# Programação com Objectos/Projecto de Programação com Objectos/Enunciado do Projecto de 2019-2020

*From Wiki**3*

< Programação com Objectos | Projecto de Programação com Objectos

**AVISOS - Avaliação em Época Normal** **[Collapse]**

Esclarecimento de dúvidas:

- Consultar sempre o corpo docente atempadamente: presencialmente ou através do endereço oficial da disciplina [1].
- Não utilizar fontes de informação não oficialmente associadas ao corpo docente (podem colocar em causa a aprovação à disciplina).
- Não são aceites justificações para violações destes conselhos: quaisquer consequências nefastas são da responsabilidade do aluno.

Requisitos para desenvolvimento, material de apoio e actualizações do enunciado (ver informação completa em Projecto de Programação com Objectos):

- O material de apoio é de uso obrigatório e não pode ser alterado.
- Verificar atempadamente (mínimo de 48 horas antes do final de cada prazo) os requisitos exigidos pelo processo de desenvolvimento.

Processo de avaliação (ver informação completa em Avaliação do Projecto):

- Datas: **2019/10/18 12:00** (inicial); **2019/11/18 12:00** (intercalar); **2019/12/09 12:00** (final); **2019/12/16-2019/12/20** (teste prático).
- **A entrega inicial, sendo crucial para o projecto, é obrigatória e sua não realização implica a exclusão da avaliação do projecto e, por consequência, da avaliação da disciplina.**
- Verificar atempadamente (até 48 horas antes do final de cada prazo) os requisitos exigidos pelo processo de avaliação, incluindo a capacidade de acesso ao repositório CVS.
- **Apenas se consideram para avaliação os projectos existentes no repositório CVS oficial.**
- Trabalhos não presentes no repositório no final do prazo têm classificação 0 (zero) (não são aceites outras formas de entrega). Não são admitidas justificações para atrasos em sincronizações do repositório. A indisponibilidade temporária do repositório, desde que inferior a 24 horas, não justifica atrasos na submissão de um trabalho.
- A avaliação do projecto pressupõe o compromisso de honra de que o trabalho correspondente foi realizado pelos alunos correspondentes ao grupo de avaliação.
- **Fraudes na execução do projecto terão como resultado a exclusão dos alunos implicados do processo de avaliação.**

**Material de Uso Obrigatório** **[Collapse]**

As bibliotecas **po-uuilib** e o conteúdo inicial do CVS são de **uso obrigatório**:

- po-uuilib (classes de base) po-uuilib-201708311009.tar.bz2 (não pode ser alterada) - javadoc
- m19-core (classes do "core") (via CVS) (deve ser completada -- os nomes das classes fornecidas não podem ser alterados)
- m19-app (classes de interacção) (via CVS) (deve ser completada -- os nomes das classes fornecidas não podem ser alterados)

A máquina virtual, fornecida para desenvolvimento do projecto, já contém todo o material de apoio.

**Uso Obrigatório: Repositório CVS**

**Apenas se consideram para avaliação os projectos existentes no repositório CVS oficial**.
Trabalhos não presentes no repositório no final do prazo têm classificação 0 (zero) (não são aceites outras formas de entrega). Não são admitidas justificações para atrasos em sincronizações do repositório. A indisponibilidade temporária do repositório, desde que inferior a 24 horas, não justifica atrasos na submissão de um trabalho.

Contents



O objectivo do projecto é desenvolver um sistema para gerir o acervo de uma mediateca. O sistema deverá permitir, entre outras operações, (i) fazer pesquisas de obras; (ii) registar dados de utentes; (iii) registar dados de obras; e (iv) registar requisições de obras para consulta domiciliária.

Neste texto, o tipo **negrito** indica um literal (i.e., é exactamente como apresentado); o símbolo ␣ indica um espaço; e o tipo *itálico* indica uma parte variável (i.e., uma descrição).

# Conceitos e Relações do Modelo

Existem vários conceitos importantes neste contexto: obras e suas categorias, utentes, requisições e tempo.

Os conceitos listados não são os únicos possíveis no modelo e as suas relações (assim como relações com outros conceitos não mencionados) podem depender das escolhas do projecto.

