Instituto Superior
de Engenharia
Politécnico de Coimbra

Programação Avançada

2021/2022

Enunciado do Trabalho Prático

Nota prévia: *O enunciado poderá não estar completo em alguns pontos. Os alunos deverão avaliar as opções e discuti-las com um docente da unidade curricular. As decisões tomadas deverão ser sucintamente explicadas no relatório.*

Pretende-se que desenvolva, em linguagem *Java*, uma aplicação de apoio ao processo de gestão de projetos e estágios do Departamento de Engenharia Informática e de Sistemas do ISEC. A aplicação será alimentada com diversos tipos de dados consoante a fase em que se encontra o processo de gestão: (1) recolha de propostas; (2) classificação dos estudantes matriculados na UC de PoE com vista à sua seriação (i.e., um valor decimal compreendido entre 0.0 e 1.0) e informação acerca dos alunos terem a possibilidade de realizar estágio ou, pelo contrário, apenas terem acesso a projetos; (3) candidaturas, em que os estudantes indicam quais os projetos ou estágios que pretendem e qual a ordem de preferência; e (4) informação acerca dos docentes que farão a orientação de cada projeto ou estágio.

As interfaces com o utilizador, detalhadas mais adiante, devem permitir a introdução dos dados referidos atrás, manualmente (teclado e rato) e por leitura de informação disponibilizada em ficheiros no formato CSV (*comma-separated values*)[1].

O código fonte da aplicação pretendida deve apresentar uma divisão clara entre o modelo e a interface com o utilizador. O modelo não deve incluir qualquer tipo de interação com o utilizador e a interface com o utilizador não deve incluir qualquer tipo de lógica de negócio. A estrutura do projeto, em termos de *packages*, deve refletir esta separação, conforme explicado mais à frente. A aplicação deve ser desenvolvida seguindo o padrão *State* (*FSM - Finite-State Machine*), segundo uma abordagem polimórfica e organização semelhante à apresentada durante as aulas. Para o efeito, considera-se a existência de fases distintas no processo de gestão dos projetos e estágios, cada uma possibilitando um leque específico de funcionalidades (note que a organização da FSM não é necessariamente um decalque *exato* do processo de gestão explicado a seguir).

- **Fase 1 – Configuração**

As funcionalidades disponíveis na fase de configuração dizem respeito à **gestão de alunos**, **docentes** e **propostas de estágio ou projeto**:

  - Alternar entre os modos de gestão de alunos, docentes e propostas de estágios ou projetos. Quando a aplicação está num destes modos não deve disponibilizar as

[1] Mais informação sobre o formato CSV em https://docs.fileformat.com/spreadsheet/csv/

funcionalidades relativas aos outros. Por exemplo, quando a aplicação está no modo de gestão de alunos não deve permitir operações relativas à gestão de docentes ou propostas.

- Inserção, consulta, edição e eliminação dos dados referentes a **alunos** matriculados na UC de PoE (modo de gestão de alunos), sendo estes caracterizados por: número de estudante (*long*); nome; endereço de email; sigla do curso ("LEI" ou "LEI-PL"); sigla do ramo ("DA", "RAS" ou "SI"); classificação (*double*); possibilidade de aceder a estágios além de projetos (*boolean*).
- Inserção, consulta, edição e eliminação dos dados referentes a **docentes** (modo de gestão de docentes), sendo estes caracterizados por um endereço de email e um nome. Os docentes poderão ter o papel de orientador ou proponente de projeto.
- Inserção, consulta, edição e eliminação dos dados referentes a **propostas** (modo de gestão de propostas), existindo três tipos:
  - T1 - Estágio. Dados a gerir: código de identificação (*String* e único, ex.: "P010"); área(s) de destino ("RAS", "DA" e/ou "SI"); título; identificação da entidade de acolhimento (ex: "ISEC", "Tecnologias do Passado, Lda."). O estágio poderá ter a indicação do aluno (número de aluno) ao qual deve ser atribuído.
  - T2 - Projeto (proposto por docente): Dados a gerir: código de identificação (*String* e único, ex.: "P010"); ramo(s) de destino ("RAS", "DA" e/ou "SI"); título; docente proponente (previamente registado). O projeto poderá ter a indicação do aluno ao qual deve ser atribuído (através de um número do aluno registado).
  - T3 - Estágio/projeto autoproposto por um aluno. Dados a gerir: código de identificação (*String* e único, ex.: "P010"); título; número do estudante proponente (o estágio/projeto fica com uma *atribuição prévia* ao aluno em questão, ou seja, ser-lhe-á automaticamente atribuído o estágio/projecto). Um aluno tem que estar registado previamente e apenas poderá apresentar uma proposta.
- Fechar a fase (bloqueio das operações desta fase, não sendo possíveis futuras alterações, mantendo-se a possibilidade de consultar os dados). Apenas deverá ser permitido o fecho da fase se, para cada ramo, o número total de propostas for igual ou superior ao número de alunos.
- Avançar para a fase seguinte (mesmo sem esta fase estar fechada).

