Generating Closed Modular Colorings of Graphs Using a SAT Solver

Jan Hlavacek and Garry L. Johns
Saginaw Valley State University

May 4, 2017

1 Introduction

We start with a graph G. To keep things simple, we will assume that G has no true twins. We color all vertices with colors from 0 to $k-1$. Then we calculate the closed modular coloring: for each vertex you add the colors of all its neighbours, add it to its color, and assign that number mod k as its new color.

A coloring is proper if no two adjacent vertices have the same color.

We are trying to see if it is possible to color a graph G so that the resulting closed modular coloring is proper.

2 Description using logic

We will have k colors, $0, 1, \cdots, k-1$. We try to describe the condition of having a closed modular coloring using logic formulas, so we can use a SAT-solver to find out if the condition is satisfiable for the graph G. We will need to introduce a number of logical variables and write a logic formula using these variables that will be equivalent to G having a proper closed modular coloring.

2.1 Vertices

2.1.1 Coloring a vertex

For each vertex v, there will be k color variables c_v^i, $i = 0, \ldots, k-1$. Exactly one of those will be true at a time, since a vertex cannot have more than one color:

$$c_v^1 \vee c_v^2 \vee c_v^3 \vee \cdots \vee c_v^k \tag{1}$$

$$\neg c_v^i \vee \neg c_v^j \text{ for } i \neq j \tag{2}$$

2.1.2 Modular coloring

Now we need to figure out what will be the new color of the vertex in the modular coloring. That will depend on the neighbouring vertices, as well as on the vertex itself.

We will take a list of all the neighbours of the vertex v, including the vertex v itself, and go through all the possible colorings of this list. Each such coloring C_s will be assigned a variable n_v^s, that will be equivalent to all the neighbouring vertices having exactly the colors from C_s.

For example, if the list of neighbours of v is $[v, w, y, x, z]$, then for coloring $C_s = [2, 1, 3, 2, 4]$ of those vertices the variable n_v^s will satisfy

$$n_v^s \equiv c_v^2 \wedge c_w^1 \wedge c_y^3 \wedge c_x^2 \wedge c_z^4 \tag{3}$$

For each color j, there will be possibly several colorings $C_{s_1}, C_{s_2}, \ldots, C_{s_{t_j}}$ of the neighbouring vertices that will produce j as the modular coloring of the vertex v. We introduce k variables m_v^j for $j = 1, 2, \ldots, k$, satisfying

$$m_v^j \equiv n_v^{s_1} \vee n_v^{s_2} \vee \cdots \vee n_v^{s_{t_j}} \tag{4}$$

2.2 Edges

The rules for edges will be simple enough. All we need to assure is that the resulting modular coloring is proper, that is for each edge (v, w), the modular coloring of the vertex v is different than the modular coloring of the vertex w. That means that for each color $j = 0, \ldots, k-1$, m_v^j and m_w^j cannot be both true:

$$\neg m_v^j \vee \neg m_w^j \text{ for } j = 0, \ldots, k-1 \tag{5}$$

3 Translation to CNF

Satisfiability solvers typically require their input to be in the Conjunctive Normal Form (CNF), that is a written as a conjunction of clauses, where each clause is a disjunction of single variables or their negations. We see that equations (1), (2) and (5) already are in CNF, while (3) and (4) need to be translated.

The formula $a \equiv (b \wedge c)$ can be translated into CNF as

$$(c \vee \neg a) \wedge (b \vee \neg a) \wedge (a \vee \neg b \vee \neg c) \tag{6}$$

while the formula $a \equiv (b \vee c)$ can be translated to

$$(a \vee \neg c) \wedge (a \vee \neg b) \wedge (b \vee c \vee \neg a) \tag{7}$$

Equation (3) will then translate to

$$(c_v^2 \vee \neg n_v^s) \wedge (c_w^1 \vee \neg n_v^s) \wedge (c_y^3 \vee \neg n_v^s) \wedge (c_x^2 \vee \neg n_v^s) \wedge (c_z^4 \vee \neg n_v^s) \tag{8}$$

$$n_v^s \vee \neg c_v^2 \vee \neg c_w^1 \vee \neg c_y^3 \vee \neg c_x^2 \vee \neg c_z^4 \tag{9}$$

In general, for each coloring C_s of the neighbours of vertex v, we will have the conjunction of

$$c_w^j \vee \neg n_v^s \tag{10}$$

where w runs through all the neighbors of v, including v, and for each w, j is the color of w in the coloring C_s, with

$$n_v^s \vee \neg c_{w_1}^{j_1} \vee \neg c_{w_2}^{j_2} \vee \cdots \tag{11}$$

where w_i are all the neighbors of v, including v, and j_i are their corresponding colors in C_s.

The formula (4) will translate to

$$m_v^j \vee \neg n_v^{s_i} \tag{12}$$

$$n_v^{s_1} \vee n_v^{s_2} \vee \cdots \vee n_v^{s_{t_j}} \vee \neg m_v^j \tag{13}$$

where j runs through all the colors 1 to k, and s_i runs through all indices s such that the corresponding coloring C_s of all the neighbours of v will produce the modular coloring of v with the color j.

