Universidade Lusófona de Humanidades e Tecnologias
Linguagens de Programação II
LEI / LIG / LEIRT
2020/21 – 1º Semestre
Segunda época
v1.0.0
Bruno Cipriano, Lúcio Studer, Rodrigo Correia, Pedro Alves

The Walking DEISI

Enunciado do projecto de Segunda época

Notas prévias

Nesta avaliação de segunda época devem ser mantidos os grupos que entregaram em primeira época. Mesmo que um dos alunos do grupo não vá a segunda época, não podem entrar novos elementos nos grupos. Elementos que não entregaram projecto em primeira época, deverão entregar sozinhos. Mediante autorização prévia do professor, é possível que os dois elementos do grupo de primeira época possam entregar individualmente em segunda época.

Para poderem entregar o projecto, os alunos têm que se inscrever obrigatoriamente em segunda época até 48 horas antes do prazo de entrega.

Objectivo

Este projecto tem como objectivo o desenvolvimento de uma aplicação (programa) em **linguagem Java** (versão Java 8) aplicando os conceitos de modelação e Programação **Orientada a Objetos** (encapsulamento, herança, polimorfismo, etc.) e os conceitos de programação Funcional (*mapping*, *filtering*, *streams, lambdas*).

O projecto de segunda época corresponde à união dos requisitos do projecto de primeira época, considerando as 3 partes do mesmo. Nos casos em que uma parte do projecto alterou requisitos de partes anteriores, devem ser considerados os requisitos mais recentes.

Para além disso, existem alguns requisitos novos específicos de 2ª época.

Assim, são mantidos para a segunda época os requisitos relativos a:

- tipos de criatura
- regras de movimentação
- regras do jogo
- cálculo de resultados
- estatísticas
- etc.

Notas:

- ao contrário da P3, nesta avaliação de segunda época serão avaliados os movimentos de todos os tipos de criatura.
- nesta segunda época não existe bónus para a criatura extra.
- mantêm-se as restrições técnicas previamente enunciadas (p.e. as estatísticas têm de ser calculadas usando programação funcional).

Tema

Como qualquer bom cliente de um sistema informático, o **Freddie M.** não está completamente satisfeito com o *software* que nos comprou, e pediu mais umas alterações.

Será que o nosso sistema é fácil de mudar?

Novos requisitos

Nesta secção são apresentados os requisitos que se acrescentam. Estes novos requisitos são de implementação obrigatória. Caso os testes respectivos falhem, o projecto terá nota zero.

Existem dois novos requisitos:

- Um novo tipo de Criatura
- Uma alteração às regras de fim de jogo

Universidade Lusófona de Humanidades e Tecnologias
Linguagens de Programação II
LEI / LIG / LEIRT
2020/21 – 1º Semestre
Segunda época
v1.0.0
Bruno Cipriano, Lúcio Studer, Rodrigo Correia, Pedro Alves

Novo tipo de Criatura:

Passa a existir um novo tipo de criatura, com os seguintes detalhes:

- Nome do Tipo: Coelho
- ID (como Vivo): 12
- ID (como Zombie): 13
- Move-se na horizontal e na vertical.
- Alcance/Deslocamento:
    - três casas nos turnos pares
    - duas casas nos turnos ímpares
    - (o primeiro turno é considerado ímpar)
- Outras restrições:
    - O Coelho apenas se pode mover para casas que não tenham equipamento.

Alteração às condições de fim de jogo:

Para além das condições de final de jogo que já existiam em primeira época, passam também a ser válidas as seguintes condições:

- Não existirem mais zombies em jogo

- Não ser possível continuar a jogar com as criaturas que existem em jogo.

A segunda regra identificada acima pode-se traduzir em duas sub-regras:

- Caso o turno actual seja d'Os Vivos, o único Vivo for um Idoso e o turno actual for um turno nocturno, não é possível continuar o jogo.
- Caso o turno actual seja d'Os Outros, o único Zombie existe seja um Vampiro e o turno actual seja diurno, não é possível continuar o jogo.

Visualizador

Considera-se válido o visualizador **v111** (previamente disponibilizado para a terceira parte do projecto de primeira época).

Universidade Lusófona de Humanidades e Tecnologias
Linguagens de Programação II
LEI / LIG / LEIRT
2020/21 – 1º Semestre
Segunda época
v1.0.0
Bruno Cipriano, Lúcio Studer, Rodrigo Correia, Pedro Alves

Testes Unitários Automáticos

Nesta componente, os alunos devem implementar **casos de teste unitários automáticos** para garantir a qualidade do seu projecto. Por "caso de teste" entende-se a definição de um método que verifique se, para certo "*input*" é obtido o "*output*" / *resultado esperado*.

Não existe um número mínimo de testes a implementar.

Esta componente será avaliada pela "percentagem de cobertura" que os testes garantam do código de cada grupo. A "percentagem de cobertura" define-se como a percentagem de código do projecto que é executada quando se corre uma bateria de testes.

