

ArtAuctions - Obras de Arte e Leilões

Algoritmos e Estruturas de Dados

Ano letivo 2023/2024

Enunciado do Trabalho Prático, **versão 1.4 – 14-11-2023**



NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTAMENTO
DE INFORMÁTICA

DEPARTMENT OF
COMPUTER SCIENCE

Conteúdo

1	Introdução e notas importantes	3
2	Desenvolvimento da aplicação <i>ArtAuctions</i>	3
2.1	Descrição do problema	3
2.2	Conceitos e definições	4
2.3	Sintaxe	4
2.4	Dados e resultados	5
2.5	Especificação do sistema	5
3	Comandos	5
3.1	Comando addUser – inserir novo utilizador	7
3.2	Comando addArtist – inserir novo artista	7
3.3	Comando removeUser - remoção de um utilizador	8
3.4	Comando addWork – inserir nova obra de arte no sistema	9
3.5	Comando infoUser - Consultar dados de um utilizador	9
3.6	Comando infoArtist - Consultar dados de um artista	10
3.7	Comando infoWork - Consultar dados de uma obra de arte	10
3.8	Comando createAuction – Criar novo leilão	11
3.9	Comando addWorkAuction – Adicionar obra de arte a leilão	11
3.10	Comando bid – Efetuar proposta de compra em Leilão	12
3.11	Comando closeAuction – Encerrar leilão	13
3.12	Comando listAuctionWorks – listar Obras adicionadas a leilão	13
3.13	Comando listArtistWorks – listar Obras de um artista	14
3.14	Comando listBidsWork – listar propostas de compra de uma obra	15
3.15	Comando listWorksByValue – listar Obras por valor	16
3.16	Comando quit	16
4	Desenvolvimento	16
4.1	Entregas e faseamento	17
4.2	Relatório	18
5	Requisitos do Mooshak	18
5.1	Regras a satisfazer pelo programa	18
5.2	Como preparar o arquivo ZIP	18
6	Avaliação	18
6.1	Testes de avaliação	19
6.2	Verificação de autoria	19
6.3	Resumo das datas importantes	19

Esclarecimentos e Correções de gralhas

1. Secção 2.4: Esclarecimento sobre o número de obras e bids em cada leilão.
2. Secção 3.3: Clarificação de restrições na remoção de utilizadores.
3. Secção 3.10: versão 1.1: Alteração nas mensagens e na ordem de identificação das situações de excepção;
Versão 1.2: Nova alteração na ordem de identificação das situações de excepção.
4. Secção 3.11: Esclarecimento sobre as ações associadas ao encerramento do leilão.
5. Secções 3.12, 3.13 e 3.15: Clarificação do valor associado a obras nas listagens.
6. Secção 3.7: Clarificação do valor associado a obras na consulta da obra.

1 Introdução e notas importantes

Este documento descreve o trabalho prático da disciplina de Algoritmos e Estruturas de Dados do 2º ano da Licenciatura/Mestrado Integrado em Engenharia Informática. Os alunos podem e devem tirar todas as dúvidas com os docentes da disciplina.

Trabalho em grupo de 2 alunos a desenvolver respeitando os princípios do *Código de Ética* do Departamento de Informática. Não são aceites grupos individuais sem autorização por escrito (por ex. por email) de um docente.

Trabalho com 2 fases de entrega. Ver detalhes sobre submissão, datas de entrega e versões possíveis nas secções 4, 5 e 6.

2 Desenvolvimento da aplicação *ArtAuctions*

O objectivo deste trabalho é o desenvolvimento de um sistema de acesso a informação sobre obras de arte, e seus autores assim como de organização de leilões da mesmas obras. Nesta secção encontra informação sobre o problema, os conceitos associados, a sintaxe utilizada, os tipos de dados primitivos e resultados de base assim como a especificação genérica do sistema. Na secção seguinte (secção 3), cada uma das funcionalidades serão descritas em forma de comandos a disponibilizar.

2.1 Descrição do problema

Pretende-se assim criar uma aplicação que facilite a exposição digital de obras de arte que serão postas à vendas em leilões. O sistema irá incluir informação sobre *Utilizadores* que serão, de base, colecionadores de arte. Estes utilizadores poderão ser também *Artistas* que pretendem divulgar e leiloar as suas *Obras* de arte.

O sistema deverá assim também permitir a organização de *Leilões* aos quais serão associadas obras para leiloar. Na sequência da criação de um leilão, podem-se adicionar obras ao mesmo, e relativamente a cada uma destas obras, os colecionadores podem fazer propostas de compra, que serão aceites, desde que sejam superiores ou iguais ao valor mínimo de compra para a obra, definido no momento de adição da obra ao leilão.

