Instituto Politécnico de Beja
Escola Superior de Tecnologia e Gestão
Licenciatura em Engenharia Informática
Programação Orientada por Objectos

Matriz de Palavras

Trabalho Prático de Avaliação

Beja, 16 de abril de 2024

Este enunciado pode sofrer pequenas alterações até à data limite de entrega. Deve estar atento ao Teams.

The place where I come from is a small town
They think so small, they use small words
But not me, I'm smarter than that,
I worked it out
– Big Time, Peter Gabriel

https://www.youtube.com/watch?v=PBAl9cchQac

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** (Secção 3)
2. **NÃO ESSENCIAIS** (Secção 4)
3. **QUE IMPLICAM PENALIZAÇÕES OU BONIFICAÇÕES** (Secção 5).

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.

2 O Jogo

Neste trabalho prático, pretende-se ficar com um jogo de um só jogador em que este tem de encontrar palavras numa grelha de letras. As palavras podem estar na vertical ou na horizontal e em qualquer dos quatro sentidos horizontais e verticais.

Para tal, tem de continuar o código fornecido como base e disponível em `https://cms.ipbeja.pt/mod/resource/view.php?id=256331`.

Caso não utilize o código fornecido como base ou não respeite a estrutura do mesmo será avaliado com zero valores neste trabalho prático.

O jogo que se pretende implementar deve ter as seguintes características:

1. O jogo decorre numa grelha em que cada uma das posições contém uma letra maiúscula;
2. As letras que constam da grelha devem conter algumas palavras escritas na vertical ou na horizontal, e em qualquer dos sentidos;
3. A quantidade de linhas e a quantidade de colunas do tabuleiro de jogo devem ser indicadas pelo jogador no início do programa;
4. O programa lê as palavras de uma lista num ficheiro de texto e utiliza essas palavras para colocar algumas na matriz de letras; tal corresponde a um nível do jogo;
5. O jogo termina quando o jogador tiver encontrado todas as palavras que o jogo conhece e que estão na matriz, ou quando o jogador decidir terminar;
6. No final o programa diz quantas palavras o jogador assinalou e quantas o programa conhecia e que estavam na grelha.

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 essenciais (RE) implica uma classificação negativa no trabalho e a não contabilização dos requisitos não essenciais(RNE).

Pode sempre acrescentar mais métodos e mais classes ao código base dado. Também pode adicionar parâmetros aos métodos dados. Também é recomendado que utilize programação conduzida por testes no que respeita à *package model* pelo que deve escrever mais métodos de teste além dos pedidos no RE3.

RE1 - Classes base utilizadas (0,5 valores) A implementação do jogo deve respeitar o código base e conter todas as classes e interfaces que nele constam.

RE2 - Leitura e colocação das palavras (2,0 valores) O programa deve ler uma lista de palavras de um ficheiro de texto. Serão estas as palavras que o programa conhecerá e deverá colocar na matriz de letras. As restantes posições devem ser preenchidas com letras aleatórias. Note que pode haver outras palavras na grelha além das que o programa selecionou (que podem surgir devido às letras aleatórias, devido à intercecção das próprias palavras colocadas ou por serem partes destas), mas essas não são consideradas neste requisito.

RE3 - Testes a completar (5,0 valores) Deve completar os testes que estão no código base e escrever código de teste para o seguinte cenário:

1. (criação de nivel de jogo com palavras indicadas) Criar um model com 4 palavras. Testar que essas palavras estão na matriz e que o jogo termina quando todas são selecionadas; estas quatro palavras devem ser lidas de um ficheiro de texto pelo método de teste.

RE4 - Detecção de cliques e palavras (1,0 valores) Quando um botão com uma letra e clicado deve mudar a sua cor para amarelo. Quando todos os botões com as letras de uma palavra tiverem sido premidos, o programa deve colocar um fundo verde nos botões com essas letras.

RE5 - Detecção de fim de jogo (1,0 valores) O jogo deve poder terminar quando o jogador tiver encontrado todas as palavras que o jogo conhece e que estão na matriz, ou quando o jogador decidir terminar; No final o programa diz quantas palavras o jogador assinalou e quantas o programa conhecia e que estavam na grelha, bem como a percentagem correspodendente.