4 Implementation

4.1 Imports and Setup

We will use the `pycosat` module, which provides a Python interface to the picosat SAT solver. We will also need few functions from the `itertools` module. We will use the `graph_tool` module to handle the graph stuff.

```
import pycosat
from itertools import count, islice, product
import graph_tool as gt
```

The `pycosat` module requires variables to be represented by integers. Each variable will have to be associated with a unique integer. To keep track which integers we already used, we will use the `count()` function from the `itertools` module.

4.2 Assigning Variables

Given a graph, we will need to assign the initial variables for original colors (c_v^j) and for colors of a modular coloring (m_v^j). We need to create two property maps for the given graph, one for original colors, and one for the closed modular coloring. Each property will be a vector of integers of length k, where k is the number of colors. The following function will create and populate one such map

```
def assign_color_vars(g, no_of_colors, vars_generator):
    """
    Create property map for graph g, containing variable numbers
    for colors.

    Args:
        g: A graph
        no_of_colors: the number of colors used for graph coloring.
        vars_generator: generator that produces numbers for logical
            variables, to be used in pycosat clauses.

    Returns:
        A vertex property map of 'vector<int>' type, filled with
        numbers of pycosat variables corresponding to possible
        colors for each vertex.
    """
    colors_map = g.new_vertex_property("vector<int>")
    for v in g.vertices():
        colors_map[v] = list(islice(vars_generator, no_of_colors))
    return colors_map
```

We will call this function twice for each graph, first time to generate variable numbers for all c_v^j variables, the second time to generate numbers for all m_v^j variables.

4.3 Generating Clauses

Now we will write some functions that will generate clauses for the SAT solver. Each clause will be a list of positive or negative integers. Positive number means that the variable with the number appear in the conjunction, while a negative number means that the negation of the variable with the opposite number appears in the conjunction.

4.3.1 Clauses from equations (1) and (2)

These clauses are grouped by individual vertices. We will write a function that will receive a vertex and generate all these clauses for that vertex. For the equation (2), we will only need the clauses for $j < i$, since disjunction is commutative.

```
def vertex_color_clauses(v, color_map):
    """
    Return a list of clauses dealing with individual vertex
    initial color.

    Args:
        v: a graph vertex
        color_map: a vertex property map that assigns variable
        numbers to colors of individual vertices.

    Returns:
        A list of lists, each of them representing one clause.
    """
    l = []
    colors = color_map[v]
    l.append(list(colors))
    for i in range(1,len(colors)):
        for j in range(i):
            l.append([-colors[i],-colors[j]])
    return l
```

4.3.2 Clauses from equation (5)

There will be a group of clauses for each edge, assuring that the resulting modular coloring is proper. The function will receive an edge, and return a list of clauses. It will need the lists of modular coloring colors for each end of the edge, and create a list of clauses assuring that the colors on the ends of the edge are not the same.

```
def edge_clauses(e, modular_color_map):
    """
    Return a list of clauses assuring that the colors of the
    modular coloring on the ends of the edge are different.

    Args:
        e: an edge of a graph
        modular_color_map: a vertex property map that assigns
        variable numbers to colors in modular coloring of
        individual vertices.

    Returns:
        A list of lists, each of which correspond to one clause.
    """
    l = []
    mc1 = modular_color_map[e.source()]
    mc2 = modular_color_map[e.target()]
    for i,j in zip(mc1,mc2):
```

```
        l.append([-i,-j])
    return l
```

4.3.3 Clauses from equations (10), (11), (12) and (13)

This is the trickiest part. For a given vertex v, we take the list of all the neighbours of v, including v. Then we will go through all the possible colorings of this list, for each coloring we allocate a new variable, and make it equivalent to a conjunction of the vertex color variables corresponding to the colors in the coloring. At the same time, we will check what color would a modular coloring produced by this coloring of neighbours of v produce at v. We will add the variable we allocated to this coloring to a list of variables for the obtained modular color. At the end we will go through all the modular color variables for vertex v, and generate clauses that will make each equivalent to the disjunction of all the variables added to their lists.

We will start with two auxiliary functions. Each will receive a variable number as the first argument, and a list of variable numbers as its second argument. The first will generate clauses making the first variable equivalent to the conjunction of all the variables in the list, the second one will do the same for a disjunction of all the variables on the list.

```
def equiv_to_conjunction(a,vars):
    """
    Return a list of clauses that assure that the variable a is
    equivalent to the conjunction of all the variables in the list
    vars.

    Args:
        a: integer, a number of a variable
        vars: a list of integers, each corresponding to a variable

    Returns:
        a list of lists of integers.  Each list of integers
        corresponds to one clause.  The conjunction of all the
        clauses will be a CNF version of the equivalence of the
        variable a with the conjunction of variables in l.
    """
    l = [[a] + [-b for b in vars]]
    for b in vars:
        l.append([b,-a])
    return l

def equiv_to_disjunction(a,vars):
    """
    Return a list of clauses that assure that the variable a is
    equivalent to the disjunction of all the variables in the list
    vars.

    Args:
        a: integer, a number of a variable
        vars: a list of integers, each corresponding to a variable

    Returns:
        a list of lists of integers.  Each list of integers
        corresponds to one clause.  The conjunction of all the
```

```
    clauses will be a CNF version of the equivalence of the
    variable a with the disjunction of variables in l.
"""
l = [[-a] + vars]
for b in vars:
    l.append([-b,a])
return l
```

