Sudoku Ödevi

Amaç: Bu ödevde, aşağıda verilen sudoku bulmacasını çözen bir Java programı geliştirmeniz beklenmektedir. Program, verilen Sudoku bulmacasını çözmeli ve sonucu ekrana yazdırmalıdır. Bulmacada 0 olarak verilen yerler boş olarak kabul edilecektir.

```
+-----------------+
|1 9 5|7 3 8|4 6 2|
|2 6 8|4 0 9|5 7 3|
|3 7 4|5 2 6|9 1 0|
+-----------------+
|7 5 1|3 8 2|0 4 9|
|4 0 9|6 5 1|2 8 7|
|8 2 6|9 0 7|3 5 1|
+-----------------+
|5 8 2|1 9 4|7 0 6|
|9 0 7|8 6 3|1 2 5|
|6 1 3|2 0 5|8 9 4|
+-----------------
```

Adımlar:

1. **Proje Dizisi Oluşturma**
    - Yeni bir dizin oluşturun ve bu dizini tercih ettiğiniz bir Java geliştirme ortamında (örneğin, Eclipse veya NetBeans) açın.
2. **SudokuGrid.java ve SudokuSolver.java Dosyalarını oluşturma**
    - SudokuGrid.java ve SudokuSolver.java dosyalarını oluşturun ve projenizin dizinine ekleyin.
3. **SudokuGrid Sınıfını Tamamlama**
    - SudokuGrid.java dosyasını açın ve aşağıdaki görevleri tamamlayın:
        - Izgara boyutu ve rakam aralığı için sabitler (örneğin, SIZE ve DIGIT RANGE) ekleyin.
        - Bir SudokuGrid nesnesinin bir kopyasını oluşturmak için SudokuGrid copy() adında bir yöntem ekleyin.
        - Bir sonraki boş hücreyi bulmak için java.awt.Point findEmptyCell() adında bir yöntem ekleyin. Bu yöntem, sırasıyla soldan sağa ve yukarıdan aşağıya doğru okuma sırasında bir sonraki boş hücreyi

bulmaya çalışmalıdır. Bulunduğunda, hücrenin koordinatlarını bir dizi olarak ({satır, sütun}) döndürün; aksi takdirde, {-1, -1} dizisi olarak döndürün.

- Izgarayı yazdırmak için print() yöntemini kullanmayın.
- Hücreleri doldurmak için fillCell(int r, int c) adında bir yöntem ekleyin.
- Bir kareye rakam d doldurmanın bir çakışma olup olmadığını belirlemek için boolean givesConflict(int r, int c, int d) adında bir yöntem ekleyin. Bu yöntemi daha fazla detaylı olarak üç yönteme bölmelisiniz: rowConflict, colConflict ve boxConflict.
- Gerekli kurucuları oluşturun.

4. SudokuSolver Sınıfını Tamamlama

- SudokuSolver.java dosyasını açın ve aşağıdaki görevleri tamamlayın:
  - SudokuGrid grid adında bir SudokuGrid nesnesi ve grid'in çözülebilir olup olmadığını kontrol eden bir boolean solve() yöntemi ekleyin.
  - solve(SudokuGrid grid) yönteminde, bir çözüm bulmak için özyinelemeli bir strateji kullanın:
    - Eğer özyinelemeli çağrı true döndürürse, bir çözümünüz vardır ve true döndürün.
    - Aksi takdirde, doldurulanları geri alın ve bir sonraki rakamla devam edin.
  - Sudoku bulmacalarının en fazla bir çözümü olduğundan emin olun.

5. Sudoku Bulmacayı Çözme

- Verilen SudokuGrid.java ve SudokuSolver.java şablonlarını kullanarak Sudoku bulmacayı çözen bir program geliştirin.
- Programınızı çalıştırın ve verilen Sudoku örneğini çözmesini sağlayın.
- Sonucu ekrana yazdırın.

6. İlerleme ve Test

- Programınızın doğru çalıştığından emin olmak için farklı Sudoku bulmacalarını test edin.
- Programınızın tüm gereksinimleri karşıladığından emin olun.

7. Proje Sunumu ve Gönderimi

- Ödevinizi kaynak kodlarıyla birlikte laboratuvar dersinde kontrol için hazır hale getirin.