Quando o leilão é encerrado, cada uma das propostas submetidas para compra de cada obra será avaliada, por ordem do momento de submissão, e a proposta mais alta que tenha sido introduzida mais cedo, será a proposta aceite. O valor mais alto alguma vez usado para comprar uma obra de arte é usado para ordenar as obras de arte na 2ª fase do trabalho.

Dado que o trabalho será entregue em duas fases e que, na primeira fase, os estudantes terão apenas à sua disposição as classes do pacote *dataStructures* suportadas pela implementação de lista duplamente

ligada, a funcionalidade, assim como a informação associada a esta primeira fase será limitada, e o trabalho a entregar será simplificado.

Esta aplicação tem como funcionalidades principais:

- registar um utilizador (coleccionador) c ou artista a ;
- remover um utilizador u e, caso este utilizador seja também artista, remover todas as obras por ele criadas;
- registar uma obra de arte o criada por um artista a ;
- consultar os dados de um utilizador u , de um artista a , ou de uma obras o ;
- registar a criação de um leilão l ;
- associar uma obra o a um leilão l ;
- registar uma proposta de compra de uma obra o no leilão l .
- encerrar o leilão l .
- Efetuar várias listagens de dados.

2.2 Conceitos e definições

Existe um conjunto de conceitos associados ao sistema a desenvolver e que se descreve a seguir:

Utilizador/coleccionador – Um utilizador de base é um coleccionador que poderá adquirir obras de arte em leilões. É identificado por um login (uma cadeia de caracteres sem espaços).

Artista – Um artista é um utilizador conhecido na aplicação e que poderá ter obras à venda em leilões. É também identificado por um login (uma cadeia de caracteres sem espaços). Um artista também é um coleccionador, ou seja, um artista também pode adquirir obras.

Um login identifica de forma única um utilizador do sistema, seja ele coleccionador e/ou artista.

Obra de arte – foi criada por um artista existente no sistema. As obras de arte são identificadas por um identificador único.

Leilão – É um evento do sistema identificado por um identificador único. A partir da sua criação, podem-se-lhe adicionar obras para leiloar. Estas podem também ser alvo de propostas de compra que, caso sejam válidas (iguais ou superiores ao valor mínimo de *bidding*, serão adicionadas à obra, por ordem de chegada, no âmbito do leilão. Uma vez fechado o leilão, a obra será adquirida pelo coleccionador que primeiro tenha submetido a proposta mais alta. Desta forma, não existe a possibilidade de empate no valor das propostas.

2.3 Sintaxe

Pretende-se que a interface da aplicação seja muito simples, de modo a poder ser utilizada em ambientes diversos e, no caso da saída de dados, para permitir automatizar o processo de teste. Por estes motivos, a entrada e a saída deverão respeitar o formato rígido que se indica na secção 3.

Convém referir que o símbolo \leftarrow representa uma mudança de linha e que **cada comando termina com duas mudanças de linha**. Poderá admitir que a entrada está sempre sintaticamente correta e que os dados satisfazem as restrições enunciadas na secção 2.4. Não existe necessidade de, durante a execução, verificar se os dados inseridos são dos tipos definidos.

2.4 Dados e resultados

Os campos *login* de utilizador e artistas, assim como os identificadores (*idObra* e *idLeilao*) e os emails são cadeias de caracteres, sem espaços.

Os nomes de utilizadores, artistas e obras, assim como os nomes artísticos, são também cadeias de caracteres, mas podem conter espaços. As cadeias de caracteres de dados a inserir no sistema não deverão conter mudanças de linha (símbolo `\n`).

A *idade* dos utilizadores, o ano de concepção de uma obra, e o valor e/ou preço de uma obra são inteiros superior a zero (0). Este facto não terá de ser controlado pelo sistema. No entanto, apenas adultos podem ser registados no sistema e isso deve ser verificado.

O número de utilizadores, obras e leilões não é limitado superiormente. Poderemos, no entanto, prever um número de 20 000 utilizadores e 10 000 obras. Num determinado momento, poderemos ter largas centenas de leilões abertos. Cada leilão terá apenas algumas (poucas) dezenas de obras e também o número de bids (propostas de compra) por obra será da ordem das dezenas. Os leilões só são conhecidos pelo sistema até encerrarem. No entanto, cada obra deve guardar o valor máximo pelo qual já foi leiloada, porque esse valor será usado para o *ranking* das obras existentes no sistema. Um artista poderá criar centenas de obras.

NOTA RELATIVA À PRIMEIRA FASE: Assuma que, na primeira fase de realização do trabalho, apenas algumas dezenas de utilizadores, obras e leilões serão consideradas.

