Licenciatura em Engenharia Informática
Programação Orientada por Objectos

Bar Chart Racer

Trabalho Prático de Avaliação

(versão inicial, 7 de maio de 2022) (versão completa, 16 de maio de 2022)

Esta é a versão completa do enunciado

For the loser now
Will be later to win
For the times they are a-changin'
– The Times They Are A-Changin' – Bob Dylan

https://youtu.be/90WD_ats6eE

https://genius.com/14139032

https://en.wikipedia.org/wiki/The_Times_They_Are_a-Changin%27_(song)

1 Introdução

Este enunciado é algo extenso. Tal significa que deve ser lido mais do que uma vez e com muita atenção. O que é pedido está classificado em três tipos de **REQUISITOS**:

1. **ESSENCIAIS**
2. **NÃO ESSENCIAIS**
3. **QUE IMPLICAM PENALIZAÇÕES.**

Os essenciais permitem obter 10 valores, mas apenas desde que não estejam presentes motivos para penalizações, ou seja, se cumprir totalmente todos os requisitos que podem implicar penalizações. Para não ter penalizações basta ter cuidado a ler o enunciado e respeitar o que é requerido. Assim, poderá garantir que o programa entregue não irá oferecer motivos para a aplicação de penalizações, as quais podem fazer baixar a nota para níveis certamente indesejados. Os requisitos não essenciais permitem obter até 17 valores. Os 18, 19 e 20 valores ficam apenas para quem fez tudo o que é pedido de forma absolutamente impecável e inventou algo mais para fazer, os chamados **EXTRAS**. Esses extras devem acrescentar algo de significativo às funcionalidades do programa e utilizando apenas conteúdos lecionados na unidade curricular. Não vale a pena fazer extras sem ter TUDO o que é pedido feito. Também não vale a pena fazer os requisitos não essenciais sem ter feito os essenciais. Em resumo:

- Só pode ter mais do 10 valores se tiver cumprido totalmente os requisitos essenciais e não tiver penalizações.
- Só pode ter mais do 17 valores se tiver cumprido totalmente os requisitos essenciais e também os não essenciais e não tiver penalizações.

Para mais informações pode e deve ler as regras de avaliação no guia de funcionamento da unidade curricular.

2 Objetivo geral

Neste trabalho prático pretende-se criar um programa para uma "corrida de gráficos de barras"(um *Bar Char Racer*). O programa proposto é uma variante do proposto por Kevin Wayne, apresentado em `http://nifty.stanford.edu/2020/wayne-bar-chart-racer/` e da qual iremos utilizar os ficheiros de dados ("data files") disponibilizados nessa mesma página.

Resumidamente, o programa deverá ser capaz de ler dados históricos sobre um determinado tema, por exemplo as maiores cidades ou as marcas mais valiosas, e apresentar gráficos de barras de uma forma animada reflectindo os dados ao longo do tempo. Cada gráfico de barras apresentará os primeiros *n* elementos (o maior/com mais valor/etc.) em cada conjunto em cada momento. Para tal, esse conjunto de elementos em cada momento (um *dataset*) terá de ser ordenado por algum critério.

Seguem-se os requisitos para o trabalho. Antes de escrever código, leia-os atentamente mais do que uma vez. Antes de entregar o trabalho leia-os, pelo menos, mais uma vez. O vídeo no endereço acima mencionado ilustra o resultado pretendido (o som não é requerido).

3 Requisitos Essenciais

A não realização de qualquer destes requisitos implica uma classificação negativa no trabalho e a não contabilização dos requisitos não essenciais.

Req. E1 - Estrutura do programa (2,0 valores) O programa deve seguir o padrão dado nas aulas. Nomeadamente deve haver um package com o nome `pt.ipbeja.po2.chartracer.gui` que conterá todo o código que depende do JavaFX, e um package `pt.ipbeja.po2.chartracer.model` que deverá conter todo o código que não depende de uma interface com o utlizador. Este código deve ler e escrever ficheiros e comunicar com o package `gui` utilizando os métodos numa interface com o nome `pt.ipbeja.po2.chartracer.model.View`.

