compress2D

Write a function compress2D() that takes as input a square 2-dimensional array of binary data, compresses each row of the array by replacing each run of 0s or 1s with a single 0 or 1 and the number of times it occurs, and prints on each line the result of compression. For example, the row with data 0011100011 may be compressed into 02130312. The prototype of the function is given as follows:

```
void compress2D(int data[SIZE][SIZE], int rowSize, int colSize);
```

A sample program to test the function is given below.

```
#include <stdio.h>
#define SIZE 100
void compress2D(int data[SIZE][SIZE], int rowSize, int colSize);
int main()
{
  int data[SIZE][SIZE];
  int i,j;
  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", &data[i][j]);
  printf("compress2D(): \n");
  compress2D(data, rowSize, colSize);
  return 0;
}
void compress2D(int data[SIZE][SIZE], int rowSize, int colSize)
{
  /* Write your code here */
}
```

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 1 1 0
0 0 1 1
1 1 1 1
0 0 0 0
compress2D():
1 3 0 1
0 2 1 2
1 4
```

0 4

(2) Test Case 2:

Enter the array size (rowSize, colSize):
5 5
Enter the matrix (5x5):
1 1 1 0 0
0 0 1 1 1
1 1 1 1 1
0 0 0 0 0
1 1 1 1 1
compress2D():
1 3 0 2
0 2 1 3
1 5
0 5
1 5

(3) Test Case 3:

Enter the array size (rowSize, colSize):
5 5
Enter the matrix (5x5):
0 0 0 0 0
1 1 1 1 1
1 1 1 1 1
0 0 0 0 0
1 1 1 1 1
compress2D():
0 5
1 5
1 5
0 5
1 5

(4) Test Case 4:

Enter the array size (rowSize, colSize):
10 10
Enter the matrix (10x10):
1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 1 0 1 0 1
1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 1 0 1 0 1
1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 1 0 1 0 1
1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 1 0 1 0 1
1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 1 0 1 0 1
compress2D():
1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1
0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1

1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1
0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1
1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1
0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1
1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1
0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1
1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1
0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1