O sistema não faz qualquer distinção entre maiúsculas e minúsculas. Por exemplo, para a aplicação, o leilão com identificador *RFS-Q101* deverá ser igual a *Rfs-Q101* e a *rfs-q101*. Também não serão incluídas palavras com acentos, cedilha ou outros caracteres especiais. No entanto, o output da informação introduzida deverá ser gerado tal como foi inserido. Isto significa que, no exemplo dado, se o identificador do leilão foi inserido utilizando a primeira forma (*RFS-Q101*), o sistema deverá listá-lo nesta forma, quando gerar o seu output.

2.5 Especificação do sistema

A interação do utilizador com o programa é feita sempre através de comandos, descritos na secção 3. O sistema deverá ler comandos da entrada padrão (`System.in`), processando-os um a um e enviando os resultados para a saída padrão (`System.out`). A execução do programa pode ser assegurada com a introdução de dados e respetiva apresentação de resultados em ficheiro, através do redirecccionamento do input e do output.

A persistência dos dados deve ser assegurada entre execuções consecutivas. Isto significa que, antes de terminar a execução, o sistema deve guardar o estado da base de dados em disco, utilizando as funcionalidades de serialização do Java. Na execução seguinte do programa, os dados armazenados deverão ser carregados do disco e o estado do sistema reconstituído.

3 Comandos

Nesta secção apresentam-se os vários comandos que o sistema deve ser capaz de interpretar e executar. Nos exemplos apresentados, diferenciamos o **texto escrito pelo utilizador** da **retroacção escrita pelo programa na consola**, ao executar o comando.

Tal como descrito nos próprios comandos, alguns comandos deverão ser apenas implementados na **2ª fase**.

Vários comandos têm parâmetros. A entrada e a saída deverão respeitar o formato rígido que se indica neste documento. Pode assumir que o utilizador não cometerá erros na introdução de argumentos nos comandos, para além dos descritos neste enunciado, ou seja, apenas tem de tratar as situações de erro descritas aqui, pela ordem em que são descritas.

Pode assumir que o utilizador apenas usa argumentos em número e de tipos correctos. No entanto, pode acontecer que os argumentos passados a um desses comandos tenham algum valor inválido no contexto do

problema. Por esse motivo, teremos de testar esses argumentos exactamente pela ordem especificada neste enunciado.

Na leitura de comandos, o interpretador não deve fazer distinção entre maiúsculas e minúsculas. Por exemplo, para o comando `ADDUSER` (comando usado para registar um novo utilizador) o interpretador deve aceitar como válidas todas as alternativas possíveis, por exemplo: `addUSER`, `adduser`, `aDdUsEr` e `ADDUser`.

Nos vários exemplos que se seguem, o símbolo ↵ denota a mudança de linha. Cada comando termina com duas mudanças de linha.

A tabela 1 apresenta os comandos do sistema, cujos detalhes são apresentados nas secções seguintes. Caso o utilizador introduza um comando inexistente, o programa ignora. Por exemplo o comando inexistente `desconhecido` teria este efeito:

```
> desconhecido↵
>
```

Tabela 1: Comandos

Comando	Descrição	Detalhes
<code>addUser</code>	inserção de novo utilizador (colecionador)	Secção 3.1
<code>addArtist</code>	inserção de novo artista	Secção 3.2
<code>removeUser</code>	remoção de um utilizador	Secção 3.3
<code>addWork</code>	inserção de nova obra de arte	Secção 3.4
<code>infoUser</code>	informação sobre um utilizador genérico	Secção 3.5
<code>infoArtist</code>	informação sobre um artista	Secção 3.6
<code>infoWork</code>	informação sobre uma obra de arte	Secção 3.7
<code>createAuction</code>	criação de Leilão	Secção 3.8
<code>addWorkAuction</code>	adiciona obra de arte a leilão	Secção 3.9
<code>bid</code>	submissão de proposta de compra de uma obra num leilão	Secção 3.10
<code>closeAuction</code>	encerramento de leilão	Secção 3.11
<code>listAuctionWorks</code>	listagem das obras para venda em leilão	Secção 3.12
<code>listArtistWorks</code>	listagem das obras de um artista	Secção 3.13
<code>listBidsWork</code>	listagem de propostas de compra de obra em leilão	Secção 3.14
<code>ListWorksByValue</code>	listagem de obras já vendidas por valor máximo de venda	Secção 3.15
<code>quit</code>	termina a execução do programa	Secção 3.16

3.1 Comando **addUser** – inserir novo utilizador

Regista um novo utilizador (colecccionador) no sistema. Para realizar um registo de utilizador deve ser introduzido sempre o login, que tem que ser único. São também introduzidos o nome do utilizador, a sua idade e o email.

O sistema poderá guardar vários utilizadores. O acesso a cada utilizador será conseguido por login. O login identifica o utilizador de forma única.

