check whether each element of that diagonal is equal to it’s head representative in the first row/column. If it isn’t, the matrix isn’t Toeplitz.

This leads to the following solution:

```
function isToeplitz(matrix):
    R, C = matrix.length, matrix[0].length

    # For each diagonal starting in the first row at (0, c)
    for c from 0 to C - 1:
        start = matrix[0][c]

        # For each element of that diagonal
        for k from 0 to min(R, C-c) - 1:
```

```
        start = matrix[r][0]
        for k from 0 to min(R-r, C) - 1:
            if matrix[r+k][k] != start:
                return false

    return true
```

Additionally, there is a hashing solution available. Elements at position (r, c) are on the same diagonal if and only if they have the same "diagonal hash" value of r - c. That idea leads to the following solution:

```
function isToeplitz(matrix):
  # seen[diagonal_hash] = value of elements on that diagonal
    seen = new HashTable()

    # For each element
    for r from 0 to matrix.length - 1:
        for c from 0 to matrix[0].length - 1:
            # If we haven't visited this diagonal before,
            # record it's value
            if r-c not in seen:
                seen[r-c] = matrix[r][c]

            # Otherwise, if the value we've recorded for
            # this diagonal differs, return False
            else if seen[r-c] != matrix[r][c]:
                return False

    return True
```

We could also have used an array for seen, as there are R+C-1 different diagonal hashes and they are contiguous.

Time Complexity: `O(R * C)` if the matrix has dimensions `R x C`. We iterate over every element exactly once and do constant work in between.

[Pra]ctice [m]akes [P]erfect!

Go Back