

**Instituto Politécnico do Cávado e do Ave**

**Escola Superior de Tecnologia**

**Licenciatura**

**em**

**Engenharia de Sistemas Informáticos**

**Relatório do Trabalho Pratico**

Fábio Rafael Gomes Costa – a22997

Luís Pedro Pereira Freitas – a23008

**Barcelos, dezembro de 2023**

****

**Instituto Politécnico do Cávado e do Ave**

**Escola Superior de Tecnologia**

**Licenciatura**

**em**

**Engenharia de Sistemas Informáticos**

**Relatório do Trabalho Pratico**

Fábio Rafael Gomes Costa (nº 22997)

Luís Pedro Pereira Fretas (nº 23008)

Unidade Curricular: Programação Orientada a Objetos

**Docente:**

Luís G. Ferreira

**Barcelos, dezembro de 2023**

# Ficha de Identificação

|  |  |
| --- | --- |
| **Elaborado por** | Fábio Rafael Gomes Costa  Luís Pedro Pereira Freitas |
| **Número Mecanográfico** | a22997  a23008 |
| **Contato** | [a22997@alunos.ipca.pt](mailto:a22997@alunos.ipca.pt)  a23008@alunos.ipca.pt |
|  |  |
| **Unidade Curricular** | Programação Orientada a Objetos |
| **Curso** | Licenciatura em Engenharia de Sistemas Informáticos |
| **Instituição** | Escola Superior de Tecnologia do Instituto Politécnico do Cávado e do Ave |
| **Docente** | Doutor Luís G. Ferreira |
| **Contato/Gabinete** | [lufer@ipca.pt](mailto:lufer@ipca.pt) /Gab: 5 |
|  |  |
| **Data de início** | 20 de outubro de 2023 |
| **Data de conclusão** | 19 de dezembro de 2023 |
| **Tempo Aplicado** | 12 horas |

# Resumo

# Palavras-Chave

* \*\*

# Índice

[Ficha de Identificação I](#_Toc150720722)

[Resumo II](#_Toc150720723)

[Palavras-Chave II](#_Toc150720724)

[Índice III](#_Toc150720725)

[Índice de Figuras IV](#_Toc150720726)

[Índice de Tabelas V](#_Toc150720727)

[Lista de siglas VI](#_Toc150720728)

[1. Introdução 1](#_Toc150720729)

[1.1 Contextualização 1](#_Toc150720730)

[1.2 Motivação e Objetivos 1](#_Toc150720731)

[1.3 Estrutura do Documento 1](#_Toc150720732)

[2. Atividade 2](#_Toc150720733)

[3. Problema 2](#_Toc150720734)

[3.1 Analise simples/básica 2](#_Toc150720735)

[3.2 Analise com aplicação de matéria 2](#_Toc150720736)

[4. Resolução de Problema 2](#_Toc150720737)

[5. Considerações Finais 3](#_Toc150720738)

[5.1 Reflexão sobre as competências adquiridas 3](#_Toc150720739)

[5.2 Aplicação dos conteúdos aprendidos em sala de aula 3](#_Toc150720740)

[5.3 Sugestão 3](#_Toc150720741)

[6. Glossário 4](#_Toc150720742)

[7. Referencias bibliográficas 5](#_Toc150720743)

# Índice de Figuras

**Não foi encontrada nenhuma entrada do índice de ilustrações.**

# Índice de Tabelas

**Não foi encontrada nenhuma entrada do índice de ilustrações.**

# Lista de Siglas/Acrónimos

IPCA – Instituto Politécnico do Cavado Ave

LESI/ESI – Licenciatura em Engenharia de Sistemas Informáticos

POO – Programação Orientada a Objetos

UC /UCs – Unidade Curricular /Unidades Curriculares

VS – Visual Studio 2022

# Introdução

## Contextualização

No Âmbito da unidade curricular de “Programação Orientada a Objetos”, foi nos solicitada a realização de um trabalho pratico, para instrumento de avaliação, pelo docente Dr. Luís G. Ferreira. Esse trabalho consiste na análise de um problema real simples e a aplicação do Paradigma Orientado a Objetos na implementação de possíveis soluções.