O formato geral do comando é o seguinte (parâmetros com estilo **bold**):

```
> addUser login nome↵  
idade email↵  
↵  
Registo de utilizador executado.↵  
↵  
>
```

Este comando só tem sucesso se não existir, no sistema, um utilizador (colecccionador ou artista) com o login dado. É de notar que apenas adultos se podem registar no sistema.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo e, nesse caso, apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- Candidato a utilizador não adulto:

```
Idade inferior a 18 anos.↵  
↵
```

- Login já existe no sistema:

```
Utilizador existente.↵  
↵
```

3.2 Comando **addArtist** – inserir novo artista

Regista um novo utilizador (artista) no sistema. Para realizar um registo de artista deve ser introduzido sempre o login, que tem que ser único. São também introduzidos o nome do utilizador, o nome artístico, a sua idade e o email.

O sistema poderá guardar vários artistas. O acesso a cada artista será conseguido por login. O login identifica o utilizador de forma única.

O formato geral do comando é o seguinte:

```
> addArtist login nome↵  
nomeArtistico↵  
idade email↵  
↵  
Registo de artista executado.↵  
↵  
>
```

Este comando só tem sucesso se não existir, no sistema, um utilizador (colecccionador ou artista) com o login dado e apenas adultos se podem registar no sistema.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- Candidato a utilizador não adulto:

Idade inferior a 18 anos.
↓

- Login já existe no sistema:

Utilizador existente.
↓

3.3 Comando `removeUser` - remoção de um utilizador

Remove um utilizador (colecccionador ou artista) do sistema. Esta operação determina a remoção do registo de um utilizador. A remoção é feita através do login do utilizador. Caso o utilizador a remover seja também um artista, as suas obras devem também ser removidas do sistema.

O formato geral do comando é o seguinte:

```
> removeUser login  
↓  
Remocao de utilizador executada.  
↓  
>
```

A remoção de um utilizador faz-se por login, que tem de existir no sistema para que a remoção tenha sucesso. Além disso, Um utilizador só pode ser removido se não tiver propostas ativas em leilões que ainda não encerraram. Um artista só pode ser removido se não tiver obras em leilão.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- login não existe no sistema:

Utilizador inexistente.
↓

- utilizador tem propostas submetidas:

Utilizador com propostas submetidas.
↓

- Artista tem obras em leilões:

Artista com obras em leilao.
↓

3.4 Comando **addWork** – inserir nova obra de arte no sistema

Regista uma nova obra de arte no sistema. Para realizar um registo de obra de arte deve ser introduzido identificador da obra, que tem que ser único. São também introduzidos o login do artista, autor da obra, o ano de elaboração da mesma e o seu nome.

O sistema poderá guardar várias obras de arte. O acesso a cada obra de arte é feito pelo seu identificador, que é único.

O formato geral do comando é o seguinte:

```
> addWork idObra loginAutor ano nome↵
↳
Registo de obra executado.↵
↳
>
```

Este comando só tem sucesso se não existir, no sistema, uma obra de arte com o identificador submetido. Além disso, o login do autor da obra tem de existir no sistema.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- identificador da obra já existe no sistema:

```
Obra existente.↵
↳
```

- Login de autor não existe no sistema:

```
Utilizador inexistente.↵
↳
```

- Login de autor não é referente a um artista:

```
Artista inexistente.↵
↳
```

3.5 Comando **infoUser** - Consultar dados de um utilizador

Consulta os dados genéricos de um dado utilizador. Quando este comando é executado, a informação do utilizador é escrita na consola. Os dados apresentados são: login, nome, idade e email do utilizador.

O formato geral do comando é o seguinte (o resultado apresentado em *italico* representa a ordem pela qual os dados deverão ser devolvidos):

```
> infoUser login↵
↳
login nomeUtilizador idade email↵
↳
>
```

Para a operação ter sucesso, o login do utilizador tem de existir no sistema. Em caso contrário, a seguinte mensagem deve ser devolvida e a operação não é executada.

```
> Utilizador inexistente.↵
↵
```

3.6 Comando **infoArtist** - Consultar dados de um artista

Consulta os dados de um artista. Quando este comando é executado, a informação do artista é escrita na consola. Os dados apresentados são: login, nome, nome artístico, idade e email do artista.

O formato geral do comando é o seguinte:

```
> infoArtist login↵
↵
login nome nomeArtistico idade email↵
↵
>
```

Para a operação ter sucesso, o login do artista tem de existir no sistema e tem de estar associado a um utilizador do tipo artista.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- Login de autor não existe no sistema:

```
Utilizador inexistente.↵
↵
```

- Login de autor não é referente a um artista:

```
Artista inexistente.↵
↵
```

3.7 Comando **infoWork** - Consultar dados de uma obra de arte

Consulta os dados de uma obra de arte. Quando este comando é executado, a informação da obra de arte é escrita na consola. Os dados apresentados são: o identificador da obra, o seu nome, ano de criação, valor de venda mais alto, login e nome do seu autor. Se a obra ainda não foi leiloadada, o último preço de venda é zero (0).

