UNIVERSIDADE D
COIMBRA

FCTUC - Faculdade de Ciências e Tecnologia

da Universidade de Coimbra

Programação Orientada aos Objetos

Projeto: Aplicação StarThrive

Professora: Karima Castro

Autores: Alexandre Ferreira * 2021236702

Duarte Oliveira * 2020238969

Coimbra, 7 de Dezembro de 2022

Índice

Introdução

No âmbito da cadeira Programação Orientada aos Objetos, este projeto tem como objetivo dar aos alunos conhecimento suficiente para modelar problemas de acordo com o paradigma da orientação a objetos, desenvolver as boas práticas da programação, nomeadamente, encapsulamento, herança, polimorfismo e tratamento de erros, como também aprender melhor como utilizar ficheiros e o GUI.

Deste modo, foi pedido a criação de uma aplicação para a empresa StarThrive. Esta empresa gere empresas do ramo da restauração e mercearias e a aplicação vai ser responsável por geri-las.

Uma vez que vai ser utilizado por um funcionário da StarThrive, a aplicação deverá ter as implementações necessárias para a correta gestão das mesmas. As funcionalidades variam desde listar todas as empresas, editá-las, apagá-las e criá-las, como também apresentar as empresas com maior receita anual, menor despesa anual, maior lucro anual e as duas empresas do ramo restauração com maior capacidade de clientes por dia.

Funcionamento da Aplicação

A lista de empresas estão armazenadas num ficheiro de texto “starthrive.txt”, contendo uma estrutura estratégica para a fácil e rápida edição do ficheiro, e analogamente simplificar o seu parsing pela aplicação.

Na primeira vez que o programa corre, lê-se o conteúdo do ficheiro para o arraylist de empresas e logo de seguida cria-se um ficheiro de objetos com o conteúdo presente no ficheiro txt. Seguidamente à criação deste último ficheiro, o programa irá ler sempre a partir dele, da mesma maneira que as alterações feitas pelo utilizador serão refletidas no mesmo. Uma vez que o arraylist de empresas já contém as empresas existentes, o resto da aplicação utiliza vários métodos importantes para o seu correto funcionamento, utilizando o arraylist, dos quais vamos realçar seguidamente.

Dado que um dos principais objetivos da empresa StarThrive é a gestão das despesas e receitas anuais, na classe Empresa existem dois métodos abstratos, “despesa” e “receita”. Estes têm como função calcular e retornar a respetiva despesa e receita de cada tipo de empresa, consoante a sua assinatura.

Na classe StarThrive, denota-se os métodos “adicionarEmpresa” ao qual é destinado a função de adicionar uma empresa ao arraylist através do método “add”, “apagarEmpresa” que apaga uma empresa no arraylist através do método “remove”, “replaceEmpresa” que substitui uma empresa por outra num determinado índice do arraylist.

Verifica-se também a existência dos métodos “atualizaMaiorReceita” “atualizaMenorDespesa” e “atualizaMaiorLucro”. Estes começam por inicializar os atributos estáticos de cada subclasse da classe empresa, responsáveis por armazenar o índice da empresa no arraylist com a maior receita anual, a menor despesa anual e o maior lucro anual, tendo assim a segurança que se compara empresas pertencentes à mesma classe. Posteriormente, para esses índices serem atualizados, percorre-se o arraylist e utiliza-se os métodos abstratos correspondentes “*comparamaiorReceita”, “comparamenorDespesa” e “comparamaiorLucro”* que levam, deste modo, a empresa ir atualizar o índice respetivo, à sua classe. Similarmente, o método “maisclientesdiarioRestauracao” é responsável por atualizar os dois atributos estáticos “maiorcapacidadeclientes1” e “maiorcapacidadeclientes2”, na classe Restauração, que contêm os índices no arraylist das duas empresas com maior capacidade de clientes.

Na classe Ficheiro, vale a pena evidenciar o método “carregaDados”, onde este, por sua vez, irá verificar se o ficheiro de objeto existe. Caso não exista, lê-se o ficheiro de texto e armazena-se o seu conteúdo no arraylist que queremos utilizar, através do método “converteEmpresa”. Se existir, lê-se o ficheiro objeto com a ajuda do método “lerFicObjeto” e armazena-se o seu conteúdo no arraylist referenciado anteriormente.

Sempre que haja alguma alteração feita pelo utilizador no arraylist, o método “gravarFicObjeto” grava num ficheiro de objeto os dados do mesmo.

Manual do utilizador

1. Visualizar a informação de todas as empresas
    1.1- Abra o programa e clique no botão “seguinte”
    1.2- Clique no botão “Listar empresas”

2. Editar uma empresa
    2.1- Abra o programa e clique no botão “seguinte”
    2.2- Clique no botão “Listar empresas”
    2.3- Selecione uma empresa e clique no botão editar
    2.4- Mude os parâmetros que deseja alterar
    2.5- Clique no botão “Gravar”

3. Apagar empresa

    3.1- Abra o programa e clique no botão “seguinte”

    3.2- Clique no botão “Listar empresas”

    3.3- Selecione uma empresa e clique no botão “apagar”

4. Visualizar todo o tipo de empresas com maior receita anual

    4.1- Abra o programa e clique no botão “seguinte”

    4.2- Clique no botão “Maior receita anual”

5. Visualizar todo o tipo de empresas com menor despesa anual

    5.1- Abra o programa e clique no botão “seguinte”

    5.2- Clique no botão “Menor despesa anual”

6. Visualizar todo o tipo de empresas com maior lucro anual

    6.1- Abra o programa e clique no botão “seguinte”

    6.2- Clique no botão “Maior lucro anual”

7. Visualizar as 2 empresas do tipo restauração com maior capacidade de clientes por dia

    7.1- Abra o programa e clique no botão “seguinte”

    7.2- Clique no botão “Visualizar as 2 empresas do tipo restauração com maior capacidade de clientes por dia”

Conclusão

Em suma, este projeto ajudou a desenvolver e consolidar consideravelmente os conhecimentos sobre Java e boas práticas de programação orientada aos objetos.

Ao longo da execução deste projeto, várias maneiras que utilizámos para implementar as funcionalidades da aplicação não eram as mais corretas perante o polimorfismo, no entanto, as soluções finais são as que mais respeitam o mesmo. Com isto, verifica-se um aumento no conhecimento sobre programação orientada aos objetos.

Denota-se que no UML a classe ButtonlistarempresaListener é só um exemplo, pois, na verdade, existem mais classes parecidas. Escolhemos não colocar todas as classes que criam janelas, porque todas tinham o mesmo princípio e assim o UML não se torna confuso. Além disso, não colocámos os getters e setters no UML, para que este não ficasse desnecessariamente grande. Por outro lado, devido ao programa utilizado na construção do UML não conseguimos sublinhar os atributos estáticos sem sublinhar todo o bloco; sendo assim, os atributos, estes, são seguidos pela expressão (static).

Referência

Para a realização deste trabalho, consultámos os slides fornecidos, a informação transmitida na aula, e exemplos de exercícios feitos pela professora Karima.

Para a realização do UML consultámos o seguinte site:

UML Class Diagram Tutorial (visual-paradigm.com)