- **Fase 2 – Opções de candidatura**

  As funcionalidades disponíveis na fase de candidaturas são as seguintes:

  - Inserção, consulta, edição e eliminação de candidaturas, as quais correspondem à indicação das propostas pretendidas (códigos de identificação das propostas). As opções devem ser indicadas por ordem de preferência (a mais preferida em primeiro lugar). Não deverão ser permitidas escolhas de projetos ou estágios que tenham uma atribuição prévia de aluno.

- Obtenção de listas de alunos:
  - Com autoproposta.
  - Com candidatura já registada.
  - Sem candidatura registada.
- Obtenção de listas de propostas de projecto/estágio de acordo com os seguintes critérios (podem ser indicados 0 ou mais filtros):
  - Autopropostas de alunos.
  - Propostas de docentes.
  - Propostas com candidaturas.
  - Propostas sem candidatura.
- Fechar a fase (inclui passagem para a fase seguinte e bloqueio das operações desta fase, não sendo possíveis futuras alterações, mantendo-se a possibilidade de consultar os dados). Esta fase não poderá ser fechada, caso a anterior ainda se mantenha em aberto.
- Regresso à fase anterior para consulta de dados ou realizar alterações (caso essa fase não esteja fechada).
- Avançar para a fase seguinte (mesmo sem esta fase estar fechada).

- **Fase 3 – Atribuição de propostas**

  As funcionalidades disponíveis na fase de atribuição de propostas são as seguintes:

  - Atribuição automática das autopropostas ou propostas de docentes com aluno associado (trata-se, no fundo, de confirmar a associação entre aluno e proposta que ficou prevista quando a proposta foi inserida). Esta será a única forma de atribuição permitida, caso a fase anterior ainda esteja aberta.
  - Atribuição automática de uma proposta disponível aos alunos ainda sem atribuições definidas, com base nas suas classificações, restrição de acesso a estágios e opções indicadas no processo de candidatura. Em situação de empate o processo deverá ser interrompido, para solicitar ao utilizador a resolução do conflito. Durante essa fase, o utilizador deverá poder consultar os dados completos dos alunos envolvidos, bem como das propostas incluídas nas candidaturas desses alunos. Resolvido o conflito, deverá ser retomado o processo de atribuição automática.
  - Atribuição manual de propostas disponíveis aos alunos sem atribuição ainda definida.
  - Remoção manual de uma atribuição previamente realizada ou de todas as atribuições (excepto as autopropostas ou propostas de docentes com aluno associado). É eliminada a associação *aluno-proposta*, mas não são eliminados os dados do aluno nem os dados da proposta.
  - A gestão manual deverá permitir operações de *undo* e *redo*.
  - Obtenção de listas de alunos (incluindo indicação dos seus dados) que respeitam um dos seguintes critérios:
    - Têm autoproposta associada.
    - Têm candidatura já registada.

        - Têm proposta atribuída (com indicação de qual a ordem de preferência, sendo uma autoproposta interpretada como ordem "1").
        - Não têm qualquer proposta atribuída.
    - Obtenção de listas de propostas de projecto estágio de acordo com os seguintes critérios (podem ser indicados 0 ou mais filtros):
        - Autopropostas de alunos.
        - Propostas de docentes.
        - Propostas disponíveis (não atribuídas a alunos).
        - Propostas atribuídas.
    - Fechar a fase (inclui passagem para a fase seguinte e bloqueio das operações desta fase, não sendo possíveis futuras alterações, mantendo-se a possibilidade de consultar os dados). Apenas será possível fechar esta fase se todos os alunos com candidaturas submetidas possuírem projeto atribuído.
    - Regresso à fase anterior para consulta de dados ou realizar alterações (caso essa fase não esteja fechada).
    - Avançar para a fase seguinte (mesmo sem esta fase estar fechada).