O formato geral do comando é o seguinte:

```
> infoWork idObra↵
↵
idObra nome ano valorVendaMaisAlto loginAutor nomeAutor↵
↵
>
```

Para a operação ter sucesso, o identificador da obra tem de existir no sistema. Em caso contrário, a seguinte mensagem deve ser devolvida e a operação não é executada.

```
Obra inexistente.↵
↵
```

3.8 Comando **createAuction** – Criar novo leilão

Cria um novo leilão. Para criar um leilão deve ser introduzido sempre o identificador do leilão, que tem que ser único. O sistema poderá gerir vários leilões ao mesmo tempo. O acesso a cada leilão será conseguido por identificador.

O formato geral do comando é o seguinte:

```
> createAuction idLeilao↵
↵
Registo de leilao executado.↵
↵
>
```

Este comando só tem sucesso se não existir, no sistema, um leilão com o identificador introduzido. Em caso contrário, a seguinte mensagem deve ser devolvida e a operação não é executada.

```
Leilao existente.↵
↵
```

3.9 Comando **addWorkAuction** – Adicionar obra de arte a leilão

Adicionar um obra de arte a um leilão. Para adicionar uma obra de arte a um leilão deve ser introduzido o identificador do leilão, o identificador da obra e o valor mínimo aceitável como proposta de compra.

O formato geral do comando é o seguinte:

```
> addWorkAuction idLeilao idObra valorMinimo↵
↵
Obra adicionada ao leilao.↵
↵
>
```

Este comando só tem sucesso se existir, no sistema, um leilão com o identificador inserido e uma obra de arte com o identificador submetido. Se a obra já tiver sido adicionada anteriormente ao leilão, o comando devolve o mesmo resultado, sem criar repetições.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- identificador da leilao não existe no sistema:

```
Leilao inexistente.↵
↵
```

- identificador da obra não existe no sistema:

Obra inexistente.
↓

3.10 Comando **bid** – Efetuar proposta de compra em Leilão

Efetuar uma proposta de compra de uma obra de arte num leilão. Para efetuar uma proposta deve ser introduzido o identificador do leilão e da obra, o login do colecionador e o valor da proposta de compra.

O formato geral do comando é o seguinte:

```
> bid idLeilao idObra login valor  
↓  
Proposta aceite.  
↓  
>
```

Este comando só tem sucesso se existir, no sistema, um leilão com o identificador inserido, uma obra de arte com o identificador submetido e um utilizador com o login dado. Um dado utilizador pode submeter várias propostas, relativas à mesma obra ou a outras, num mesmo leilão.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- login não existe no sistema:

Utilizador inexistente.
↓

- identificador da leilao não existe no sistema:

Leilao inexistente.
↓

- identificador da obra não existe no leilão:

Obra inexistente no leilao.
↓

- valor de proposta é inferior ao valor mínimo definido no momento de adição da obra ao leilão:

Valor proposto abaixo do valor minimo.
↓

3.11 Comando **closeAuction** – Encerrar leilão

Encerra leilão. O encerramento do leilão tem várias partes:

1. Para cada uma das obras que fazem parte do leilão, as propostas submetidas para compra da obra são analisadas, sendo aceite a primeira proposta mais alta submetida;
2. Se o valor final de venda de uma obra for superior ao valor mais alto de venda associado à mesma (ou seja se for o valor de venda mais alto que já alguma vez foi aplicado a esta obra), este será substituído, passando a ser o valor de referência para *ranking* da obra;
3. Na segunda fase do trabalho, a alteração efetuada no ponto anterior poderá envolver uma alteração de *ranking* da obra;
4. Os valores de venda de cada obra são listados associados ao login e nome do colecionador comprador;
5. o Leilão é apagado do sistema.

O formato geral do comando é apresentado abaixo. A lista das obras vendidas no leilao é apresentada por ordem de entrada das obras no leilão. As obras não vendidas por falta de propostas, são também listadas dentro da mesma ordem com mensagem diferente.

```
> closeAuction idLeilao↵
←
Leilao encerrado.↵
Mensagem-Listagem-Encerramento-Leilao↵
Mensagem-Listagem-Encerramento-Leilao↵
...
Mensagem-Listagem-Encerramento-Leilao↵
←
>
```

Possibilidades para *Mensagem-Listagem-Encerramento-Leilao*:

- *idObra nomeObra loginComprador nomeComprador valorCompra*
 - Para obras vendidas;
- *idObra nomeObra* sem propostas de venda.
 - Para obras não vendidas.

