UNIVERSIDADE DE COIMBRA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
***Departamento de Engenharia Informática***

# Programação Orientada aos Objetos

Projeto 2016/17
**Gestão de Exames no DEI**

**Datas de Entrega:**
1ª meta: 5 de Novembro
Entrega final: 15 de Dezembro

## Descrição do Problema

O DEI está a planear desenvolver uma nova aplicação para a gestão dos Exames (salas, inscrições, vigilâncias, etc.).

Um Exame é caracterizado pelos seguintes elementos: disciplina, data e hora, duração, sala, docente responsável, lista de vigilantes (que devem ser docentes), lista de funcionários não docentes de apoio à realização do exame, lista de alunos inscritos e respetivas classificações obtidas.

Esta aplicação tem em consideração todas as pessoas envolvidas no processo: funcionários (docentes e não docentes) e alunos. Qualquer pessoa é caracterizada pelo nome e e-mail.

Os docentes têm um número mecanográfico, categoria (ex.: assistente, auxiliar, associado ou catedrático) e área de investigação (ex.: sistemas de informação, comunicação e telemática, engenharia de software). Os funcionários não docentes têm um número mecanográfico, categoria (ex.: assistente operacional, assistente técnico, técnico superior, técnico de informática ou especialista de informática) e cargo (ex.: secretaria, financeiro, apoio técnico). Os alunos têm um número de aluno, curso, ano de matrícula no curso (1.º, 2.º, …), regime (normal, trabalhador-estudante, atleta, dirigente associativo ou aluno de Erasmus).
Uma disciplina tem um nome, docente responsável, outros docentes e lista de alunos inscritos. O curso tem um nome, duração (em anos), grau que confere (Licenciatura, Mestrado, Doutoramento) e lista de disciplinas que compõem o curso.

Os exames podem ser de três tipos: época normal, de recurso e época especial. Os exames de época normal e recurso estão acessíveis todos os alunos. Os exames de época especial estão acessíveis apenas a alunos com estatutos de trabalhador-estudante, atletas, dirigentes associativos ou alunos que frequentem o último ano do curso (corresponde ao 3º ano de matrícula).

O seu programa deverá permitir gerir todos os dados que compõem a aplicação e dar resposta especificamente às seguintes operações:

1. Criar Exames e associar docentes da disciplina à lista de vigilantes, assegurando que nenhum docente terá vigilâncias sobrepostas;
2. Configurar a sala do exame e assegurar que não existe mais nenhuma marcação para essa sala nesse horário;
3. Convocar vigilantes e funcionários – associar docentes à vigilância de exames, assegurando que não têm vigilâncias sobrepostas;
4. Inscrever alunos – associar alunos ao exame, assegurando que estão inscritos na disciplina e que têm acesso à época do exame em causa;
5. Lançar notas de um exame;

6. Listar Exames – Época, disciplina, data, hora, duração, sala, número de vigilantes convocados e de alunos inscritos;
7. Listar alunos inscritos num exame e classificações obtidas, caso existam;
8. Listar exames em que um aluno está inscrito e classificações obtidas, caso existam;
9. Listar docentes e funcionários associados a um exame;
10. Listar exames em que um docente ou funcionário está envolvido;
11. Listar notas de um exame.

Todos os dados da aplicação (ex.: alunos, funcionários, docentes, cursos e disciplinas) deverão ser guardados em ficheiros de texto, e carregados para as estruturas de dados adequadas no arranque do programa.

A interação com o utilizador deverá ser realizada através de uma consola em modo de texto para permitir a entrada de dados e a apresentação de resultados. Os resultados (ex.: listagens e pesquisas) deverão, também, poder ser armazenados em ficheiros de texto (a pedido do utilizador).

## Implementação

A aplicação deve ser implementada na linguagem Java e deverá ter em conta os princípios de programação orientada a objetos e toda a matéria dada na disciplina, salientando os seguintes aspetos:

1. Ao entrar na aplicação, devem persistir todos os dados que se encontravam no fim da última utilização (incluindo exames, vigilantes associados, alunos inscritos, notas, etc.).
2. Cada classe deve gerir internamente os seus dados, pelo que deverá cuidar da proteção das suas variáveis e métodos;
3. Cada objeto deverá ser responsável por uma tarefa ou objetivo específico, não lhe devendo ser atribuídas funções indevidas;
4. Utilize a *keyword* **static** apenas quando tal se justifique e não para contornar erros do compilador.