## Obras e Categorias

O sistema mantém um registo de obras da mediateca. Cada obra é identificada por um número de obra. O identificador é atribuído automaticamente e incrementalmente (a partir de 0 ou do último valor atribuído, caso o estado do sistema tenha sido recuperado). As obras registam ainda o número de exemplares existentes no acervo da mediateca (várias cópias da mesma obra). Todas as obras têm um título (cadeia de caracteres) e um preço (número inteiro, por simplicidade).

Cada obra tem uma categoria, de acordo com o assunto nela tratado. Inicialmente, consideram-se as seguintes categorias: (i) obras de referência: onde se incluem dicionários, gramáticas, enciclopédias e documentários; (ii) obras de ficção; e (iii) obras técnicas e científicas. Deve ser possível criar novas categorias, com um impacto mínimo sobre o sistema desenvolvido.

As obras a considerar inicialmente são livros e DVDs. As propriedades específicas de cada um (além das gerais) são as seguintes:

- Livros – O sistema deverá manter, para cada livro, a seguinte informação: autor (apenas um, por simplicidade), e ISBN (cadeia com dez caracteres).
- DVDs – Para cada DVD, o sistema deverá manter: realizador (apenas um, por simplicidade), e o número de registo na IGAC (Inspecção-Geral das Actividades Culturais) (cadeia de caracteres).

Deve ser possível criar novos tipos de obras. O impacto da introdução dos novos tipos na implementação desenvolvida deve ser mínimo.

Existem inicialmente três categorias de obra: obras de referência, obras de ficção; e obras técnicas e científicas. Deve ser possível adicionar novas categorias com impacto mínimo na aplicação já desenvolvida.

# Utentes

O sistema mantém um registo de utentes da mediateca. Cada utente é identificado por um número de utente. O identificador é atribuído automaticamente e incrementalmente (a partir de 0 ou do último valor atribuído, caso o estado do sistema tenha sido recuperado).

O sistema mantém ainda, para cada utente, o seu nome, endereço de correio electrónico. Estes campos não podem ser vazios. É ainda mantida informação sobre a situação do utente perante a mediateca: (i) **activo**, i.e., o utente pode fazer requisições; (ii) **suspenso**, i.e., o utente não pode fazer novas requisições.

Um utente é suspenso se não devolver uma obra requisitada dentro do prazo estipulado; permanece suspenso até devolver a obra e pagar a multa referente ao atraso na entrega.

A mediateca distingue entre utentes que cumprem as regras de funcionamento e utentes que violam sistematicamente os compromissos. Existe ainda uma classificação intermédia para novos utentes ou para utentes com comportamento misto.

Um utente que nas últimas 3 requisições não tenha cumprido os prazos de devolução, é classificado como faltoso. Um utente faltoso que proceda a 3 devoluções consecutivas dentro do prazo é considerado normal. Um cliente que tenha cumprido rigorosamente os prazos de entrega nas últimas 5 requisições é classificado como cumpridor. Em todos os outros casos, o utente não tem classificação especial e é considerado normal. Como se verá adiante, a classificação influencia a capacidade de requisição do utente.

O comportamento (no que diz respeito a entrega, dentro ou fora do prazo, de obras requisitadas antes da suspensão) de um utente suspenso influencia a sua classificação após o pagamento da multa. Por exemplo, se, antes de pagar a multa, o utente entregar 3 obras dentro do prazo e regularizar a sua situação, então passará a ser um utente que pode efectuar requisições com a classificação de normal. Deve ser possível discriminar os utentes quanto à sua conduta, considerando outros critérios ou novas classificações, com um impacto mínimo na implementação desenvolvida.

# Requisições

O sistema garante o cumprimento de regras para a requisição de obras. As regras dependem das características da obra que se pretende requisitar e da conduta passada do utente. Utentes cumpridores estão sujeitos a regras permissivas enquanto que a utentes faltosos são aplicadas regras restritivas. Os restantes utentes seguem um conjunto de regras de base. As regras gerais a respeitar pelos utentes são:

1. Não pode requisitar duas vezes a mesma obra (i.e., em duas requisições diferentes e simultâneamente abertas).
2. Não pode requisitar obras um utente que esteja suspenso;
3. Não pode requisitar obras cujos exemplares tenham sido já todos requisitados;
4. Não pode ter mais que *n* obras requisitadas em cada momento (valor base: 3; utentes cumpridores: 5; utentes faltosos: 1);
5. Não pode requisitar obras de referência;
6. Não pode requisitar obras com um preço superior a €25,00 (não aplicável a utentes cumpridores);

No caso de violação da regra 3, o utente pode pedir para ser notificado assim que algum exemplar seja devolvido. A notificação consiste na indicação de que a obra já se encontra disponível.

