Homework 2 – AI

Leonard Christopher Limanjaya – 20225087

1. Implement Sudoku solver with CSP and any suitable search methods.

Variables (X): each empty cell on the sudoku board

Domains (D): A set number (1-9) which already used in a row, column, or 3x3 squares

Constraints (C): No repeating number in rows, columns, and the 3x3 squares

I watch this video explanation for the reference of making this code and making my own approach:

<https://towardsdatascience.com/solving-nonograms-with-120-lines-of-code-a7c6e0f627e4>

How the code works:

1. Get input from the terminal with the following format:

Table

Description automatically generated

The following function will get input from the terminal and store it to the list

Text

Description automatically generated

1. Find any zero value on the sudoku board. The code will run solve() function to solve the sudoku problem.

Text

Description automatically generated

At first, this function will run find\_zero() function to get the position of 0 value as the return value. This function will go through the entire list and search where is the 0. If there is no 0 value in the list, then the function will return -1, -1

Text

Description automatically generated

1. Try to choose the number that available on that point

Text

Description automatically generated

In this step the code will try to fill the empty one with 1-10, and the program will check if the value is available or not from the check\_value() function. If it is available, the code will assign that number to the sudoku matrix. And the program will continue to run solver() function recursively. The following picture will show the code to check\_value

Text

Description automatically generated

1. If at some point it can not continue to fill until no zero left, the code will go back to the previous step/tiles
2. Repeat steps b – d until no remaining zero left. This happened when the find\_zero() function return -1,1 and the solve() function will return False.
3. Print the solved sudoku
4. Implement a Nonogram solver with CSP and any suitable search methods.

Variables (X): start cells of row or columns segments

Domains (D): -1, 0, and 1. -1 denoted as white, 1 as black, and 0 as unknown

Constraints (C): Relationship between adjacent segments, each segment will have black space(s) between them if the segments is more than 2

I read this article to understand the algorithm how to solve this problem, and I make my own approach:

<https://towardsdatascience.com/solving-nonograms-with-120-lines-of-code-a7c6e0f627e4>

How the code works

1. Get input from the terminal with the following format:

Text

Description automatically generated with low confidence

The following picture is the code to get the input from terminal

Text

Description automatically generated

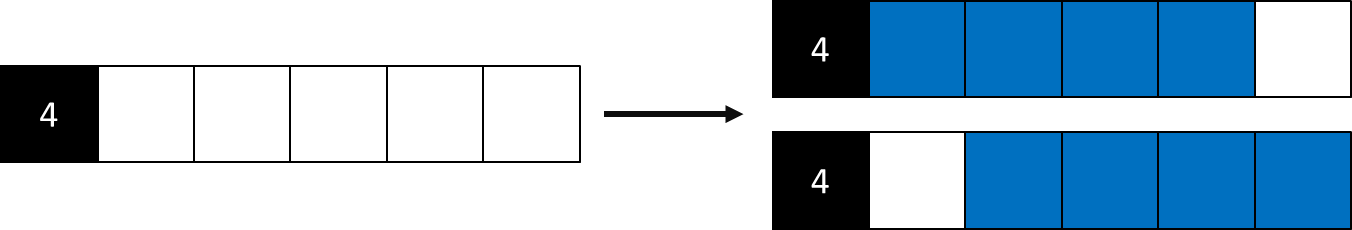
From the terminal input, the code will be store on the variable with list format

1. First, after the program gets the input, the program will generate the possible combination of each row and column. To make this, this program will utilize itertools library to make the combination. These three functions will help to make the combination.

Text

Description automatically generated

This step will be visualized in picture shown below:

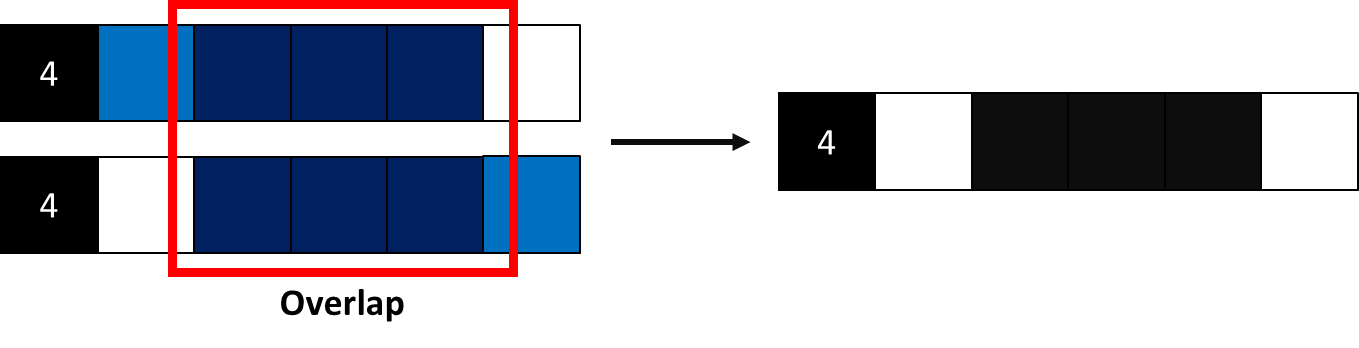


1. After that, the program gets the overlap tiles from each row and column. This process included the black tiles and white tiles.

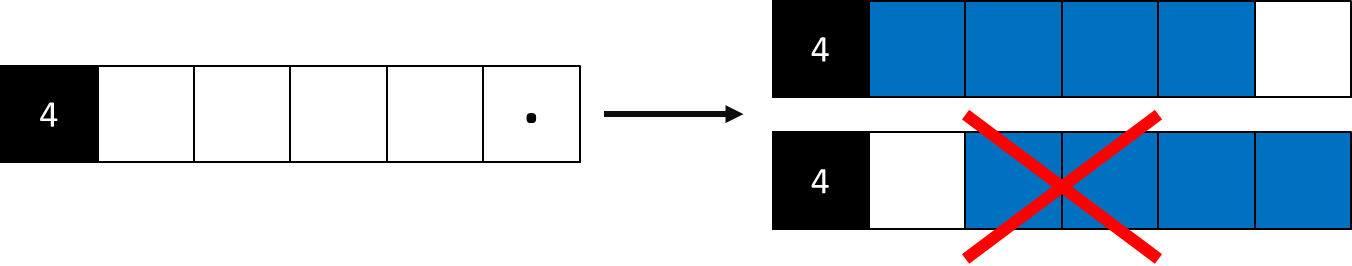
Text

Description automatically generated

This picture below will visualized this step:

****

1. The program will check if the possible tiles in each row violate known tiles (black and white). If the possible rows or columns violate the known tiles, the possible rows or columns will be removed.



The below possible tiles are violating the known tiles, so this possible row will be removed.

1. The program will start again from step 2 until the remaining possible rows or columns in each row and column is 1.
2. In the end, the code will print the result, 1 will represent “\*”, and -1 will be represented by “ “.