1. ЇЖАЧОК ТЕСТ (C#)

В лісі живе популяція їжаків. Їжачки тут можуть бути лише одного з трьох кольорів - червоного, зеленого та синього. Коли зустрічаються два їжачки різних кольорів, вони можуть змінити свій колір на третій (тобто, коли зустрічаються червоний і синій їжачок, вони обидва можуть стати зеленими). Іншого способу змінити свій колір у їжаків немає (зокрема, коли зустрічаються червоний і синій їжачок, вони не можуть стати обоє червоними, можна припустити лише третій колір).
Їжачки хочуть стати одного певного кольору. Вони можуть планувати свої зустрічі, щоб досягти цієї мети. Їжачки хочуть знати, як швидко можна досягти своєї мети (якщо її взагалі можна досягнути).

Вхідні дані:
Колір задано цілим числом, 0 - червоний, 1 - зелений, 2 - синій. Початкова популяція їжаків задана у вигляді масиву з трьох цілих чисел з індексом, що відповідає кольору (тобто [8, 1, 9] означає 8 червоних, 1 зелених і 9 синіх їжачка). Всі числа невід'ємні, їх сума знаходиться між 1 та int.MaxValue (максимально можливе значення для типу int, іншими мовами). Бажаний колір задається цілим числом від 0 до 2.

Виведіть:
Код повинен повернути мінімальну кількість зустрічей, необхідних для зміни всіх їжачків у заданий колір, або -1, якщо це неможливо (наприклад, якщо всі їжачки спочатку одного кольору).

2. SUDOKU TEST (JS)

Судоку - це гра, в яку грають на сітці 9x9. Мета гри - заповнити всі клітинки сітки цифрами від 1 до 9 так, щоб кожен стовпчик, кожен рядок і кожна з дев'яти підсіток 3x3 (також відомих як блоки) містили всі цифри від 1 до 9.

(Більше інформації за посиланням: http://en.wikipedia.org/wiki/Sudoku)

Валідатор розв'язків Судоку

Напишіть функцію validSolution(), яка отримує двовимірний масив, що представляє дошку Судоку, і повертає true, якщо це правильний розв'язок, або false у протилежному випадку. Клітинки дошки судоку також можуть містити 0, які означають порожні клітинки. Дошки, що містять один або більше нулів, вважаються невірними розв'язками.

Дошка завжди має розмір 9 клітинок на 9 клітинок, і кожна клітинка містить лише цілі числа від 0 до 9.

```
validSolution([
  [5, 3, 4, 6, 7, 8, 9, 1, 2],
  [6, 7, 2, 1, 9, 5, 3, 4, 8],
  [1, 9, 8, 3, 4, 2, 5, 6, 7],
  [8, 5, 9, 7, 6, 1, 4, 2, 3],
  [4, 2, 6, 8, 5, 3, 7, 9, 1],
  [7, 1, 3, 9, 2, 4, 8, 5, 6],
  [9, 6, 1, 5, 3, 7, 2, 8, 4],
  [2, 8, 7, 4, 1, 9, 6, 3, 5],
  [3, 4, 5, 2, 8, 6, 1, 7, 9]
]); // => true

 validSolution([
  [5, 3, 4, 6, 7, 8, 9, 1, 2],
  [6, 7, 2, 1, 9, 0, 3, 4, 8],
  [1, 0, 0, 3, 4, 2, 5, 6, 0],
  [8, 5, 9, 7, 6, 1, 0, 2, 0],
  [4, 2, 6, 8, 5, 3, 7, 9, 1],
  [7, 1, 3, 9, 2, 4, 8, 5, 6],
  [9, 0, 1, 5, 3, 7, 2, 1, 4],
  [2, 8, 7, 4, 1, 9, 6, 3, 5],
  [3, 0, 0, 4, 8, 1, 1, 7, 9]
]); // => false
```