function SudokuCreate(maxNum){ // maxNum=9

let numSet = [];

let sudokuArray = [];

for (let i=1 ; i <= maxNum ; i++){

numSet.push(i); // numSet = [1,2,3,4,5,6,7,8,9]

sudokuArray.push(new Array(maxNum)); // sudokuArray = [Array(9), Array(9), Array(9), Array(9), Array(9), Array(9), Array(9), Array(9), Array(9)]

}

הפונקציה מתחילה לייצר את התשתית ההתחלתית של הסודוקו.

בעצם הפונקציה מתחילה ליצור מערך של 9 ספרות (numSet = [1,2,3,4,5,6,7,8,9]) בנוסף מייצרת מערך שמכיל 9 תת-מערכים(sudokuArray = [Array(9), Array(9), Array(9), Array(9), Array(9), Array(9), Array(9), Array(9), Array(9)]), כל תת-מערך מכיל 9 תאים, (התאים ריקים בהתחלה)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

let horizontalBoxSize = 3; מגדירים שגודל תת-קבוצה שווה ל-3 תאים עבור כל שורה אופקית//

let verticalBoxSize = 3; מגדירים שגודל תת-קבוצה שווה ל-3 תאים עבור כל עמודה//

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function getRandomInt(max){ max=9

return Math.floor(Math.random()\*max);

הפונקציה מייצר מספרים באופן רנדומלי מ - 0 עד 9.

חשוב להבין איך מתקבל ש max שווה ל 9... הסבר בהמשך!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function placeNumber(num, arr){

let lastRowIndex = arr.length - 1;

let lastRow = arr[lastRowIndex];

let rowToCheck = lastRowIndex % verticalBoxSize;

let safeIndexes = [];

let randomSafeIndex;

הפונקציה placeNumber(num, arr) נעזרת במספר משתנים כדי לייצר את החוקיות של הסודוק...

הפרמטר num שווה ל- numSet[i] (זוכרים שהוגדר בהתחלה: numSet = [1,2,3,4,5,6,7,8,9]) , כך ש:

numSet[8]=9

numSet[7]=8

numSet[6]=7

numSet[5]=6

numSet[4]=5

numSet[3]=4

numSet[2]=3

numSet[1]=2

numSet[0]=1

למה num שווה ל- numSet[i] , והסדר הינו הפוך?

(נקפוץ לרגע אחת לסוף הקוד) בגלל שבלולאה האחרונה:

for (let i=numSet.length - 1 ; i >=0 ; i--){ // numSet.length - 1 = 9 - 1 = 8

let workingArray = [];

let possible = true;

while (sudokuArry.length > 0){

workingArray.push(sudokuArray.shift());

possible = placeNumber(numSet[i] , workingArray);

. . . . . . . . . . . .

ואז כפי שאתם מבחינים הלולאה רצה בסדר הפוך מהמספר 8 עד 0 כולל.

**מזכיר לכם שאנחנו עדיין נמצאים עם הפונקציה Function placeNumber(num, arr)... במקרה שהלכתם לאיבוד ☺**

הפרמטר arr שווה ל- workingArray, למה?

גם כן, מופע בסוף הקוד בלולאה האחרונה כאשר:

workingArray=[] >> workingArray.push(sudokuArray.shift());

בעצם ה- arr שווה ל- מערך של 9 תאים (מה שהוגדר בהתחלה)... Array[9]

לאחר שהבנו את ההשוואה של num ו- arr , נחזור לשאר הפרמטרים בפונקציה:

Function placeNumber(num, arr)

let lastRowIndex = arr.length - 1; // =workingArray[]-1 = 8

let lastRow = arr[lastRowIndex]; // = arr[8] = 9 … arr[7]=8 … arr[6]=7…… so on….

let rowToCheck = lastRowIndex % verticalBoxSize; // 8%3=2, 7%3=1 , 6%3=0, 5%3=2……. הסבר מפורט בהמשך

let safeIndexes = [];

let randomSafeIndex;

let rowToCheck = lastRowIndex % verticalBoxSize; // 8%3=2, 7%3=1 , 6%3=0, 5%3=2……. הסבר מפורט בהמשך

בעצם מייצרים 9 קופסאות על ידי המודולו (%) של האינדקסים של השורות והעמודות ב-3:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|  |  |  |  |  |  |  |  | 7 |
|  |  |  |  |  |  |  |  | 6 |
|  |  |  |  |  |  |  |  | 5 |
|  |  |  |  |  |  |  |  | 4 |
|  |  |  |  |  |  |  |  | 3 |
|  |  |  |  |  |  |  |  | 2 |
|  |  |  |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  | 0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 |
|  |  | 1 |  |  | 1 |  |  | 1 |
|  |  | 0 |  |  | 0 |  |  | 0 |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

אם הבנתם... מכאן הדרך פשוטה....