## Pretensões/Motivação e Objetivos

Com este trabalho pratico pretende-se melhorar a estruturação de dados e a forma de programar em um Paradigma Orientado a Objetos, e com este relatório pretende-se justificar e mostrar como foi pensada, realizada e estruturada a resolução do problema real selecionado. Pretende-se que sejam desenvolvidas soluções em C# para problemas reais de complexidade moderada. Serão identificadas classes, definidas estruturas de dados e implementados os principais processos que permitam suportar essas soluções. Pretende-se ainda contribuir para a boa redação de relatórios.

Objetivos:

* Consolidar conceitos basilares do Paradigma Orientado a Objetos;
* Analisar problemas reais;
* Desenvolver capacidades de programação em C#;
* Potenciar a experiência no desenvolvimento de software;
* Assimilar o conteúdo da Unidade Curricular.

## Estrutura do Documento

Este relatório foi pensado/estruturado no seguinte formato, primeiro a capa e a ficha de identificação, depois o resumo e o índice (índice de tópicos e de imagens e/ou tabelas), apos iniciasse a introdução, de seguida o desenvolvimento, posteriormente a conclusão, e por fim a Webgrafia e glossário, onde são listados todos os links consultados na elaboração do trabalho, e onde são apresentadas expressões e os seus significados.

# Problema

O problema real selecionado para desenvolvimento deste trabalho pratico foi um dos indicados pelo docente, sendo o seguinte:

“Gestão de rendas/imoveis: sistema que permita a uma empresa/proprietário gerir os seus imoveis e as respetivas rendas mensais; Keywords: senhorios, inquilinos, imoveis (apartamento, vivenda, terreno), recibos, contratos, despesas, documentos, distritos (código e descrição nas finanças), concelhos (código e descrição nas finanças), freguesias (código e descrição nas finanças), estados.”

## Analise simples/básica

Uma empresa de gestão de rendas e imoveis “desempenha um papel crucial ao administrar propriedades em nome de proprietários. Suas responsabilidades incluem atrair inquilinos, cobrar aluguéis, cuidar da manutenção, gerenciar finanças, resolver conflitos, realizar vistorias e garantir conformidade legal. O objetivo é proporcionar aos proprietários uma gestão eficiente e tranquila de suas propriedades, otimizando o retorno financeiro e minimizando as preocupações operacionais”. <https://blog.kurby.ai/pt/the-role-of-property-management-in-real-estate-development/>

Assim sendo, podem ser identificados três intervenientes, o proprietário, o cliente, e o operador da empresa. Para alem destes também podemos considerar o próprio imóvel.

Mas quais informações a empresa possui destes intervenientes? E, porque possui essas informações?

Do proprietário possui:

* Um nome;
* Um número de contribuinte;
* Um IBAN;
* Um contato (email/Telemóvel).

Do Cliente possui:

* Um nome;
* Um número de contribuinte;
* Data de nascimento (ainda a determinar);
* Um contato (email/Telemóvel).

Do operador da empresa possui:

* Um nome;
* Um número de Operador;
* Um contato (email/Telemóvel).

Do imóvel possui:

* Contribuinte proprietário
* Uma morada \*2;
* Um número de identificação predial.

\*2 Da morada possui:

* Número de porta;
* Código postal;
* Rua;
* Freguesia;
* Concelho;
* Distrito.

A partir destes dados a empresa pode originar documentos, tais como o contrato, para originar estes documentos a empresa precisa:

- Um contrato de arrendamento

* Um operador
  + Um número de Operador;
* Um cliente
  + Um nome;
  + Um número de contribuinte;
* Um proprietário
  + Um nome;
  + Um número de contribuinte;
* Um imóvel
  + Uma morada
  + Um número de identificação predial
* Um valor da mensalidade
* Nº de contrato
* Taxa da empresa

O objetivo do software, é permitir a uma empresa gerir os seus imoveis e as respetivas rendas mensais, ou seja, tem de permitir a:

Consulta de:

* Proprietário
* Cliente
* Imóvel
* Contrato (Exportar contrato em PDF)

Inserir:

* Proprietário
* Cliente
* Imóvel

Criação de:

* Contrato

## Analise na ótica de Orientação a Objetos

A partir da análise básica, foi possível avançar para uma análise aplicando os conceitos adquiridos no decorrer da UC.