The next function will receive a vertex v, and produce all the clauses from equations (10), (11), (12) and (13).

```
def modular_coloring_clauses(v,color_map,modular_color_map,
                             vars_generator):
    """
    Return the list of clauses that will assure that the colors in
    the modular_coloring_map are created as a closed modular
    coloring from the colors in the color_map.

    Args:
        v: a vertex of a graph
        color_map: a vertex property map that assigns variable
            numbers to colors of individual vertices.
        modular_color_map: a vertex property map that assigns
            variable numbers to colors in modular coloring of
            individual vertices.
        vars_generator: generator that produces numbers for logical
            variables, to be used in pycosat clauses.

    Returns:
        A list of lists, each of which correspond to one clause.

    Notes:
        May allocate new pycosat variables using the vars_generator.
    """
    modular_colors = list(modular_color_map[v])
    k = len(modular_colors) # number of colors
    m_lists = {m:[] for m in modular_colors}
    neighbours = list(v.out_neighbours()) + [v]
    colorings = product(range(k),repeat=len(neighbours))
    l = []
    for cs in colorings:
        new_var = next(vars_generator)
        modular_color = sum(cs) % k
        m_lists[modular_colors[modular_color]] += [new_var]
        neighbour_color_vars = []
        for w,c in zip(neighbours,cs):
            neighbour_color_vars.append(color_map[w][c])
        l += equiv_to_conjunction(new_var,neighbour_color_vars)

    for m,s in m_lists.items():
        l += equiv_to_disjunction(m,s)

    return l
```

4.3.4 Putting all clauses together

Now we need to put all the clause generating functions together. The following function will take a graph and a number of colors, and generate all the clauses for the graph. The result can then be fed directly to the pycosat solver.

The steps will be:

1. Create a new variable generator.
2. Assign color and modular color variables to all vertices.
3. For each vertex:
   (a) Generate all the clauses for equations (1) and (2).
   (b) Generate all the clauses for equations (10), (11), (12) and (13).
4. For each edge, generate all clauses for equation (5).
5. Return a complete list of all the generated clauses. Also return the two vertex property maps, since we will need them to decode the result.

```
def graph_clauses(g,num_of_colors):
    """
    Return the complete lists of all the clauses describing
    conditions for a proper closed modular coloring of g.

    Args:
       g: a graph_tool graph
       num_of_colors: number of colors for the modular coloring

    Returns:
       a list of lists of integers: each list of integers
       corresponds to a disjunction of variables or their
       negations, and two vertex property maps containing
       the colors for the original coloring and the corresponding
       modular coloring.
    """
    l = []
    vars_generator = count(1)
    color_map = assign_color_vars(g,num_of_colors,vars_generator)
    modular_color_map = assign_color_vars(g,num_of_colors,
                                          vars_generator)

    for v in g.vertices():
       l += vertex_color_clauses(v,color_map)
       l += modular_coloring_clauses(v,color_map,modular_color_map,
                                     vars_generator)

    for e in g.edges():
       l +=  edge_clauses(e, modular_color_map)

    return l, color_map, modular_color_map
```

4.4 Making sense of the result

The pycosat solver returns a long list of positive or negative integers, one for each variable. Each number corresponds to one variable, and is positive if the corresponding variable is true, and negative if the variable is false. We are only interested in the variables on the beginning, the ones that correspond to the color variables and modular color variables for each vertex. Out of these, we only want to know, for each vertex, which of its color variables and which of its modular color variables has a positive number in the result.

A safe way (see below for some ideas that could make things easier if we could rely on some assumptions) would be:

1. Discard all negative entries from the list
2. Create a set from the positive entries, to make lookup faster
3. Iterate through vertices of the graph:
   (a) For each vertex, iterate through the colors, asking if each is in the set, until you find one
   (b) Do the same for modular colors
4. For each vertex, remember the found color and modular color.

If we could assume that:

- Iteration through vertices of a graph always happens in the same order. (that is probably true)
- The variables returned by pycosat solver always come ordered by absolute value. (that is also probably true. We could also sort the list ourselves)
- The first (number of colors)*(number of vertices) variables correspond to the colors, the next to modular colors (that is true, but it would be too hackish to rely on it. Changing one part of the code could completely mess up the other part, introducing messy hard to detect errors.)

we could simply discard all negative variables, zip together the first (number of vertices) of the remaining numbers with the vertices of the graph to get the colors, and zip together the next (number of vertices) of the numbers with the vertices to get the modular coloring.

```
def extract_colors(g,vertex_map,result_list):
    """
    Extract colors of each vertex from the result list.

    Args:
        g: a graph_tool graph
        vertex_map: a vertex property map.  Each vertex has a list
            of its color variables. Could be original color map or
            modular map.
        result_list: the list of variable values returned by the
            pycosat solver.

    Returns:
        a new vertex property map: one integer for each vertex, the
        color of the vertex in the coloring found by the solver.
    """
    s = set(i for i in result_list if i > 0)
    coloring = g.new_vertex_property('int')
    for v in g.vertices():
```

```
        for i,n in enumerate(vertex_map[v]):
            if n in s:
                coloring[v] = i
                break

    return coloring
```

