

23-11-2017

Algoritmos e Estruturas de Dados

Docente Ana Paula Rocha

Leonardo Gomes Capozzi – up201503708@fe.up.pt

Ricardo Carvalho – up201503717@fe.up.pt

Verónica Fradique – up201506440@fe.up.pt

**Relatório do Projeto Castings TV**

Índice

[Descrição do projeto 2](#_Toc499331309)

[Introdução 3](#_Toc499331310)

[Descrição da Implementação 4](#_Toc499331311)

[Diagramas de UML 0](#_Toc499331312)

[Casos de Utilização do Castings TV 0](#_Toc499331313)

[Conclusão 1](#_Toc499331314)

# Descrição do projeto

O projeto a que este relatório se refere foi desenvolvido com base no seguinte enunciado:

“Uma empresa organiza castings para programas de televisão e pretende desenvolver um sistema para suportar esses processos. O processo envolve um conjunto de jurados, caracterizados pelo nome, morada, telemóvel e o género de arte performativa que é sua especialidade. Esses jurados são responsáveis pela avaliação de candidatos nas sessões de casting. Cada sessão possui 3 jurados, sendo um deles o responsável pela sessão. Cada sessão incide sobre um género de arte performativa que será avaliada. A empresa nunca organiza duas sessões de casting de uma arte performativa no mesmo dia. Finalmente, é preciso registar informação sobre os candidatos, incluindo o nome, morada, data de nascimento, género de arte performativa em que se consideram mais aptos, um número único de inscrição que lhes é atribuído quando se inscrevem pela primeira vez e também as suas participações em sessões de casting. Cada candidato pode participar em várias sessões e, para cada uma delas, é registada a ordem (1o, 2o, 3o, etc.) e a classificação final (número de 1 a 10) atribuída pelo jurado responsável pela sessão. O processo de seleção do candidato vencedor ocorre em duas fases. Numa primeira fase, a pontuação é calculada como a média das pontuações dadas por cada jurado. Numa segunda fase, a pontuação do jurado responsável pela sessão tem o dobro do peso das restantes. Na segunda fase, participam os 5 candidatos com maior pontuação da primeira fase. O sistema deve gerir todo o processo de organização dos castings, incluindo inscrições, seleção de jurados e registo de resultados.“

O objetivo do grupo foi criar uma plataforma intuitiva e que cumprisse todas as funcionalidades que o projeto exigia. Assim sendo, findo o tempo para a realização do mesmo, o grupo considera que alcançou os objetivos, com sucesso.

# Introdução

No âmbito da unidade curricular de Algoritmos e Estruturas de Dados foi desenvolvido o projeto Castings TV, na liguagem de programação C++.

Este projeto consiste na estruturação e implementação de um sistema que faça a gestão de todo o processo de casting, desde a criação de candidatos, jurados e sessões, até à posterior criação do casting em si, onde se conhece o vencedor do casting.

O relatório encontra-se dividido nas quatro partes seguintes:

* Descrição da Implementação : nesta secção aborda-se a solução utilizada pelo grupo para cumprir o objetivo do trabalho, de forma detalhada;
* Diagramas de UML: nesta secção encontram-se os diagramas UML do código desenvolvido;
* Casos de Utilização do Castings TV: nesta secção descrevem-se todas as funcionalidades do programa;
* Conclusão: nesta última secção é feita uma descrição das principais dificuldades encontradas durante o desenvolvimento do projeto e o resumo da contribuição dos membros para o projeto.

# Descrição da Implementação

O código que constitui o projeto Castings TV encontra-se dividido em dezassete ficheiros .cpp e .hpp e, para além destes, conta com quatro ficheiros de base que funcionam como bases de dados do programa.

Inicialmente, optou-se por criar uma Super Classe Pessoa, que se encontra nos ficheiros Pessoa.hpp e Pessoa.cpp, que guarda os dados que são comuns a candidatos e a jurados. Esses dados são o nome, a morada, o género de arte a que pertencem e também um booleano. Este booleano é utilizado para quando o utilizador da plataforma Castings TV pretende eliminar um candidato ou um jurado. Essa eliminação consiste em colocar esse booleano com o valor *false* para que o utilizador não lhe tenha mais acesso, mas todas as participações e as suas posições nos castings que já efetuou permanecem guardadas.