Este comando só tem sucesso se existir, no sistema, um leilão com o identificador introduzido. Em caso contrário, a seguinte mensagem deve ser devolvida e a operação não é executada.

```
Leilao inexistente.↵
←
```

3.12 Comando **listAuctionWorks** – listar Obras adicionadas a leilão

Lista obras em leilão. Este comando lista todas as obras adicionadas a um determinado leilão pela ordem em que estas foram adicionadas ao mesmo.

O formato geral do comando é apresentado abaixo. A lista das obras é apresentada por ordem de entrada das obras no leilão.

```
> listAuctionWorks idLeilao
  ↴
  idObra nome ano valorVendaMaisAlto loginAutor nomeAutor
  idObra nome ano valorVendaMaisAlto loginAutor nomeAutor
  ...
  idObra nome ano valorVendaMaisAlto loginAutor nomeAutor
  ↴
>
```

Este comando só tem sucesso se existir, no sistema, um leilão com o identificador introduzido e se já tiverem sido adicionadas obras ao leilão.

As situações de exceção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de exceção:

- Leilão não existe no sistema:

```
Leilao inexistente.
  ↴
```

- Leilão não tem obras adicionadas:

```
Leilao sem obras.
  ↴
```

3.13 Comando **listArtistWorks** – listar Obras de um artista

Lista todas as obras criadas por um artista. Este comando lista todas as obras de um determinado artista ordenadas por nome da obra, e só deverá ser implementado na 2^a fase do trabalho.

O formato geral do comando é apresentado abaixo. A lista das obras é apresentada por ordem do nome da obra.

```
> listArtistWorks login
  ↴
  idObra nome ano valorVendaMaisAlto
  idObra nome ano valorVendaMaisAlto
  ...
  idObra nome ano valorVendaMaisAlto
  ↴
>
```

Para a operação ter sucesso, o login do artista tem de existir no sistema e tem de estar associado a uma utilizador do tipo artista.

As situações de exceção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- Login de artista não existe no sistema:

Utilizador inexistente.↵
↳

- Login dado não é referente a um artista:

Artista inexistente.↵
↳

- Artista não tem obras:

Artista sem obras.↵
↳

3.14 Comando **listBidsWork** – listar propostas de compra de uma obra

Lista as propostas de compra de uma obra num leilão. Todas as propostas de compra de uma obra num leilão são listadas por **ordem de submissão das mesmas**. Os dados apresentados são o login e nome do coleccionador e o valor de compra proposto pelo mesmo.

O formato geral do comando é apresentado abaixo.

```
> listBidsWork idLeilao idObra ↵
↳
login nome valor↵
login nome valor↵
...
login nome valor↵
↳
>
```

Para a operação ter sucesso, o identificador da obra e do leilão têm de existir no sistema. A obra tem de ter sido adicionada ao leilão e devem já existir propostas de compra da obra nesse leilão.

As situações de excepção descritas devem ser detectadas na ordem apresentada abaixo. Apenas a mensagem associada à primeira situação detetada deve ser devolvida e a operação não é executada.

Situações de excepção:

- Leilão não existe no sistema:

Leilao inexistente.↵
↳

- Obra não existe no Leilão:

Obra inexistente no leilao.↵
↳

- Obra não tem propostas de compra no leilão:

```
Obra sem propostas.↵
↳
```

3.15 Comando `listWorksByValue` – listar Obras por valor

Lista todas as obras no sistema por **ordem descendente** do valor mais alto pelo qual já foram vendidas. Este comando só deverá ser implementado na 2^a fase do trabalho. Apenas as obras que já foram vendidas devem fazer parte desta listagem. As obras com o mesmo valor máximo de venda associado devem ser ordenadas por **nome da obra**.

O formato geral do comando é apresentado abaixo.

```
> listWorksByValue↵
↳
idObra nome ano valorVendaMaisAlto loginAutor nomeAutor ↵
idObra nome ano valorVendaMaisAlto loginAutor nomeAutor ↵
...
idObra nome ano valorVendaMaisAlto loginAutor nomeAutor ↵
↳
>
```

Para a operação ter sucesso, deve existir pelo menos uma obra no sistema que já tenha sido vendida em leilão. Em caso contrário, a seguinte mensagem deve ser devolvida e a operação não é executada.

```
Nao existem obras ja vendidas em leilao.↵
↳
```

3.16 Comando `quit`

Termina a execução do programa. O comando não tem parâmetros e tem sempre sucesso.

O formato do comando é o seguinte:

```
> quit↵
↳
Obrigado. Ate a proxima.↵
↳
>
```

4 Desenvolvimento

Todos os estudantes que realizam o trabalho prático têm de estar registados em turno prático.

