symmetry2D

Write the C function that takes in a square two-dimensional array of integer numbers M and the array sizes for rows and columns as parameters, and returns 1 if M is symmetric or 0 otherwise. A square two-dimensional matrix is symmetric iff it is equal to its transpose. It means that M[i][j] is equal to M[j][i] for 0<=i<=rowSize and 0<=j<=colSize. For example, if rowSize and colSize are 4, and *M* is {{1,2,3,4},{2,2,5,6},{3,5,3,7}, {4,6,7,4}}, then M will be symmetric. The function prototype is given as follows:

```
int symmetry2D(int M[][SIZE], int rowSize, int colSize);
```

A sample program template is gven below to test the function:

```
#include <stdio.h>
#define SIZE 10
#define INIT_VALUE 999
int symmetry2D(int M[][SIZE], int rowSize, int colSize);
int main()
{
  int M[SIZE][SIZE],i,j, result = INIT_VALUE;
  int rowSize, colSize;

  printf("Enter the array size (rowSize, colSize): \n");
  scanf("%d %d", &rowSize, &colSize);
  printf("Enter the matrix (%dx%d): \n", rowSize, colSize);
  for (i=0; i<rowSize; i++)
    for (j=0; j<colSize; j++)
      scanf("%d", &M[i][j]);
  result=symmetry2D(M, rowSize, colSize);
  if (result == 1)
    printf("symmetry2D(): Yes\n");
  else if (result == 0)
    printf("symmetry2D(): No\n");
  else
    printf("Error\n");
  return 0;
}
int symmetry2D(int M[][SIZE], int rowSize, int colSize)
{
  /* Write your code here */
}
```

```
int symmetry2D(int M[][SIZE], int
rowSize, int colSize)
{
   int i;
   int j;

   for(i=0;i<rowSize;i++)
   {
      for(j=0;j<colSize;j++)
      {
         if(M[i][j]!=M[j][i])
         {
            return 0;
         }
      }
   }
   return 1;
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter the array size (rowSize, colSize):
4 4
Enter the matrix (4x4):
1 2 3 4
2 2 5 6
3 5 3 7
```

```
4 6 7 4
symmetry2D(): Yes
```

(2) Test Case 2:

```
Enter the array size (rowSize, colSize):
4 4
Enter the matrix (4x4):
1 2 3 4
2 2 5 6
3 5 3 7
5 6 7 4
symmetry2D(): No
```

(3) Test Case 3:

```
Enter the array size (rowSize, colSize):
3 3
Enter the matrix (3x3):
1 2 3
2 6 7
3 7 3
symmetry2D(): Yes
```

(4) Test Case 4:

```
Enter the array size (rowSize, colSize):
5 5
Enter the matrix (5x5):
1 2 3 4 5
2 2 5 6 7
3 5 3 7 8
4 6 7 4 5
5 7 8 5 5
symmetry2D(): Yes
```