# Tema 3: Atribuição de projetos de dissertação

# Unidade curricular:

Concepção e análise de algoritmos

# Título:

Atribuição de projetos de dissertação

# Data:

28 – 4 – 2014

# Elementos do grupo:

Mário Ferreira -> ei12049 – [ei12049@fe.up.pt](mailto:ei12049@fe.up.pt)

Pedro Sousa -> ei12179 – [ei12179@fe.up.pt](mailto:ei12179@fe.up.pt)

Vítor Teixeira ->

# Tema do Problema

O objetivo deste trabalho é construir um programa que proceda à atribuição de projetos de dissertação a alunos seguindo várias regras, as quais passo a enunciar:

1º Fase:

* Tanto os estudantes como as entidades externas (empresas, institutos de investigação, etc.) e os docentes expressam as suas preferências, sendo o número das mesmas variável, desde apenas uma até ao número total de projetos propostos.
* A atribuição imediata dos projetos deve ser feita tendo em atenção as preferências de todos os elementos e utilizando o critério do problema dos casamentos estáveis.
* O número de propostas deve ser no mínimo igual ao número de estudantes.
* Podem ficar estudantes por colocar.

2º Fase:

* Caso o projeto não tenha um supervisor associado é necessário atribuir um docente supervisor.
* Cada docente tem um número máximo de projetos que poderá supervisionar e uma lista de preferências, e mais uma vez a colocação dos mesmos deve ser efetuada tendo em atenção as preferências evidenciadas pelos docentes de forma a maximizar as mesmas.

Em adição aos dados acima expostos o programa deve também ser capaz de ler dados de entrado em formato de ficheiros de texto e apresentar os resultados no standard output.

# Estratégia de resolução do problema

Para a resolução do problema proposto foi escolhida a seguinte solução:

- Construir a classe Pessoa, a classe mãe dos elementos do problema a organizar (Supervisores, Estudantes e Proponentes).

- Contruir ainda a classe Projeto, à qual estaria sempre ligado um proponente.

- As classes Supervisores, Estudantes e Proponentes possuem vetores de preferências, sendo o elemento de índice zero a primeira opção.

- Criar a classe MIEIC aonde estão guardados em vetores todos os elementos previamente carregados a partir de ficheiros de texto e que futuramente irão ser organizados.

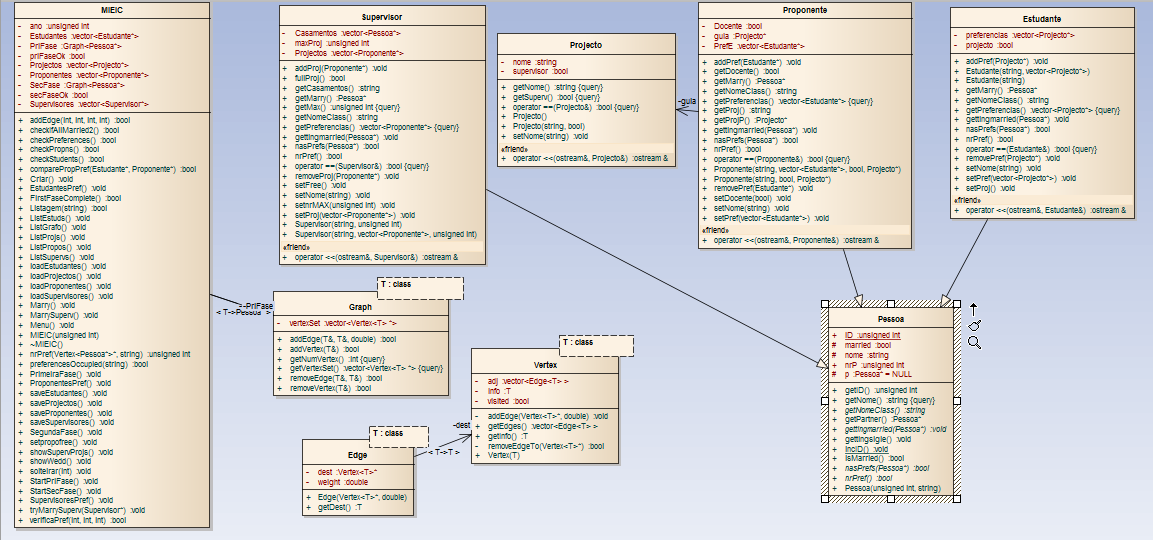
- De seguida inserir esses elementos num grafo também ele elemento da classe MIEIC e organiza-los segundo o método de Gale-Shapley, também conhecido por método dos casamentos estáveis.

- Os dados de entrada serão então os ficheiros de texto, ou os próprios comandos do utilizador que fornecerão ao programa os dados necessários para criar uma base de dados adequada.

# Casos de Utilização

* Construção da base de dados: O caso de utilização é iniciado através da criação da base de dados, seja através de inputs diretos pelo utilizador ou pelo carregamento dos ficheiros de texto. Só depois de verifica a condição de o número de projetos ser superior ao numero de alunos é possível passar para a primeira fase.
* Primeira fase: são atribuídos projetos ao alunos tendo em conta as preferências dos mesmos e as dos proponentes dos vários projetos, usando o método dos casamentos estáveis.
* Segunda fase: caso os proponentes dos projetos não sejam nocentes é necessário atribuir um supervisor. Tal atribuição é feita de forma a que cada supervisor não esteja ligado a um número de projetos superior ao número máximo expresso pelo mesmo. São também tidas em conta as preferências dos supervisores em relação aos projetos

# Diagrama de classes



# Principal algoritmo implementado

O principal algoritmo usado foi o dos casamentos estáveis de Gale-Shapley, que segue o raciocínio expresso no pseudo-código apresentado de seguida:

While( exista um proponente cujo projeto é preferenciado por algum aluno)

P = projeto preferido pelo estudante E

If( P está livre)

P casa-se com E

Else( P já está casado com o Estudante X)

If ( P prefere E a X)

P fica solteiro

X fica solteiro

P e E casam-se

Else

P e X continuam casados

A complexidade do algoritmo original é O(n^2), mas projeto em questão foi necessário acrescentar mais uma condição o que tornou a complexidade O(n^3)

# Principais dificuldades

* Adaptar os algoritmos ao caso do tema proposto
* Encontrar testes diversificados
* Analisar e detetar os vários casos possíveis no decorrer do programa