Practice Arena

Practice problems aimed to improve your coding skills.

- PRACTICE-02_SCAN-PRINT
- PRACTICE-03_TYPES
- LAB-PRAC-02_SCAN-PRINT
- LAB-PRAC-01
- PRACTICE-04_COND
- BONUS-PRAC-02
- LAB-PRAC-03_TYPES
- PRACTICE-05_COND-LOOPS
- LAB-PRAC-04_COND
- LAB-PRAC-05_CONDLOOPS
- PRACTICE-07_LOOPS-ARR
- LAB-PRAC-06_LOOPS
- LAB-PRAC-07_LOOPS-ARR
- LABEXAM-PRAC-01_MIDSEM
- PRACTICE-09_PTR-MAT
- LAB-PRAC-08_ARR-STR
- PRACTICE-10_MAT-FUN
- LAB-PRAC-09_PTR-MAT
- LAB-PRAC-10_MAT-FUN
- PRACTICE-11_FUN-PTR
- LAB-PRAC-11_FUN-PTR
- LAB-PRAC-12_FUN-STRUC
  - Point Pairing Party
  - Verify the family tree of Mr C
  - Simple Sodoku
  - The Family Tree of Mr C Part Three
  - The Post offices of KRville
  - Matrix Mandala
  - Mango Mania
  - Recover the Rectangle
  - Crazy for Candy
  - A Brutal Cipher Called Brutus
  - Triangle Tangle
  - Basic Balanced Bracketing
- LABEXAM-PRAC-02_ENDSEM
- LAB-PRAC-13_STRUC-NUM
- LAB-PRAC-14_SORT-MISC

Matrix Mandala

LAB-PRAC-12_FUN-STRUC

Matrix Mandala [20 marks]

Problem Statement

In the input we will give you two strictly positive single digit numbers N and C, separated by a space. Thus, both N and C strictly greater than 0 and strictly smaller than 10. The question will work with N X N matrices and we have to color each location in the matrix using one of the colors from 1, 2, ..., C.

Now if there were no other constraints, there would have simply been C^{N^2} such colorings. However, we want beautiful colorings. A coloring is beautiful if no location in the matrix has the same color as one of its neighbors. Every location in the matrix has four neighbors, the location immediately to its right, the location immediately to its left, the location immediately above it and the location immediately below it. If a cell is on the edge of the matrix or at one of the corners of the matrix, then its neighbors are found by wrapping around our search.

For example, consider the following 3 x 3 matrix colored using 9 colors (N = 3, C = 9)
1 2 3
4 5 6
7 8 9
The neighbors of the location with color 5 are the locations with colors 4, 2, 6 and 8. The neighbors of the location with color 2 are the locations with colors 1, 8, 3, and 5, The neighbors of the location with color 7 are the locations with colors 9, 4, 8 and 1. Thus, the search for neighbors simply wraps around the matrix if we reach and edge or a corner.

We want to find and output all beautiful colorings. To output a coloring, simply output all colors of the first row (from left to right), followed by all colors of the second row (from left to right) and so on. Thus, the above coloring would be output as 123456789. Your output must list all beautiful colorings in lexicographically increasing order (i.e. view each coloring as a long number and output the numbers in increasing order). In the last line of the output, print how many beautiful colorings did you find.

Caution

1. Warning: note that there can be no 1 x 1 beautiful coloring since the only location will become its own neighbor from every direction so we cannot give it any color without giving it the same color as one of its neighbors.
2. Be careful about extra/missing lines and extra/missing spaces in your output.

HINTS: You should try to solve this problem recursively. Write a function
int gridColour(int** mat, int rowNext, int colNext, int N, int C);
which takes in a partially filled in matrix, the location (rowNext, colNext) in the matrix where the next color needs to be filled, the size of the matrix and the number of colors allowed. The function prints all possible beautiful colorings possible from this partially filled in matrix and returns the number of ways this partially filled matrix can be completed to form a beautifully colored grid.

The routine should start filling in colors to locations in the first row from left to right, then after the first row is completed, to the second row from left to right and so on. Take care to fill smaller colors first

and bigger colors later to preserve the lexicographic ordering. The base case can be when we have filled in all the entries of the matrix i.e. when the next entry to be filled lies in row n+1 i.e. when rowNext = n and colNext = 0. This means that the matrix is completely colored and if beautifully colored, can be printed.

EXAMPLE 1:
INPUT
2 2

OUTPUT:
1221
2112
2

Explanation: There are only two beautifully colored matrices using 2 colors
1 2
2 1

2 1
1 2

Notice that the colorings have been output in increasing order.

EXAMPLE 2:
INPUT
2 1

OUTPUT:
0

Explanation: We cannot beautifully color a 2 x 2 matrix using just one color

EXAMPLE 3:
INPUT
1 5

OUTPUT:
0

Explanation: A 1 x 1 matrix cannot be beautifully colored no matter how many colors are given since the lone element in the matrix ends up being its own neighbor.

Grading Scheme:
Total marks: **[20 Points]**

There will be partial grading in this question. There are several lines in your output. Printing each line correctly, in the correct order, carries equal weightage. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

Start Solving! (/editor/practice/6235)