RE6 - Gravação da pontuação para ficheiro (0.5 valores) No final de cada jogo, a pontuação obtida deve ser acrescentada ao ficheiro de pontuações. Este é um ficheiro de texto com o nome `scores.txt`. A pontuação é a percentagem de palavras descobertas.

4 Requisitos Não Essenciais

Estes requisitos não essenciais (RNE) só são contabilizados se os requisitos essenciais (RE) estiverem totalmente correctos.

RNE1 - Apresentação dos Movimentos(1,0 valores) Adicione num `Pane` ao lado do mapa e na mesma janela deste, um campo de texto onde são colocadas as jogadas efectuadas. Para tal o tabuleiro deve ter as letras a indicar as colunas e os números a indicar as linhas. Quando a última letra de uma palavra é seleccionada e todas as restantes já tinha sido selecionadas, a palavra deve ser escrita no campo de texto. Por exemplo:

```
(2, C) -> S
(2, B) -> A
(2, A) -> C
(2, D) -> A
casa
```

RNE2 - Escrita de posições (1,0 valores) Adicione a possibilidade de, em qualquer momento, o jogador escrever num ficheiro de texto o seguinte:

1. A data e hora (*timestamp*) da escrita;

2. A matriz de letras;
3. O conteúdo previsto para o campo de texto do RNE 1.

RNE3 - Iniciar novo jogo (1,0 valores) Adicione a possibilidade de, em qualquer momento, o jogador reiniciar o jogo. O programa deverá produzir uma nova matriz de letras com todas as palavras lidas de um ficheiro de texto. O formato do ficheiro é livre mas sempre em texto.

Apenas para época normal

RNE4 - Palavras na diagonal (2,0 valores)
Considere que as palavras também podem estar na diagonal em qualquer dos sentidos e que tal é uma opção disponível num menu do jogo.

RNE5 - Pontuação implementada com Polimorfismo (2,0 valores)
Algumas letras podem ter um bónus associado. Quando uma letra com um bónus associado é detectada surge uma pontuação numa Label no canto superior direito da interface gráfica. Quando uma palavra é totalmente selecionada, essa palavra e o somatório das pontuações associadas a cada letra dessa palavra surgem numa label abaixo da matriz. Por exemplo:

```
"casa" = 20 pontos
```

Utilize uma classe ou interface `Cell` (já está uma classe incompleta no código base) e duas outras classes para definir dois tipos de `Cell`. A estrutura com a matriz no model deverá conter objectos `Cell`.

Apenas para época de recurso

RNE4 - Palavras na diagonal (1,0 valores) — Considere que as palavras também podem estar na diagonal em qualquer dos sentidos e que tal é uma opção disponível num menu do jogo.

RNE5 - Pontuação implementada com Polimorfismo (1,0 valores)
Algumas letras podem ter um bónus associado. Quando uma letra com um bónus associado é detectada surge uma pontuação numa Label no canto superior direito da interface gráfica. Quando uma palavra é totalmente selecionada, essa palavra e o somatório das pontuações associadas a cada letra dessa palavra surgem numa label abaixo da matriz. Por exemplo:

```
"casa" = 20 pontos
```

Utilize uma classe ou interface `Cell` (já está uma classe incompleta no código base) e duas outras classes para definir dois tipos de `Cell`. A estrutura com a matriz no model deverá conter objectos `Cell`.

RNE6 - Matrizes aleatórias (2,0 valores) Através de uma opção no início do jogo, deve ser possivel gerar uma grelha com letras aleatórias. No final o jogo deve ser capaz de indicar que palavras há nessa grelha. Para tal terá de procurar todas as palavras que conhece na grelha.

Exemplos de extras:

- Atráves de uma opção num menu, adicione a possibilidade do programa mostrar onde estão as palavras assinalando uma a uma as letras, como faria um jogador.
- Adicione a possibilidade do programa reproduzir, automaticamente, no final do jogo, os cliques do jogador.
- Detecção de palavras considerando *wildcards*. Por exemplo: se a matriz contiver "apart**ento", a palavra "apartamento"deve ser contada como estando na matriz.

