Chapter 7: Sudoku Maker

Description: This program will create and verify a sudoku puzzle.

Why?: Learning backtracking. We have used regression before but not to this level. You can make a program that tries every possibility. There are two types of this type of regression: depth first search and breadth first search. We will be doing a depth first search here.

To-do:

- Create grid
- Make difficulty
- Check if solvable

Note: I borrowed some code from here, also this video is where I got the idea.
[Python Sudoku Solver - Computerphile]()

Code:

```
import random
import numpy as np

grid = []
fin_grid = []
happened = False


def make_sudoku(difficulty):
    global grid, fin_grid, happened
    values = [1, 2, 3, 4, 5, 6, 7, 8, 9]
    happened = False
    grid = [[0 for i in range(9)] for j in range(9)]

    random.shuffle(values)
    for i in range(9):
        grid[i][0] = values[i]

    generate()

    print("CREATED")
    squares_to_remove = 0
    if difficulty == 0:
        squares_to_remove = 36
    elif difficulty == 1:
```

```python
        squares_to_remove = 46
    elif difficulty == 2:
        squares_to_remove = 52
    else:
        return

    print("STARTING SUDOKU")
    while squares_to_remove > 0:
        x = random.randint(0, 8)
        y = random.randint(0, 8)
        if fin_grid[x][y] != 0:
            fin_grid[x][y] = 0
            squares_to_remove -= 1
    print(np.matrix(fin_grid))


def generate():
    global grid, happened, fin_grid
    if happened:
        return
    for y in range(9):
        for x in range(9):
            if grid[y][x] == 0:
                for n in range(1, 10):
                    if possible(y, x, n):
                        grid[y][x] = n
                        generate()
                        grid[y][x] = 0
                return

    if not happened:
        fin_grid = list(map(list, grid))
        happened = True


def possible(y, x, n):
    global grid

    # check row
    for i in range(0, 9):
        if grid[y][i] == n:
            return False
    # check column
    for i in range(0, 9):
        if grid[i][x] == n:
            return False
    # check square
    x0 = (x // 3) * 3
    y0 = (y // 3) * 3
    for i in range(0, 3):
        for j in range(0, 3):
            if grid[y0 + i][x0 + j] == n:
                return False
    return True


if __name__ == '__main__':
    difficulty = input("Enter difficulty 0-2: ")
    make_sudoku(int(difficulty))
```

```
Import random to create random numbers and shuffle function
Import numpy to print 2d arrays

Create 2 lists grid and fin_grid (holds changing sudoku puzzle and final sudoku puzzle)
Create happened variable (A flag to tell if sudoku puzzle was generated

Define a function called make_suoku with parameter of difficulty
    Fetch global variables grid, fin_grid, and happened
    Create a list with numbers from 1-9
    Set happened to false (If you were to set it on an endless loop, this would need to be reset)
    Fill grid with 2d array of 9x9 that are all zeros
    Shuffle around values list
    Put in shuffled values
    For loop up to 9
        Add values list to grid
    Call generate function
    Remove squares based on difficulty
    While there are squares left
        Create random x and y coordinates
        Turn that value to zero if a number is there
    Print finished grid

Define generate
    Call global variables grid, happened and fin_grid
    If happened is true
        Return
    For x and y coordinates up to 9
        If grid value is zero
            Try 1 to 9 values
                If possible
                    Add to grid
                    Call generate
                    Failed remove addition
            Return
    If happened is false
        Set fin_grid to current grid
        Set happened to true

Define possible with parameters y, x and n
    Fetch global variable grid
    Check if column contains n already
    Check if row contains n already
    Check if current square contains n already
    If all checks passed return True

Enter difficulty
Call make_sudoku with difficulty
```

Extra:
For extra stuff look here:
https://github.com/DownRamp/Game/blob/master/sudoku.py

****

THIS IS THE IMPORTANT PART PLEASE DON'T SKIP

****

Next steps:

- Visual
- 5 x sudoku