Os casos de teste devem ser implementados usando **JUnit 4**, tal como demonstrado nas aulas. Cada caso de teste deve ser implementado num método anotado com `@Test`.

Os testes devem ser implementados em uma ou mais classes com o nome `TestXYZ` (em que `XYZ` é o nome da classe que está a ser testada). Por exemplo, uma classe chamada `ContaBancaria` teria os seus testes numa classe chamada `TestContaBancaria`. As classes de teste devem estar no mesmo package que as classes que estão a ser testadas.

Entrega

O que entregar

Os alunos têm de entregar:

- Um diagrama de classes em **UML** (em formato `png`), assim como eventuais comentários que os alunos decidam fazer no sentido de justificarem as suas escolhas de modelação;

- Ficheiro README.md
    - (ver instruções mais abaixo)

- Ficheiros Java onde seja feita a implementação das diversas classes que façam parte do modelo.

- Ficheiros Java que implementem os testes unitários automáticos (JUnit).

- Um **vídeo** que demonstre o funcionamento do jogo no visualizador.
    - Este vídeo vai funcionar como teste de integração, de forma a mostrar que o código dos alunos interage bem com o visualizador fornecido.
    - O vídeo deve apresentar um jogo completo.
        - Caso os alunos não implementem a 100% a função gameIsOver(), então o vídeo deve apresentar pelo menos 7 turnos do jogo.
    - O vídeo não pode exceder 3 minutos de duração
    - O vídeo deve ser carregado para o youtube e configurado como "**unlisted**" de forma a não aparecer nos resultados de pesquisa. O título do vídeo deve incluir os números dos alunos (ex: "fandeisia-21701234-21704567").
    - O URL do vídeo deve estar no ficheiro `README.md` do projecto.
    - Vídeos com som serão valorizados (ex: narração do que está a acontecer) mas não é obrigatório ter som.

Nota importante: O programa submetido pelos alunos tem de compilar e executar no contexto do visualizador e no contexto do Drop Project.

No visualizador:

- Carregar nos botões **não pode** resultar em erros de *runtime*. Projectos que não cumpram esta regra serão considerados como não sendo entregues. Isto significa que todos os métodos obrigatórios terão de estar implementados e a devolver valores que cumpram as restrições indicadas.

No Drop Project:

- Projectos que não passem as fases de **estrutura** e de **compilação** serão considerados como não entregues.
- Projectos que não passem a fase de **CheckStyle** serão penalizados em 3 valores.

Universidade Lusófona de Humanidades e Tecnologias
Linguagens de Programação II
LEI / LIG / LEIRT
2020/21 – 1º Semestre
Segunda época
v1.0.0
Bruno Cipriano, Lúcio Studer, Rodrigo Correia, Pedro Alves

Ficheiro README.md

O ficheiro README.md do repositório github deve contar os seguintes artefactos:

- Apresentação da imagem do diagrama UML;
- Eventuais comentários que os alunos decidam fazer no sentido de justificarem as suas escolhas de modelação.

Para incluirem a imagem do vosso diagrama no README.md devem usar o código seguinte:

```
![](diagrama.png?raw=true "Diagrama UML")
```

Nota: Projectos que não tenham o ficheiro README.md ou cujo ficheiro README.md não inclua o diagrama UML terão uma penalização de 3 valores na nota final.

Estrutura do projecto

O projecto deve estar organizado na seguinte estrutura de pastas:

```
AUTHORS.txt  (contém linhas NUMERO_ALUNO;NOME_ALUNO, uma por aluno do
grupo)
diagrama-2a-epoca.png

README.md
+ src
|---+ pt
|------+ ulusofona
|---------+ lp2
|------------+ theWalkingDEISIGame
|--------------- TWDGameManager.java

|--------------- Creature.java
|--------------- ...   (outros ficheiros java do projecto)
|--------------- TestXXX.java
|--------------- ... (outras classes de testes)

+ test-files
```

```
|--- somefile1.txt
|--- ...   (outros ficheiros de input para os testes unitários)
```

Como entregar - Repósitorio git (github)

A entrega deste projecto terá que ser feita usando o mesmo repositório git que foi usado para entrega das partes anteriores do projecto. Não serão aceites outras formas de entrega do projecto.

Nota: alunos que tenham feito grupo em primeira época e que pretendam fazer o projecto em separado devem contactar o Professor respectivo para receberem instruções de como proceder em relação ao repositório.

Todos os ficheiros a entregar (seja o relatório / UML, seja código) devem ser colocados no repositório git.

Relembra-se que o user id / username de cada aluno no *github* tem de incluir o respectivo número de aluno.

Os alunos terão acesso a uma página no **Drop Project** a partir da qual poderão pedir para que o estado actual do seu repositório (ou seja, o *commit* mais recente) seja testado.

O DP de **segunda época** estará disponível no URL seguinte:

https://deisi.ulusofona.pt/drop-project/upload/lp2-2021-projecto-2a-epoca

Apenas se consideram entregues projectos que estejam associados ao DP.

