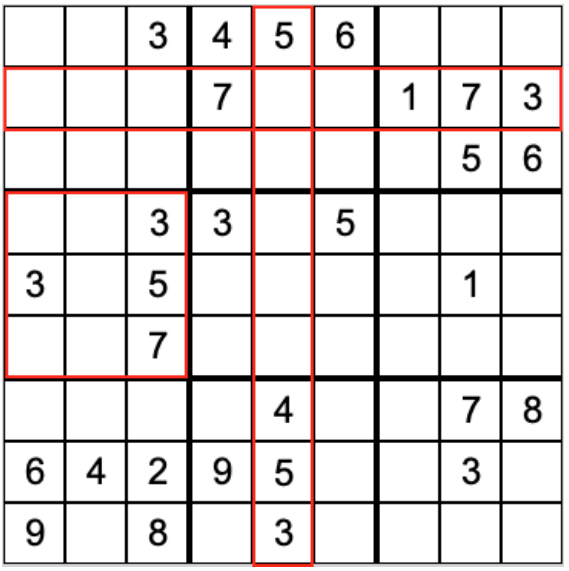
INSTITUTO UNIVERSITÁRIO DE LISBOA

Introdução à Programação | 2023/2024

Projeto Individual - Sudoku



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

- Data de entrega: até às 24h de dia 10 de Dezembro de 2023
  (por *upload* na plataforma do *moodle*)
- Data da discussão: semana de 11 a 15 de Dezembro de 2023
  (na respetiva aula teórico-prática)

Aviso: Os alunos podem partilhar e/ou trocar ideias entre si, sobre os trabalhos e/ou resolução dos mesmos. No entanto, o trabalho entregue deve corresponder ao esforço individual de cada aluno. O projeto é individual, e em nenhum caso deve ser copiado código que será entregue. A deteção de código copiado será realizada por software especializado bastante sofisticado. Casos de plágio óbvio serão penalizados com a anulação do projeto, o que implica a reprovação à Unidade Curricular (UC). Adicionalmente, a situação será reportada à Comissão Pedagógica da ISTA/Conselho Pedagógico do ISCTE-IUL. Serão penalizados da mesma forma tanto os alunos que fornecem código como os que copiam código de outros.

Introdução

O objetivo deste projeto é desenvolver o jogo “Sudoku” em Java, demonstrando a execução no ambiente *PandionJ*, usando as classes *Color* e *ColorImage* fornecidas. Nesse sentido, deverão ser desenvolvidas três classes:

- uma classe estática SudokuAux com funções e procedimentos úteis para criação e manipulação de tabuleiros e sua visualização numa imagem;
- uma classe de objetos SudokuBoard para a lógica e estado do jogo;
- uma classe de objetos Sudoku para a gestão do jogo, incluindo o carregamento e gravação de ficheiros.

A solução desenvolvida deverá fazer uso das classes *ColorImage, ImageUtil,* e *Color* disponibilizadas nas aulas praticas, e da classe *String* do Java. Não é permitida a utilização de outras classes.

Recomenda-se que o desenvolvimento do projeto decorra por etapas, as quais apresentamos em seguida. Não será apropriado avançar para uma fase posterior sem ter a(s) anterior(es) minimamente funcionais.

Parte 1: SudokuAux

Objetivo: Desenvolver uma classe estática SudokuAux com procedimentos e/ou funções para a criação, manipulação, e visualização do tabuleiro de jogo.

Esta classe deve ter funções e/ou procedimentos para:

1. Validar se uma matriz de inteiros 9x9 representa um Sudoku válido, sendo que o valor zero representa uma posição não jogada.

2. Gerar uma matriz de jogo dada uma solução completa válida (sem zeros), alterando-a de forma a que parte dos números da mesma sejam substituídos por zero. Deverá ser fornecida uma percentagem que determina qual a proporção de posições a zerar.

3. Produzir uma String com o conteúdo de uma matriz de inteiros.
   {{1,2,3},{4,5,6}} → “1 2 3
   4 5 6”

4. Produzir uma imagem a cores com o desenho do tabuleiro Sudoku a partir de uma matrix de inteiros 9x9. Poderá ser útil desenvolver procedimentos auxiliares para abordar partes do objetivo (p.e. desenho de grelha, desenho de números).

5. Dada uma imagem resultante de (4), alterar uma posição da imagem do tabuleiro, fornecendo uma coordenada e o número a colocar.

6. Dada uma imagem resultante de (4), marcar uma linha do tabuleiro com contorno vermelho (para sinalizar que está inválida), fornecendo o índice da linha.

7. Análogo a (6), mas para colunas.

8. Dada uma imagem resultante de (4), marcar o segmento com um contorno vermelho (para sinalizar que está inválido), fornecendo o índice do segmento. Um segmento corresponde a um dos nove quadrados 3x3.

Parte 2: SudokuBoard

Objetivo: Desenvolver uma classe de objetos para manipulação do jogo Sudoku. Esta classe deve ter funções e/ou procedimentos para:

1. Criar um jogo fornecendo uma matriz de inteiros 9x9 (a ser validada).

2. Obter o número que está numa coordenada, ou zero caso não esteja preenchida.

3. Efetuar uma jogada dando a coordenada e o valor. Não é permitida uma jogada que se sobreponha a um número da matriz inicial. Por outro lado, a jogada pode resultar num tabuleiro inválido.

