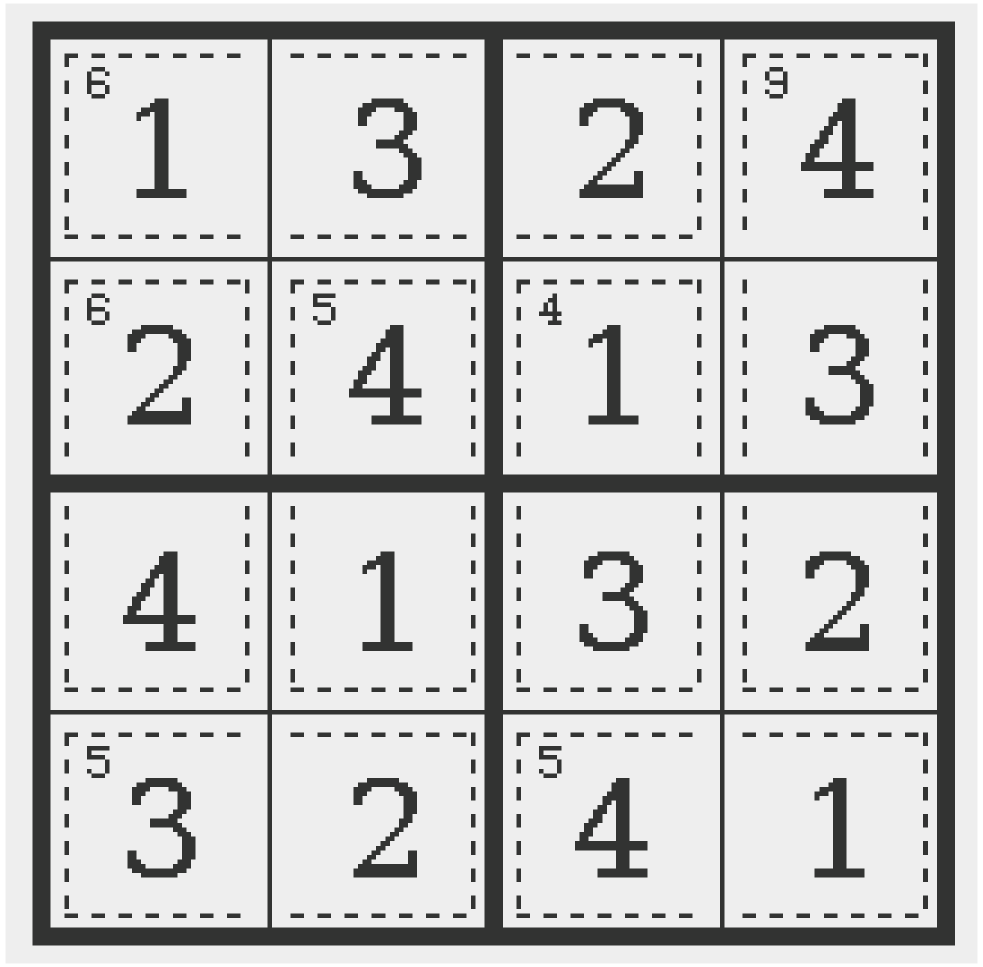
KILLER SUDOKU

KNOWLEDGE REPRESENTATION PROJECT

TEAM 7 – MARCO BELLIZZI, DIEGO DE BARTOLO, DOMENICO SPAGNOLO

https://github.com/MarcoBellizzi/Killer-Sudoku

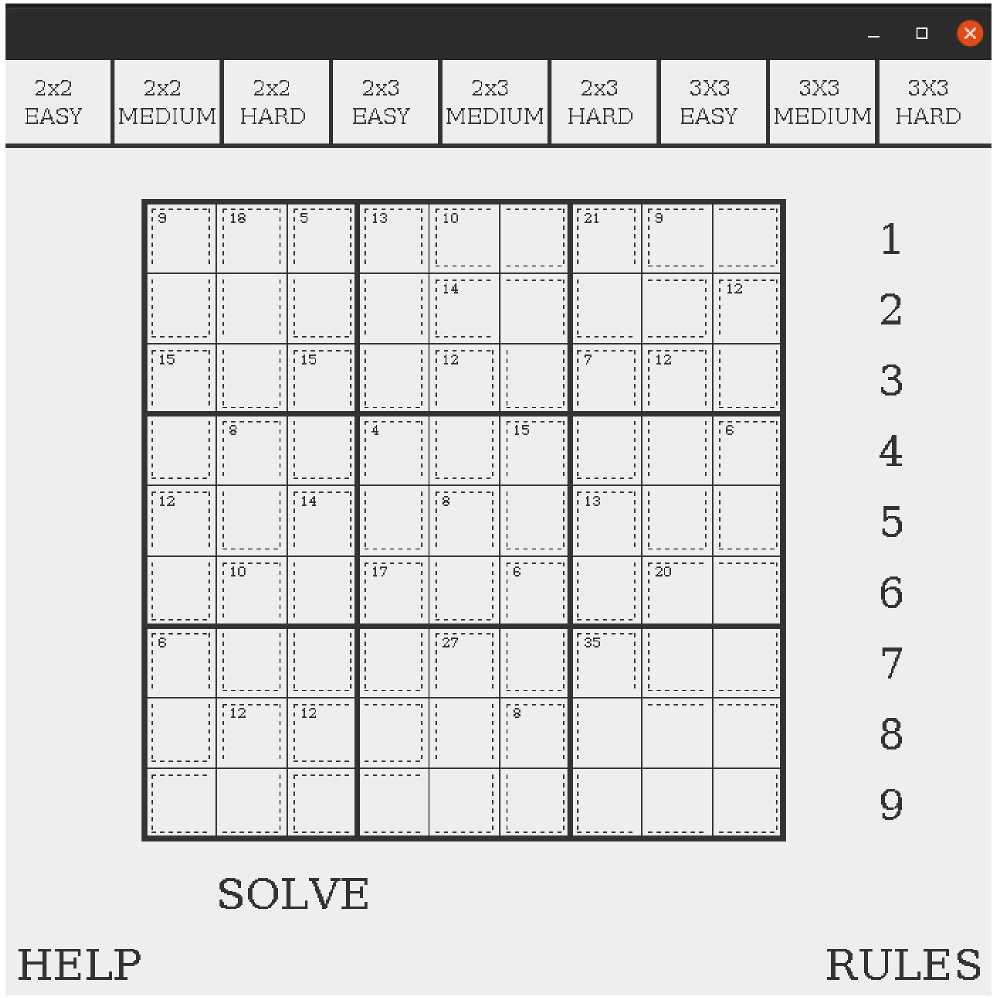


KILLER SUDOKU - RULES

Killer sudoku is a puzzle that combines elements of sudoku and kakuro.

The rules are the following:

- Each row, column, and sector contains each number exactly once.
- The sum of all numbers in a cage must match the small number printed in its corner.
- No number appears more than once in a cage.



REALIZATION

This project is made in Java, using Java Swing. The project uses Minizinc to check the solutions. It requires Internet connection to get the instances from Internet.

MINIZINC MODEL SNEAK PEEK

```
constraint forall(i in 1..num_of_regions) (    % elements in each region are different
  forall(j in 1..regions[i, 2]) (
    alldifferent([matrix[regions[i, j*2 +1], regions[i, j*2 +2]]])
  )
);

constraint forall(i in 1..num_of_regions) (    % the sum of each region is equal to the sum
of each element in that region
  regions[i, 1] == sum(j in 1..regions[i, 2]) (matrix[regions[i, j*2 +1], regions[i, j*2 +2]])
);
```

