Instituto Politécnico de Beja — Escola Superior de Tecnologia e Gestão

Simulador de Pandemia (Parte 1)

Trabalho Prático de Avaliação 1

15 de abril de 2020

Everyone seeks damages and everyone agrees
That these are 'classic symptoms of a monetary squeeze'
On ITV and BBC they talk about the curse
Philosophy is useless theology is worse
History boils over there's an economics freeze
– Industrial Disease – Dire Straits

https://youtu.be/g3X3rKtruSg

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 têm de ter uma dificuldade superior ao que é pedido embora devam continuar na linha do que é pedido e utilizando apenas os conteúdos que já foram dados nas aulas. 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 O Jogo

Neste trabalho prático, pretende-se ficar com um programa que permita simular o contágio numa epidemia/pandemia. O trabalho prático de avaliação 2 consistirá em continuar este enunciado.

Deve ficar atento ao fórum da unidade curricular no moodle. Será lá que surgirão os esclarecimentos sobre este enunciado e é lá que deve colocar as dúvidas. Nunca deve é colocar código feito por si.

As regras gerais a que o programa simulador deve obedecer neste primeiro trabalho prático são as seguintes:

1. É uma simulação de contágio e consequente cura;
2. Cada pessoa (`Person`) será representada por uma célula (`Cell`) numa grelha invisível;
3. Cada pessoa desloca-se aleatoriamente nessa grelha;
4. Cada pessoa é visualizada como um pequeno quadrado numa janela (`Pane`) de fundo liso; o quadrado terá uma cor diferente conforme represente uma pessoa saudável, doente ou recuperada;
5. Esses quadrados deslocam-se na janela;
6. As quantidades iniciais de pessoas saudáveis e de pessoas doentes deve ser um parâmetro do programa;
7. O utilizador pode indicar outros parâmetros antes de iniciar a simulação (indicados no restante enunciado);
8. Em cada célula (`Cell`) só pode estar uma pessoa;
9. A simulação funciona passo a passo;
10. Em cada passo de execução qualquer quantidade de pessoas se pode deslocar para uma posição adjacente;
11. No final de cada passo de execução, as pessoas saudáveis que estão ao lado de uma pessoa doente passam a estar doentes antes do passo de execução seguinte;
12. Há dois parâmetros relacionados com o tempo em que cada pessoa pode estar doente: o tempo mínimo e o tempo máximo; o tempo em que cada pessoa está doente é um valor aleatório mas dentro desse intervalo;
13. A simulação inicia-se premindo um botão ou escolhendo uma item (`MenuItem`) de um menu (`Menu`);

14. A simulação termina quando terminam as iterações definidas ou quando o utilizador prime o botão "Stop".

Seguem-se os requisitos para o trabalho. Antes de escrever o código, leia-os atentamente mais do que uma vez. Antes de entregar o trabalho, leia-os, pelo menos, mais uma vez.

3 Requisitos Essenciais

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

Req. E1 - Classes a definir (0,5 valores) A implementação do jogo deve respeitar os seguintes pontos:

1. Na `package model` deve haver, pelo menos, as seguintes classes: `World`, `Cell`, `EmptyCell`, `Person`, `HealhyPerson`, `SickPerson`, `ImmunePerson`;
2. Na `package model` deve haver, pelo menos, a interface `View`;
3. Na `package gui` deve haver, pelo menos, as seguintes classes: `GuiStart`, `ContagiousBoard`, `WorldBoard`;

Req. E2 - Grelha no model (0.75 valores) A classe `World` deve conter uma grelha de Cell; algumas destas Cell serão pessoas pelo que serão objectos de classes que herdam do tipo Person;

Req. E3 - Código de teste movimento (0.75 valores Escrever código de teste para o seguinte cenário: Movimento de uma pessoa para um célula vazia; tentativa de de movimento de uma pessoa para fora do tabuleiro em quatro casos: (1) demasiado para a esquerda, (2) demasiado para a direita, (3) demasiado para cima e (4) demasiado para baixo;

Req. E4 - Código de teste contágio (3,5 valores Escrever código de teste para o seguinte cenário: movimento de uma pessoa doente para uma célula vazia e contágio de todas as células com pessoas saudáveis, e só essas, que estão justapostas; o `setup` do método deve garantir que após o movimento da pessoa doente ela (1) fica em contacto com, pelo menos, uma pessoa saudável; (2) fica em contacto com, pelo menos, uma outra pessoa doente e que (3) fica em contacto com, pelo menos, uma pessoa recuperada; as pessoas saudáveis em contacto devem ficar doentes; as pessoas doentes em contacto devem manter-se doentes, as pessoas recuperadas não devem ficar doentes; este teste deve ser feito para pelo menos dois `setups` (dois movimentos);

Req. E5 - Apresentação dos Movimentos(2,0 valores) Deve ser possível as pessoas mudaram de posição e ver a janela com os quadrados correspondentes às pessoas a moverem-se em cada passo de execução de forma aleatória; este movimento deve ser parametrizado pelo menos na rapidez e tipo de movimento (com mais ou menos mudanças de sentido e direcção);

Req. E6 - Interface (1,5 valores) A interface do programa deve ter um menu com itens que permite iniciar e terminar o movimento das pessoas no *model* e dos correspondentes quadrados na *view*, bem como indicar a quantidade inicial de pessoas; est interface deve estar funcional (os comandos terão de funciona correctamente);

Req. E7 - Relatório (1,0 valores) O programa deve ser entregue com um relatório que explique, por palavras, o funcionamento do programa, incluindo um texto para cada classe e cada método.

