Faculty of Engineering,
Ain Shams university

AI Project: Kenken Puzzle Solver

Submitted to: Dr. Manal Morad Zaki

Eng. Ahmed Mostafa

Student Personal Information for Group Work

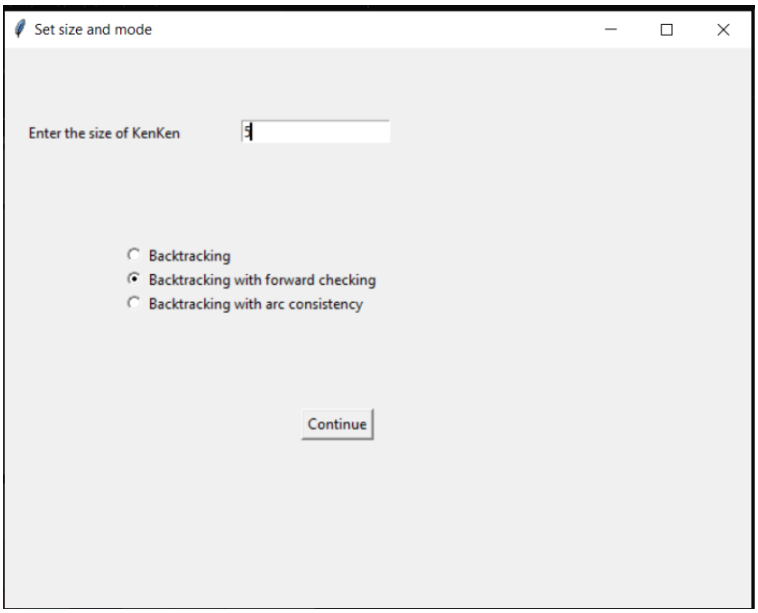
Student Names	Student Codes and Section:
آلاء ابراهيم محمد ابراهيم عامر	1700267, Section 1
آلاء شعبان حسين على شتات	1700271, Section 1
سلمى على عبدالحليم	1600654, Section 2
مروة فؤاد حسين	1701394, Section 4
مريم عادل	1701402, Section 4

Faculty of Engineering,
Ain Shams university

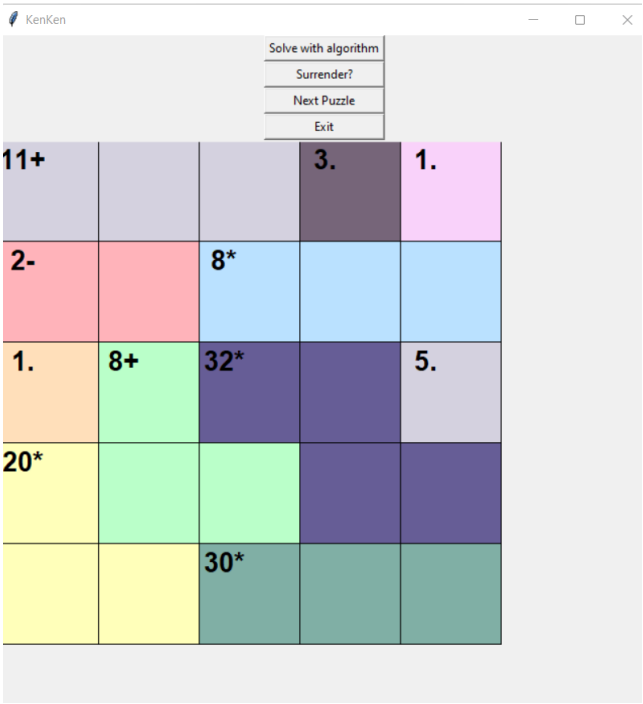
Table of Contents

GUI screenshots:

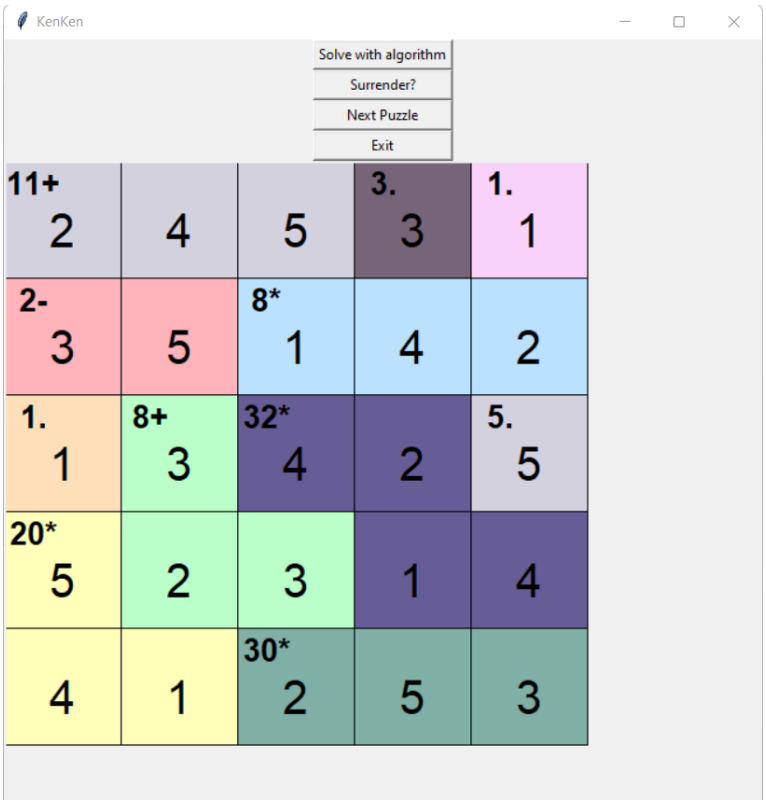
- The first GUI:



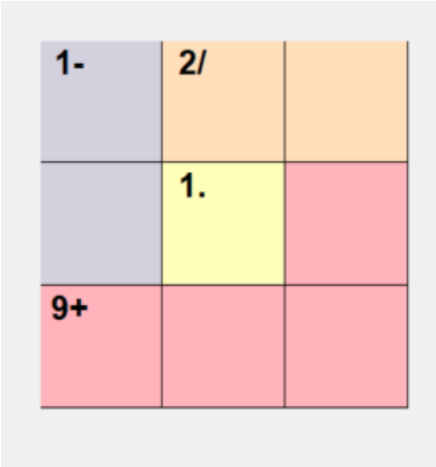
- The second GUI:
  - As we can see the cells within the same cage have the same color.
  - If the operator is “.”, then the cage contains one cell ”kind of clue”.

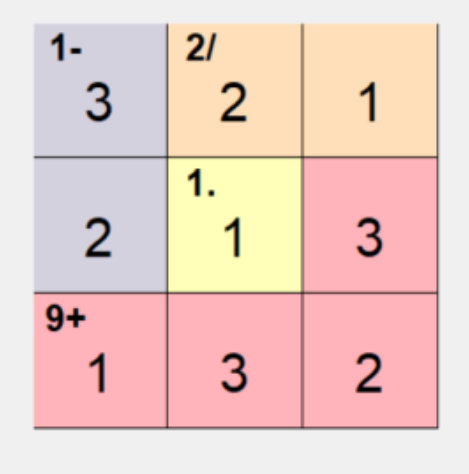


- Solved:

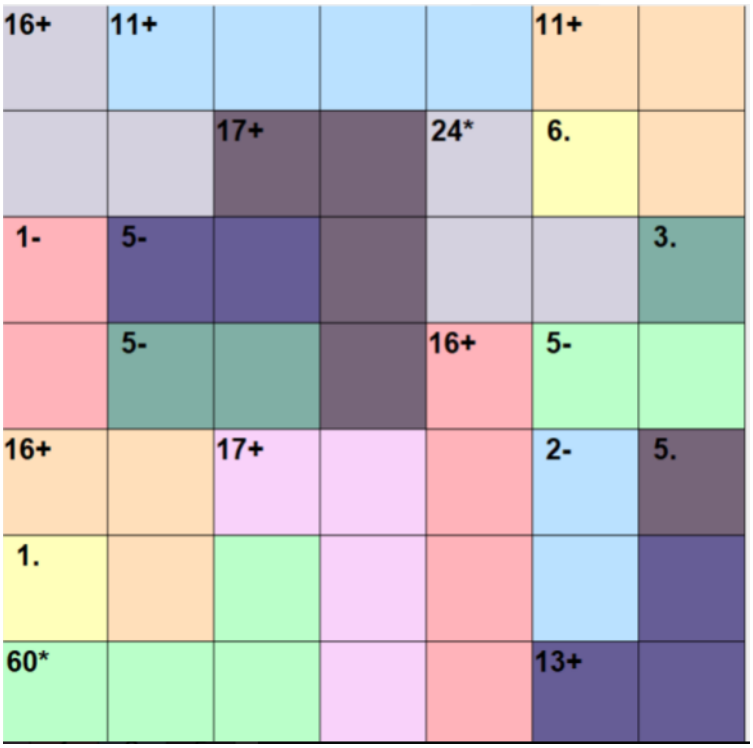


- Screenshots samples:
- 3*3





- 7*7 example:



- After solving:

Classes used:

GUI

- First, in our application the first GUI that will pop up, the user will enter the size of the kenken puzzle, the algorithm needed:
    - Function that contains the size entry and the Radiobuttons, and return the size.
    - Function to return the selected algorithm.
- The second GUI will pop up containing the kenken puzzle with some buttons, you can solve with the selected algorithm, you can solve without any algorithm, select new puzzle, or exit the game:
    - The class mainly gets the size, and the mode of the selected algorithm.
    - Function to create the grid and gets the value from the Generator class to show the complete board.
    - The cages are colored, so there is a function to color the cells within the same cage with the same color.
    - Surrender function: to solve the puzzle.
    - Modsolver function: to call the required function and solve using the selected algorithm.
    - Next function: to generate the next puzzle.

Generator

- The main goal of this class is to generate the board:
- Functions to achieve this goal:
    - Init function that takes the size from the gui, initialize the solution.
    - Adjacent function: to ensure that each adjacent cells are not the same number.
    - Operation function: decide the operation of the cage.
    - Generate function: to generate the puzzle and returns the solution.
    - Op_gui function: to formulate the puzzle so it can be represented at the GUI.
    -

Problem

A formal problem's abstract class.

CSP

- Class describes finite-domain Constraint Satisfaction Problems. A CSP is specified by the following inputs:
    - Variables: A list of variables; each is atomic (e.g. int or string).
    - Domains: A dictionary of {var:[possible_value, ...]} entries.
    - Neighbors: A dictionary of {var:[var,...]} that for each variable list the other variables that participate in constraints.

- Constraints: A function f(A, a, B, b) that returns true if neighbors. A, B satisfy the constraint when they have values A=a, B=b.

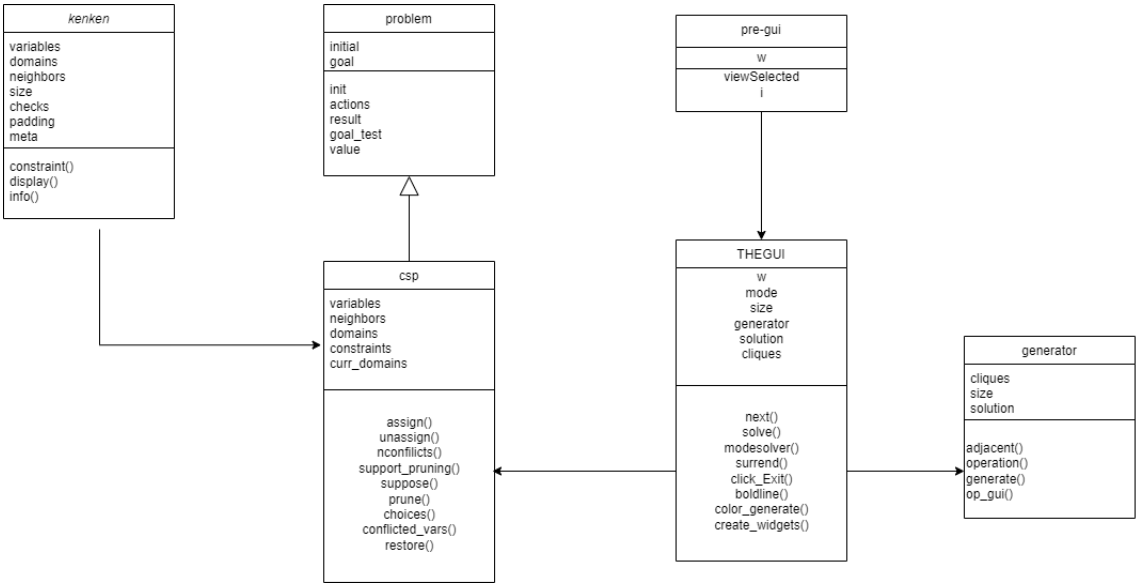
Kenken

- Cliques are input to the class:
- In our implementation, we consider the cliques themselves as variables.
- A clique is of the format (((X1, Y1), ..., (XN, YN)), <operation>, <target>)

  Where

  - (X1, Y1), ..., (XN, YN) are the members of the clique
  - <operation> is either addition, subtraction, division or multiplication
  - <target> is the value that the <operation> should produce
  - when applied on the members of the clique.
- Prepare the required data for the CSP and the algorithms: variables, domain, neighbors, constraints.