Req. E2 - Testes do model (3,0 valores) Deve escrever o código de teste para os seguintes cenários. Cada um dos métodos deve ter um nome com o formato `testN` em que `N` é o número do cenário na seguinte lista (teste1, teste2, etc.):

1. Ler um ficheiro de dados e verificar que está bem lido (os dados lidos são os esperados);

2. Ordenar os dados lidos relativos a um dado momento no tempo. Esta ordenação deve ser a "ordenação natural", ou seja, a que está definida para os objetos utilizados para guardar os dados do ficheiro. Corresponde à implementaçao da interface Comparable<E>. O teste deve ser feito para o primeiro e também para o último momento (conjunto de dados) no ficheiro de dados `cities.txt`, disponível em `http://nifty.stanford.edu/2020/wayne-bar-chart-racer/cities.txt`:

   ```
   1500,Beijing,China,672,East Asia
   1500,Cairo,Egypt,400,Middle East
   1500,Cuttack,India,140,South Asia
   1500,Fez,Morocco,130,Middle East
   1500,Gauda,India,200,South Asia
   1500,Guangzhou,China,150,East Asia
   1500,Hangzhou,China,250,East Asia
   1500,Istanbul,Turkey,200,Europe
   1500,Nanjing,China,147,East Asia
   1500,Paris,France,185,Europe
   1500,Tabriz,Iran,250,Middle East
   1500,Vijayanagar,India,500,South Asia
   ```

   e

   ```
   2018,Beijing,China,22674,East Asia
   2018,Cairo,Egypt,19850,Middle East
   2018,Delhi,India,27890,South Asia
   2018,Dhaka,Bangladesh,19633,South Asia
   2018,Karachi,Pakistan,18185,South Asia
   2018,Mexico City,Mexico,21520,Latin America
   2018,Mumbai,India,22120,South Asia
   2018,New York,United States,18713,North America
   2018,Osaka,Japan,20409,East Asia
   2018,Shanghai,China,25779,East Asia
   2018,São Paulo,Brazil,21698,Latin America
   2018,Tokyo,Japan,38194,East Asia
   ```

3. Escrever os primeiros cinco dados já ordenados para um ficheiro e verificar que estão bem escritos lendo o ficheiro e comparando. O teste deve ser feito para o primeiro e também para o último momento (conjunto de dados) no ficheiro de dados cities.txt.

Req. E3 - Interface com o utilizador (2,0 valores) O programa deve ter uma interface gráfica utilizando JavaFX 17 e apresentar um gráfico de barras para o primeiro conjunto de dados lidos do ficheiro indicado pelo utilizador. Os gráficos têm obrigatoriamente de ser desenhados com objectos do tipo Shape, por exemplo, Rectangle, Line e Text.

Req. E4 - Animação (3,0 valores) O programa deve apresentar, em sucessão, os gráficos de barras para todos os conjuntos de dados lidos do ficheiro escolhido pelo utilizador, com uma apresentação gráfica semelhante à que surge no vídeo em http://nifty.stanford.edu/2020/wayne-bar-chart-racer/cities-sound.mp4. Note que os dados, em cada gráfico, devem estar ordenados.

4 Requisitos Não Essenciais

Estes requisitos só são contabilizados se os requisitos essenciais estiverem todos feitos.

Req. NE1 - Aspeto visual dos gráficos (2,5 valores) O programa deve ter um menu com o nome `Skin` e com um *check menu item* para cada aspeto visual dos gráficos. Note que para cada animação, com base num mesmo ficheiro de dados, deve utilizar um único *skin*. Quando o utilizador escolhe o ficheiro, será iniciada a visualização com a skin que esteja escolhida no item respetivo do menu.

Req. NE2 - Criação de ficheiro de dados estatísticos (2,5 valores) O programa deve ter um menu com o nome `Data` e com um check menu item com o nome `Generate File`. Quando este item de menu está seleccionado, o programa gera um ficheiro de texto de dados estatísticos logo que o ficheiro de dados é indicado pelo utilizador (*vide* E3). O ficheiro de dados estatísticos deve apresentar os seguintes dados e formato em que os ?? devem ser substituídos pelos números respetivos:

> Number of data sets in file: ??
> First date: YYYY/MM/DD
> Last date: YYYY/MM/DD
> Average number of lines in each data set: ??
> Number of columns in each data set: ??
> Maximum value considering all data sets: ??
> Minimum value considering all data sets: ??

Req. NE3 - Utilização de polimorfismo (2,0 valores)
No pacote `pt.ipbeja.po2.chartracer.gui` utilize várias classes e polimorfismo para definir as várias skins requeridas em NE1. Não pode testar o tipo dos objetos. Por exemplo, não pode utilizar os métodos `instanceOf`, `getClass` ou algo semelhante.

