The following are the sudoku rules according to the following website: <https://sudoku.com/sudoku-rules>:

“

* Sudoku grid consists of 9x9 spaces.
* You can use only numbers from 1 to 9.
* Each 3×3 block can only contain numbers from 1 to 9.
* Each vertical column can only contain numbers from 1 to 9.
* Each horizontal row can only contain numbers from 1 to 9.
* Each number in the 3×3 block, vertical column or horizontal row can be used only once.
* The game is over when the whole Sudoku grid is correctly filled with numbers.

”

The only difference to their sudoku rule and the sudoku rule of this program is that the game is over when the user chooses to get the solved sudoku, when the user chooses to exit or when the sudoku is filled with the right numbers.  
  
The application functions as follows:

1. The initial sudoku starts by creating a 9x9 sudoku board, with the help of NumPy that is populated with the integer 0. (def\_\_init\_(self): method)
2. After which it goes to the next method in the SudokuBoard Class, (def print\_board(self). This method is responsible for the look of the sudoku board.
   1. The board will have the letters from A-I on top of the board.
   2. After every 3rd row. It will have 25 of the following symbol: “-“
   3. It prints numbers incrementally add adds the following symbol: “|”
   4. It prints this symbol “|” after every 3rd column
   5. And lastly if there is an empty cell on the board it will change them to asterisks.
3. Now that it has a look the program will get a list of keys from the puzzle dictionary (that is in the file called sudoku\_puzzles.py. this is imported into the main sudoku.py file), chooses a random puzzle and uses this to populate the board.
4. The program proceeds to a conditional statement (while loop) where the program will continue to run while the conditions mentioned are met.
   1. It will ask the user to input the position of the cell that they want to add a number to
   2. After that it will ask for the number the user would like to add.
   3. When this is done,

* the program will check (place\_num method) if the position input has the length of 2 characters and if the first character is an alphabet and if the second character is an integer.
* If it is the right input, it will convert the first character to the right column and the second character to the right row.
* When this is done, it will check if the second user input, for the number has the right parameter (a number from 1-9)
* If all this is correct it will check if the inputted number in the specified position is a valid number to be placed according to the sudoku rules
  + (check\_valid): checks if it is valid number in the row and column of the board, checks if the number is valid for that 3x3 block (loops through row and column again) if it is a valid number, it will return as True, meaning that it is all right to place the number.
* The program will now go back to placing the number (place\_num method) and in turn will go back to the while contidional statement
  1. (While loop) Here it checks if the placed number results in a solved sudoku board.
     + The program jumps to the is\_solved method, where it compares the board with the solution\_key , that is found in the sudoku\_puzzles.py file.
  2. It goes back to the while statement and it prints out a statement as well as the completed board if the inputted number resulted in the completion of the board. If the placement of the number did not result in the completion of the board it will print out a statement that it was the right number.
  3. If the inputted number is wrong it will print out a message informing the user that it was a wrong input
  4. After either right or wrong number placement the board is printed.
  5. After the first number input (wrong or right), the user will be asked if it wants the sudoku to be solved.
  6. If yes, the program will go to solve\_sudoku method and
     + solves the sudoku by going row by row, column by column and checks if the value is 0, tries the numbers from 1 to 9 ,
* calls the check\_valid method and checks if it is a valid move, and checks the next cell, and returns a solved sudoku if it is solvable.
  + - Goes back to the previous code and prints the solved sudoku board if the sudoku is solvable and exits the game
  1. If the sudoku is not solvable it will print that there is no solution to the sudoku puzzle
  2. If the user chose not to get the sudoku puzzle solved, it will ask the user if it wants to exit the game. If yes(Y) it will exit the game if no (N) it will go back to the beginning of the while loop.

Def \_\_init\_\_(self):

Print\_board(self)

Import random sudoku puzzle sudoku\_puzzles.py

Input the position and number

place\_num

check\_valid

While loop:Uppdate  
 +   
 input for autosolve puzzle

Is\_solved

Solution\_key

solve\_sudoku

While loop: exit?

exit program

yes

yes

no