Casos de teste

Test Case outline

Version 2.0

12/11/2019

|  |  |
| --- | --- |
| Grupo\_TP1\_Q: | |
| Rui Bessa | 8150521 |
| Miguel Guimarães | 8150520 |
| José Fernando | 8150498 |

**Histórico de Versões**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | *Grupo TP1 Q* | 12/11/2019 | *Grupo TP1 Q* | *Grupo* *TP1 Q* | 1º versão |
| 2.0 | *Grupo TP1 Q* | 24/11/2019 | *Grupo TP1 Q* | *Grupo TP1 Q* | Entrega |

Tabela de Conteúdos

[1. Introdução 4](#_Toc25524795)

[1.1. Identificador do documento 4](#_Toc25524796)

[1.2. Âmbito 4](#_Toc25524797)

[1.3. Repositório 4](#_Toc25524798)

[1.4. Referências 4](#_Toc25524799)

[1.5 Glossário 4](#_Toc25524800)

[2. Casos de Teste 6](#_Toc25524801)

# 1. Introdução

## 1.1. Identificador do documento

Test\_Case\_Outline\_TP1\_Q

## 1.2. Âmbito

Este relatório foi elaborado para a disciplina de Engenharia de Software II, e tem como objetivo apresentar e explicar os testes de software, caixa preta, realizados pelo grupo “TP1 Q” no primeiro trabalho prático. Pretende-se, também, apresentar a documentação detalhada de todas as definições, planos, abordagens efetuadas e os resultados retornados.

O trabalho desenvolvido passou por realizar os testes unitários de software fornecido pelo docente da disciplina. Este software, consiste na realização de um teste, ou seja, adicionar perguntas a uma coleção (Test) para formar um conjunto de questões. Estas podem ser de três tipos, questões numéricas, questões de verdadeiro ou falso e questões de múltipla escolha.

Para a realização dos testes, foi necessário planear e testar com testes de caixa preta, equivalência e limites, para os métodos addQuestion, numberQuestions, removeQuestion, isComplete, correctAnswers, incorrectAnswers, correctAnswerPecentage, incorrectAnswerPecentage, evaluateAnswer e answer.

Serão de seguida, apresentados todos os dados para o desenvolvimento dos testes de software, tais como, inputs para cada método, outputs esperados e os obtidos, e uma avaliação feita pelos membros do grupo e em alguns casos, acrescentadas observações para possíveis resoluções ou melhorias de problemas encontrados.

## 1.3. Repositório

A hiperligação para o repositório do gitlab, onde foi efetuado o desenvolvimento deste trabalho, é o seguinte:

<http://gitlab.estg.ipp.pt/8150521/esii_tp1_q.git>

## 1.4. Referências

* Slides disponibilizados na plataforma moodle da unidade curricular ESII;
* IEEE Standard for Software Unit Testing;
* S.W.E.B.O.K.;

## 1.5 Glossário

**ECP** (equivalency class partitioning) – este método procura definir um caso de teste com o intuito de descobrir classes com erros, reduzindo assim o número tota de casos de teste a serem desenvolvidos.

Uma classe de equivalência representa um conjunto de estados válidos para condições de entrada.

**BVA** (boundary value analysis) – esta análise leva à escolha de casos de teste que põem à prova os valores limites. Serve de complemento aos testes de partição de equivalência. Em vez de selecionar qualquer elemento de uma classe de equivalência, o BVA leva à seleção de casos de testes nos extremos da classe.

**q1:** Iquestion q1 = new QuestionIQuestion("title", "testDescription", "no")

Como não foi possível instanciar a questão desta forma, deteta-se uma falha no construtor pois não recebe como argumentos o parametro “correct\_answer”.