5 Requisitos que podem implicar penalizações ou bonificações

O incumprimento de um ou mais dos seguintes requisitos implica a atribuição da penalização ou bónus especificado. No caso de uma penalização, será impossível obter uma clasificação superior a 17.

Req. PB1 - Métodos com mais de 30 pontos e vírgulas (-4,0 a -2,0 valores)
Nenhum método deve ter mais de trinta pontos e vírgulas (";"). Note que um ciclo `for` tem dois pontos e vírgulas. Deve preferir métodos pequenos. Deve optar por mais métodos pequenos em lugar de menos métodos grandes.

Req. PB2 - Regras de estilo e elegância do código (-4,0 a 2,0 valores) O código entregue deve respeitar as regras de estilo e a utilização de programação funcional easserts poderá ser valorizada:

programação funcional Utilização de funções *filter*, *map* e *reduce*.

Utilização de asserts Sempre que conveniente, os métodos devem utilizar asserts para testar os valores dos parâmetros ou valores de outras variáveis.

Identificadores em inglês Os nomes de todas as variáveis, constantes, métodos e classes devem estar em inglês.

Nomes das variáveis, métodos, constantes e classes Utilização de letras minúsculas/maiúsculas e informação transmitida pelos nomes; por exemplo, `box` é um melhor nome para uma caixa do que `b` ou `xyz`.

Constantes Não deve utilizar constantes literais (por exemplo, 20, -300, 45.4) para representar valores que podem ser melhor representados por um nome. Nesses casos deve definir constantes utilizando a palavra reservada **`final`**.

Os espaçamentos Depois das vírgulas e antes e depois dos operadores.

Indentação coerente Para cada bloco, incluindo posicionamento e utilização coerente das chavetas.

Utilização de parâmetros Sempre que conveniente, os métodos devem utilizar parâmetros de modo a evitar duplicação de código e a utilização de variáveis partilhadas externas para passar informação entre métodos.

Repetição de código Deve ser evitada a repetição de código.

Comentários Antes de cada método deve escrever um comentário javadoc (`/** */`) que explique o que o método faz, os respectivos parâmetros e valor devolvido (se existentes). Os comentários podem estar em português mas tente colocá-los em inglês. Os comentários têm de incluir *tag*s `@param` para cada parâmetro e `@return` quando necessário.

Utilização do `this` Utilização explícita do **`this`**na chamada de métodos e na utilização de atributos. Por exemplo, **`this`**`.add(line); `**`this`**`.total++;`.

Ocultação de informação Todos os atributos devem ser definidos como `private`.

Req. PB03 - Auto-avaliação (-2,0 a 0,0 valores) Num ficheiro "auto-aval.txt", deve indicar a sua classificação em cada requisito e a classificação resultante. No mesmo ficheiro **deve indicar quantas horas gastou a fazer este trabalho, incluindo o tempo em aulas e fora das aulas**.

Req. PB04 - Identificação (-1,0 valores) Todos os ficheiros com código (ficheiros *.java) têm de conter, em comentário no início, o nome e número de aluno do autor.

Req. PB05 - Quantidade de autores (-20,0 valores) O trabalho deve ser realizado individualmente ou em grupos de dois; quando for feito por dois alunos ambos os alunos terão de entregar a mesma resolução no prazo definido.

Req. PB06 - Nome do projecto (-1,0 a 0,0 valores) O nome do projecto no InteliJ tem de respeitar um dos seguintes formato conforme o trabalho seja feito por um ou dois autores.

```
Numero_PrimeiroUltimo_TP_PO2_2021-2022
Por exemplo: 1232_VanessaAlbertina_TP_PO2_2021-2022
Numero_PrimeiroUltimo_Numero_PrimeiroUltimo_TP_PO2_2021-2022
```

Por exemplo:

```
12345_VanessaAlbertina_TP_PO2_2021-2022
12345_VanessaAlbertina_12346_AnaLee_TP_PO2_2021-2022
```

Note que `Primeiro` e `Ultimo` representam nomes de autores. `Numero` representa um número de aluno do autor. O trabalho entregue deve ser um zip gerado pelo IntelliJ com o comando em File->Export->Project to Zip File... .

