Picross

Description of the problem

Picross is a popular game of a similar nature to Sudoku.

It consists of a grid of square dimension $n \times n$ and of a list of integers for each row and each column.

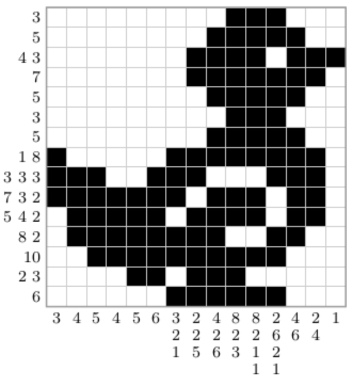

Figure 1: Solution for the duck problem

We can use the following notations:

- $b_{i,j} \in \{0,1\}$ represents a bit, indicating a column or a row where $i, j \in \{1, ..., n\}$ represents a column or row index;
- $m_i \in \{1, ..., \lceil n/2 \rceil\}$ represents the size of the list associated to row/column i;
- $a_{i,k} \in \{1, ..., n\}$ is the k-th integer of the list associated to row/column i.

The goal is to darken some cells of the grid, such that in each row/column i the number of successive black cells form the list $\{a_{i,1}, ..., a_{i,m_i}\}$.

Paper submission

Before heading to the code, you shall answer the following questions.

Please **handwrite** your answers on a separate sheet of paper. You will submit a pdf scan of this document together with your notebook.

(You may take a picture with your phone if need be but the pdf option is preferred.)

1. There are two ways to model the problem:
   - binary variables (1) associated to the color of each cell, or
   - integer variables (2) associated to the bottom/left coordinate of each black cluster.

   For **both** ways, specify the decision variables and their associated domain.

2. Write the constraint stating that on each line and column, two blocks do not overlap.
   Choose **one** of the two ways of modelling the problem (i.e. decision variables from (1) or from (2)) to express this constraint.

3. Write the constraint that puts a limit to the number of black cells on each line/column.
   Choose **one** of the two ways of modelling the problem (i.e. decision variables from (1) or from (2)) to express this constraint.

4. Write the constraint that links decision variables from (1) to decision variables from (2), stating the conditions under which a cell must be coloured in black.

Before you start coding, **please take time to understand** what happened: we stated two different ways to model the same problem (question 1.). Both are convenient for stating one part of the problem, but cannot state the whole set of constraints (question 2. and 3.).

One constraint (question 4.) helps linking both decision variables.

You are provided with a set of problems in the Python dictionary `picross`. Your code will be evaluated on all these problems, but you should start with small problems like `moon`, `star` and `cat`. The most time-consuming should be `house` so do not try it before your code is stable. You may also try more complicated example (e.g. `artist`, referring to the file `picross/artist.non`)

Have a look at variables `lines` and `columns`, since you will have to feed this data into your problem. From now, it should be (almost) direct to code the constraints you have written on paper into Python.

Code submission

Return a file named `picross_solve.py` with a function with the following signature. Your code will be automatically evaluated on many problems: you will not get any point if you do not follow this simple convention.

```python
def picross_solve(
    lines: list[list[int]],
    columns: list[list[int]],
) -> tuple[facile.Solution, facile.Array]:
    n, m = len(lines), len(columns)
    grid = facile.Array.binary((n, m))

    return sol, grid
```

A template is given for your convenience. You may run it as:

```
python picross_solve.py  # moon by default
python picross_solve.py house
```