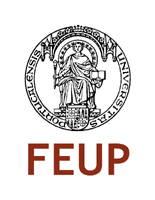
Faculdade de Engenharia da Universidade do Porto



**GESTÃO DE OBRAS**

*Ana Isabel Neves Alves de Sousa,* [*201108026*](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201108026)*,* [*ei11068@fe.up.pt*](mailto:ei11068@fe.up.pt)

*Gabriel Cardoso Candal,* [*201108021*](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201108021)*,* [*ei11066@fe.up.pt*](mailto:ei11066@fe.up.pt)

*João Guilherme Rodrigues Marques de Oliveira,* [*201108000*](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201108000)*, ei11114@fe.up.pt*

Mestrado Integrado em Engenharia Informática e Computação

Algoritmos e Estruturas de Dados

4 de Novembro de 2012

# Índice

[Índice 2](#_Toc339103835)

[1. TEMA DO TRABALHO 3](#_Toc339103836)

[2. SOLUÇÃO IMPLEMENTADA 5](#_Toc339103837)

[3. DIAGRAMAS UML 7](#_Toc339103838)

[4. LISTA CASOS UTILIZAÇAO 8](#_Toc339103839)

[5. PRINCIPAIS DIFICULDADES 9](#_Toc339103840)

[6. ESFORÇO DE CADA ELEMENTO 11](#_Toc339103841)

# INTRODUÇÃO

No âmbito da disciplina de Algoritmos e Estruturas de Dados foi-nos proposto o desenvolvimento de uma aplicação em C++ para a gestão de obras de uma empresa Construtora. A aplicação iria permitir a manipulação de trabalhos e obras, sendo que a Construtora deveria ter um conjunto de obras encomendadas e, para cada obra, os trabalhos contratados.

# 2. TEMA DO TRABALHO

A aplicação de gestão das obras de uma empresa construtora deveria permitir registar toda a informação essencial parar gerir esta.

Cada empresa possuiria uma ou mais obras e, por sua vez, cada obra possui um ou mais trabalhos. Cada trabalho poderia ser de dois tipos: infra-estruturas, subdividido em arruamentos e saneamentos, ou domésticos, subdivido em trolha, carpinteiro e electricista.

Em relação às operações de gestão, deveria ser possível realizar a manutenção das obras e respectivos trabalhos, a manutenção de uma lista de empresas contratadas para cada trabalho bem como efectuar diversos tipos de cálculos relativos à construtura, às obras e às empresas contratadas.

# 3. SOLUÇÃO IMPLEMENTADA

Em primeiro lugar, a aplicação lê e trata convenientemente a informação da construtora de um ficheiro (“obras.txt”) que, se existente, deve ter a seguinte formatação: nome da construtora seguida das obras que a constituem. Cada obra começa por “+”, o nome da obra, uma linha em branco, posteriormente vem a lista dos trabalhos que a obra possui. Cada trabalho é apresentado assim: (uma característica em cada linha) duração, custo, id rua (para infra-estruturas) ou habitação (para domésticos), quantidade material especifico, nome da empresa contratada. Se o trabalho não for o último da obra em seguida vem uma linha em branco, se for vem um “\*”. Seguidamente começa a nova obra e assim sucessivamente até acabarem as obras e respectivos trabalhos.

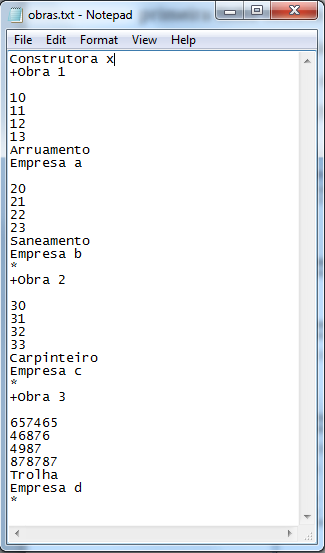


Figura 1: modelo do ficheiro “obras.txt”

A interacção com o utilizador é expressa através de um menu em que são apresentadas várias opções (que são devidamente explicadas na secção 5 correspondente aos casos de utilização).

(imagem do menu inicial)

A aplicação termina quando o utilizador quiser e a informação actualizada da construtora é escrita no ficheiro inicialmente referido com a mesma formatação.

# 4. DIAGRAMA UML

# 5. LISTA CASOS UTILIZAÇAO

# 6. PRINCIPAIS DIFICULDADES

As principais dificuldades sentidas foram:

- Escolher uma ferramenta adequada para o diagrama UML, já que o plug-in do Eclipse para o efeito tem limite de caracteres, o que torna impossível escrever os tipos de retorno, assim como os argumentos. Acabámos por optar pelo *Enterprise Architect*;

- Conseguirmos distinguir quais as funções úteis de implementar e quais as supérfluas;

- Para a função que lista, ou elimina, trabalhos por tipo foi necessário ter algo que identificasse o trabalho (dado que estávamos a usar polimorfismo). A ideia inicial foi ter um vector para cada tipo de trabalho ou uma estrutura semelhante, mas isso seria contornar o polimorfismo, então decidimos usar um enumerador para o efeito (apesar de também poder ser usado para contornar o polimorfismo, não o fizemos, usando-o apenas para o efeito descrito neste parágrafo);

- A dada altura, tivemos um problema em que o Eclipse compilava (sem dar erros), mas não gerava os ficheiros binários. Após experimentar um compilador diferente e alguma pesquisa, chegámos à conclusão que o problema estava em *include*’s cíclicos das *header files*, após reorganização do código o problema desapareceu;

- Os membros-função da classe Obra que retornavam vectores davam erro quando a corríamos o programa, com *invalid arguments*, viemos a concluir que tal se devia ao facto de estarmos a passar uma Obra como *const* nos argumentos de uma função, que no seu corpo chamava as funções que estavam a dar o tal erro, como na altura os membros-função que retornavam o vector de Trabalhos não eram *const*, ao aplicá-los a um objecto que o era dava erro (apesar de não alterar o objecto);

- *Doxygen* ao gerar documentação não incluía o que escrevíamos (apesar de utilizarmos o formato correcto, como por exemplo *@param*), mas apenas o diagrama das classes.

Pensamos que, concluido o projecto, os entraves que nos surgiram foram ultrapassados com sucesso.

# 7. ESFORÇO DE CADA ELEMENTO

O trabalho foi dividido por cada elemento do grupo de forma equitativa. (........)