**המשך הסבר יישלח בשעות הקרובות! (לוקח זמן לכתוב....) ☺**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function findSafeIndex(boxesUsed){ // = findSafeIndex(horizontalBoxesUsed)

function boxSafe(index){ // = boxSafe(indexInlastRow)

let indexBox = Math.floor(index / horizontalBoxSize);

if (boxesUsed.indexOf(indexBox) >= 0) {

return false;

} else {

Return true;

}

}

//rowLen = lastRow.length=9

For(let indexInlastRow = 0, rowLen = lastRow.length ; indexInlastRow < rowLen ; indexInlastRow++){

Let columSafe = true; //מניחים שהעמודה בטוחה לשימוש...

For(let rowIndex = arr.length - 1 ; rowIndex >= 0 ; rowIndex--){ // arr.length - 1 = 8 , rowIndex = 8,7,6….1,0

If(arr[rowIndex][indexInLastRow] == num) { // if(arr[8][0] == numSet[i]

columSafe = false;

}

}

הלולאה בודקת:

אם arr[8][0] שווה ל num ( ז"א numSet[i] , זוכרים שהוגדר בהתחלה והוא מייצג את המספרים מ 1 עד 9 במערך - במקרה שלנו ה ה i מייצג את האינדקס של המספרים (1,2,3,4,5,6,7,8,9) במערך numSet[i]

במערך numSet שמתחיל מ 8 עד 0 (הלולאה הסופית בקוד - הוסבר בהתחלה)

במקרה וקיים מספר כלשהו מ 1 עד 9 בתא שמספרו arr[8][0] הרי אין טעם להציב מספר בתא, אחרת אם התא ריק מצבים מספר רנדומלי שיגע בהמשך.... וכך עבור התא הבא בתור: arr[8][1]….. arr[8][2]……

**בנוסף, הלולאה בודקת שהמספר לא חוזרת פעמיים!!!!**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| arr[0][8] | arr[0][7] | arr[0][6] | arr[0][5] | arr[0][4] | arr[0][3] | arr[0][2] | arr[0][1] | arr[0][0] |
|  |  |  |  |  |  |  |  | . |
|  |  |  |  |  |  |  |  | . |
|  |  |  |  |  |  |  |  | . |
|  |  |  |  |  |  |  |  | . |
|  |  |  |  |  |  |  |  | . |
| arr[6][8] | arr[6][7] | arr[6][6] | arr[6][5] | arr[6][4] | arr[6][3] | arr[6][2] | arr[6][1] | arr[6][0] |
| arr[7][8] | arr[7][7] | arr[7][6] | arr[7][5] | arr[7][4] | arr[7][3] | arr[7][2] | arr[7][1] | arr[7][0] |
| arr[8][8] | arr[8][7] | arr[8][6] | arr[8][5] | arr[8][4] | arr[8][3] | arr[8][2] | arr[8][1] | arr[8][0] |

צריך להבטח באמת שהתא ריק ולכן הלולאה הבאה פותרת את הבעיה או במילים אחרות מבטיחה לנו שהתא ריק והוא מוכן להכיל מספר בתוכו.... פשוט ☺:

If (lastRow[indexInLastRow] == undefined && columSafe && boxSafe(indexInLastRow)){

safeIndexes.push(indexInLastRow);

1. lastRow[indexInLastRow] ז"א ה- lastRow = arr[lastRowIndex] או כל מערך מ 0 עד 8

2. boxSafe(indexInLastRow), נחזור לפונקציה:

function boxSafe(index){ // = boxSafe(indexInlastRow)

let indexBox = Math.floor(index / horizontalBoxSize); //Math.floor(index = indexInlastRow = 0,1,2,3,4,…..8)/3) >>>> 0,0,1,1,1,2,2,2

if (boxesUsed.indexOf(indexBox) >= 0) { //

return false;

} else {

Return true;

בגדול הפונקציה בודקת אם נעשה שימוש באינדקסים של תתי הקבוצה אם כן לא נוגעים באינדקס לכן זה false אחרת זה true וצריך לנצל את אותם אינדקסים כדי להציב בהם מספרים...

