INSTITUTO UNIVERSITÁRIO DE LISBOA

Introdução à Programação | 2022/2023

# Projeto Individual

- Data de entrega: até às 24h de dia 11 de Dezembro de 2022
  (por *upload* na plataforma do *moodle*)
- Data da discussão: até à semana de 12 a 16 de Dezembro de 2022
  (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 criar o jogo “Wordle” (ou “Termo”, versão em português) para ser visualizado no ambiente do PandionJ.

Nesse sentido, deverão ser desenvolvidas três classes:

- uma classe estática com funções e procedimentos uteis para criação e manipulação de imagens a cores (usando a classe dada C*olorImage*);
- uma classe de objetos *Game* para visualização/manipulação do jogo;
- uma classe de objetos *Stats* para visualização das estatísticas do jogo;

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

Recomenda-se que o desenvolvimento do projeto decorra por etapas, as quais devem corresponder às classes que se apresentam em seguida. Não será apropriado avançar para a segunda parte sem ter a primeira minimamente completa. A terceira parte pode ser desenvolvida e testada separadamente.

# Parte 1: Funções e procedimentos úteis para a criação e manipulação de imagens e lógica do jogo

**Objetivo:** Desenvolver uma classe estática com procedimentos e/ou funções para a criação e manipulação de imagens a cores (*ColorImage*). Esta classe deve ter funções e/ou procedimentos para:

(desenvolva as funções e/ou procedimentos adicionais que achar necessário)

1. desenhar numa imagem dada, um *ícone* com um tamanho fixo, dados uma posição x , y , uma cor de fundo (ou um código de estado (int) correspondente à cor), e uma letra (char).
2. desenhar numa imagem dada, uma grelha de jogo a partir de uma matriz de char [ ] [ ] e de uma *String* (palavra puzzle) dados, centrada na imagem.
3. desenhar numa imagem dada, uma representação de um teclado representado por uma matriz char [ ] [ ] e por um vector de inteiros int [ ] dados, centrada horizontalmente na parte inferior da imagem.
4. dada uma *String*, devolver uma *String* equivalente sem acentuação nem cedilhas.

# Parte 2: A classe de objetos *Game*

**Objetivo:** Desenvolver uma classe de objetos para visualização e manipulação do jogo. O objeto do tipo *Game* pode ser criado de duas formas: i) fornecendo apenas uma imagem de background (caso em que usa o dicionário por omissão fornecido "*pt_br.txt*"); ii) fornecendo apenas um objeto do tipo *Dictionary*. Esta classe deve ter entre outros atributos para guardar: a matriz de jogo e do teclado (ambos char [ ] [ ]); a estatística; a palavra puzzle; o dicionário; e um vetor de inteiros representando o estado das letras do teclado, devem ser desenvolvidas funções e/ou procedimentos para:

1. criar um objeto da classe *Dictionary*, e uma função que obtém uma palavra puzzle aleatoriamente do dicionário.
2. criar um vetor de inteiros ( int [ ] ) que guarde o estado de cada letra no teclado ao longo do jogo (e.g.: 0 – não testada; 1 – não existe; 2 – existe mas fora de posição; 3 – posição correta).
3. atualizar o estado das letras do teclado no vetor de inteiros, sendo dadas a palavra a inserir e a palavra puzzle.
4. criar uma função que insira uma palavra (*String*) numa linha da matriz de jogo.

## Parte 3: A classe de objetos *Stats*


**Objetivo:** Desenvolver uma classe de objetos que representa um registo de estatísticas do utilizador do jogo. O objeto do tipo *Stats* pode ser criado apenas fornecendo o número máximo de tentativas possíveis.

Um objeto do tipo *Stats* é composto pelos seguintes atributos: um vetor int [ ] que regista as vitórias (na respetiva linha) e inteiros que guardam: o número de jogadas, o número de vitórias, o número de vitórias consecutivas, e a maior sequência de vitórias consecutivas. Após a sua criação, deve ser possível:

1. atualizar o registo devido a vitória, dada a respetiva linha;
2. atualizar o registo devido a derrota;
3. visualizar as estatísticas devolvendo um objeto do tipo *ColorImage*;

## Exceções

No projeto, deverá lançar uma exceção de cada um dos seguintes tipos:

1. IllegalArgumentException;
2. NullPointerException;
3. IllegalStateException.

## Critérios

No projeto, as opções visuais/estéticas do jogo não serão valorizadas. Deve haver constantes para definir o tamanho da letra e do ícone, o espaçamento entre letras e o número de tentativas do puzzle.

## Avaliação e Entrega

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. O projeto será classificado da seguinte forma:

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

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% | classe estática |
| 40% | classe *Game* |
| 25% | classe *Stats* |
| 5% | Exceções |

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, e.g.:

- Classificação funcional C com má qualidade de código, é despromovida para D;
- Classificação funcional A com má qualidade de código, é despromovida para B.

Os alunos poderão obter feedback juntos dos professores das respetivas turmas sobre o progresso do projeto e qualidade do código, e deverão seguir as recomendações dadas.

Os projetos só poderão ser entregues em suporte eletrónico ficheiro comprimido .zip (contendo somente os ficheiros .java desenvolvidos 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.