4 Requisitos Não Essenciais

Estes requisitos só são contabilizados se os requisitos essenciais estiverem totalmente correctos.

Req. N1 - Simulação gráfica e contágio (3,0 valores) O programa efectua o movimento das células no model e o correspondente movimento dos quadrados na view; quando as células doentes tocam nas saudáveis estas ficam doentes no passo de execução seguinte; tal deve ser visualizado na interface gráfica;

Req. N2 - Simulação gráfica e cura (1,0 valores) passado o tempo aleatório de doença as pessoas doentes ficam recuperadas no passo de execução seguinte; tal deve ser visualizado na interface gráfica;

Req. N3 - Gráfico (3,0 valores) O programa mostra, na interface gráfica, um gráfico com a quantidade de doentes em cada passo de execução.

5 Requisitos que podem implicar penalizações ou bonificações

O incumprimento de um ou mais dos seguintes requisitos implica a atribuição da penalização especificada e a automática impossibilidade de obter uma nota superior a 17.

Req. PB1 - Controlo de versões (-20,0 a 0.0 valores) antes do dia 26 de abril a resolução deste trabalho terá de estar num repositório privado no github (`https://github.com/`) e os três docentes terão de ter sido já convidados como colaboradores para esse repositório utilizando os seguintes *emails*:
jpbarros@acm.org;
diogo.manique@ipbeja.pt;
amarantejoao@hotmail.com O trabalho será penalizado se não existirem actualizações consideradas regulares para o repositório; por exemplo, a maior parte do programa não deve surgir no repositório muito perto da data de submissão para avaliação.

Req. PB2 - *View/controller* (*gui*) separada do *model* (-4,0 a -1.0 valores) A *view/controller* (*gui*) apenas deve tratar de comunicar ao *model* o que o jogador fez (por exemplo um clique num botão, uma opção de menu, etc.) e fazer na interface gráfica as alterações pedida pelo *model*. Cada view também pode utilizar *getters* do *model* para obter informação, mas **toda** a informação e lógica do jogo deve estar presente no *model*.

Req. PB3 - Packages (-1,0 valores) Todo o código deve estar definido num *package* com o nome `pt.ipbeja.estig.po2.pandemic`. O código de interface deve estar num *package* com o nome `pt.ipbeja.estig.po2.pandemic.gui`.

O restante código deve estar num *package* com o nome `pt.ipbeja.estig.po2.pandemic.model`.

Req. PB4 - 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. PB5 - Utilização de testes de tipo (-4,0 valores) A utilização do operador `instanceof`, do método `getClass` (`https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#getClass--`), ou algo idêntico que permita saber o tipo de dados de um objecto, será penalizado.

Req. PB6 - Regras de estilo e elegância do código (-4,0 a 1,0 valores) O código entregue deve respeitar as regras de estilo, nomeadamente **todas** as seguintes:

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, métodos e classes devem estar em inglês.

Nomes das variáveis, métodos, constantes e classe 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 e 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.

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 das referências antes do nome das operações. Por exemplo, **`this`**`.add(line)`.

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

Req. PB7 - Auto-avaliação (-2,0 a 0,0 valores) Num ficheiro "auto-aval.txt", deve indicar quais os requisitos que estão **totalmente** cumpridos (os únicos que contam como cumpridos) 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 (trabalho autónomo)**.

Req. PB8 - 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. PB9 - Quantidade de autores (-20,0 valores) O trabalho deve ser realizado **INDIVIDUALMENTE**, apenas pelo aluno que submete a resolução.

Req. PB10 - Nome do projecto (-1,0 a 0,0 valores) O nome do projecto no inteliJ tem de respeitar o seguinte formato. Note que `Primeiro` e `Ultimo` representam o nome do autor. `Numero` representa o número de aluno do autor ou com origem noutras fontes.

```
Numero_PrimeiroUltimo_TP01_PO2_2019-2020
Por exemplo: 1232_VanessaAlbertina_TP01_PO2_2019-2020
```

O trabalho é entregue compactando a directoria do projecto eclipse num ficheiro .zip de forma a que este fique com o nome `Numero_PrimeiroUltimo_TP01_PO2_2019-2020.zip`.

Req. PB11 - 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; se utilizar código encontrado na internet pode referir esse facto indicando a fonte e autor como forma e diminuir ou até evitar totalmente este penalização;

Req. PB12 - 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 e o relatório (3 elementos). 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 por email) 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 PB10 e sempre por **duas** vias:

1. Na página da disciplina.
2. Por *email*, respeitando as seguintes regras: A entrega por *email* é feita para `trabalhos.p2ARROBAgmail.com`. O *email* a enviar deve conter em *attach* um único ficheiro no formato *zip*. O *subject* do *email* deve ser o `nomeDoProjecto`.
3. Pode entregar mais do que uma vez, desde que dentro do prazo. A última entrega dentro do prazo é a única que conta.

Req. PB13 - Data limite de entrega em época normal O trabalho deve ser entregue no moodle e também por email até às **23:55 da data que ficar definida no calendário de avalaiações.**. 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 necessita de realizar mais do que o pedido para obter mais de 17 valores. A criatividade também pode justificar uma melhor classificação pelo que extras originais e sofisticados serão uma boa aposta. Note que estas adições só contam para a classificação do trabalho se forem consideradas suficientemente significativas e se **todos** os requisitos estiverem completamente cumpridos e sem penalizações.

Lembre-se que a originalidade (Req. PB10 do 5) é outra 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 e contabilizada 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