Elabore um diagrama com as suas classes e objetos (em UML), antes de iniciar a implementação, para prever a estrutura do projeto.

Tenha ainda em conta os seguintes pontos que serão importantes na avaliação:

1. Comentar as classes, métodos e variáveis públicas segundo o formato Javadoc. Isto permitir-lhe-á gerar automaticamente uma estrutura de ficheiros HTML, descritivos do seu código, que deve incluir no seu relatório;
2. Comentar o restante código sempre que a leitura dos algoritmos não seja óbvia;
3. Tal como sugerido acima, evitar o uso abusivo de **static** e de variáveis e métodos **public**;
4. Na escolha de nomes para variáveis, classes e métodos, seguir as convenções adotadas na linguagem **Java**;
5. Na organização das classes deverá ser evitada a redundância dos dados.

## Prazos de entrega

A entrega do trabalho compreende duas metas distintas:
**Meta 1** (1 valor) – Análise do projeto e diagrama de classes (entrega até **5 de Novembro**);
**Meta 2** (4 valores) – Versão final (entrega até **15 de Dezembro**).
Os trabalhos serão comparados (tanto entre os trabalhos da disciplina como com código disponível na Internet), no sentido de detetar eventuais fraudes por cópia. Nos casos em que se verifique que houve cópia de trabalho total ou parcial, os grupos envolvidos terão os projetos anulados, reprovando à disciplina.

## Material a entregar

**Cada grupo deve entregar obrigatoriamente:**

**Meta 1**: Diagrama de classes em UML

1. Upload em pdf do diagrama de classes no InforEstudante;
2. Entrega de uma cópia impressa do diagrama no momento da defesa na aula teórico-prática.

**Meta 2**: Aplicação

1. Upload de zipFile no InforEstudante com:
   - Todas as classes .java;
   - Executável (JAR);
   - Ficheiros de dados para teste;
   - JavaDoc;
   - Relatório (em formato pdf).
2. Relatório em papel, entregue no cacifo do docente da turma teórico-prática a que os alunos pertencem, descrevendo o programa do ponto de vista técnico e que deve incluir:
   - Estrutura geral do programa;
   - Diagramas de classes inicial e final;
   - Descrição das principais estruturas de dados e de ficheiros usados;
   - Breve explicação de como o programa se executa.

## Avaliação do trabalho

Para a avaliação do trabalho contam fatores de dois tipos:

- Caixa preta (tal como é percecionado pelo utilizador):
  - Conjunto de funcionalidades implementadas;
  - Robustez do programa;
  - Qualidade da interface.
- Caixa branca (a forma como está construído):
  - Qualidade das soluções técnicas encontradas para os problemas em causa;
  - Estruturação do código;
  - Qualidade dos comentários.

A avaliação em cada uma das metas é feita individualmente, independentemente dos grupos serem constituídos por 2 alunos. Aos 5 valores atribuídos ao projeto, acrescidos dos 3 valores de avaliação contínua nas aulas, estão associados mínimos de 40%. Os alunos que tenham um resultado de avaliação (projeto + aulas) inferior a 40% não serão admitidos ao exame da disciplina.

***Nota**: Não se aceitam trabalhos que apresentem erros de compilação no momento da defesa e que não estejam corretamente estruturados do ponto de vista da Programação Orientada aos Objetos.*

## Composição dos grupos

O trabalho deve ser realizado em grupos de 2 elementos, tendo os elementos do grupo que pertencer à mesma turma teórico-prática. A defesa dos trabalhos é feita individualmente, bem como a respetiva avaliação.

## Defesa final do trabalho

O trabalho deve ser defendido através de uma discussão presencial e individual. Para isso, cada grupo deve inscrever-se num horário que esteja disponível para essa defesa no InforEstudante.