INPUT VALIDATION

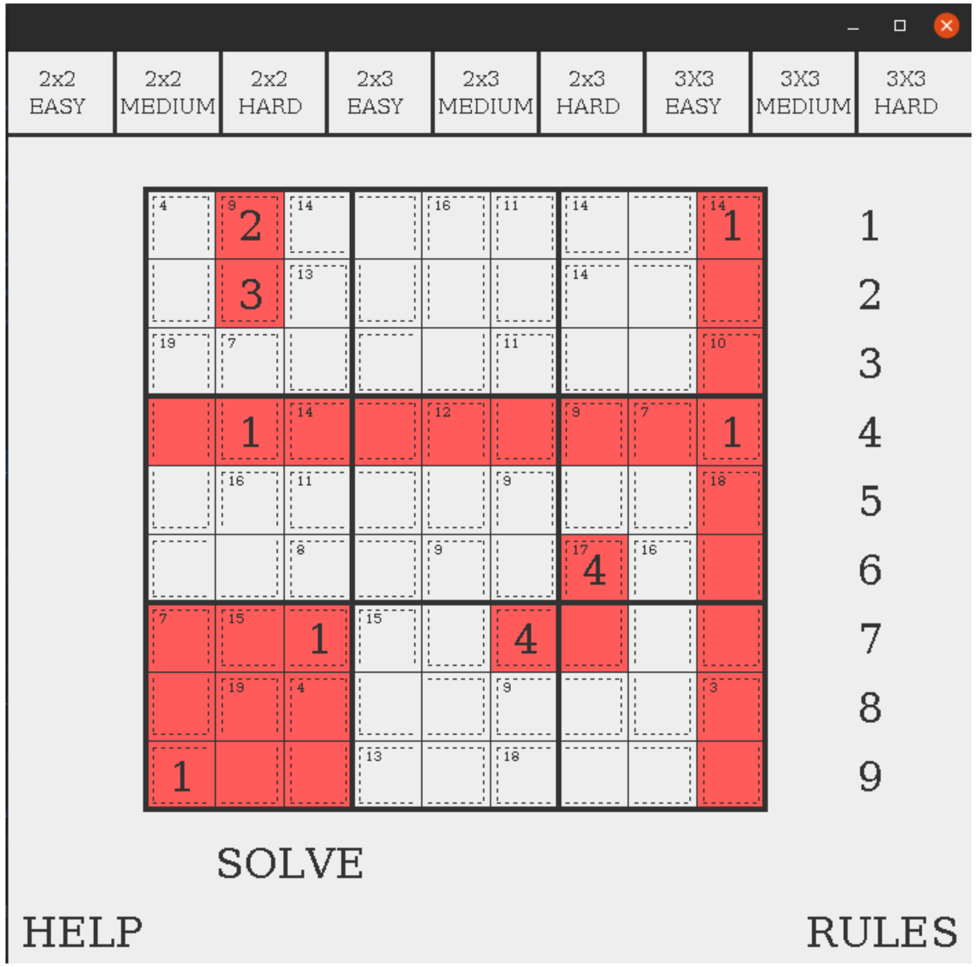
```
constraint assert(forall(i in 1..n*m, j in 1..n*m) (   % user input validation
    matrix[i,j] > 0 /\ matrix[i,j] <= n*m
), "wrong user input value");

constraint assert(forall(i in 1..num_of_regions) (    % site input validation
  forall(j in 1..regions[i, 2]) (
    regions[i, j*2 +1] > 0 /\ regions[i, j*2 +2] <= n*m
  )
), "wrong site input value");
```

INSTANCES

- curl https://www.puzzle-killer-sudoku.com/ | grep task | sed \"s/.*task = '//\" | sed \"s/'.*//\

- The project retrives the instances trought the command shown above. Parse the response, extracting the information required.



SUGGESTIONS

The GUI tells you if you do any mistakes:

- Color the row, the column, the sector or the cage in red if find at least two elements equal.
- Color the cage in red if the sum doesn't match the value on its corner.

THANKS FOR THE ATTENTION