TATU BOGDAN
19.02.2021

```
2 void color_map( pattern [2][2] , rows , cols)
      for ( i=0; i<rows ; i++) {
          for (j=0; j< cols ; j++) {
              Map[i][j] = pattern [i%2][j%2];
          }
      }
  }

  main() {
      pattern = ~~red~~ read_pattern();
      cin >> rows >> cols;
      coler_map(pattern , rows , cols);
  }
```

Example $Map[5][5] = \begin{bmatrix} \_ & \_ & \_ & \_ & \_ \\ \_ & \_ & \_ & \_ & \_ \\ \_ & \_ & \_ & \_ & \_ \\ \_ & \_ & \_ & \_ & \_ \\ \_ & \_ & \_ & \_ & \_ \end{bmatrix}$

$Pattern[2][2] = \begin{bmatrix} \overset{00}{R} & \overset{01}{G} \\ \overset{10}{B} & \overset{11}{W} \end{bmatrix}$

$Map \Rightarrow \begin{bmatrix} R & G & R & G & R \\ B & W & B & W & B \\ R & G & R & G & R \\ B & W & B & W & B \\ R & G & R & G & R \end{bmatrix}$

rows 0, 2, 4 => pattern of row 0

rows 1, 3 => pattern of row 1

cols 0, 2, 4 => pattern of col 0

cols 1, 3 => pattern of col 1

4

TATU BOGDAN
19.02.2021

Complexity : O(n^2) - for in for

Data structs: 2D Arrays - pattern and map
2x2 nxm
rows x cols

```
backtrack ( x, y, ~~pattern[2][2]{~~ colors[4] , rows, cols)
for ( i=0 ; i<4; i++)
 col ~~Map[x][y]~~= colors [i];
    if (valid(x,y, ~~Map~~ col)){
       ~~if (is~~
       ~~{~~ Map[x][y]= col;
          if ( solution () {
               print_map()
          }
          else {
               if (x <rows-1)
               bactracking (x+1, y, colors,rows,cols)
               else
               backtracking (0, y+1), colors... )
          }
          Map[x][y]=-1
    }
 }

solution( ~~{~~ x, y, rows, cols)
    return (x== cols-1, && y=rows-1);

valid() ;
```