⌛⌛ Sudoku

There is material for this question in the remote material repository:

- The material is located at the `round4/sudoku` directory.
- `SudokuTest.java`: a test program to test your implementation.
- `input.txt`: a test input for the program.
- `output.txt`: an example output matching `input.txt`.

Exercise: Sudoku

Place your code into a file named **Sudoku.java**.

Implement `Sudoku` class that maintains a Sudoku grid and can, for example, check if it is legal. If you have somehow managed to avoid knowing what Sudoku is, see first its Wikipedia article.

To be more precise, you should implement the `Sudoku` class to have public members given below. All the other members should be private.

- A constructor `Sudoku()` that initializes the Sudoku object to hold a 9 × 9 Sudoku grid whose all cells are empty. Empty cells are expressed using space characters `' '`.
  - You are free to decide the internal (private) details of how you store the grid.
- Member function `set(int i, int j, char c)`: sets the charecter `c` into the grid cell (`i`, `j`). By this we mean the cell in column `j` of row `i`.
  - Both the rows and columns have index values from 0 to 8. The function must check that `i` and `j` are legal row and column indices. If this is not the case, the function prints a message of form "`Trying to access illegal cell (i, j)!`".
  - A Sudoku grid is allowed to hold only space characters `' '` (empty cells) and the digit characters `'1'`, `'2'`, `'3'`, `'4'`, `'5'`, `'6'`, `'7'`, `'8'` and `'9'`. If the value pf the parameter `c` is some other character, it will not be set and a message of form "`Trying to set illegal character c to (i, j)!`" should be printed.
- Member function `check()`: checks if the current values in the Sudoku grid are legal. Returns `true` (is legal) or `false` (is not legal).
  - A Sudoku grid is legal if no row, column or 3 x 3 sub-block contains a digit multiple times (multiple spaces may naturally occur).
    - Check first the rows, then the columns and finally the sub-blocks (inspect the sub-blocks in a row-wise manner, starting with the upper left corner sub-block).
    - Record the smallest recurring digit from a row, column or sub-block. This ensures that the messages described below will be unique even if there were several recurring digits. When the smallest recurring digit has been found, print one of the messages desribed below and return immediately from the function with the `false` value.
      - Row: "`Row i has multiple c's!`", where `i` is the row index.
      - Column: "`Column j has multiple c's!`", where `j` is the column index.

    - Sub-block:
      "`Block at (x, y) has multiple c's!`",
      where `x` and `y` are the row and column index of the upper left corner cell of the sub-block.
- Member function `print()`: prints the Sudoku grid. The grid borders are expressed using the characters `'#'`, `'-'`, `'|'` and `'+'` and each cell value has a space on its both sides. See further details from the example output file.

Testing

You may test your class by using the test program given in the file `SudokuTest.java`, the test data given in the file `input.txt`, and the example output given in the file `output.txt`. Place these files and your own class implementation into the same directory, compilte the program e.g. as `javac *.java`, and run the test as `java SudokuTest input.txt`. The program should produce exactly the same output as shown in the file `output.txt`.