הלולאה חשובה מאד - אחרת נקבל כל הזמן תאים ריקים לכן יש לאבטח שבכל תא\שורה\עמודה\תת קבוצה אין תאים ריקים או מנוצלים...

3. safeIndexes.push(indexInLastRow) >>> הגדרנו לפני כן את safeIndexes כמערך... שימו לב שהפעולה מייצרת את המערך הבא:

**safeIndexes**.push(indexInLastRow) >>> indexInLastRow=0,1,2,3,4,5,6,7,8 >>> **safeIndexes** = [0,1,2,3,4,5,6,7,8]

תראו איזה יפה זה... אורך המערך של safeIndexes שנוצר שווה ל- 9..... היופי כאן שאנו מנצלים של המספרים במערך כאנדקסים בהמשך ואת אורך המערך שהוא 9 מנצלים בפונקציה הבאה:

Return safeIndexes[getRandomInt(**safeIndexes**.length)]; // safeIndexes.length = 9

ופה נכנסת הפונקציה לייצור מספר רנדומלי - מזכיר לכם אותה: function getRandomInt(max)

כאשר ה max שווה ל- 9!!!!

למה כל זה נעשה בצורה מורכבת משום שרצינו ליצור סודוקו דינאמי מההתחלה... היה חשוב להביא קוד אחד שיעבוד על כל גודל של מטריצה אך הבעייתיות היא בהגדרת התת-קבוצה שככל הנראה עבור מטריצה 16X16 ישנם חוקים אחרים ולא מקבילים לאותם חוקים עבור סודוקו של 9X9...

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**לא נשאר הרבה חברים... אחרי שהבנו את כל הבלגן הדרך מכאן תהיה ממש קלה ופשוטה (מקווה שמבינים אותי למה לא הצלחתי להסביר בזום)**

**המשך ההסבר במהל היום...**

**בינתיים לילה טוב...**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Let horizontalBoxesUsed = [];

If (rowsToCheck > 0) {

For (let i = rowsToCheck ; i > 0 ; i--) {

Let horizontalBoxes = Math.floor(arr[lastRowIndex - 1].indexOf(num) / horizontalBoxSize);

horizontalBoxesUsed .push(horizontalBoxes );

}

}

Let horizontalBoxes = Math.floor(arr[lastRowIndex - 1].indexOf(num) / horizontalBoxSize) = Math.floor(arr[8 - 1].indexOf(numSet[i]) / 3) =

Let horizontalBoxes = Math.floor(arr[7].indexOf(numSet[8]) / 3 ) = Math.floor(9 / 3) = 3…3…2…2…2…1…1…1

horizontalBoxesUsed .push(horizontalBoxes)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **3** | | | 2 | | | 1 | | |
| **3** | | | 2 | | | 1 | | |
| 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 |
|  |  | 2 |  |  | 2 |  |  | 2 |
|  |  | 3 |  |  | 3 |  |  | 3 |

מנח שהקוד יותר ברור ברור עכשיו... וגם לגבי השורות הבאות:

randomSafeIndex = findSafeIndex(horizontalBoxesUsed);

if (randomSafeIndex == undefined){

return num;

} else {

lastRow[randomSafeIndex] = num;

return true;

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

נעבור ללולאה האחרונה....

for (let i=numSet.length - 1 ; i >=0 ; i--){ // numSet.length - 1 = 9 - 1 = 8

let workingArray = [];

let possible = true;

while (sudokuArry.length > 0){

workingArray.push(sudokuArray.shift()); // הסבר למטה

possible = placeNumber(numSet[i] , workingArray);// השורה מפעילה את הקוד

. . . . . . . . . . . .

workingArray.push(sudokuArray.shift()); //

השורה שולפת מערך שלם מ- sudokuArray ודוחפת אותו למטריצה חדשה workingArray -

הלולאה עוצרת ברגע ש i=0 כוונה כאשר עוברים על כל המספרים 1 עד 9 במערך שנוצר בתחילת הקוד (numSet)...

זה כל הקוד - תשלחו ואשמח לענות לכם...