Sudoku[1]

Sudoku, por vezes escrito Su Doku, (em japonês: 数独, sūdoku) é um jogo baseado na colocação lógica de números. O objetivo do jogo é a colocar de números de 1 a 9 em cada uma das posições vazias numa grade quadrada de 9x9, constituída por subgrades 3x3 denominadas regiões, como apresentado na figura a seguir:

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

O quebra-cabeça contém algumas pistas iniciais, ou seja, os números já dispostos em algumas posições. A partir dos números inseridos é possível deduzir os números que deveriam estar nas posições vazias, obedecendo a seguinte regra: cada coluna, linha e região só pode ter um número de 1 à 9. Resolver o problema requer apenas raciocínio lógico e algum tempo. A solução para o problema acima é:

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

[1] **Importante**: A especificação desse trabalho pode sofrer modificações de acordo com discussões que tivermos em sala de aula.

Objetivo

O objetivo deste trabalho é implementar o jogo Sudoku em modo texto, o seu programa irá ajudar o usuário a achar a solução do quebra-cabeça. Inicialmente o programa fará a leitura da grade disposta em um arquivo texto e armazenando em uma matriz em memória.

O jogo deve ser implementado baseado nas rotinas do **Jogo da Velha** visto em sala. Assim o seu programa deverá ter pelo menos as funções a seguir**: game()**, **initialize()**, **print(), step()** e **status()**, caso precise você escrever outras funções.

- **game()**: Essa função irá executar a lógica do jogo, a função **game()** chama todos as funções descritas a seguir. A cada jogada, a função deve imprimir a grade do Sudoku na tela e solicitar uma jogada pelo teclado. A jogada pode ser colocar um número em determinada posição na matriz ou limpar uma posição, pois dependendo do estado em que se encontra o jogo, o jogador deverá rever algumas jogadas que o deixaram sem possibilidade de fazer jogadas válidas.

- **initialize()**: A função **initialize()** deverá fazer a leitura da grade armazenada em um arquivo texto e devolver uma matriz 9x9 já com os valores iniciais. Para que o jogo fique desafiador tente criar arquivos com configurações com grau de dificuldade diferentes. Abaixo segue um exemplo de arquivo de entrada as posições estão separadas por espaço em branco e a posição vazia é definida usando o caractere '_' underline.
  Exemplo de arquivo de entrada:

```
5 3 _ _ 7 _ _ _ _
6 _ _ 1 9 5 _ _ _
_ 9 8 _ _ _ 6 _
8 _ _ _ 6 _ _ _ 3
4 _ _ 8 _ 3 _ _ 1
7 _ _ _ 2 _ _ _ 6
_ 6 _ _ _ 2 8 _
_ _ _ 4 1 9 _ _ 5
_ _ _ _ 8 _ _ 7 9
```

- **print()**: Essa função imprime o grade do Sudoku na tela em modo texto, não se esqueça que a impressão deve ser o mais informativa possível, para o que jogador consiga decidir claramente onde e qual será o próximo número colocado no Sudoku.

- **step()**: Essa função recebe uma posição e um número que o usuário deseja inserir na solução do Sudoku, caso a posição esteja ocupada ou inválida (linha coluna que não existe) a função deve retornar -1, se o valor ser inserido já estiver presente na linha, coluna ou região a função retorna 0, e por fim, se for possível colocar o número na posição solicitada a função retorna 1. Ao final a matriz atualizada é retornada. Essa função também é responsável em limpar uma posição na matriz do Sudoku e retorna 1 caso tenha sucesso, caso não consiga a função retorna -1.

- **status ()**: A função **status()** verifica se o jogador já solucionou o quebra-cabeça, ou seja, não existe posições vazias na matriz. Caso o jogador tenha cumprido o objetivo do jogo a função retorna **true**, ou **false** caso contrário

Critérios de avaliação:

O programa entregue **será avaliado** de acordo com os seguintes itens:

- Funcionamento do programa;
- O programa deve estar na linguagem **Java**.
- O quão fiel é o programa quanto à descrição do enunciado;
- Comentários e legibilidade do código;
- Clareza na nomenclatura de variáveis e funções.

Grupo
A atividade pode ser feita em grupo de no **máximo dois integrantes**, para entregar basta que somente um dos integrantes submeta o trabalho no **Blackboard.** O arquivo fonte do trabalho deve ter o seguinte cabeçalho.

```
/*
Entrega de trabalho
Nós,

Nome completo 1
Nome completo 2

declaramos que

todas as respostas são fruto de nosso próprio trabalho,
não copiamos respostas de colegas externos à equipe,
não disponibilizamos nossas respostas para colegas externos à equipe e
não realizamos quaisquer outras atividades desonestas para nos beneficiar ou prejudicar
outros.
*/
```

Importante
Como este trabalho pode ser feito em grupo, evidentemente você pode “discutir”" o problema dado com outros grupos, inclusive as “dicas”" para chegar às soluções, mas você deve ser responsável pela solução final e pelo desenvolvimento do seu programa. Assim não repasse para e nem copie o programa de outro grupo. Trabalhos considerados plagiados receberão nota 0 (zero).