O trabalho é realizado em **grupos de dois alunos**, preferencialmente do mesmo turno prático. (NOTA: tal como referido nas aulas, **não são aceites grupos individuais** sem autorização por escrito dos docentes. Esta autorização não será dada na véspera da entrega. Portanto, se um aluno pretender fazer o trabalho individual porque tem uma justificação de força maior, deve contactar o docente do seu turno prático o mais brevemente possível.)

O trabalho é implementado em Java, nomeadamente em JDK, instalado nos laboratórios das aulas práticas, em Windows e Linux.

Há duas versões base do trabalho, tal como descrito abaixo:

- **Na Versão I completa**, avaliada entre 0 e 20 valores, não é permitido fazer uso do package *java.util*, à excepção da classe *Scanner*;
- **Na Versão J mínima**, avaliada entre 0 e 15 valores, podem ser usadas todas as interfaces e classes do Java.

Os estudantes têm também a opção de submeter uma versão com algumas interfaces e classes implementados em Versão I e outras em versão J (versão mista), o que lhes permitirá entregar um trabalho que será cotado para um valor máximo entre 15 e 20. Neste caso, durante a avaliação do trabalho, os docentes irão verificar quais os Tipos Abstractos de Dados que foram implementados sem recurso ao package *java.util*, atribuindo uma nota de acordo com as referências encontradas. Os alunos que optarem por esta versão mista, devem referir na secção I do relatório que o trabalho é uma versão mista e quais os Tipos Abstractos de Dados que foram implementados sem recurso ao *java.util*.

4.1 Entregas e faseamento

Como já descrito acima, o trabalho será desenvolvido incrementalmente, em duas fases diferentes, com entregas já marcadas.

Na primeira fase, os estudantes deverão desenhar a sua solução utilizando os tipos abstractos de dados e respectivas estruturas de dados apresentados até ao final da 5^a aula teórica (ver tabela de aulas teóricas no moodle para confirmação). **Há ainda duas operações que não são implementadas (secções 3.13 e 3.15) nesta fase.**

Todas as operações deverão ser implementadas na 2^a fase. A escolha dos tipos abstratos de dados deve ser revista na 2^a fase para melhorar a performance do trabalho da 1^a para a 2^a fase, tendo em conta os requisitos descritos na secção 2.4. É expectável que alguns dos tipos abstratos de dados usados na solução da 1^a fase sejam alterados na 2^a fase (Será dada mais informação sobre este assunto nas aulas práticas e teóricas).

O programa deve ser entregue nos concursos do Mooshak para o efeito. Adicionalmente, na última fase deverá ainda ser entregue, no Moodle, um relatório final do trabalho (secção 4.2).

Cada grupo de **dois alunos** (equipa de trabalho) deve registar-se nesses concursos. O nome do utilizador (login no mooshak) deve ser a concatenação dos números dos alunos que compõem o grupo, separados por _ (o primeiro número a colocar é o menor). Por exemplo, os alunos nº 3435 e nº 3434 do devem ter como nome utilizador no Mooshak **3434_3435**. **Só serão aceites, como entregues para avaliação, os trabalhos cujo utilizador no concurso do Mooshak siga estas regras.**

A entrega no Mooshak, é constituída por um zip contendo o código fonte devidamente comentado utilizando as regras para geração de javadoc, tal como descrito na secção 5. O nome e número dos alunos que compõem o grupo deverá ser inserido no cabeçalho de todos os ficheiros entregues, da seguinte forma:

```
/**  
 * @author NOMEALUNO1 (NUMEROALUNO1) emailALUNO1  
 * @author NOMEALUNO2 (NUMEROALUNO2) emailALUNO2  
 */
```

Para os programas e relatório serem avaliados, os alunos devem ter cumprido o código de ética do DI, e ter, no final do prazo, uma entrega de um trabalho no concurso do Mooshak associado a essa entrega (fase e concurso de avaliação). Cada grupo pode submeter o seu código mais que uma vez. **APENAS** a última submissão será avaliada.

4.2 Relatório

Na última fase deverá ainda ser entregue, no Moodle, um relatório final do trabalho, contendo o seguinte:

- Para cada um dos Tipos Abstratos de Dados (Interfaces) definidos no trabalho, uma justificação curta para as implementações realizadas para os mesmos, especificamente para as estruturas de dados escondidas;
- Para cada uma das operações descritas na secção 3, uma descrição do comportamento do trabalho, em termos das operações efetuadas sobre as estruturas de dados;
- O estudo das complexidades temporais das operações descritas na secção 3, no melhor caso, no pior caso e no caso esperado;
- O estudo da complexidade espacial da solução proposta.

Não são definidos templates nem formatos para o relatório. Toda a informação necessária a incluir no relatório está aqui indicada, para além da referência à versão associada ao trabalho (I, J ou mista).

