# Solution for "Modern Art" Bronze Open 2017

by Arnav Joshi

First, let’s look through the USACO sample case:

```
2230
2737
2777
0000
```

The first key observation is that we should assume that each rectangle is as small as possible. The reason for this is that we want to minimize overlap because if two rectangles overlap, one of them could not have been painted first. Thus to maximize the number of rectangles that could have been painted first, we must make the size of each rectangle as small as possible. For example, in this case, we would assume that the size of the rectangle 2 is 2 by 3, the size of rectangle 7 is also 2 by 3, and the size of rectangle 3 is 1 by 2.

Furthermore, we realize that a rectangle A could only have been first if there is no other rectangle B such that there is a grid square of color A within the area of B. For example, rectangle 3 could not have been first because there is a bit of rectangle 3 inside the area for rectangle 7. Below is the function to check this condition for colors a and b.

```
check (a, b):
    //find the rectangle of color b
    top=N, bottom=0, left=N, right=0;
    for i from 0 through n-1:
        for j from 0 through n-1:
            if grid[i][j] = b:
                top = min(top, i)
                bottom = max(bottom, i)
                left = min(left, j)
                right = max(right, j)

    //check if the rectangle of color a enters the boundary of color b
    for i from top to bottom:
        for j from left to right
            if grid[i][j] = a:
                return true
    return false
```

Below, we have the main code for checking the number of colors that could have been put down first (based on the explanation above).

```
read n
```

```
array grid[][]
for i from 0 through n-1:
     for j from 0 through n-1:
          read grid[i][j]

answer = 0
for i from 1 through 9:
     if colorExists(i):
          first = true
          for j from 1 through 9
               if j!=i and colorExists(j) and check(i,j)
                    first = false
          if first = true:
               answer += 1
print answer
```

The colorExists function demonstrated above checks if a color exists in the array. The method is implemented below

```
colorExists(a):
     for i from 0 through n-1:
          for j from 0 through n-1:
               if grid[i][j]  = a:
                    return true

     return false
```