We will try to plot the graph to verify that the solver indeed found a coloring with the right properties. We can either use colors to show the original coloring and numbers to show the modular coloring (default), or numbers to show the original coloring and colors to show the modular coloring (set `color_orig` to `False`). The default setting will make it easier to see patterns in the original coloring that lead to a closed modular coloring, while the alternative setting will make it easier to quickly verify that the resulting coloring is indeed a proper closed modular coloring generated by the original coloring.

```
from graph_tool.draw import graph_draw

def plot_graph(g,coloring,modular_coloring,
               color_names,layout,
               filename,size=(600,600),
               color_orig=True):
    fcolors = g.new_vertex_property('string')
    labels = g.new_vertex_property('string')
    if color_orig:
        for v in g.vertices():
            fcolors[v] = color_names[coloring[v]]
            labels[v] = str(modular_coloring[v])
    else:
        for v in g.vertices():
            fcolors[v] = color_names[modular_coloring[v]]
            labels[v] = str(coloring[v])

    graph_draw(g,vertex_fill_color=fcolors,
               vertex_text=labels,
               pos=layout,
               output_size=size,
               output=filename)
```

5 Check the coloring

Just to make sure, we will make an independent check if

1. the produced coloring is indeed a closed modular coloring obtained from the original coloring.
2. the produced coloring is proper.

These can be done the following way:

1. Loop through vertices, calculating modular coloring and comparing it with the assigned modular color.
2. Loop through edges, making sure the two colors at the ends of the edge are different.

```python
def is_modular(g,coloring,modular_coloring,k):
    """
    Check if the coloring described by modular_coloring is really a
    modular coloring.

    Args:
        g: a graph-tool graph
        coloring: a vertex property map describing the original
            coloring
        modular_coloring: a vertex property map describing the
            coloring that is tested for modularity
        k: an integer, the number of colors available

    Returns:
        True if the coloring in modular_coloring is really modular,
        False otherwise
    """
    for v in g.vertices():
        neighbours = list(v.out_neighbours()) + [v]
        true_modular_color = sum(coloring[w]
                                 for w in neighbours) % k
        if modular_coloring[v] != true_modular_color:
            return False

    return True

def is_proper(g,coloring):
    """
    Check if the coloring of g described by coloring is proper

    Args:
        g: a graph-tool graph
        coloring: a vertex property map describing a coloring of g

    Return:
        True if the coloring is proper, False otherwise
    """
    for e in g.edges():
        if coloring[e.source()] == coloring[e.target()]:
            return False

    return True
```

5.1 Test a given graph

Now we take the generated clauses and feed them into the SAT-solver. Then we decode the result, and return a tuple whose first element will be `True` if we found a solution, or `False` if we did not.

In the case a solution was found, the second element will be the string `'SAT'`, and the third and fourth element will be the original coloring and the modular coloring, respectively.

If the solution was not found, the second element of the tuple will be a string explaining what happened:

UNSAT the solver decided that there is no solution.

UNKNOWN the solver was not able to find a solution, and was not able to prove that there is none.

NOT MODULAR the solver found a solution, but the supposedly modular coloring returned by the solver is not a closed modular coloring generated by the original coloring returned by the solver. This should *never* happen!

NOT PROPER the solver found a solution, but the supposedly modular coloring returned by the solver is not a proper. This should *never* happen!

```
def g_has_pcmc(g,k):
   l, c_m, m_c_m = graph_clauses(g,k)
   result = pycosat.solve(l)

   if (result == 'UNSAT') or (result == 'UNKNOWN'):
      return (False,result)

   coloring = extract_colors(g,c_m,result)
   modular_coloring = extract_colors(g,m_c_m,result)

   if not is_modular(g,coloring,modular_coloring,k):
      return(False,'NOT MODULAR',coloring,modular_coloring)
   if not is_proper(g,modular_coloring):
      return(False,'NOT PROPER',coloring,modular_coloring)

   return(True,'SAT',coloring,modular_coloring)
```

5.2 Iteration over all colorings

The Pycosat solver makes it possible to iterate over all solutions, using the `itersolve` function. We will use this to attempt to find all colorings of a given graph that produce a proper closed modular coloring. For graphs that have a large number of such colorings, this can be rather lengthy process, and, especially for larger graphs, will require a large amount of memory. We will therefore provide an option to limit the number of coloring found. We will also provide an option to check if the resulting colorings are proper and closed modular.

```
def g_all_pcmc(g,k,limit=0,check=True):
   l, c_m, m_c_m = graph_clauses(g,k)
   if limit==0:
      results = list(pycosat.itersolve(l))
   else:
      results = list(islice(pycosat.itersolve(l),limit))

   colorings = [extract_colors(g,c_m,r) for r in results]
   modular_colorings = [extract_colors(g,m_c_m,r) for r in results]

   combined = zip(colorings, modular_colorings)

   return [(c,m) for c,m in combined if (not check) or
      (is_modular(g,c,m,k) and is_proper(g,m))]
```

6 Try it on K_5

Since K_5 has true twins, and since we do not check for true twins, there will not be a proper modular coloring of K_5.

```
g = gt.Graph(directed=False)

vs = [g.add_vertex() for i in range(5)]
for v in vs:
   for w in vs:
      if v != w:
         g.add_edge(v,w)

res = g_has_pcmc(g,3)

if not res[0]:
   print(res[1])
else:
   print('Modular coloring is proper')
```

The result was UNSAT .

7 Grid Graphs

We will see if the solver can find closed modular colorings for some grid graphs.

