

# Trabalho Prático de Programação II

## Relatório

### Licenciatura em Engenharia Informática

Ano Letivo 2024/2025

Trabalho realizado por:  
André Gonçalves – 58392  
André Zhan - 58762

## ÍNDICE

<b><i>Introdução.....</i></b>	<b><i>3</i></b>
<b><i>Parte I – O jogo .....</i></b>	<b><i>4</i></b>
<b><i>Parte II – Validar palavras encontradas.....</i></b>	<b><i>6</i></b>
<b><i>Parte II – Gerar novos jogos.....</i></b>	<b><i>7</i></b>
<b><i>Parte III – Interface Gráfica.....</i></b>	<b><i>8</i></b>
<b><i>Conclusão.....</i></b>	<b><i>10</i></b>

## INTRODUÇÃO

O objetivo deste trabalho é implementar em Java o jogo de palavras chamado “Palavra GURU”, apresentado no enunciado.

Para realizar o trabalho nós seguimos as indicações do enunciado e decidimos fazer todas as partes do trabalho.

Iremos apresentar duas versões do jogo, uma que possui todas as funcionalidades propostas pela professora e que é apresentado na consola, e outra versão em que o jogo é apresentado em uma interface gráfica mas por outro lado está incompleto uma vez que conseguimos implementar apenas algumas das funcionalidades propostas nesta versão com o GUI.

De seguida iremos explicar como implementámos o jogo e a lógica usada para cada uma das partes.

## PARTE I – O JOGO

Na primeira parte do trabalho devemos implementar o jogo solicitado. Para tal criámos uma classe interna “Nível” que é usada para representar um nível do jogo. Esta classe possui dois atributos:

- “letras” – é uma string que representa as letras disponíveis do respetivo nível para formar palavras;
- “palavras” – é um array de strings que contém as palavras que devem ser adivinhadas no respetivo nível.

De seguida criámos o método “carregarNiveis” usado para ler os níveis a partir do ficheiro que contenha os níveis a serem jogados. Para carregar os níveis, o ficheiro é lido linha por linha, linhas em branco serão ignoradas e, as letras assim como as palavras serão agrupadas em um objeto *Nível*.

Para implementar a funcionalidade de salvar o progresso do jogo e recuperá-lo da próxima vez que o utilizador jogar, criámos os métodos “salvarProgresso” e “carregarProgresso”. No primeiro caso, para guardar o progresso do jogo, é criado um ficheiro em que se escreve o nível atual, o número de moedas e as palavras adivinhadas no nível atual. Já o segundo método lê o ficheiro em que foi salvo o jogo e é retornado um `Map<Integer, Set<String>>` em que a chave (`Integer`) representa o índice do nível e o valor (`Set<String>`) contém as palavras já adivinhadas nesse nível.

Para incluir o sistema de dicas criámos o método “darDica” que percorre as palavras do nível atual e verifica se há alguma que ainda não foi adivinhada. Quando encontrar uma palavra elegível, o método fornece ao utilizador a primeira letra da mesma, funcionando como uma dica.

Por fim, para a execução do jogo criámos o método “jogarGuru”. Neste método começamos por verificar se há algum ficheiro com progresso salvo e é pedido ao utilizador que responda se pretende recuperar o progresso ou começar de início. Se o utilizador responder que quer recuperar o progresso, então o ficheiro com o progresso é lido e é usado o método “carregarProgresso” para carregar o nível, as moedas e as palavras adivinhadas do jogo salvo.

De seguida, enquanto o número do nível atual for menor que o tamanho da lista que contém os níveis, o jogo decorre. Neste momento é solicitado um input ao utilizador e o mesmo pode escolher entre tentar adivinhar uma palavra, ou solicitar uma dica e se for o caso, será usado o método “darDica” se o utilizador tiver moedas suficientes, ou o utilizador pode escolher salvar o progresso atual, onde será usado o método “salvarProgresso”.

Sempre que o utilizador acerta uma palavra do nível, é lhe concedido 10 moedas, cujas são armazenadas na variável “coins”, que é definida dentro do método “jogarGuru”.