Req. PB07 - Originalidade (-20,0 a +3,0 valores) A originalidade das soluções encontradas para resolução dos requisitos essenciais e não essenciais, comparativamente com outros trabalhos entregues ou disponíveis na internet, pode ser valorizada num máximo de 3 valores e penalizada num máximo de 20,0 valores. Para efeitos de aplicação desta valorização ou penalização, a originalidade é determinada pelas diferenças ou semelhanças entre o trabalho a ser avaliado e os restantes trabalhos entregues por outros alunos e disponíveis noutras fontes; pode ser utilizado código encontrado na internet desde que o mesmo seja referenciado no trabalho entregue e os professores autorizem a sua utilização;

Req. PB08 - Entrega do trabalho e eventual apresentação (-20,0 a 0,0 valores) O trabalho entregue tem de ser uma directoria do projecto IntelliJ pronto a funcionar, sob a forma de um único ficheiro zip contendo toda a directoria mais o ficheiro de auto-avaliação. Antes de entregar, verifique que sabe pôr a funcionar o código no ficheiros zip entregue. Para tal parta desse ficheiro, descompacte-o, e leia-o no IntelliJ. Tal poderá ser requerido numa eventual apresentação individual do trabalho. Se não conseguir pôr a funcionar o conteúdo do ficheiro zip entregue (no moodle e no Teams) a classificação no trabalho poderá ser de zero valores. A entrega tem de ser feita num ficheiro no formato zip com o nome indicado no Requisito PB07 e sempre por **duas** vias:

1. No moodle na página da disciplina em `https://cms.ipbeja.pt/mod/assign/view.php?id=223421`.
2. Por mensagem directa para os três docentes da unidade curricular no Teams.

Note que pode entregar mais do que uma vez, desde que dentro do prazo. A última entrega dentro do prazo é a única que conta.

Req. PB9 - Data limite de entrega em época normal O trabalho deve ser entregue no moodle e também por chat no Teams até às **24:00 de 22 de junho de 2022.**. Não deve assumir que o relógio do sistema está igual a qualquer outro. Assim, a entrega depois da uma hora (da madrugada) do dia seguinte ao prazo de entrega implicará zero valores no trabalho.

Req. PB10 - Data limite de entrega em época de recurso O trabalho deve ser entregue no moodle e também por chat no Teams até às **24:00 de 7 de julho de 2022.**. Não deve assumir que o relógio do sistema está igual a qualquer outro. Assim, a entrega depois da uma hora (da madrugada) do dia seguinte ao prazo de entrega implicará zero valores no trabalho.

Finalmente, note que a originalidade (Req. PB07 da Secção 5) é uma forma de subir a pontuação e contribuir para obter mais do que 17 valores.

6 Nota importante

Todas as contribuições para o trabalho que não sejam da exclusiva responsabilidade dos autores têm de ser identificadas (com comentários no código e referências no relatório) e

a sua utilização bem justificada e expressamente autorizada pelo professor responsável. Justificações insatisfatórias, ausência de autorização, ou ausência de referências para trabalhos ou colaborações externas utilizadas serão consideradas fraude sempre que o júri da unidade curricular considere os trabalhos demasiado semelhantes para terem sido criados de forma independente e **tal terá como consequência a reprovação direta na unidade curricular de todos os alunos envolvidos sem possibilidade de realizar época de recurso ou especial**. Esta decisão de reprovação pode suceder até final do semestre logo que seja detectada a situação e independentemente de já ter sido ou não publicitada uma nota aos alunos. Assim, nenhum aluno deve dar cópia do seu código (ainda que em fase inicial) a outro. Por essa mesma razão não é boa ideia partilhar código com os colegas, quer directamente quer através do fórum. Naturalmente, cada aluno pode e deve trocar impressões e esclarecer dúvidas com todos os colegas, mas deve saber escrever todo o código sozinho. A classificação neste trabalho prático fica dependente de uma eventual apresentação individual do mesmo, tal como previsto no guia da unidade curricular e pode ser alterada em resultado do desempenho do aluno nessa mesma apresentação.

Caso o júri detecte, antes do final do semestres, algum tipo de ocorrência que considere anómala, a avaliação do trabalho poderá ser repetida ou substituída por um novo elemento de avaliação sendo, em qualquer dos casos, contabilizado apenas esta segunda avaliação.

Finalmente, antes de entregar leia com MUITA atenção todo o enunciado. A falha de parte do exigido num requisito implica o não cumprimento desse requisito e consequente penalização. A programação é também a atenção aos detalhes.

Bom trabalho!

João Paulo Barros