|  |
| --- |
| Programação Orientada Por Objetos  Curso Técnico Superior Profissional  CTeSP  2018/2019 |
| Relatório Técnico |

Docente: Patrícia Dinis

nº180269016 João Almeida

nº181210022 Pedro Grilo

Índice

[1. Introdução 3](#_Toc536107886)

[2. Problema 3](#_Toc536107887)

[3. Solução proposta 3](#_Toc536107888)

[a. Descrição de aspetos particulares relevantes 4](#_Toc536107889)

[4. Utilização da aplicação 4](#_Toc536107890)

[5. Discussão – análise da solução proposta 6](#_Toc536107891)

[6. Conclusão e trabalhos futuro 6](#_Toc536107892)

[7. Referências 6](#_Toc536107893)

[8. Anexos 6](#_Toc536107894)

# Introdução

Este relatório foi realizado no âmbito da disciplina de Progamação Orientada Por Objetos, com o objetivo de descrever o problema que foi construído, ou seja, a Gestão de Recursos Humanos

Tem também como objetivo mostrar as soluções propostas para o problema, por exemplo, como a Gestão de Recursos Humanos foi resolvido ou com que programa foi criado, descrever um resumo da utilização da aplicação, uma discussão da análise da solução proposta, e uma conclusão com objetivos de futuros trabalhos.

Os conceitos fundamentais da programação orientadas por objetos está bastante alicerçada na solução que se pretende que responda às necessidades do projeto, nomeadamente os conceitos de classe, objeto, abstração de dados, proteção e encapsulamento. Para além disto estão presentes os conceitos avançados tais como o polimorfismo, hierarquia, herança e agregação.

# Problema

O objetivo deste problema é desenvolver uma aplicação, utilizando a linguagem JAVA, de forma a implementar uma aplicação que gerisse os recursos humanos de uma empresa. No problema, destacam-se as seguintes entidades:

* **A Empresa;**
* **Os Tipos de Empregados;**
* **Ficheiros**
* **JavaFX**

# Solução proposta

Este **projeto** foi dividido em **duas** **fases**:

* A **primeira** **fase**, consistia em realizar o projeto em consola
* E a **segunda** **fase,** consistia em utilizar os recursos feitos da primeira fase e implementares num interface gráfica com o **JavaFX**.

O **projeto** na **primeira fase** foi realizado inicialmente pela a base do programa, ou seja, criar as **classes com as respetivas interfaces, menus, opções, etc**. Após concluir, de seguida, foram feitas as opções do **Menu Gestão**, pois era onde continha a maior parte do código, tal como, Inserir uma nova ficha de empregado; Verificar se existe algum empregado com um código dado como parâmetro; Obter a ficha de empregado de código dado como parâmetro, se existir; Dada uma lista de empregados contratados, inseri-la no registo atual; Determinar o número atual de empregados de uma dada categoria; Devolver o conjunto atual de fichas de empregados; Devolver o conjunto atual de fichas de empregados, filtrados por categoria; Calcular, a qualquer momento, o total de salários a pagar; Calcular e apresentar os custos trimestrais, semestrais e anuais com salários, devendo ter em conta os subsídios de natal e de férias;

**Após concluir a etapa anterior** e em seguimento com a aula de exceções, começaram a ser **criadas as exceções, com o respetivo enumerado onde contem todos os erros**.

**Para testes e afins**, foi criado **o Menu Empregados**, onde era possível registar o dia trabalhado, introduzir o valor de vendas num determinado mês ou ate mesmo adicionar a distancia percorrida.

De seguida, foi feito a parte dos **ficheiros**, onde guarda o **objeto** **empresa** num ficheiro e após o início do programa, este é lido automaticamente, e com isto damos como concluído este tópico: Guardar a lista atual de empregados num ficheiro de texto;

Para **finalizar** a **primeira** **fase** do projeto, foi analisado o código todo, corrigindo os *bugs*, e organizando e otimizando o código.

Na segunda fase, o objetivo era criar uma interface gráfica utilizando o JavaFX.