Ao requisitar uma obra, o utente deve ser informado da data limite para a devolução. O tempo de requisição permitido para cada obra depende do número total de exemplares que constem do acervo da mediateca e da conduta do utente. Os prazos, em dias, são os seguintes:

- Obras com apenas um exemplar – valor de base: 3; utentes cumpridores: 8; utentes faltosos: 2;
- Obras com 5 exemplares ou menos – valor de base 8; utentes cumpridores: 15; utentes faltosos: 2;
- Obras com mais de 5 exemplares – valor de base 15; utentes cumpridores: 30; utentes faltosos: 2.

Se o utente não entregar as obras requisitadas no prazo devido, fica imediatamente suspenso, não podendo requisitar mais obras até regularizar a situação. Por cada dia de atraso, o utente fica sujeito ao pagamento de uma multa de €5,00 (cinco euros). A situação só se considera regularizada após a devolução das obras em atraso e o pagamento da multa. Para efeitos de pagamento de multas, fracções de dia contam como um dia (a unidade de tempo do sistema é o dia).

Deve ser possível alterar ou acrescentar regras para a requisição de obras, bem como fazer alterações aos tempos de requisição permitidos. As alterações devem ter impacto mínimo na implementação desenvolvida.

## Pesquisas

Para permitir que o sistema ajude os utentes a determinar a existência de uma obra, deverá ser possível efectuar pesquisas. As pesquisas consideram os campos relevantes das várias obras. Deve ser possível introduzir novos métodos de pesquisa de uma obra com um impacto mínimo na implementação desenvolvida.

## Gestão de Tempo

A unidade de tempo do sistema é o dia. A data do sistema começa no dia 0 (zero) e faz parte do estado persistente. Sempre que a data é alterada, deve ser verificada a situação dos utentes.

## Notificações

Deve existir um mecanismo de notificações que permita avisar eventuais interessados quando as obras ficam em determinadas situações:

- Quando uma obra é emprestada, quer-se enviar uma notificação a todos as entidades que demonstraram interesse nessa operação.
- Quando uma obra é devolvida, quer-se enviar uma notificação a todos as entidades que demonstraram interesse nessa operação.

A apresentação das notificações enviadas para um dado utilizador deve ser feita quando se visualiza o utilizador (ver abaixo). Quando se faz a visualização, são anexadas à descrição do utilizador (no final) todas as notificações recebidas. Após esta visualização, considera-se que o utilizador fica sem notificações. As notificações devem ser apresentadas pela mesma ordem em que foram enviadas pelo sistema.

Note-se que existem diferença relativamente aos vários pedidos de notificações. No caso de notificações relativamente a obras emprestadas, quer-se receber uma notificação sempre que a obra indicada é requisitada. No caso dos pedidos de notificação relativos à devolução de obras, apenas se quer receber uma notificação quando a obra indicada for devolvida.

# Requisitos de Desenho

Devem ser possíveis extensões ou alterações de funcionalidade com impacto mínimo no código já produzido para a aplicação. O objectivo é aumentar a flexibilidade da aplicação relativamente ao suporte de novas funções. Assim, deve ser possível:

- Adicionar novos tipos (e.g., CDs ou VHS) e novas categorias de obras;
- Definir novas entidades que desejem ser notificadas da requisição ou devolução de obras;
- Introduzir alterações nas regras para requisição de obras ou nos prazos de requisição permitidos;
- Definir novas classificações para os utentes, para além de faltoso, cumpridor ou normal.

Embora na especificação actual não seja possível a remoção de obras ou de utilizadores, a inclusão destas funcionalidades deve ser prevista, por forma a minimizar o impacto da sua futura inclusão.

# Funcionalidade da aplicação

A aplicação permite manter informação sobre as entidades do modelo, permitindo, em particular, gerir utentes, obras e empréstimos. Possui ainda a capacidade de preservar o seu estado (não é possível manter várias versões do estado da aplicação em simultâneo).

A base de dados com os conceitos pré-definidos é carregada no início da aplicação. Não é possível remover utentes ou obras durante a execução da aplicação.

