4W5D

▼ Судоку

▼ 0 Release

Ветки в git:

1 createLoc, convertStringBoard and equalBoards
2 copyBoard, checkHorizontal and checkVertical
3 checkSquare

Функции:

* Создать location(createLoc)

1. Преобразовываем строку в массив и массивов.
    - (ConvertStringBoard)Функция которая преобразовывает строку в массив (9×9)
2. Написать функцию которая решает судоку с очевидными решениями (trySolveWithSingleMatch)
    - CopyBoard копируем нашу доску изначальную (oldBoard)
    - checkHorizontal
    - checkVertical
    - checkSquare
    - equalBoards - Когда функция ничего не может изменить, возвращает тру или фолс

```
// Пункт 1 ConvertStringBoard - примерный код
const string = "1-58-2----9--764-52--4--819-19--73-6762-83-9-----61-5---76---3-43--2-5-16--3-89---"

console.log(string.match(/.{9}/g).map((el) => el.split('')))
```

```
// Пункт 2 CopyBoard (oldBoard - старая доска )

const string = "1-58-2----9--764-52--4--819-19--73-6762-83-9-----61-5---76---3-43--2-5-16--3-89---"


const board = string.match(/.{9}/g).map((el) => el.split(''))


const copyBoard = JSON.parse(JSON.stringify(board))

copyBoard[0][0] = '9'

console.table(board)
console.table(copyBoard)
```

```
// Пункт 3 equalBoards (сравнивает старую и новую доску)
const string = "1-58-2----9--764-52--4--819-19--73-6762-83-9-----61-5---76---3-43--2-5-16--3-89---"


const board = string.match(/.{9}/g).map((el) => el.split(''))
```

```
const copyBoard = JSON.parse(JSON.stringify(board))

copyBoard[0][0] = '9'

console.log(JSON.stringify(board) == JSON.stringify(copyBoard) )
```

Псевдокод(Flow)

```
board = convertStringToBoard(stringBoard);

trySolveWithSingleMatch(board);

[//TODO](//todo) check sudoku in tests

trySolveWithSingleMatch(board)

{

oldBoard = copyBoard(board)

for(row = 0; row < board.length; row += 1 )

{

if(emptyElement(board,createLoc(row,col))) // return board[row][col] == '-'

{

for(col = 0; col < board.length; col += 1)

{

   let arr = ['1','2','3','4','5','6','7','8','9'];

   checkHorizontal(arr,board, createLoc(row,col));

   checkVertical(arr,board, createLoc(row,col);

   checkSquare(arr,board, createLoc(row,col);

  if(arr.length == 1)

  {

board[row][col] = arr[0];

  }

}

}

}

if(!equalBoards(oldBoard,board))

{

  trySolveWithSingleMatch(board);

}

}
```