Dessa forma, é necessário instanciar assim:

“

QuestionIQuestion q1 = **new** QuestionIQuestion(**"title"**, **"testDescription"**);  
q1.setCorrect\_answer(**"no"**);

“

# 2. Casos de Teste

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_01 | **Dependências** | - | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  | - |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP1 | QuestionIQuestion | null | Throws: TestException | Fail |

**Observações:**

Output: “Devia de dar TestException ==> Expected interfaces.exceptions.TestException to be thrown, but nothing was thrown.”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_02 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() **50 vezes** | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP2, ECP3, ECP6 | QuestionIQuestion | q1 | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_03 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() **101 vezes** | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP4, ECP5, BVA5 | QuestionIQuestion | q1 | False | Fail |

**Observações:**

Output: “java.lang.ArrayIndexOutOfBoundsException: 100

FAILURE: Build failed with an exception.”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_04 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA1 | QuestionIQuestion | q1 | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_05 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() **2 vezes** | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA2 | QuestionIQuestion | q1 | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_06 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() **99 vezes** | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA3 | QuestionIQuestion | q1 | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | AQ\_07 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() **100 vezes** | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA4 | QuestionIQuestion | q1 | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_01 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() **50 vezes**  4. Invocar método assertTrue(removeQuestion(49))  5. Invocar método assertNull(getQuestion(49)) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP1 | pos(int) | 49 | True | Pass |

**Observações:**

Para além de testar a remoção da questão na posição 49, confirmou-se também que a referida posição passou a “null”, através do método de teste “assertNull”.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_02 | | **Dependências** |  | |
|  |  |  | |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  |  |
|  | | | | | |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1. Instanciar Test  2. Invocar método assertThrows(removeQuestion(100)) | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP2, ECP4, BVA6 | Pos(int) | 100 | | ArrayIndexOutOfBoundsException | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_03 | | **Dependências** | |  | |
|  |  |  | |  | | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  | |  |
|  | | | | | | |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método assertFalse(removeQuestion(49)) | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | | **Pass/Fail/Untested** |
| ECP3 | Pos(int) | pos = 49 | | False | | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_04 | | **Dependências** | |  | |
|  |  |  | |  | | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  | |  |
|  | | | | | | |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion()  4. Invocar método assertTrue(removeQuestion(0)) | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | | **Pass/Fail/Untested** |
| BVA1 | Pos(int) | pos = 0 | | True | | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_05 | | **Dependências** | |  | |
|  |  |  | |  | | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  | |  |
|  | | | | | | |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() 2 vezes  4. Invocar método assertTrue(removeQuestion(1)) | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | | **Pass/Fail/Untested** |
| BVA2 | Pos(int) | pos = 1 | | True | | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_06 | | **Dependências** | |  | |
|  |  |  | |  | | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  | |  |
|  | | | | | | |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() 99 vezes  4. Invocar método assertTrue(removeQuestion(98)) | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | | **Pass/Fail/Untested** |
| BVA3 | Pos(int) | pos = 98 | | True | | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_07 | | **Dependências** | |  | |
|  |  |  | |  | | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  | |  |
|  | | | | | | |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Instanciar QuestionIQuestion  3. Invocar método addQuestion() 100 vezes  4. Invocar método assertTrue(removeQuestion(99)) | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | | **Pass/Fail/Untested** |
| BVA4 | Pos(int) | pos = 99 | | True | | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | RQ\_08 | | **Dependências** |  | |
|  |  |  | |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  | |  |  |
|  | | | | | |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1. Instanciar Test  2. Invocar método assertThrows(removeQuestion(-1)) | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA5 | Pos(int) | -1 | | ArrayIndexOutOfBoundsException | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_01 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar TestStatistics  2. Criar arrayIQuestions[100]  3. Adicionar ao arrayIQuestions 50 vezes a question Q1 com a resposta do utilizador correta  4. Adicionar arrayIQuestions à instancia TestStatistics  5. assertArrayEquals(arrayIQuestions, demoTestStatistics.correctAnswers()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP1 | IQuestion[100] | 50 Q1 certas | Iquestions[50] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_02 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. add arrayIQuestions[101]  3. assertThrows(ArrayIndexOutOfBoundsException.class, ()-> demoTest.getTestStatistics().correctAnswers()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP2, BVA9 | IQuestion[101] | 101 Q1 | ArrayIndexOutOfBoundsException | Pass |