- **Fase 4 – Atribuição de orientadores**

  As funcionalidades disponíveis na fase de atribuição de orientadores são as seguintes:

    - Associação automática dos docentes proponentes de projetos como orientador dos mesmos. Tal como no caso da atribuição automática de alunos às suas autopropostas, trata-se de confirmar a associação proposta-docente que ficou prevista na altura da criação da proposta.
    - Atribuição, consulta, alteração e eliminação de um orientador do ISEC (docente) aos alunos com propostas atribuídas.
    - A gestão da atribuição manual de orientações deve permitir operações de *undo* e *redo*.
    - Obtenção de dados diversos sobre atribuição de orientadores, incluindo:
        - lista de estudantes com proposta atribuída e com orientador associado.
        - lista de estudantes com proposta atribuída mas sem orientador associado.
        - número de orientações por docente, em média, mínimo, máximo, e por docente especificado.
    - Fechar fase (inclui passagem para a fase seguinte e bloqueio das operações desta fase, não sendo possíveis futuras alterações)
    - Regresso à fase anterior para consulta de dados ou realizar alterações (caso essa fase não esteja fechada)..

- **Fase 5 – Consulta**

Atingida esta fase não será possível regressar a qualquer uma das fases anteriores. As funcionalidades disponíveis na fase de consulta são as seguintes:

  - Obtenção de dados diversos sobre todo o processo:
    - lista de estudantes com propostas atribuídas.
    - lista de estudantes sem propostas atribuídas e com opções de candidatura.
    - conjunto de propostas disponíveis.
    - conjunto de propostas atribuídas.
    - número de orientações por docente, em média, mínimo, máximo, e por docente especificado.
    - outros dados (poderão ser incluídas outras listagens que sejam consideradas pertinentes pelo grupo).

A aplicação não deve permitir a definição de propostas com o mesmo código ou previamente associadas ao mesmo estudante (número de aluno). Também não deverá ser permitida a existência de docentes e alunos com endereços de e-mail iguais, nem alunos com números repetidos. Estes campos, que servem de identificador para as diversas entidades, não podem ser passíveis de alteração (por exemplo, quando se edita um docente apenas se podem efectuar correções ao seu nome. Também devem ser identificadas e tratadas situações de inconsistência (ex.: opção de candidatura sem correspondência na lista de propostas).

Para além das diversas listagens e possibilidades de informação indicadas para as diversas fases, poderão ser incluídas outras consultas que sejam consideradas relevantes.

As interfaces com o utilizador desenvolvidas, tanto em modo texto (primeira meta) como gráfico (segunda meta), devem possibilitar:

- Fase 1 – importar a partir de ficheiro CSV os dados de alunos inscritos na UC de PoE, de propostas de projeto e estágio e de docentes. Também deve ser possível exportar esta informação para ficheiros CSV;
- Fase 2 – importar a partir de ficheiro CSV os dados referentes às opções indicadas pelos estudantes nas suas candidaturas. Também deve ser possível exportar esta informação para ficheiros CSV.
- Fase 3 – exportar para ficheiro CSV os dados referentes aos alunos inscritos na UC de PoE, incluindo opções de candidatura, proposta atribuída e ordem desta nas suas opções de candidatura;
- Fases 4 e 5 – exportar para ficheiro CSV os dados referentes aos alunos inscritos na UC de PoE, incluindo opções de candidatura, proposta atribuída, ordem desta nas suas opções de candidatura e orientador atribuído.

A importação referida nos pontos anteriores corresponde à inserção de novos elementos sem eliminação dos eventualmente já existentes, devendo ser observadas as restrições quanto à não duplicação de

códigos, endereços de email, atribuições prévias de propostas a alunos, etc., já anteriormente referidas. Deverão ser disponibilizadas opções para eliminar todos os dados, permitindo a especificação da operação por tipos (alunos, propostas, docentes, ...). Todas as operações devem garantir a consistência dos dados (um exemplo entre outros que deverá ser identificado: eliminação de um aluno que realizou uma autoproposta - neste caso a proposta também deverá ser eliminada).

Não é admitido o recurso a bibliotecas externas à Java API ou JavaFX para implementação das funcionalidades da aplicação (por exemplo, bibliotecas para ler/escrever ficheiros CSV). A utilização de bibliotecas externas implicará uma forte penalização no trabalho.

Regras gerais

O trabalho deve ser realizado por grupos de dois elementos, de acordo com as propostas de grupo registadas no Nónio. A realização individual é permitida, mas não aconselhada.