We will start by some imports and some code for properly laying out grid graphs. Grid graphs are available in graph-tool in the `graph_tool.generation` module, and are called 'lattice' graphs. Automatic layout algorithms do not work well for them, so we define our own layout.

```
from graph_tool.generation import lattice

def grid_layout(g,rows,cols):
   grid_pos = g.new_vertex_property("vector<float>")

   for i in range(cols):
      for j in range(rows):
         grid_pos[g.vertex(rows*i + j)] = [i,j]

   return grid_pos
```

The following function will receive a list of pairs (m, n) and an integer k, and return a dictionary containing, for each pair (m, n), the result of `g_has_pcmc` for the grid graph $G_{m,n}$. It will only return the Boolean result and the string result, without the actual colorings.

```
def check_grids(pairs,k):
   return {p:g_has_pcmc(lattice(p),k)[0:2] for p in pairs}
```

Since the graphs $G_{m,n}$ and $G_{n,m}$ are equivalent from our point of view, we only need to check $G_{m,n}$ for $n \leq m$. The following function will generate all such pairs:

```
def triangle(K):
   return ((i,j) for i in range(2,K) for j in range(2,i+1))
```

The following function will then check all the grid graphs $G_{m,n}$ for $2 \leq m, n < K$ (using the above mentioned symmetry), and print the results.

```
def check_all_grids(K,k):
   def format_g(pair):
      return '$G_{{{:d},{:d}}}$'.format(*pair)

   results = check_grids(triangle(K),k)
   graphs_without = [p for p in results if not results[p][0]]
   if len(graphs_without) == 0:
      print('All of the graphs have proper {:d}-modular coloring'.format(k))
   else:
      print('Proper {:d}-modular coloring not found for '.format(k) +
         ', '.join(format(g) for g in sorted(graphs_without)))
```

7.1 $k = 3$

First we will look at grid graphs $G_{m,n}$ for $2 \leq m, n < 21$:

```
check_all_grids(21,3)
```

All of the graphs have proper 3-modular coloring

7.2 $k = 2$

Now we will try with $k = 2$:

```
check_all_grids(21,2)
```

Proper 2-modular coloring not found for (3, 2), (4, 4), (5, 2), (8, 3), (8, 5), (8, 6), (9, 2), (9, 8), (11, 2), (11, 8), (13, 2), (13, 8), (14, 3), (14, 4), (14, 5), (14, 9), (14, 11), (14, 13), (14, 14), (16, 14), (16, 16), (17, 2), (17, 8), (17, 14), (19, 2), (19, 8), (19, 14), (20, 3), (20, 5), (20, 8), (20, 9), (20, 11), (20, 13), (20, 17), (20, 19)

7.3 All Colorings for $3 \times m$

We will look at coloring by 2 colors for $3 \times m$ graphs.

```
def all_for_grid(m,n,k,color_names=['black','white']):
   g = lattice([m,n])
   results = g_all_pcmc(g,k)
   print('There are {} proper closed modular colorings mod {} for the graph $G_{{{:d},{:d}}}$.'.forma

   for i,r in enumerate(results):
      filename = 'grid_{:d}_{:d}_{:d}_{:d}.pdf'.format(m,n,k,i)
      plot_graph(g,r[0],r[1],
                    color_names,
                    grid_layout(g,m,n),
                    filename,size=(50*(n-1),50*m))
      frm = 'width=\\linewidth' if n > 10 else ''
      print('\n\n\\includegraphics[' + frm + ']{' + filename + '}\n\n')
```

?? PythonTeX ??

```
for n in range(2,21):
   print('\n\\subsection{{Colorings for $G_{{3,{:d}}}$}}'.format(n))
   all_for_grid(3,n,2)
```

7.4 Colorings for $G_{3,2}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{3,2}$.

7.5 Colorings for $G_{3,3}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,3}$.

7.6 Colorings for $G_{3,4}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,4}$.

7.7 Colorings for $G_{3,5}$

There are 16 proper closed modular colorings mod 2 for the graph $G_{3,5}$.

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

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

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

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

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

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

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

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

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

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

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

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

7.8 Colorings for $G_{3,6}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,6}$.

7.9 Colorings for $G_{3,7}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,7}$.

7.10 Colorings for $G_{3,8}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{3,8}$.

7.11 Colorings for $G_{3,9}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,9}$.

7.12 Colorings for $G_{3,10}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,10}$.

7.13 Colorings for $G_{3,11}$

There are 16 proper closed modular colorings mod 2 for the graph $G_{3,11}$.

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

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

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

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

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

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

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

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

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

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

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

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

7.14 Colorings for $G_{3,12}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,12}$.

7.15 Colorings for $G_{3,13}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,13}$.

7.16 Colorings for $G_{3,14}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{3,14}$.

7.17 Colorings for $G_{3,15}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,15}$.

7.18 Colorings for $G_{3,16}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,16}$.

7.19 Colorings for $G_{3,17}$

There are 16 proper closed modular colorings mod 2 for the graph $G_{3,17}$.