 Note-se que não é necessário implementar de raiz a aplicação: já existem classes que representam e definem a interface geral da funcionalidade do *core* da aplicação, tal como é visível pelos comandos da aplicação.

 A interface geral do *core* já está parcialmente implementada na classe **m19.LibraryManager** e outras fornecidas (cujos nomes devem ser mantidos), devendo ser adaptadas onde necessário. É ainda necessário criar e implementar as restantes classes que suportam a operação da aplicação.

## Serialização

É possível guardar e recuperar o estado actual da aplicação, preservando toda a informação relacionada com a mediateca e que foi descrita acima.

# Interacção com o utilizador

Descreve-se nesta secção a **funcionalidade máxima** da interface com o utilizador. Em geral, os comandos pedem toda a informação antes de proceder à sua validação (excepto onde indicado). Todos os menus têm automaticamente a opção **Sair** (fecha o menu).

As operações de pedido e apresentação de informação ao utilizador **devem** realizar-se através dos objectos *form* e *display*, respectivamente, presentes em cada comando. As mensagens são produzidas pelos métodos das bibliotecas de suporte (**po-uuilib** e **m19-app**). As mensagens não podem ser usadas no núcleo da aplicação (**m19-core**). Além disso, não podem ser definidas novas. Potenciais omissões devem ser esclarecidas antes de qualquer implementação.

As excepções usadas na interacção, excepto se indicado, são subclasses de **pt.tecnico.po.ui.DialogException**, são lançadas pelos comandos e tratadas por **pt.tecnico.po.ui.Menu**. Outras excepções não devem substituir as fornecidas nos casos descritos.

 Note-se que o programa principal e os comandos e menus, a seguir descritos, já estão parcialmente implementados nas *packages* **m19.app**, **m19.app.main**, **m19.app.requests** e **m19.app.users**. Estas classes são de uso obrigatório e estão disponíveis no CVS (módulo **m19-app**).

## Menu Principal

As acções deste menu permitem gerir a salvaguarda do estado da aplicação e abrir submenus. A lista completa é a seguinte: Abrir, Guardar, Mostrar data actual, Avançar data actual, Menu de Gestão de Utentes, Menu de Gestão de Obras e Menu de Gestão de Requisições. Inicialmente, a aplicação apenas tem informação sobre as entidades que foram carregados no arranque.

As etiquetas das opções deste menu estão definidas na classe **m19.app.main.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **m19.app.main.Message**.

 Estes comandos já estão implementados nas classes da *package* **m19.app.main** (disponível no CVS), respectivamente: **DoOpen**, **DoSave**, **DoDisplayDate**, **DoAdvanceDate**, **DoOpenUsersMenu**, **DoOpenWorksMenu**, **DoOpenRequestsMenu**.

### Salvaguarda do estado actual da aplicação

O conteúdo da aplicação (toda a informação detida pela mediateca actualmente em memória) pode ser guardado para posterior recuperação (via serialização Java: **java.io.Serializable**). Na leitura e escrita do estado da aplicação, devem ser tratadas as excepções associadas. A funcionalidade é a seguinte:

- **Abrir** -- Carrega os dados de uma sessão anterior a partir de um ficheiro previamente guardado (ficando este ficheiro associado à aplicação, para futuras operações de salvaguarda). Pede-se o nome do ficheiro a abrir (**openFile()**). Caso ocorra um problema na abertura ou processamento do ficheiro, deve ser lançada a excepção **FileOpenFailedException**. A execução bem sucedida desta opção substitui toda a informação da aplicação.
- **Guardar** -- Guarda o estado actual da aplicação no ficheiro associado. Se não existir associação, pede-se o nome do ficheiro a utilizar, ficando a ele associado. Esta interacção realiza-se através do método **newSaveAs()**. Não é executada nenhuma

acção se não existirem alterações desde a última salvaguarda.

Note-se que a opção **Abrir** não permite a leitura de ficheiros de texto (estes apenas são utilizados na inicialização da aplicação).

A opção **Sair** nunca guarda o estado da aplicação, mesmo que existam alterações.

## Mostrar data actual

A data actual do sistema é apresentada através da mensagem **currentDate()**.

## Avançar data actual

O número de dias a avançar é pedido através de **requestDaysToAdvance()**. O valor indicado deve ser positivo. Caso contrário, a operação não tem efeito.