A elaboração do trabalho está dividida em duas metas separadas.

As datas de entrega do trabalho nas duas metas são as seguintes:

- Primeira meta, que inclui o modelo da aplicação pretendida utilizando os padrões apresentados nas aulas e interface com o utilizador em modo texto (consola): **2 de maio (8h00)**;
- Segunda meta, correspondente à funcionalidade completa da aplicação, incluindo uma interface com o utilizador em modo gráfico baseada em *JavaFX*: **20 de junho (8h00)**.

Mais à frente são dados mais pormenores acerca dos requisitos a cumprir em cada meta.

As entregas correspondentes às duas metas do trabalho devem ser feitas através do *Nónio-InforEstudante* num ficheiro compactado no formato **ZIP** e apenas neste formato. O nome deste ficheiro deve obrigatoriamente incluir o primeiro nome, o último nome e o número de estudante (do Nónio) dos elementos do grupo.

Exemplo: `Antonio-Ferreira-202006104_Nelson-Pacheco-202007205.zip`)

O ficheiro **ZIP** deve conter, pelo menos:

- O projeto com todo o código fonte produzido (projeto *IntelliJ* ou *Visual Studio Code*, sem diretórios de *output* da compilação: `bin`, `out` ou similares).
- Eventuais ficheiros adicionais de dados e recursos auxiliares necessários à execução do programa, caso não estejam incluídos nos diretórios do projeto referidos no ponto anterior).
- O relatório em formato **PDF**.

O relatório deve incluir em ambas as fases:

- Uma descrição sintética acerca das opções e decisões tomadas na implementação (máximo de uma página).

- O diagrama da máquina de estados que controla o apoio ao processo de gestão da UC de PoE, devidamente explicado; neste diagrama, o nome atribuído às transições de estado deve corresponder ao nome dado às funções da hierarquia de estados a que correspondem e o nome dos estados deve também corresponder ao nome das classes que os representam.
- Diagramas de outros padrões de programação que tenham eventualmente sido aplicados no trabalho.
- Descrição sucinta das classes utilizadas no programa (o que representam e os objetivos).
- Descrição sucinta do relacionamento entre as classes (podem ser usados diagramas UML).
- Em ambas as metas, para cada funcionalidade da aplicação, a indicação de cumprido/implementado totalmente ou parcialmente (especificar o que foi efetivamente cumprido neste caso) ou não cumprido/implementado (especificar a razão). O uso de uma tabela pode simplificar a elaboração desta parte.

A primeira meta será sujeita a defesa para 1/3 dos grupos. Na segunda meta todos os trabalhos serão sujeitos a defesa, as quais incluem a apresentação, explicação e discussão do trabalho apresentado, podendo ainda incluir a realização de alterações ao que foi entregue.

À primeira meta do trabalho corresponde um coeficiente (fator multiplicativo) igual a **0.8** ou **1.0** e à segunda fase **12** valores**.** A classificação final será o resultado da multiplicação do coeficiente da primeira meta com o resultado obtido na segunda.

Objetivos e requisitos

Os objetivos das duas metas do trabalho prático são os seguintes:

- **Primeira meta**:
    - Implementação das 5 fases da aplicação em modo texto, sem as operações de *undo* e *redo* mencionadas. A introdução dos dados é realizada apenas por importação de informação a partir de ficheiros CSV, não sendo necessário disponibilizar as funcionalidades de edição e remoção de dados.
    - Gravação/carregamento do estado da aplicação usando um formato binário (deverá incluir toda a informação necessária à continuação do normal funcionamento da aplicação após a sua interrupção).

- **Segunda meta** (adicionalmente ao implementado na primeira meta):
    - Inclusão das operações de *undo* e *redo*. Não é necessário manter a possibilidade de fazer *undo*/*redo* após interrupção da aplicação (“*sair e voltar a entrar*”).
    - Implementação da aplicação com uma interface com o utilizador (IU) em modo gráfico (*JavaFX*).
    - Introdução e gestão de dados realizada de forma interactiva na IU (mantendo-se a possibilidade de importar os dados a partir de ficheiros CSV).

