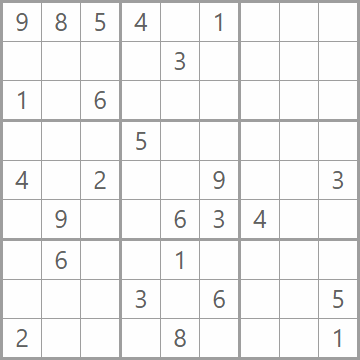
BUDOWA:

Przykład:



Kolumna:

Przykład: Kolumna 1



Wiersz:

Przykład: Wiersz 1



Sektor:

Przykład: Sektor 1/A

Obliczanie Sektora : (Int wiersz / 3) \* 3 +(Int kolumna /3)

Obraz zawierający zrzut ekranu, kwadrat, linia, Prostokąt

Opis wygenerowany automatycznie

Obraz zawierający tekst, numer, zrzut ekranu, kwadrat

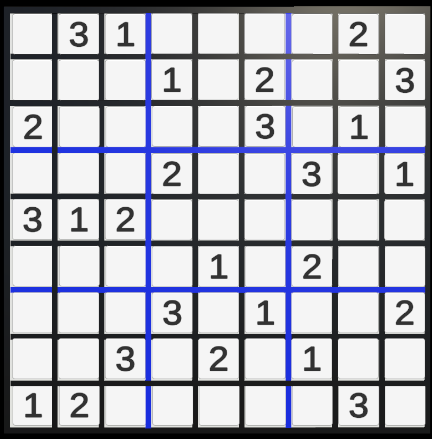
Opis wygenerowany automatycznie

~~-TABLICA NA START BĘDZIE WYPEŁNIONA ZERAMI, W KOLEJNYM ALGORYTMIE LOSOWO WYBIERA LICZBY I PRZYPISUJE WARTOŚĆ PILNUJĄC ZGODNOŚĆ Z KOLUMNĄ, WIERSZEM I SEKTOREM~~

Tablica generuje się strasznie długo i pozostawia mnóstwo pustych pól z uwagi na niezgodność z zasadami(10% tablicy jest puste)

~~-TABLICA NA START WYPEŁNIONA ZERAMI, KOLEJNY ALGORYTM WYPISUJE LICZBY OD 1 DO 9 W LOSOWYM WIERSZU I KOLUMNIE~~

Tablica generuje się krócej niż 5 sekund, ale dochodzi do ściany, gdy nie możemy zapełnić reszty z uwagi na niezgodność z zasadami gry



Backtracking?

Znalezienie konkretnej działającej planszy i mieszanie jej, co da 836,075,520 różnych wariantów?

<https://www.sudokuoftheday.com/creation>

-LICZBY BĘDĄ WYBIERANE MYSZKĄ Z UI, CO PRZYPISUJE PLAYERPREFS, BY UNITY ROZRÓŻNIAŁ NASZE INTENCJE

-PLANSZA BĘDZIE 2 STRONNA – NUMERY SĄ ZNANE DLA GRY (Tablica tablic typu int), ALE ZAKRYTE DLA GRACZA, DO MOMENTU PRZYPISANIA PRZEZ NIEGO ODPOWIEDZI(Tablica guzików Button)

NUMER NA PLANSZY NA CZARNO, ZNACZY ŻE GRACZ MIAŁ RACJĘ, NUMER I PODŚWIETLENIE NA CZERWONO, ZNACZY ŻE GRACZ SIĘ POMYLIŁ PRZY WPISYWANIU LICZBY

-GRACZ MA PRAWO DO 3 POMYŁEK, PO CZYM GRA SIĘ RESETUJE, BY UNIEMOŻLIWIĆ „STRZELANIE NA ŚLEPO”

-AI BĘDZIE ZBIERAŁO PUNKTY ZA PODANIE DOBREJ ODPOWIEDZI I DUŻY BONUS ZA UKOŃCZENIE PLANSZY, GDY PODA BŁĘDNĄ ODPOWIEDŹ TRACI CZĘŚĆ PUNKTÓW

CO DO SAMEGO BADANIA:

REINFORCEMENT LEARNING – przez doświadczenie agent uczy się podejmowania decyzji otrzymując kary i nagrody

Fajnie byłoby na końcu zrobić porównanie jak szybko AI rozwiąże zagadkę przy 1szej udanej próbie, 1000 udanej próbie i ostatecznej wytrenowanej wersji

Według mnie projekt jest w pełni możliwy do wykonania, chociaż metody bazujące bardziej na patternach i wykrywaniu konkretnych stanów typu sieci neuronowe byłyby znacznie lepsze, szczególnie na trudniejszych planszach, gdzie nie można polegać tylko na „wykluczaniu opcji”

Według projektów jakie wyszukałem w internecie, MLAgent radzi sobie dobrze z trudniejszymi łamigłówkami typu Mahjong.

Linki które mogą się przydać:

<https://norvig.com/sudoku.html>

<https://github.com/dimitri/sudoku>

<https://www.sudoku-solutions.com>

https://www.sudokuwiki.org/sudoku.html

PROBLEMY:

-LITERATURA

CIĘŻKO ZNALEŹĆ COŚ NAUKOWEGO OPARTEGO NA MATEMATYCE, A NIE NA WIEDZY POWSZECHNEJ.

-PLANSZA

ZROBIENIE W UI MOŻE BYĆ PROBLEMATYCZNE, ZROBIENIE W PRZESTRZENI 3D MOŻE POWODOWAĆ BŁĘDY Z KAMERĄ,CIENIOWANIEm, SKALOWANIEM I UMIESZCZANIEM ODPOWIEDZI W ODPOWIEDNICH POLACH + JAK „GUZIKI” WTEDY UMIEŚCIĆ

WYBÓR ZŁEJ OPCJI BĘDZIE SIĘ ODBIJAŁ NA MNIE DO KOŃCA PRACY

-ROZWIĄZANIE ŁAMIGŁÓWKI

JEST SZANSA, ŻE SUDOKU BĘDZIE NIEMOŻLIWE DLA ZWYKŁEGO UŻYTKOWNIKA UŻYWAJĄC BAZOWYCH METOD, A NAUCZENIE AI SKOMPLIKOWANYCH STRATEGII MOŻE POWODOWAĆ ZBĘDNE WYDŁUŻANIE TRENINGU ORAZ ZMIANĘ METOD UCZENIA.

ZOSTAWIAJĄC KILKA UZUPEŁNIONYCH PÓL LOSOWO, NASZA PLANSZA SUDOKU BĘDZIE NARAŻONA NA 2 DUŻE NIEDOGODNOŚCI:

-kilka możliwych rozwiązań do 1 planszy(zaawansowane łamigłówki posiadają 1 rozwiązanie)

-brak jednoznacznej odpowiedzi(mogą pojawić się takie sytuacje, gdzie do 1 pola będą pasowały 2 lub więcej odpowiedzi na raz w danym momencie,bez natychmiastowej informacji czy ruch jest dobry czy zły, problem może wyjść pod sam koniec planszy)

-ALGORYTM

POSZUKUJĘ MATEMATYCZNEGO ALGORYTMU NA TWORZENIE PLANSZY, ZAMIAST SKUPIAĆ SIĘ NA UZUPEŁNIENIU PLANSZY I USUWANIE NADMIARU LICZB NA POTRZEBY PLANSZY DLA GRACZA