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

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

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

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

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

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

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

7.20 Colorings for $G_{3,18}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,18}$.

7.21 Colorings for $G_{3,19}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{3,19}$.

7.22 Colorings for $G_{3,20}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{3,20}$.

7.23 All Colorings for $4 \times m$

The same code can be used to generate all coloring by 2 colors for $4 \times m$ graphs, for $m \leq 20$:

```
for n in range(2,21):
   print('\n\\subsection{{Colorings for $G_{{4,{:d}}}$}}'.format(n))
   all_for_grid(4,n,2)
```

7.24 Colorings for $G_{4,2}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,2}$.

7.25 Colorings for $G_{4,3}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,3}$.

7.26 Colorings for $G_{4,4}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{4,4}$.

7.27 Colorings for $G_{4,5}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,5}$.

7.28 Colorings for $G_{4,6}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,6}$.

7.29 Colorings for $G_{4,7}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,7}$.

7.30 Colorings for $G_{4,8}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,8}$.

7.31 Colorings for $G_{4,9}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{4,9}$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.32 Colorings for $G_{4,10}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,10}$.

7.33 Colorings for $G_{4,11}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,11}$.

7.34 Colorings for $G_{4,12}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,12}$.

7.35 Colorings for $G_{4,13}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,13}$.

7.36 Colorings for $G_{4,14}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{4,14}$.

7.37 Colorings for $G_{4,15}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,15}$.

7.38 Colorings for $G_{4,16}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,16}$.

7.39 Colorings for $G_{4,17}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,17}$.

7.40 Colorings for $G_{4,18}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,18}$.

7.41 Colorings for $G_{4,19}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{4,19}$.

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

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

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

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

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

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

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

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

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

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

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

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

7.42 Colorings for $G_{4,20}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{4,20}$.

7.43 All Colorings for $5 \times m$

The same code can be used to generate all coloring by 2 colors for $5 \times m$ graphs, for $m \leq 40$:

```
for n in range(2,41):
   print('\n\\subsection{{Colorings for $G_{{5,{:d}}}$}}'.format(n))
   all_for_grid(5,n,2)
```

7.44 Colorings for $G_{5,2}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,2}$.

7.45 Colorings for $G_{5,3}$

There are 16 proper closed modular colorings mod 2 for the graph $G_{5,3}$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.46 Colorings for $G_{5,4}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,4}$.

7.47 Colorings for $G_{5,5}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{5,5}$.

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

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

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

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

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

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

7.48 Colorings for $G_{5,6}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,6}$.

7.49 Colorings for $G_{5,7}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{5,7}$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.50 Colorings for $G_{5,8}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,8}$.

7.51 Colorings for $G_{5,9}$

There are 4 proper closed modular colorings mod 2 for the graph $G_{5,9}$.

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

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

7.52 Colorings for $G_{5,10}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,10}$.

7.53 Colorings for $G_{5,11}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{5,11}$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.54 Colorings for $G_{5,12}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,12}$.

7.55 Colorings for $G_{5,13}$

There are 4 proper closed modular colorings mod 2 for the graph $G_{5,13}$.

7.56 Colorings for $G_{5,14}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,14}$.

7.57 Colorings for $G_{5,15}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{5,15}$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.58 Colorings for $G_{5,16}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,16}$.

7.59 Colorings for $G_{5,17}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{5,17}$.

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

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

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

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

7.60 Colorings for $G_{5,18}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,18}$.

7.61 Colorings for $G_{5,19}$

There are 16 proper closed modular colorings mod 2 for the graph $G_{5,19}$.

7.62 Colorings for $G_{5,20}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,20}$.

7.63 Colorings for $G_{5,21}$

There are 4 proper closed modular colorings mod 2 for the graph $G_{5,21}$.

7.64 Colorings for $G_{5,22}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,22}$.

7.65 Colorings for $G_{5,23}$

There are 64 proper closed modular colorings mod 2 for the graph $G_{5,23}$.

7.66 Colorings for $G_{5,24}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,24}$.

7.67 Colorings for $G_{5,25}$

There are 4 proper closed modular colorings mod 2 for the graph $G_{5,25}$.

7.68 Colorings for $G_{5,26}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,26}$.

7.69 Colorings for $G_{5,27}$

There are 16 proper closed modular colorings mod 2 for the graph $G_{5,27}$.

7.70 Colorings for $G_{5,28}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,28}$.

7.71 Colorings for $G_{5,29}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{5,29}$.

7.72 Colorings for $G_{5,30}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,30}$.

7.73 Colorings for $G_{5,31}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{5,31}$.

7.74 Colorings for $G_{5,32}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,32}$.

7.75 Colorings for $G_{5,33}$

There are 4 proper closed modular colorings mod 2 for the graph $G_{5,33}$.

7.76 Colorings for $G_{5,34}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,34}$.

7.77 Colorings for $G_{5,35}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{5,35}$.

7.78 Colorings for $G_{5,36}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,36}$.

7.79 Colorings for $G_{5,37}$

There are 4 proper closed modular colorings mod 2 for the graph $G_{5,37}$.

7.80 Colorings for $G_{5,38}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{5,38}$.

7.81 Colorings for $G_{5,39}$

There are 32 proper closed modular colorings mod 2 for the graph $G_{5,39}$.