(…)

Uma imagem com texto, diagrama, captura de ecrã, Paralelo

Descrição gerada automaticamente

(…)

### Heranças, Classes, método e variáveis

Para realizar-se uma solução para este problema, foi necessário utilizar-se as seguintes classes, métodos e variáveis.

(…)

#### Classe Proprietário

A classe proprietária é responsável por criar e mostrar os proprietários (…)

Variáveis:

* + nomeProp : string
  + contribuinteProp: int
  + ibanProp: string
  + emailProp: string
  + telemovelCliente: int

Métodos:

* CriarProprietario – este método tem como função receber e arquivar em um array, as variáveis que definem um proprietário, utilizando o número de contribuinte como índice do array.
* MostrarProprietario – ao receber o número de contribuinte este método iguala-o ao valor de índice do array, o que permitira a exportação dos dados do proprietário em uma Spring.

#### Classe Cliente

A classe cliente é responsável por criar e mostrar clientes (…).

Variáveis:

* + nomeCliente: string
  + contribuinteCliente: int
  + datanascimentoCliente: date
  + emailCliente: string
  + telemovelCliente: int

Métodos:

* CriarCliente – este método tem como função receber e arquivar em um array, as variaveis que definem um cliente, utilizando o número de contribuinte como índice do array.
* MostrarCliente –ao receber o número de contribuinte este método iguala-o ao valor de índice do array, o que permitira a exportação dos dados do Cliente em uma Spring.

#### Classe Operador

A classe Operador é responsável por criar e mostrar operadores (…).

Variáveis:

* + nomeOperador: string
  + numOperador: int
  + emailOperador: string
  + telemovelOperador: int

Métodos:

* MostrarOperador - ao receber o número de operador este método iguala-o ao valor de índice do array, o que permitira a exportação dos dados do proprietário em uma Spring.

#### Classe Morada

A classe morada é responsável por criar a morada (…)

Variáveis:

* + numeroPorta: int
  + codigoPostal: int
  + rua: string
  + freguesia: string
  + concelho: string
  + distrito: string

#### Classe Imóvel

Variáveis:

* + morada: string
  + numidentificacaoPredial: int

Métodos:

* CriarImovel – este método tem como função receber e arquivar em um array, as variáveis que definem um Imóvel, utilizando o número identificação predial como índice do array.
* MostarImovel – ao receber o número Identificação Predial este método iguala-o ao valor de índice do array, o que permitira a exportação dos dados do Imóvel em uma Spring.

#### Classe Contrato Arrendamento

Variáveis:

* + valorMensalidade: double
  + numcontrato: int
  + taxaEmpresa: int

Métodos:

* CriarContrato – este método tem como função receber e arquivar em um array, as variáveis que definem um contrato, utilizando o número contrato como índice do array.
* MostarContrato - ao receber o número de contrato este método iguala-o ao valor de índice do array, o que permitira a exportação dos dados do contrato em uma Spring.

# Resolução do Problema

(Aqui será inserido o relatório exportado diretamente do Visual Studio)

# Considerações Finais

## Reflexão sobre as competências adquiridas

## Aplicação dos conteúdos aprendidos em sala de aula

## Sugestão

# Glossário

|  |  |
| --- | --- |
| **Expressão** | **Significado** |
| *\*\*\*\*\*\*\** | “” |
| Algoritmo | “Um algoritmo representa uma sequência finita e não ambígua de instruções de modo a obter a resolução de um problema sob a forma de resultado (saída de dados) tendo por base uma prévia entrada de dados. Um algoritmo é representado através de uma linguagem com uma determinada sintaxe e semântica associada. Por fim, um algoritmo deve ser eficaz na resolução do problema subjacente assim como eficiente para resolver o problema com o melhor desempenho (performance) possível”. |

# Referencias bibliográficas

[1] L. Ferreira, ‘LESI-POO-2023-2024’, Material das aulas de Programação Orientada a Objetos do Curso de LESI, do IPCA. Accessed: Nov. 01, 2023. [Online]. Available: https://github.com/luferIPCA/LESI-POO-2023-2024