Por sua vez, da classe Pessoa derivam as classes Candidato e Jurado. As mesmas encontram-se, nos ficheiros Candidato.cpp e Candidato.hpp, no caso do Candidato e Jurado.cpp e Jurado.hpp no caso do Jurado.

A classe Candidato acrescenta aos membros que herda da classe Pessoa uma data de nascimento, um número de inscrição que é sequencial e não pode ser alterado pelo utilizador, e as participações que o candidato tem em sessões, guardadas num *vector* de apontadores para Participação, classe que se encontra abaixo explicada.

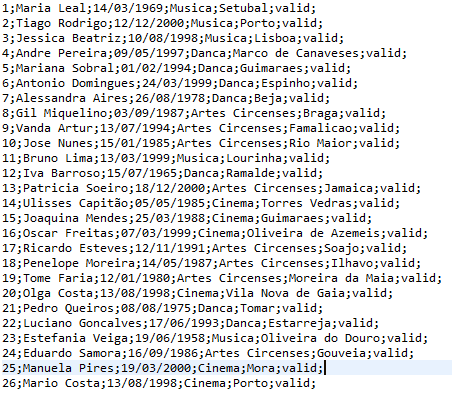
A classe Jurado contém, para além dos membros que são comuns a Pessoa, um atributo que é um número de telemóvel. Este número de telemóvel é único, pelo que funciona também como *ID* do jurado em questão.

A classe Participação, já anteriormente referida, pode ser encontrada nos ficheiros Participacao.cpp e Participacao.hpp. Os atributos desta classe são os seguintes: um apontador para uma sessão, que aparece descrita no decorrer da descrição da implementação, um *array* com quatro posições que guarda a pontuação final do candidato na fase da sessão na primeira posição e as pontuações atribuídas por cada jurado nas três últimas posições do *array*. Destas três, a primeira delas é a pontuação referente ao jurado responsável pela sessão. Um outro atributo é a posição em que o candidato terminou e, para além disto, a classe tem um outro membro que é a fase, para que se saiba a que fase da sessão se refere este resultado.

A classe Sessão está declarada nos ficheiros Sessao.cpp e Sessao.hpp. Esta classe é a que define em que consiste a base do projeto, sessões de casting com pontuações atribuídas por jurados a candidatos e no final da qual se apura um vencedor. Esta classe tem quatro membros dado: um *vector* de apontadores para Jurado, que terá três posições cada uma a apontar para um jurado, que o utilizador da plataforma previamente escolheu quando resolveu agendar uma sessão,  uma *string* que será a arte performativa a que se refere a sessão, a data da sessão no formato que é pedido ao utilizador - dd/mm/aaaa - e que é verificada através da função *isValidDate()*, e por fim um booleano que será o que permite saber se já foram geradas as fases da sessão, primeira e segunda. Se já tiverem sido geradas este booleano toma o valor *false* e não é permitido ao utilizador editar a sessão, caso contrário ele pode fazer alterações, como por exemplo trocar a data em que se irá realizar.

O programa desenvolvido conta também com uma outra classe que designamos como Empresa que está implementada nos ficheiros Empresa.cpp e Empresa.hpp.  É esta a classe responsável por fazer a leitura e a escrita para os ficheiros e que guarda estas informações em três estruturas do tipo *vector*, sendo elas jurados, candidatos e sessoes. São estes vetores que vão sendo atualizados durante o decorrer do programa, em todas as ações que envolvem acessos a estas bases de dados e, no final, ao fechar o programa são todas guardadas nos respetivos ficheiros. A classe Empresa é também responsável por imprimir na consola todos os jurados, todos os candidatos, todas as sessões, ou ainda as sessões detalhadas. Esta ação é verificada quando o utilizador escolhe estas opções no menu que o programa lhe apresenta.