Filosofia do uso do DP

O objectivo do DP não é servir de guião àquilo que os alunos têm que implementar. Os alunos devem implementar o projecto de forma autónoma tendo apenas em conta o enunciado e usar o DP apenas para validar que estão no bom caminho. Nas empresas nas quais um dia irão trabalhar não vão ter o DP para vos ajudar. Nesse sentido, os alunos

apenas podem fazer uma submissão a meia cada hora (isto é, têm que esperar meia hora entre cada duas submissões consecutivas).

Prazo de entrega

A entrega deverá ser feita através de um *commit* no repositório git previamente criado e partilhado com o Professor das aulas práticas. Recomenda-se que o repositório git seja criado (e partilhado) o mais rápido possível, de forma a evitar que surjam problemas de configuração no envio.

Para efeitos de avaliação do projecto, será considerado o último *commit* feito no repositório e sincronizado com o Drop Project.

A data limite para fazer o último *commit* (**e respectiva sincronização com o DP**) é o dia **12 de Julho de 2021 (Segunda-feira)**, pelas **23h55m** (hora de Lisboa, Portugal). Recomenda-se que os alunos verifiquem que o *commit* foi enviado (*pushed*), usando a interface *web* do github. Não serão considerados *commits* feitos após essa data e hora.

Avaliação

- A nota final mínima (antes da defesa) é de 9,5 valores.
    - Projectos que não tenham esta nota não poderão ser defendidos em segunda época.

- Para garantir que os alunos que vão à defesa do projecto têm um conjunto mínimo de funcionalidade implementada para que o projecto possa ser defendido com sucesso, alguns dos testes vão ser **obrigatórios**.
    - Estes testes estarão identificados com o sufixo "_OBG".
    - Os alunos que entreguem projectos que não passem **todos os testes obrigatórios** serão reprovados em segunda época.

Universidade Lusófona de Humanidades e Tecnologias
Linguagens de Programação II
LEI / LIG / LEIRT
2020/21 – 1º Semestre
Segunda época
v1.0.0
Bruno Cipriano, Lúcio Studer, Rodrigo Correia, Pedro Alves

Segue-se uma tabela de resumo dos itens de avaliação para a segunda época do projecto:

Tema	**Descrição**	**Cotação (valores)**
Modelo de Classes	Análise do Diagrama UML por parte do Professor. O diagrama deve seguir as regras UML apresentadas nas aulas. Deve também estar em conformidade com o código apresentado.	1
Avaliação funcional	O DP irá testar o Simulador dos alunos considerando situações de abertura de ficheiro, situações de jogada (quer válidas, quer inválidas), situações de detecção da condição de paragem (vitória, empate, etc.).	12
Avaliação dos professores		5
Testes automáticos	Os alunos definem os casos de teste automáticos indicados para esta parte do projecto. A nota será calculada em função da percentagem cobertura obtida pelos testes dos alunos (quanto maior a percentagem de cobertura, maior a nota). Só terão cotação de cobertura os projectos nos quais os testes dos alunos estejam todos a passar.	2

Cópias

Trabalhos que sejam identificados como cópias serão anulados e os alunos que os submetam terão nota zero. Uma cópia numa das entregas intermédias afastará os alunos da restante avaliação de primeira época.

Trabalhos que sejam identificados como cópias de outros projectos de primeira época terão nota zero e a nota dada ao respectivo projecto em primeira época será anulada. Por essa razão recomendamos que não partilhem os projectos entregues em primeira época.

A decisão sobre se um trabalho é uma cópia cabe exclusivamente aos docentes da unidade curricular.

Outras informações relevantes

– Não deve ser implementado qualquer menu (ou interface gráfica) para interação manual com o utilizador. A única interface gráfica prevista é o Visualizador Gráfico disponibilizado pelos docentes.

– Trabalhos que não leiam ou escrevam os ficheiros nos formatos indicados não serão avaliados.

– Recomenda-se que eventuais dúvidas sobre o enunciado sejam esclarecidas (o mais depressa possível) no piazza de Linguagens de Programação II 2020/2021:

- https://piazza.com/class/kfgyzeotqqa3hv

– Existirá uma defesa presencial e individual do projeto. A defesa será feita após a entrega. Durante esta defesa individual, será pedido ao aluno que faça alterações ao código do projecto, de forma a dar resposta a alterações aos requisitos. Se o aluno não conseguir realizar pelo menos metade das alterações pedidas, terá nota zero (0) no projecto.

- A qualquer altura os Professores podem convocar um ou mais elementos de cada grupo para uma prova oral sobre o seu projecto. Esta prova tem como objectivo determinar a participação de cada aluno no projecto, e pode resultar na atribuição da nota zero (0) a um ou mais alunos.

– É possível que sejam feitas pequenas alterações a este enunciado, durante o tempo de desenvolvimento do projeto. Por esta razão, os alunos devem estar atentos ao **Moodle de LP II**.

- Qualquer situação omissa será resolvida pelos docentes da cadeira.