Inicialmente, começou por ser feito a maquete da interface numa folha A4, para mais tarde ser copiada para o *Scene Builder* ( programa que ajuda a desenvolver a interface ).

Com o *Scene Builder*, foi utilizado um pacote(.JAR) para o mesmo que continha estilos de no

## Descrição de aspetos particulares relevantes

O uso de ficheiros binários para armazenar as pontuações e utilizadores de forma duradoura;

Utilização de algoritmos como *bubble sort* para a organização dos *top scores*;

Funções de animação após a vitória/derrota do jogador;

Confirmação de frase chave correta.

# Utilização da aplicação

Para utilização da aplicação, é recomendado o sistema operativo *Windows*.

Após a abertura da aplicação, o utilizador depara-se com um **Menu Inicial**, onde lhe é apresentado três opções, as quais são:

* **Entrar;**
* **Registar;**
* **Sair;**

Caso seja a primeira vez que o utilizador entra na aplicação, é recomendado registar-se primeiro, ou seja, escolher a opção 2.

Se o utilizador escolher a opção entrar e iniciar sessão corretamente, é-lhe apresentado o **menu Principal**, onde encontrará quatro opções, as quais são:

* **Jogador contra PC;**
* **Jogador contra Jogador;**
* **Ranking;**
* **Editar Informações;**
* **Sair;**

Se o utilizador selecionar a primeira opção (**Jogador contra PC**), o utilizador acede a um **submenu,** onde será possível escolher o tema da palavra para mais tarde adivinhar a mesma relacionada com o tema. O objetivo é adivinhar a palavra que o computador “escolheu”.

Se o utilizador selecionar a segunda opção (**Jogador contra Jogador**), o utilizador acede ao **Menu Entrar**, que serve para o segundo utilizador fazer o *login*.

Após este utilizador fazer *login*, o utilizador um irá escolher a palavra/frase para o utilizador dois adivinhar.

Se o utilizador selecionar a terceira opção (**Ranking**), o utilizador terá acesso a um **submenu**, onde será possível consultar o *ranking* do jogo (**Jogador contra PC e Jogador contra Jogador**).

Se o utilizador selecionar a quarta opção, será possível, editar as informações da conta, tais como, nome de utilizador, palavra-passe, *resetar* os pontos, e até eliminar a conta.

Se o utilizador selecionar a quinta e última opção (**Sair**), o jogador faz *log out*, e volta ao **menu Inicial**.

# Discussão – análise da solução proposta

O facto de terem sido utilizados ficheiros, facilita bastante, não só para o programador, como também ao utilizador, pois desta forma o utilizador não tem que se registar sempre que entra no programa, nem tem que voltar a zero pontos sempre que se regista novamente, e também torna as pontuações algo muito mais competitivo, devido a não serem apagadas no fecho do programa.

Para o programador, torna-se muito mais fácil testar o programa em si e controlar, o que acontece com o programa em geral.

Infelizmente com tantas funcionalidades, vem uns pequenos inconvenientes que é o facto de o ficheiro binário irá ocupar espaço no computador, porem não é muito, outro inconveniente é que se o utilizador alterar algo no ficheiro, o mesmo fica corrompido.

Caso seja preciso alterar algo manualmente, é bastante complicado, sendo que o ficheiro é escrito em binário e os leitores de texto comuns não conseguem ler binário com facilidade mostrando muitas vezes caracteres especiais no seu lugar.

# Conclusão e trabalhos futuro

Com este trabalho, foi-nos possível retirar várias conclusões, como a aprendizagem e utilização de ficheiros binários e de algoritmos de ordenação (bubble sort).

Conclui-se também, que não foi um trabalho fácil, pois tivemos que fazer várias verificações, as quais deram algum trabalho, e tivemos que manipular ficheiros, mas após algum estudo obtivemos sucesso ao utilizá-los.

Concluindo, em futuros trabalhos teremos de melhorar o nosso código fonte, para um código mais reduzido e mais eficiente.

# Referências

<https://stackoverflow.com/>

<http://www.codeblocks.org>

<https://www.geeksforgeeks.org>

# Anexos

Manual de Instruções.