4. Efetuar uma jogada aleatória numa das posições vazias, apenas respeitando a regra de não duplicar números num segmento.

5. Reiniciar o tabuleiro para a configuração inicial.

6. Obter os segmentos que não estão válidos para a solução do Sudoku, ou seja, que contêm números repetidos.

7. Obter as linhas/colunas que não estão válidas para a solução do Sudoku, ou seja, as que contêm números repetidos.

8. Saber se o Sudoku já está concluído.

9. Permitir anular jogadas com um mecanismo de *undo*, com memória suficiente para o número de posições iniciais vazias.

Parte 3: Sudoku

Objetivo: Desenvolver uma classe de objetos para gerir um jogo Sudoku, o seu carregamento e gravação em ficheiros. Esta classe será a que controla a execução do jogo, coordenando as alterações ao tabuleiro (SudokuBoard) e a respetiva visualização numa imagem.

Ficheiros

Existem dois tipos de ficheiros, descritos na seguinte tabela. Poderão ser utilizadas outras formas de representação, desde que seja guardada a mesma informação.

formato *.sud	**formato *.sudgame**
Jogo Sudoku completo, que será utilizado para gerar jogos diferentes aleatórios.	Estado atual de um jogo Sudoku, onde é também guardado o tabuleiro inicial.
1 2 3 4 5 6 7 8 9 4 5 6 7 8 9 1 2 3 7 8 9 1 2 3 4 5 6 2 1 4 3 6 5 8 9 7 3 6 5 8 9 7 2 1 4 8 9 7 2 1 4 3 6 5 5 3 1 6 4 2 9 7 8 6 4 2 9 7 8 5 3 1 9 7 8 5 3 1 6 4 2	1 2 3 **0 0 0** 7 8 9 4 **0 0** 7 8 9 1 2 3 7 8 9 **0 0 0** 4 5 6 2 **0** 4 3 6 5 8 9 7 3 6 5 8 9 7 2 1 4 8 **0** 7 2 1 **0 0** 6 5 5 **0 0** 6 4 2 9 7 8 6 **0 0** 9 7 **0** 5 3 **0** 9 7 8 5 3 1 6 **0 0** 1 2 3 4 **0 0** 7 8 9 4 **0 0** 7 8 9 1 2 3 7 8 9 **0 0 0** 4 5 6 2 **0** 4 3 6 5 8 9 7 3 6 5 8 9 7 2 1 4 8 **0** 7 2 1 4 3 6 5 5 **0 0** 6 4 2 9 7 8 6 **0 0** 9 7 **0** 5 3 **0** 9 7 8 5 3 1 6 4 2

Um objeto Sudoku deverá ser criado fornecendo o nome do ficheiro e uma percentagem para a dificuldade. Os objetos deverão ter dois atributos:

1. Um objeto SudokuBoard com o estado atual do jogo;
2. Uma imagem (ColorImage) que representa o “ecrã” do jogo, que vai sendo alterada à medida que são feitas as jogadas.

```
public class Sudoku {
      private SudokuBoard sudokuBoard;
      public ColorImage boardImage; // para ver no PandionJ

      public Sudoku(String file, double difficulty) {
```

Deverão existir operações para:

1. Efetuar e anular jogadas.
2. Gravar o estado do jogo para um ficheiro (*.sudgame).
3. Carregar um jogo fornecendo o nome do ficheiro previamente gravado (*.sudgame). Não é relevante se o ficheiro gravado corresponde ao Sudoku inicialmente carregado.

Execução

Para testar o jogo no PandionJ, recomenda-se a abordagem ilustrada no código seguinte.

```
public class SudokuTest {
     public static void main() {
           Sudoku sudoku = new Sudoku("sudoku1.sud", 0.5);
           return; // Colocar breakpoint e correr
     }
}
```

Avaliação

No projeto, as opções visuais/estéticas do jogo não serão valorizadas. O projeto será inicialmente avaliado em termos funcionais, i.e., se as funções e procedimentos produzem os resultados esperados e os objetos dos tipos das classes a desenvolver têm o comportamento esperado, independentemente da forma como estão implementados, de acordo com os seguintes pesos:

30%	Parte 1
40%	Parte 2
30%	Parte 3

A realização do projeto é obrigatória para obter aprovação à UC. Não haverá qualquer possibilidade de obter aprovação à UC sem realizar o projeto. A classificação no projeto define o limite máximo para a nota final na UC.

A	Muito bom (> 80%)	nota máxima 20
B	Bom (<= 80%)	nota máxima 16
C	Suficiente (<= 60%)	nota máxima 12
D	Não aprovado (< 45%)	reprovado

Desta primeira avaliação resultará uma classificação (A, B, C ou D). Em função da qualidade do código poderá ser aplicada uma penalização que implica descer um nível na classificação, p.e. classificação funcional A com má qualidade de código, é despromovida para B.

Os projetos só poderão ser entregues em suporte eletrónico ficheiro comprimido .zip (contendo somente os ficheiros .java desenvolvidos e exemplos de ficheiros com tabuleiros Sudoku ou alterados), por upload na plataforma *moodle* até à data-limite de entrega, e diretamente ao professor da turma a que o aluno pertence no dia da discussão. O projeto só será considerado entregue caso tenha sido rececionado corretamente por estas duas vias.

A falha no upload dentro do prazo ou a não apresentação/discussão do projeto, implicam a reprovação direta na UC.