Além da data, o sistema deve actualizar, caso seja necessário, outros aspectos que dela dependam, designadamente, a situação dos utentes relativa a prazos.

## Gestão e consulta de dados da aplicação

- **Menu de Gestão de Utentes** -- Abre o menu de gestão de utentes e operações associadas.
- **Menu de Gestão de Obras** -- Abre o menu de gestão de obras e operações associadas.
- **Menu de Gestão de Requisições** -- Abre o menu de gestão de requisições e operações associadas.

# Menu de Gestão de Utentes

Este menu permite efectuar operações sobre a base de dados de utentes da biblioteca. A lista completa é a seguinte: Registar utente, Mostrar utente, Mostrar utentes, Mostrar notificações do utente, Pagar multa.

As etiquetas das opções deste menu estão definidas na classe **m19.app.users.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **m19.app.users.Message**.

Sempre que for pedido o identificador de um utente (**requestUserId()**) e o utente não existir, é lançada a excepção **NoSuchUserException**.

Estes comandos já estão implementados nas classes da *package* **m19.app.users** (disponível no CVS), respectivamente: **DoRegisterUser**, **DoShowUser**, '**DoShowUserNotifications**, **DoShowUsers**, **DoPayFine**.

## Registar utente

Pede o nome (**requestUserName()**) e o endereço de correio electrónico (**requestUserEMail()**). O registo bem sucedido é assinalado através da mensagem **userRegistrationSuccessful()**; caso contrário, é lançada a excepção **UserRegistrationFailedException**.

Note-se que a atribuição do identificador do utente é automática e que utentes diferentes são registados em cada operação de registo.

## Mostrar utente

É pedido o identificador do utente, sendo apresentadas as informações sobre esse utente, de acordo com o seguinte formato (e variações descritas abaixo). A multa a apresentar, para utentes suspensos, é um valor inteiro.

**Formatos de apresentação de um utente (activos e suspensos)** **[Collapse]**

```
id - nome - email - comportamento - ACTIVO
id - nome - email - comportamento - SUSPENSO - EUR multa
```

**Exemplos de apresentação de utentes** **[Collapse]**

```
2 - Alberto Meireles - ameireles@mymail.com - CUMPRIDOR - ACTIVO
1 - Fernando Meireles - fmeireles@mymail.com - FALTOSO - SUSPENSO - EUR 10
3 - Fernando Meireles - ffm@mymail.com - NORMAL - ACTIVO
```

## Mostrar notificações do utente

É pedido o identificador do utente, sendo apresentadas as notificações para esse utente, de acordo com o seguinte formato (correspondente aos casos descritos acima):

**Formatos de apresentação das notificações de um utente** **[Collapse]**

```
ENTREGA: descrição de obra entregue
REQUISIÇÃO: descrição de obra requisitada
```

**Exemplos de notificações** **[Collapse]**

```
ENTREGA: 4 - 2 de 4 - DVD - Casamento Real - 8 - Ficção - António Fonseca - 200400500
REQUISIÇÃO: 5 - 4 de 22 - Livro - Dicionário - 45 - Referência - Pedro Casanova - 1234567893
```

Note-se que a descrição é idêntica à que é realizada para mostrar cada obra. No entanto, a solução deve ser suficientemente flexível para permitir outros formatos de apresentação das notificações (sem impacto no código do domínio da aplicação).

### Mostrar utentes

Apresenta informações sobre todos os utentes, ordenando-os lexicograficamente pelo nome. Caso existam utentes com o mesmo nome, devem ser ordenados por ordem crescente dos seus identificadores. O formato é o descrito em Mostrar utente.

### Pagar multa

Pede o identificador do utente cuja multa deve ser paga. Se o utente estiver suspenso, a multa é saldada e o utente passa a poder requisitar obras, de acordo com as regras gerais. Se o utente não estiver suspenso, i.e., não tem multas por saldar, deve lançar-se uma excepção **UserIsActiveException**.

## Menu de Gestão de Obras

Este menu apresenta as operações disponíveis sobre obras. A lista completa é a seguinte: Mostrar obra, Mostrar obras e Efectuar pesquisa.

As etiquetas das opções deste menu estão definidas na classe **m19.app.works.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **m19.app.works.Message**.

