Daily Coding Problem #315

Problem

This problem was asked by Google.

In linear algebra, a Toeplitz matrix is one in which the elements on any given diagonal from top left to bottom right are identical.

Here is an example:

```
1 2 3 4 8
5 1 2 3 4
4 5 1 2 3
7 4 5 1 2
```

Write a program to determine whether a given input is a Toeplitz matrix.

Solution

Although this matrix has a fancy name, implementing this solution is not terribly complicated.

We must check each descending diagonal to ensure that all its entries are the same. To perform this check, we can create a helper function that takes in the matrix and a starting location, and moves step by step to the bottom right. If at any point it finds a value

location, and moves step by step to the bottom right. If at any point it finds a value different from what has been found so far, we know to return `False`.

All that remains is to find out where the starting positions should be. These consist of every cell on the left and top borders of the matrix, which can be found by constructing every coordinate of the form `(row, 0)` and `(0, col)`.

```
def has_equal_values(matrix, row, col):
    m, n = len(matrix), len(matrix[0])

    start = matrix[row][col]
    while row < m and col < n:
        if matrix[row][col] != start:
            return False
        row += 1; col += 1

    return True

def is_toeplitz(matrix):
    m, n = len(matrix), len(matrix[0])

    for row in range(m):
        if not has_equal_values(matrix, row, 0):

            return False
    for col in range(1, n):
        if not has_equal_values(matrix, 0, col):
            return False

    return True
```

An alternative approach is to note that each diagonal is uniquely represented by the term `i - j`, where `i` and `j` are the row and column values respectively. For example, in the input given above, every cell in the ones diagonal will satisfy the property `i - j = 0`.

With this in mind, we can create a dictionary that maps `i - j` keys to their respective integers. As we iterate over each cell in the matrix, we find out what key it corresponds to, and determine if it has a matching value. If at any point the values does not match, we will return `False`.

```
def is_toeplitz(matrix):
    values = {}
```

```
    values = {}

    for i, row in enumerate(matrix):
        for j, col in enumerate(row):
            if i - j not in values:
                values[i - j] = col
            elif values[i - j] != col:
                return False

    return True
```

Both of these solutions will take `O(M * N)` time, for a matrix with `M` rows and `N` columns, since we must check every cell in the worst case.