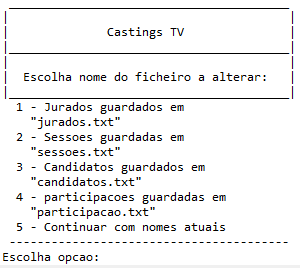
Os ficheiros que servem de base de dados têm um formato predefinido. A seguir apresenta-se um exemplo de um ficheiro de candidatos:



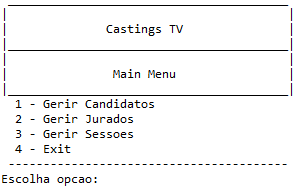
Os métodos *escolher\_candidato(), escolher\_sessao(), escolher\_jurado(), adicionar\_candidato(), adicionar\_jurado(), adicionar\_sessao(), remover\_jurado(), remover\_sessao(), adicionar\_candidato\_sessao(), remover\_candidato\_sessao(), gerarPrimeiraFase(), gerarSegundaFase() , alterarDataSessao()* encontram-se definidos nesta mesma classe também. Os métodos têm nomes auto explicativos de forma a que o código se torne fácil de compreender e de reutilizar.

Estes métodos existem de forma a que a plataforma se apresente com os menus que cumpram as funcionalidades do programa. Os menus estão definidos nos ficheiros menus.cpp e menus.hpp. A plataforma tem cinco menus essenciais divididos da seguinte forma: um menu inicial em que se permite ao utilizador escolher outros ficheiros como base de dados, ou então continuar com os já existentes, um segundo menu que é o principal e que dá acesso aos três seguinte menus: o de gestão de candidatos, o de gestão de sessões e ainda o de gestão de jurados.

De seguida apresenta-se o menu inicial de escolha dos ficheiros:



O menu principal tem o seguinte aspeto:

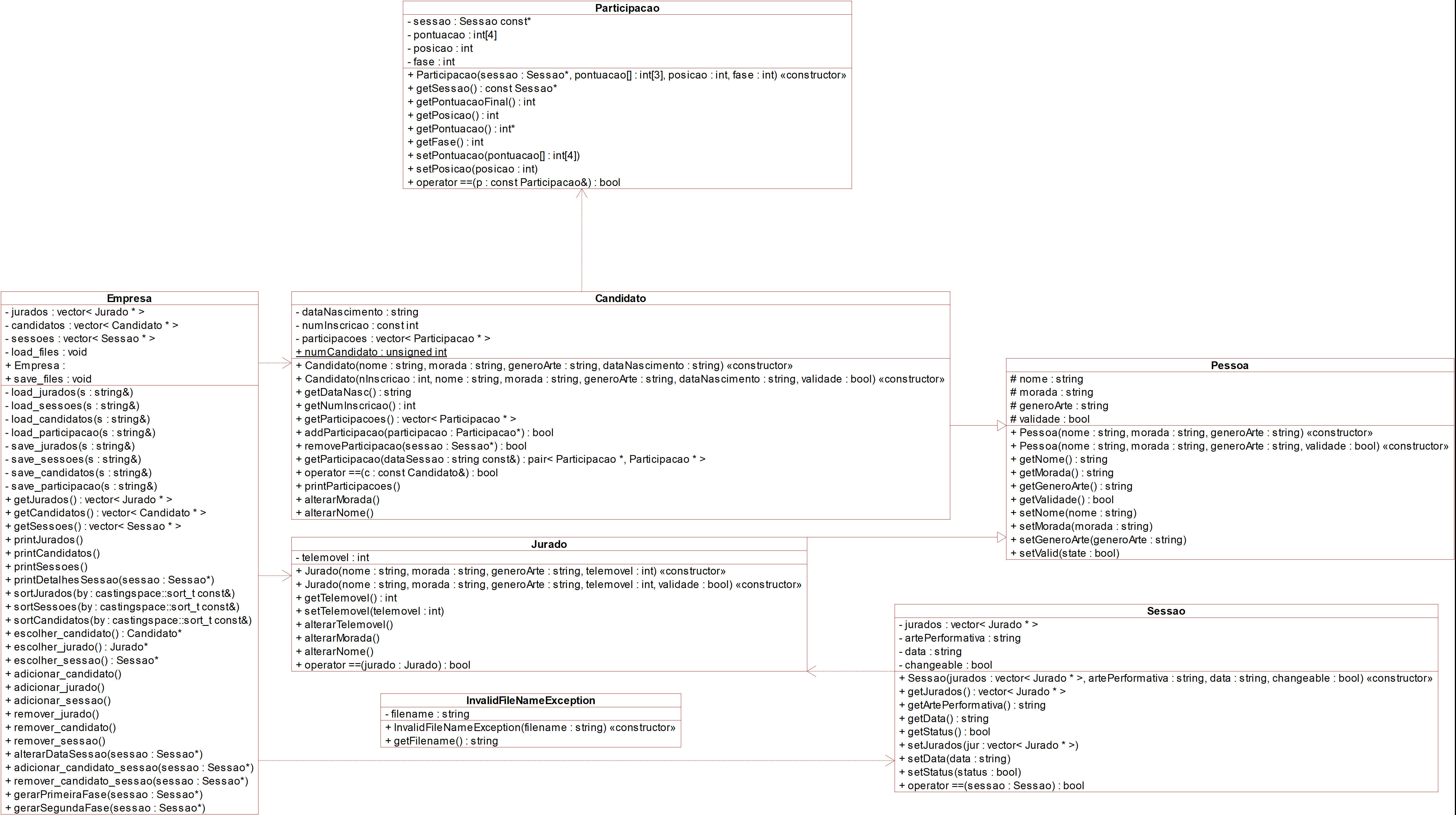


Os ficheiros utils.cpp e utils.hpp são os que contêm os *includes* das bibliotecas necessárias ao bom funcionamento do programa e também alguma funções mais gerais como:

* *read\_number\_Input()* que verifica se o que o utilizador escreveu são caracteres numéricos;
* *clear\_scrn()* que limpa o ecrã,
* *numbr\_size()* que devolve o número de dígitos de um número, e é utilizada, por exemplo na validação da introdução dos números de telemóvel dos jurados;
* *lowerThan()* que faz a comparação entre duas datas, e verifica se a primeira é anterior à segunda;
* *isValidDate()* já anteriormente referida;
* *pressKeyToContinue()* que é a função que coloca o programa em espera, até que o utilizador prima uma tecla;
* a classe *InvalidFileNameException* que é usada para exceções se um dos ficheiros não for encontrado.

Por último o ficheiro main.cpp é o responsável por inicializar todo o programa e por, quando este termina, chamar a função que guardará todos os dados nos ficheiros e também por eliminar a empresa criada no início, de forma a libertar a memória alocada para este efeito.

# Diagramas de UML



# Casos de Utilização do Castings TV

O programa desenvolvido conta com diversas funcionalidades:

* Candidato:
  + Criar, visualizar, editar e eliminar candidatos;
  + Ordenar alfabeticamente por nome, por nome e agrupados por arte ou em ordem crescente de idades e agrupados por arte.
* Jurado
  + Criar, visualizar, editar e eliminar jurados;
  + Ordenar alfabeticamente por nome ou por nome e agrupados por arte.
* Sessão
  + Criar, visualizar e eliminar sessões;
  + Gerar 1ª e 2ª fases;
  + Adicionar e remover candidatos;
  + Alterar data;
  + Ordenar cronologicamente agrupadas por arte.

# Conclusão

Ao longo das semanas destinadas ao desenvolvimento do projeto foram encontradas algumas dificuldades, que tornaram o projeto ainda mais desafiante do que aquilo que se pensava no início. Por exemplo, a interpretação do enunciado gerou algumas dúvidas, e também a gestão de tempo, que a dividir com mais projetos de outras unidades curriculares se veio a tornar escasso. À parte disso, o projeto decorreu como esperado, sendo o resultado final aquilo que o grupo esperava alcançar.

A distribuição do trabalho foi feita de forma equitativa por todos os elementos do grupo.

Em suma, o trabalho foi apresentado cumprindo todas as especificações e o grupo encontra-se motivado para o desenvolver e melhorar aquando do segundo projeto.