Sempre que é pedido o identificador da obra (**requestWorkId()**), é lançada a excepção **NoSuchWorkException**, se a obra indicada não existir.

Estes comandos já estão implementados nas classes da *package* **m19.app.works** (disponível no CVS), respectivamente: **DoShowWork**, **DoShowWorks**, **DoPerformSearch**.

### Mostrar obra

É pedido o identificador da obra (**requestWorkId()**). Se a obra existir, é apresentada de acordo com os seguintes formatos (para livros e DVDs).

O formato genérico de apresentação da obras é como se segue:

**Formato de apresentação de obras** **[Collapse]**

*id* - *número de exemplares disponíveis* de *número de exemplares* - *tipo* - *título* - *preço* - *categoria* - *informação adicional*

Para livros, a informação adicional corresponde ao autor e ao ISBN; para DVDs, a informação adicional corresponde ao realizador e ao número de registo no IGAC.

**Exemplos de apresentação de obras** **[Collapse]**

```
3 - 20 de 23 - Livro - Casa Azul - 15 - Ficção - João Fonseca - 1234567891
4 - 2 de 2 - DVD - Casamento Real - 8 - Ficção - António Fonseca - 200400500
5 - 0 de 4 - Livro - Dicionário - 45 - Referência - Pedro Casanova - 1234567893
6 - 1 de 21 - Livro - Enciclopédia - 100 - Técnica e Científica - Zé Fonseca - 1234567894
```

## Mostrar obras

Apresenta informações sobre todas as obras, ordenando-as pelos seus identificadores. O formato de apresentação é como descrito em Mostrar obra.

## Efectuar pesquisa

Esta opção realiza uma procura por termo (cadeia de caracteres), pedido através de **requestSearchTerm()**. Como resultado, deve ser apresentada uma lista das obras encontradas pela pesquisa, ordenadas por ordem crescente do seu identificador, utilizando o formato descrito para Mostrar obra.

O termo de pesquisa deve ser comparado (sem distinção entre letras maiúsculas e minúsculas) com os campos relevantes de cada obra: para DVDs, o realizador e o título; para livros, o autor e o título. Só devem ser apresentadas obras que contenham o termo de pesquisa num dos campos relevantes.

Assim, considerando as quatro obras no exemplo acima, uma pesquisa pelo termo **casa** retornaria as obras com os identificadores 3, 4 e 5.

**Resultados de pesquisa pelo termo "casa"** **[Collapse]**

```
3 - 20 de 23 - Livro - Casa Azul - 15 - Ficção - João Fonseca - 1234567891
4 - 2 de 2 - DVD - Casamento Real - 8 - Ficção - António Fonseca - 200400500
5 - 0 de 4 - Livro - Dicionário - 45 - Referência - Pedro Casanova - 1234567893
```

Caso não sejam encontradas obras, não deve ser produzido qualquer resultado.

# Menu de Gestão de Requisições

Este menu apresenta as operações relacionadas com requisições de obras. A lista completa é a seguinte: Requisitar obra, Devolver obra.

As etiquetas das opções deste menu estão definidas na classe **m19.app.requests.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **m19.app.requests.Message**.

Sempre que é pedido o identificador do utente (**requestUserId()**), é lançada a excepção **NoSuchUserException**, se o utente indicado não existir. Sempre que é pedido o identificador da obra (**requestWorkId()**), é lançada a excepção **NoSuchWorkException**, se a obra indicada não existir.

Estes comandos já estão parcialmente implementados nas classes da *package* **m19.app.requests** (disponível no CVS), respectivamente: **DoRequestWork**, **DoReturnWork**.

## Requisitar obra

No processo de requisição de uma obra, o sistema pede, primeiro, a identificação do utente e, de seguida, o identificador da obra a requisitar. Se o utente não puder requisitar a obra (considerando-se as regras definidas acima), deve ser lançada a excepção **RuleFailedException** (excepto regra 3: ver a seguir).

Se a requisição não for possível por falta de exemplares (violação da regra 3), deve-se perguntar ao utente, utilizando a mensagem **requestReturnNotificationPreference()**, se deseja ser notificado acerca da devolução. Utiliza-se a mensagem **workReturnDay()** para comunicar o prazo de devolução, em caso de requisição bem sucedida.