## PARTE II – VALIDAR PALAVRAS ENCONTRADAS

Nesta parte do trabalho pretende-se implementar uma forma de validar palavras que não estão listadas no nível, mas que também se formam com as letras disponíveis no mesmo e que estão presentes no dicionário fornecido. Para tal efeito, juntamente com o ficheiro que contém o dicionário, criámos os seguintes métodos:

- O método “carregarDicionario”, que guarda as palavras todas que estão contidas no ficheiro do dicionário utilizando uma *HashSet*, de forma a evitar palavras repetidas, e utilizando o *BufferedReader* para uma leitura mais eficiente. O mesmo remove os espaços em brancos entre as palavras e converte todas as palavras para letras maiúsculas visando uniformidade;

- O método “formaPalavra” que é utilizado para verificar se a palavra inserida pelo utilizador é ou não formada pelas letras disponíveis no nível atual.

Desta forma, com os métodos anteriormente mencionados implementados, já é possível verificar se a palavra inserida pelo utilizador pertence ao dicionário fornecido e se é constituída pelas letras disponíveis no nível, concedendo assim 10 moedas de bônus ao utilizador, embora a palavra não corresponda a nenhuma palavra do nível.

## PARTE II – GERAR NOVOS JOGOS

Nesta parte do trabalho o objetivo é implementar a possibilidade de gerar novos jogos a partir das palavras presentes no dicionário fornecido. Para isso, criámos o método “gerarNiveisPeloDicionario”:

- Este método gera 5 novos níveis através das palavras contidas no ficheiro dicionário. Para tal finalidade, primeiro é escolhida uma palavra do dicionário aleatoriamente, servindo como palavra principal, e de seguida armazena-se num mapa cada letra da mesma, juntamente com a sua quantidade total de vezes que aparece na palavra. A partir daí procuram-se palavras adicionais que possam ser formadas com as letras e as suas quantidades totais respetivas, da palavra principal. No fim, o método escreve as letras embaralhadas e as palavras encontradas em um novo ficheiro, que vai conter os novos níveis gerados, separando cada nível com uma linha em branco.

## PARTE III – INTERFACE GRÁFICA

Por fim, para apresentar o jogo, anteriormente implementado, com uma interface gráfica fizemos o seguinte:

Criámos a classe “GuruGUI” e definimos um construtor “GuruGUI” onde o mesmo inicializa os dados do jogo e configura a interface gráfica utilizando os métodos “inicializarGui” e “carregarNivel”.

No método “inicializarGui” criamos a janela “JFrame”, definimos o seu tamanho e adicionamos painéis “JPanel” que organizam os componentes. Temos então:

- Painel superior - Exibe o nível, letras disponíveis e moedas;
- Área central – Um campo de texto onde o utilizador recebe as mensagens;
- Painel inferior - Contém o campo de texto onde é possível escrever o input do utilizador e um botão para submeter a resposta;
- Painel lateral - Contém os botões para obter uma dica e salvar o progresso do jogo.

Já o método “carregarNivel” é usado para obter e carregar as informações do nível atual.

Além dos métodos anteriormente mencionados, temos também o método “darDica”, usado para conceder uma dica ao utilizador e implementado igualmente como na classe “Guru” (jogo sem a interface gráfica). Temos o método “addEventListeners” usado para configurar os eventos dos



botões, nomeadamente do botão “Submeter”, do botão “Dica” e do botão “Salvar Progresso”.

Acrescentando, temos ainda o método “submeterPalavra” que atua quando o utilizador tenta adivinhar uma palavra e clica no botão “Submeter”. Este método vai determinar se o utilizador acertou uma palavra do nível, se escreveu uma palavra válida não listada no nível mas presente no dicionário ou se escreveu uma palavra inválida.

Por fim, criámos ainda o método “atualizarLabels” que vai atualizar, durante o decorrer do jogo, o número de moedas do utilizador.

Nesta versão, com a interface gráfica, o jogo funciona, porém não tem as funcionalidades de carregar o dicionário e consequentemente de gerar novos níveis através do dicionário; de verificar se a palavra inserida pode ser formada com as letras disponíveis no caso do utilizador escrever uma palavra não listada no nível; de salvar e carregar o progresso do jogo; de fazer a escolha inicial do jogo, em que o utilizador escolhe se quer jogar os níveis originais, os níveis gerados através do dicionário ou se quer gerar novos níveis.

## CONCLUSÃO

Concluindo, para a realização deste trabalho decidimos implementar todas as funcionalidades propostas pela professora e, acabámos por ficar com duas versões do jogo, uma em que o jogo é apresentado na consola e tem todas as funcionalidades propostas, e outra versão em que o jogo é apresentado com uma interface gráfica porém incompleto, com falta de várias funcionalidades.