7.82 Colorings for $G_{5,40}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{5,40}$.

7.83 All Colorings for $6 \times m$

The same code can be used to generate all coloring by 2 colors for $6 \times m$ graphs, for $m \leq 40$:

```
for n in range(2,41):
   print('\n\\subsection{{Colorings for $G_{{6,{:d}}}$}}'.format(n))
   all_for_grid(6,n,2)
```

7.84 Colorings for $G_{6,2}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,2}$.

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

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

7.85 Colorings for $G_{6,3}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,3}$.

7.86 Colorings for $G_{6,4}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,4}$.

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

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

7.87 Colorings for $G_{6,5}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,5}$.

7.88 Colorings for $G_{6,6}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,6}$.

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

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

7.89 Colorings for $G_{6,7}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,7}$.

7.90 Colorings for $G_{6,8}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{6,8}$.

7.91 Colorings for $G_{6,9}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,9}$.

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

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

7.92 Colorings for $G_{6,10}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,10}$.

7.93 Colorings for $G_{6,11}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,11}$.

7.94 Colorings for $G_{6,12}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,12}$.

7.95 Colorings for $G_{6,13}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,13}$.

7.96 Colorings for $G_{6,14}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,14}$.

7.97 Colorings for $G_{6,15}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,15}$.

7.98 Colorings for $G_{6,16}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,16}$.

7.99 Colorings for $G_{6,17}$

There are 128 proper closed modular colorings mod 2 for the graph $G_{6,17}$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.100 Colorings for $G_{6,18}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,18}$.

7.101 Colorings for $G_{6,19}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,19}$.

7.102 Colorings for $G_{6,20}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,20}$.

7.103 Colorings for $G_{6,21}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,21}$.

7.104 Colorings for $G_{6,22}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,22}$.

7.105 Colorings for $G_{6,23}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,23}$.

7.106 Colorings for $G_{6,24}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,24}$.

7.107 Colorings for $G_{6,25}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,25}$.

7.108 Colorings for $G_{6,26}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{6,26}$.

7.109 Colorings for $G_{6,27}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,27}$.

7.110 Colorings for $G_{6,28}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,28}$.

7.111 Colorings for $G_{6,29}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,29}$.

7.112 Colorings for $G_{6,30}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,30}$.

7.113 Colorings for $G_{6,31}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,31}$.

7.114 Colorings for $G_{6,32}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,32}$.

7.115 Colorings for $G_{6,33}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,33}$.

7.116 Colorings for $G_{6,34}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,34}$.

7.117 Colorings for $G_{6,35}$

There are 128 proper closed modular colorings mod 2 for the graph $G_{6,35}$.

7.118 Colorings for $G_{6,36}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,36}$.

7.119 Colorings for $G_{6,37}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,37}$.

7.120 Colorings for $G_{6,38}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,38}$.

7.121 Colorings for $G_{6,39}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,39}$.

7.122 Colorings for $G_{6,40}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{6,40}$.

7.123 A Matrix

Next we will calculate how many closed 2-modular colorings are there for grid graphs $G_{m,n}$ for $2 \leq m, n < 18$. Since the calculation is very long for graphs that have large number of such colorings, we will stop after finding 33 colorings for each graph. We will list these as "> 32" in the table.

```
k = 2
L = 18
M = [[0]*(L-2) for i in range(2,L)]
for m in range(2,L):
   for n in range(2,m+1):
      g = lattice([m,n])
      l, c_m, m_c_m = graph_clauses(g,k)
      M[m-2][n-2] = len(list(islice(pycosat.itersolve(l),33)))
      M[n-2][m-2] = M[m-2][n-2]

print('\\begin{tabular}{r|' + 'c'*(L-2) + '}\n')
print('&' + '&'.join(str(i) for i in range(2,L)) + '\\\\\\hline\n')
for i,R in enumerate(M):
   print(str(i+2) + '&' + '&'.join(str(j) if j < 33 else '$>32$' for j in R) + '\\\\\n')
print('\\end{tabular}')
```

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2	2	0	2	0	2	8	2	0	2	0	2	0	2	8	2	0
3	0	2	2	16	2	2	0	2	2	16	2	2	0	2	2	16
4	2	2	0	2	2	2	2	32	2	2	2	2	0	2	2	2
5	0	16	2	8	2	32	0	4	2	32	2	4	0	32	2	8
6	2	2	2	2	2	2	0	2	2	2	2	2	2	2	2	> 32
7	8	2	2	32	2	2	8	2	2	> 32	2	2	8	2	2	32
8	2	0	2	0	0	8	2	0	2	0	2	0	2	8	2	0
9	0	2	32	4	2	2	0	> 32	2	4	2	2	0	2	2	4
10	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
11	0	16	2	32	2	> 32	0	4	2	> 32	2	4	0	> 32	2	32
12	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
13	0	2	2	4	2	2	0	2	2	4	2	2	0	2	2	> 32
14	2	0	0	0	2	8	2	0	2	0	2	0	0	8	0	0
15	8	2	2	32	2	2	8	2	2	> 32	2	2	8	2	2	32
16	2	2	2	2	2	2	2	2	2	2	2	2	0	2	0	2
17	0	16	2	8	> 32	32	0	4	2	32	2	> 32	0	32	2	8

7.124 All Colorings for $10 \times m$

```
for n in range(2,41):
   print('\n\\subsection{{Colorings for $G_{{10,{:d}}}$}}'.format(n))
   all_for_grid(10,n,2)
```