5 Requisitos do Mooshak

5.1 Regras a satisfazer pelo programa

Para submeter o trabalho ao Mooshak, é necessário que o programa fonte respeite as quatro regras seguintes:

- O código fonte completo (ficheiros *.java) tem de ser guardado num único arquivo ZIP;
- A classe principal tem de se chamar Main e tem de estar localizada na raiz do arquivo (i.e., tem de pertencer ao pacote principal (*default*)).
- As pastas correspondentes aos pacotes do trabalho têm de estar localizadas na raiz do arquivo.
- O programa só pode criar ficheiros na directória corrente.

5.2 Como preparar o arquivo ZIP

Para evitar problemas causados pela dimensão do ficheiro submetido, somente o código fonte (ficheiros *.java) deve ser enviado para o servidor. Assuma, para exemplificar, que o código fonte do trabalho está dentro de uma pasta chamada /home/me/aulas/AED/trabalhoFinal/src e que, dentro dessa pasta, há um ficheiro e duas sub-pastas.

```
Main.java  
dataStructures  
UniBedrooms
```

Nesse caso, o arquivo poderia ser construído (em Linux) com o seguinte comando:
`zip aed.zip Main.java dataStructures/*.java socialNet/*.java`
executado na pasta /home/me/aulas/AED/trabalhoFinal/src.

6 Avaliação

O trabalho é obrigatório para todos os alunos que não têm frequência e dá frequência. A avaliação incidirá sobre todos os aspetos: concepção, qualidade e eficiência da solução, modularidade, estrutura e documentação do código, qualidade do relatório final, etc. A 1^a fase vale 10% da nota final da unidade curricular e a 2^a fase vale 20% da mesma nota. Se uma das fases não for entregue dentro do prazo definido, a nota associada a essa fase será 0 (zero).

6.1 Testes de avaliação

O trabalho terá de satisfazer todos os requisitos especificados na secção 3. Essa verificação será efetuada automaticamente pelo sistema Mooshak, com os Testes de Avaliação.

Em cada fase:

- se o programa não passar nenhum dos Testes de Avaliação, os seus autores obterão a nota de 0 (zero) na fase em questão;
- se o programa obtiver o número mínimo de pontos requerido para avaliação da fase, será objeto de avaliação preliminar que deverá ser confirmada no final do semestre, numa discussão final.

6.2 Verificação de autoria

O código submetido pelos alunos como fazendo parte do seu trabalho deve ser desenvolvido **de raiz pelos mesmos**, expressamente para o trabalho em questão. As exceções a esta regra serão os interfaces e classes disponibilizados na página Moodle da disciplina. Se por uma razão de força maior, **o trabalho contemplar interfaces ou classes que não foram desenvolvidas de raiz pelos alunos**, estas deverão ser devidamente referidas no código e no relatório final e poderão condicionar a nota do trabalho.

Dica: Para iniciar o desenvolvimento do trabalho deve criar-se um projeto vazio no IDE utilizado, inserindo, conforme as necessidades, os TADs e classes disponíveis na página da disciplina.

No final do semestre, todos os grupos em posição de obter frequência serão submetidos a uma discussão do trabalho, para verificação de autoria. Nesta discussão, os membros do grupo poderão ser questionados sobre **qualquer das classes e interfaces submetidas ao Mooshak em qualquer das fases de submissão**.

Se se detetar:

- que um trabalho não foi realizado apenas pelos alunos que o assinaram;
- que um aluno assinou um trabalho que não realizou; ou
- que a distribuição das tarefas pelos membros do grupo não foi equilibrada,

esse trabalho será anulado e **nenhum** dos elementos do(s) grupo(s) envolvido(s) obterá frequência. Se um aluno faltar à discussão do trabalho que assinou, não obterá frequência, reprovando à disciplina.

6.3 Resumo das datas importantes

- **Submissão da 1ª fase do trabalho, no Mooshak:** entre 30 de outubro e 3 de novembro de 2023. No dia 3 de novembro, a entrega eletrónica deve ser realizada até às **17h**. Nesta fase haverá uma tolerância de entrega até 5 de novembro às 23:59, a pedido da Comissão Pedagógica da LEI/MIEI. Durante o período de tolerância não serão esclarecidas dúvidas relativas à fase em questão.
- **Submissão da 2ª fase do trabalho, no Mooshak:** entre 27 e 30 de novembro de 2023 (até às 17h). No dia 30 de novembro, a entrega eletrónica tem de ser realizada até às **17h**.
- **Marcação das discussões finais:** 3 de dezembro de 2023, no Moodle.
- **Realização das discussões:** entre 4 e 8 de dezembro, **em princípio**, dentro do horário da disciplina (das aulas teóricas ou práticas). Os alunos devem verificar todos os horários possíveis que têm em AED para confirmar a data e hora da discussão.