5 Requisitos que podem implicar penalizações e bonificações (RPB)

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.

RPB 1 - View/controller separada do Model e IMPLEMENTAÇÃO do padrão Observer-Observable (-1,0 a -4.0 valores) Cada view/controller apenas deve tratar de comunicar ao model o que o jogador fez (por exemplo um clique num botão) e fazer na interface as alterações pedida pelo *Model*. Cada view também pode utilizar *getters* do *Model* para obter informação. Toda a informação e lógica do jogo deve estar presente no *Model*. Todos o código que utiliza JavaFX deve estararrumado na `package pt.ipbeja.app.ui`).

RPB 2 - Testes de tipos (-2, e polimorfismo -2,0 a 2,0 valores) Não pode utilizar testes ao tipo dos objectos. Em seu lugar, deve utilizar herança e polimorfismo com ligação dinâmica.

RPB 3 - Packages (-1,0 valores) Todo o código deve estar definido num *package* com o nome `pt.ipbeja.estig.po2.sokoban`. O código de interface deve estar num *package* com o nome `pt.ipbeja.app.ui`. O restante código deve estar num *package* com o nome `pt.ipbeja.app.model`.

RPB 4 - Métodos com mais de 30 pontos e vírgulas (-2,0 a -4,0 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. 5 - Utilização do operador `instanceof` ou do método `getClass` (-4,0 valores) A utilização do operador `instanceof`, do método `getClass` ou algo idêntico que permita saber o tipo de dados de um objecto, será penalizado.

RPB 6 - Regras de estilo e elegância do código (-1,0 a -4,0 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`.

RPB 7 - Auto-avaliação (-2,0 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)**.

RPB 8 - 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.

RPB 9 - Quantidade de autores (-20,0 valores) O trabalho deve ser realizado **individualmente** OU em **grupos de dois**. Recomenda-se que seja feito em grupos de dois.

RPB 10 - Nome do projecto (-1,0 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_TP_PO2_2023-2024
Por exemplo: 12345_VanessaAlbertina_TP_PO2_2023-2024
```

O trabalho é entregue compactando a directoria do projecto inteliJ num ficheiro .zip de forma a que este fique com o nome `Numero_PrimeiroUltimo_TP_PO2_2023-2024.zip`.

RPB 11 - 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.

RPB 12 - Entrega e apresentação do trabalho (até -20,0 valores) O trabalho entregue tem de ser uma directoria do projecto eclipse 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 na 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 17 e no moodle.

RPB 13 - Data limite de entrega em época normal O trabalho deve ser entregue no moodle e também por email até às **23:59 de 31 de maio de 2024**. Não deve assumir que o relógio do sistema está igual a qualquer outro pelo que deve entregar sempre pelo menos 15 min ANTES do tempo limite. Assim, a não entrega até à 01:00 de 1 de Junho de 2024 implica zero valores no trabalho.

RPB 14 - Data limite de apresentação em época de recurso O trabalho deve ser entregue no moodle e também por email até às **23:59 de 28 de junho de 2024**. Não deve assumir que o relógio do sistema está igual a qualquer outro pelo que deve entregar sempre pelo menos 15 min ANTES do tempo limite. Assim, a não entrega até à 01:00 de 29 de junho de 2024 implica zero valores no trabalho.

RPB 15 - Código base (-20,0 valores a 3,0 valores) O código entregue tem obrigatoriamente de partir do código base fornecido.

RPB 16 - Relatório (-20,0 valores a 2,0 valores) // TODO

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. 11 da Secção 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**. Assim, **nenhum aluno deve dar código (ainda que em fase inicial) a colegas, mas pode ajudar dizendo O QUE FEZ e COMO FEZ, ambora SEM mostrar ou dar o código**. Nunca 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. Lembre-se que será avaliado em testes prático em frente a um computador. A

classificação neste trabalho prático fica dependente de uma eventual defesa individual do mesmo, tal como previsto no guia da unidade curricular e pode ser alterada em resultado do desempenho do aluno nessa mesma defesa.

Caso o júri detecte algum tipo de ocorrência que considere anómala, a defesa do trabalho poderá ser anulada sendo repetida e contabilizada apenas esta segunda defesa.

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