7.125 Colorings for $G_{10,2}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,2}$.

7.126 Colorings for $G_{10,3}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,3}$.

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

7.127 Colorings for $G_{10,4}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,4}$.

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

7.128 Colorings for $G_{10,5}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,5}$.

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

7.129 Colorings for $G_{10,6}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,6}$.

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

7.130 Colorings for $G_{10,7}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,7}$.

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

7.131 Colorings for $G_{10,8}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,8}$.

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

7.132 Colorings for $G_{10,9}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,9}$.

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

7.133 Colorings for $G_{10,10}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,10}$.

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
1 0 1 0 1 0 1 0 1 0

7.134 Colorings for $G_{10,11}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,11}$.

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

7.135 Colorings for $G_{10,12}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,12}$.

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

7.136 Colorings for $G_{10,13}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,13}$.

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

7.137 Colorings for $G_{10,14}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,14}$.

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

7.138 Colorings for $G_{10,15}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,15}$.

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

7.139 Colorings for $G_{10,16}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,16}$.

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

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

7.140 Colorings for $G_{10,17}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,17}$.

7.141 Colorings for $G_{10,18}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,18}$.

7.142 Colorings for $G_{10,19}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,19}$.

7.143 Colorings for $G_{10,20}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,20}$.

7.144 Colorings for $G_{10,21}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,21}$.

7.145 Colorings for $G_{10,22}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,22}$.

7.146 Colorings for $G_{10,23}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,23}$.

7.147 Colorings for $G_{10,24}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,24}$.

7.148 Colorings for $G_{10,25}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,25}$.

7.149 Colorings for $G_{10,26}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,26}$.

7.150 Colorings for $G_{10,27}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,27}$.

7.151 Colorings for $G_{10,28}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,28}$.

7.152 Colorings for $G_{10,29}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,29}$.

7.153 Colorings for $G_{10,30}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{10,30}$.

7.154 Colorings for $G_{10,31}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,31}$.

7.155 Colorings for $G_{10,32}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,32}$.

7.156 Colorings for $G_{10,33}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,33}$.

7.157 Colorings for $G_{10,34}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,34}$.

7.158 Colorings for $G_{10,35}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,35}$.

7.159 Colorings for $G_{10,36}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,36}$.

7.160 Colorings for $G_{10,37}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,37}$.

7.161 Colorings for $G_{10,38}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,38}$.

7.162 Colorings for $G_{10,39}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,39}$.

7.163 Colorings for $G_{10,40}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{10,40}$.

7.164 All Colorings for $14 \times m$

```
for n in range(2,41):
   print('\n\\subsection{{Colorings for $G_{{14,{:d}}}$}}'.format(n))
   all_for_grid(14,n,2)
```

7.165 Colorings for $G_{14,2}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,2}$.

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

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

7.166 Colorings for $G_{14,3}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,3}$.

7.167 Colorings for $G_{14,4}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,4}$.

7.168 Colorings for $G_{14,5}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,5}$.

7.169 Colorings for $G_{14,6}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,6}$.

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

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

7.170 Colorings for $G_{14,7}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{14,7}$.

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

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

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

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

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

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

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

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

7.171 Colorings for $G_{14,8}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,8}$.

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

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

7.172 Colorings for $G_{14,9}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,9}$.

7.173 Colorings for $G_{14,10}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,10}$.

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
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

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
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

7.174 Colorings for $G_{14,11}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,11}$.

7.175 Colorings for $G_{14,12}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,12}$.

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

7.176 Colorings for $G_{14,13}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,13}$.

7.177 Colorings for $G_{14,14}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,14}$.

7.178 Colorings for $G_{14,15}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{14,15}$.

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

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

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

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

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

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

7.179 Colorings for $G_{14,16}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,16}$.

7.180 Colorings for $G_{14,17}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,17}$.

7.181 Colorings for $G_{14,18}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,18}$.

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

7.182 Colorings for $G_{14,19}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,19}$.

7.183 Colorings for $G_{14,20}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,20}$.

7.184 Colorings for $G_{14,21}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,21}$.

7.185 Colorings for $G_{14,22}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,22}$.

7.186 Colorings for $G_{14,23}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{14,23}$.

7.187 Colorings for $G_{14,24}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,24}$.

7.188 Colorings for $G_{14,25}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,25}$.

7.189 Colorings for $G_{14,26}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,26}$.

7.190 Colorings for $G_{14,27}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,27}$.

7.191 Colorings for $G_{14,28}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,28}$.

7.192 Colorings for $G_{14,29}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,29}$.

7.193 Colorings for $G_{14,30}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,30}$.

7.194 Colorings for $G_{14,31}$

There are 8 proper closed modular colorings mod 2 for the graph $G_{14,31}$.

7.195 Colorings for $G_{14,32}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,32}$.

7.196 Colorings for $G_{14,33}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,33}$.

7.197 Colorings for $G_{14,34}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,34}$.

7.198 Colorings for $G_{14,35}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,35}$.

7.199 Colorings for $G_{14,36}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,36}$.

7.200 Colorings for $G_{14,37}$

There are 0 proper closed modular colorings mod 2 for the graph $G_{14,37}$.

7.201 Colorings for $G_{14,38}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,38}$.

7.202 Colorings for $G_{14,39}$

There are 128 proper closed modular colorings mod 2 for the graph $G_{14,39}$.

7.203 Colorings for $G_{14,40}$

There are 2 proper closed modular colorings mod 2 for the graph $G_{14,40}$.