## Devolver obra

No processo de devolução de uma obra, o sistema pede, primeiro, o identificador do utente e, de seguida, o da obra a devolver. Se a obra não tiver sido requisitada pelo utente indicado, deve-se lançar uma excepção **WorkNotBorrowedByUserException**. Caso contrário, o sistema processa a entrega e, caso haja lugar ao pagamento de multa, é apresentada a mensagem **showFine()**.

O utente pode entregar uma obra sem pagar a multa, continuando suspenso até regularizar a situação, sem prejuízo da obra ser assinalada como entregue. Antes de liquidar a multa, o sistema interroga o utente sobre o desejo de pagamento, através da mensagem **requestFinePaymentChoice()**. Se a resposta for positiva, a multa é liquidada e o utente fica activo, caso não tenha nenhuma obra por entregar fora de prazo.

# Leitura de Dados a Partir de Ficheiros Textuais

Além das opções de manipulação de ficheiros descritas no menu principal, é possível iniciar a aplicação com um ficheiro de texto especificado pela propriedade Java **import**.

As obras da mediateca têm o formato descrito abaixo, respectivamente, para DVDs e livros. Assume-se que os títulos das obras não podem conter o carácter **:** e que o preço é um número inteiro (sugere-se a utilização do método **String.split** para o processamento preliminar destas linhas). Não existem entradas mal-formadas.

Cada linha tem tem uma descrição distinta mas que segue o seguinte formato geral. O campo "exemplares" indica o número de exemplares da obra disponíveis na mediateca.

```
DVD:título:realizador:preço:categoria:númeroIGAC:exemplares
BOOK:título:autor:preço:categoria:ISBN:exemplares
```

É ainda possível definir utentes, de acordo com o seguinte formato:

```
USER:nome:email
```

Um exemplo de conteúdo do ficheiro inicial é como se segue:

**Exemplo de ficheiro de entrada textual** **[Collapse]**

```
DVD:Era uma vez na Amadora:Fernando Fonseca:20:FICTION:200505550:10
BOOK:A arte de sobreviver no 36:Joao Fonseca:20:FICTION:1234567892:22
BOOK:Bairro Alto e o Budismo Zen:Zun Tse Fonseca:20:FICTION:1234567891:50
DVD:48 Horas para o Exame:Orlando Fonseca:12:FICTION:200505553:10
BOOK:Analise Matematica sem Mestre:Carlos Fonseca:20:SCITECH:1234567890:5
DVD:Lumiar Selvagem:Pedro Fonseca:20:FICTION:200505551:5
BOOK:Dicionário de Programação:Odete Fonseca:20:REFERENCE:1234567890:50
USER:Obi-Wan Kenobi:obiwan@jedi.org
```

A codificação dos ficheiros a ler é garantidamente UTF-8.

# Execução dos Programas e Testes Automáticos

Usando os ficheiros **test.import**, **test.in** e **test.out**, é possível verificar automaticamente o resultado correcto do programa. Note-se que é necessária a definição apropriada da variável **CLASSPATH** (ou da opção equivalente **-cp** do comando **java**), para localizar as classes do programa, incluindo a que contém o método correspondente ao ponto de entrada da aplicação (**m19.app.App.main**). As propriedades são tratadas automaticamente pelo código de apoio.

```
java -Dimport=test.import -Din=test.in -Dout=test.outhyp m19.app.App
```

Assumindo que aqueles ficheiros estão no directório onde é dado o comando de execução, o programa produz o ficheiro de saída **test.outhyp**. Em caso de sucesso, os ficheiros das saídas esperada (**test.out**) e obtida (**test.outhyp**) devem ser iguais. A comparação pode ser feita com o comando:

```
diff -b test.out test.outhyp
```

Este comando não deve produzir qualquer resultado quando os ficheiros são iguais. Note-se, contudo, que este teste não garante o correcto funcionamento do código desenvolvido, apenas verificando alguns aspectos da sua funcionalidade.

# Notas de Implementação

Tal como indicado acima, algumas classes fornecidas como material de apoio, são de uso obrigatório e não podem ser alteradas. Outras dessas classes são de uso obrigatório e têm de ser alteradas.

A serialização Java usa as classes da *package* **java.io**, em particular, a interface **java.io.Serializable** e as classes de leitura **java.io.ObjectInputStream** e escrita **java.io.ObjectOutputStream** (entre outras).

Categories: **Ensino** **PO** **Projecto de PO**