Class diagram

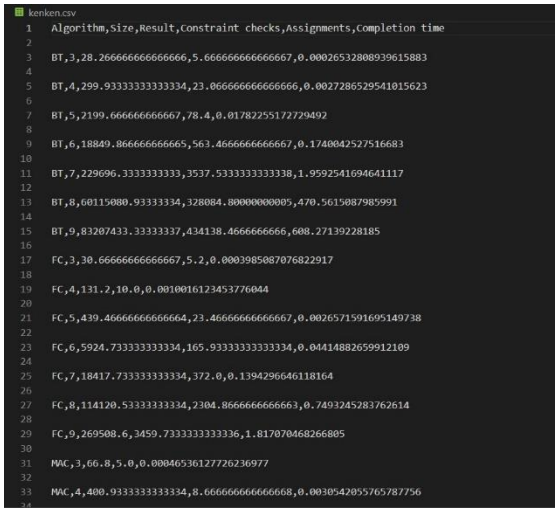


Faculty of Engineering,
Ain Shams university

Performance analysis:

on running the three algorithms on 100 boards with different sizes it's found that backtracking with forward checking is the fastest and that's because of the fact that backtracking with Forward checking detects the inconsistency earlier than simple backtracking and thus it allows branches of the search tree that will lead to failure to be pruned earlier than with simple backtracking.

Sample of the output:



kenken.csv

```
Algorithm,Size,Result,Constraint checks,Assignments,Completion time

BT,3,28.266666666666666,5.666666666666667,0.0002653208939615883

BT,4,299.93333333333334,23.066666666666666,0.0027286529541015623

BT,5,2199.666666666667,78.4,0.01782255172729492

BT,6,18849.866666666665,563.4666666666667,0.1740042527516683

BT,7,229696.3333333333,3537.5333333333338,1.9592541694641117

BT,8,60115080.93333334,328084.80000000005,470.5615087985991

BT,9,83207433.33333337,434138.4666666666,608.27139228185

FC,3,30.66666666666667,5.2,0.0003985087076822917

FC,4,131.2,10.0,0.0010016123453776044

FC,5,439.46666666666664,23.46666666666667,0.002657159169514973 8

FC,6,5924.733333333334,165.93333333333334,0.04414882659121 09

FC,7,18417.733333333334,372.0,0.1394296646118164

FC,8,114120.53333333334,2304.8666666666663,0.749324528376261 4

FC,9,269508.6,3459.7333333333336,1.817070468266805

MAC,3,66.8,5.0,0.0004653612772623697 7

MAC,4,400.9333333333334,8.666666666666668,0.003054205576578775 6
```

Notes:

- The run command can be using debugging mode so we can trace the performance time for the three algorithms:
    - Run this command to get the performance: python main.py 1.
    - Run this command to get the GUI: python main.py 0.
- Performance function is added so we can test the performance on 100 boards.
- Screenshots sample from performance:
    - The forward is the least time, and the best performance.

```
Alaa@DESKTOP-J47U1EQ MINGW64 /d/4thCSE/2ndTerm/AI/project/KENKEN-Puzzle (main)
$ python main.py 1
the debug mode is1
the backtracking algorithm 3.162520408630371
the backtracking with forward checking algorithm 2.8190009593963623
the backtracking with arc consistency  algorithm 3.979001045227051
```

Faculty of Engineering,
Ain Shams university

Working files

- You can find the code in the github link below:

  https://github.com/Alaa-Ibrahim-Amer/KENKEN-Puzzle

- You can find the video in the link below:
  https://drive.google.com/drive/folders/1LZKOxFHSdyIOK3cuAx836fdiyC7UACUv?usp=sharing

- You can find the executable file of the GUI in the repository.
- The class diagram link for better resolution:
  https://drive.google.com/file/d/133DrFf_FmI2IOZ4ujvHspWC7mIpKj8cf/view?usp=sharing