- No contexto da última fase, deverá permitir a apresentação de gráficos de resumo do processo, incluindo gráficos circulares (i.e., “queijo”, “pizza”, “pie”, …) para visualizar a distribuição dos estágios/projetos por ramos (“DA”,”RAS” e “SI”), percentagem e valores absolutos relativos às propostas atribuídas e não atribuídas, e gráficos de barras com as empresas com mais estágios (*top 5*) e docentes com mais orientações (*top 5*).
- A interface com o utilizador em modo gráfico deve permitir ter um painel de resumo da entidade que se encontra a ser gerida em cada momento. Por exemplo, quando se faz a gestão das candidaturas deverá existir um painel que informe sobre o número total de alunos, número de alunos com candidatura, etc. A atualização deste tipo de painel, bem como outras atualizações de informação, deverão estar de acordo com o padrão de notificações assíncronas estudado nas aulas (deve obrigatoriamente ser usado *JavaFX*).
- Soluções básicas não significativamente distintas de uma interface com o utilizador em modo texto serão penalizadas. Espera-se que as interfaces com o utilizador sejam apelativas, intuitivas e funcionais.
- Código devidamente comentado usando *JavaDoc*.

A implementação do trabalho deve ainda respeitar os seguintes requisitos:

- Devem ser seguidos os padrões apresentados nas aulas.
- Deve ser aplicado, de forma adequada, o padrão *State* (“máquina de estados”) para concretizar a lógica de controlo do processo de gestão dos projetos e estágios.
- A aplicação deve apresentar toda a informação necessária ao acompanhamento e verificação do bom funcionamento da aplicação.
- Em ambas as fases do trabalho, o código deve ser estruturado de maneira que a lógica não dependa da forma de interação com o utilizador (na lógica não pode ser pedida ou mostrada informação ao utilizador). Nas classes relacionadas com a interação com o utilizador não pode haver processamento de regras nem alteração direta dos dados internos da lógica.

Estrutura de projeto

A aplicação deve estar organizada em *packages*, incluindo os seguintes (podem existir outros):

- ***pt.isec.pa.apoio_poe*** – *package* que abrange toda a aplicação.
- ***pt.isec.pa.apoio_poe.model*** – tem a classe que constitui a fachada de acesso à lógica, que internamente distribui as responsabilidades que lhe são pedidas, gerindo a dinâmica de processamento através de uma máquina de estados. Esta classe apenas existirá na segunda meta. Na primeira meta a interface deverá aceder às funcionalidades através da classe *Context* da FSM.
- ***pt.isec.pa.apoio_poe.model.fsm*** – contém a hierarquia dos estados.

- ***pt.isec.pa.apoio_poe.model.data*** – contém as classes que representam as estruturas de dados e que disponibilizam toda a funcionalidade.
- ***pt.isec.pa.apoio_poe.ui.text*** – classe(s) que implementa(m) a interface em modo texto.
- ***pt.isec.pa.apoio_poe.ui.gui*** – classes que implementam a interface em modo gráfico em *JavaFx* (apenas na segunda meta).

Exemplo do formato dos ficheiros CSV

Exemplo de ficheiro CSV com dados dos alunos:

```
2022987654,Carlos Picoto,a2022987654@isec.pt,LEI-PL,SI,0.123,true
2015123456,Fernanda Gaspar,a2015123456@isec.pt,LEI,DA,0.37,true
2019999999,Mário Tavares,a2019999999@isec.pt,LEI,DA,0.37,false
2020111111,Rute Miranda,a2020111111@isec.pt,LEI,RAS,0.37,true
```

Exemplo de ficheiro CSV com dados dos docentes:

```
Álvaro Santos,ans@isec.pt
João Durães,jduraes@isec.pt
José Marinho,fafe@isec.pt
Ricardo Silva,rmc.silva@isec.pt
Sarah Cunha,sarah.cunha@isec.pt
```

Exemplo de ficheiro CSV com dados dos projetos (nota: assume-se que não existem vírgulas no contexto das Strings; caso queiram suportar a vírgula, poderão obrigar a que as Strings onde isso acontece sejam indicadas entre aspas “”):

```
T1,P010,DA,Aplicação para gestão de condomínios,Caves e Sótãos.Lda
T2,P027,RAS|SI|DA,Shared Wallet,ans@isec.pt,2022987654
T3,P007,Comunicação através de sinais de fumo,2020111111
T2,P023,DA,Registo de presenças e ausências,jduraes@isec.pt
T1,P031,DA|SI,Controlo de estacionamento,Rotundas.SA
```

Exemplo de ficheiro CSV com dados das candidaturas:

```
2015123456,P010,P031,P023
2019999999,P010,P023
```