**Observações:**

Output: “java.lang.ArrayIndexOutOfBoundsException: 100”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_03 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. add arrayIQuestions[100]  3. assertArrayEquals(arrayIQuestions[0], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA1 | IQuestion[100] | 0 Q1 certas | Iquestion[0] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_04 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a primeira correta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA2 | IQuestion[100] | 1º Q1 certa | Iquestion[1] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_05 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a segunda correta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA3 | IQuestion[100] | 2º Q1 certa | Iquestion[1] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_06 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a 99º correta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA4 | IQuestion[100] | 99º Q1 certa | Iquestion[1] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_07 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a ultima correta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA5 | IQuestion[100] | 100º Q1 certa | Iquestion[1] | Fail |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_08 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, as primeiras 99 corretas  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA6 | IQuestion[100] | 99 Q1 certas | Iquestion[99] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_09 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, as ultimas 99 corretas  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA7 | IQuestion[100] | 99 Q1 certas | Iquestion[99] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | CA\_10 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 22-11-2019 | 17h00 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões corretas  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().correctAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA8 | IQuestion[100] | 100 Q1 certas | Iquestion[100] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_01 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar TestStatistics  2. Criar arrayIQuestions[100]  3. Adicionar ao arrayIQuestions 50 vezes a question Q1 com a resposta do utilizador incorreta  4. Adicionar arrayIQuestions à instancia TestStatistics  5. assertArrayEquals(arrayIQuestions, demoTestStatistics.incorrectAnswers()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP1 | IQuestion[100] | 50 Q1 erradas | Iquestions[50] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_02 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. add arrayIQuestions[101]  3. assertThrows(ArrayIndexOutOfBoundsException.class, ()-> demoTest.getTestStatistics().incorrectAnswers()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP2, BVA9 | IQuestion[101] | 101 Q1 | ArrayIndexOutOfBoundsException | Pass |

**Observações:**

Output: “java.lang.ArrayIndexOutOfBoundsException: 100”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_03 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. add arrayIQuestions[100]  3. assertArrayEquals(arrayIQuestions[0], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA1 | IQuestion[100] | 0 Q1 erradas | Iquestion[0] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_04 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a primeira incorreta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA2 | IQuestion[100] | 1º Q1 errada | Iquestion[1] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_05 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a segunda incorreta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA3 | IQuestion[100] | 2º Q1 errada | Iquestion[1] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_06 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a 99º incorreta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA4 | IQuestion[100] | 99º Q1 errada | Iquestion[1] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_07 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, a ultima incorreta  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA5 | IQuestion[100] | 100º Q1 errada | Iquestion[1] | Fail |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_08 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, as primeiras 99 incorretas  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA6 | IQuestion[100] | 99 Q1 erradas | Iquestion[99] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_09 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões, as ultimas 99 incorretas  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA7 | IQuestion[100] | 99 Q1 erradas | Iquestion[99] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | IA\_10 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 23-11-2019 | 18h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar Test  2. Adicionar 100 questões incorretas  3. assertArrayEquals(arrayIQuestions[1], demoTest.getTestStatistics().incorrectAnswers()) | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA8 | IQuestion[100] | 100 Q1 erradas | Iquestion[100] | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_01 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 0 do json  2. Colocar a user\_answer a null  3. assertThrows(Throwable.class, () -> {  question.evaluateAnswer();  }, "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP1 | user\_answer | null | Throw Exception | Fail |
| correct\_answer | “Polymorphism” |

**Observações:**

Output: “O utilizador respondeu null mas o resultado e: Polymorphism ==> Expected java.lang.Throwable to be thrown, but nothing was thrown.”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_02 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 2 do json  2. Colocar a user\_answer a “no”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionYesNo) question).getUser\_answer() + " mas o resultado e: " + ((QuestionYesNo) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP2 | user\_answer | “no” | True | Pass |
| correct\_answer | “no” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_03 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 3 do json  2. Colocar a user\_answer a “2”  3. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| ECP3 | user\_answer | “2” | False | Pass |
| correct\_answer | “1” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_04 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 0 do json  2. Colocar a user\_answer a “”  3. assertThrows(Throwable.class, () -> {  question.evaluateAnswer();  }, "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA1 | user\_answer | “” | Throw Exception | Fail |
| correct\_answer | “Polymorphism” |

**Observações:**

Output: “O utilizador respondeu mas o resultado e: Polymorphism ==> Expected java.lang.Throwable to be thrown, but nothing was thrown.”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_05 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 4 do json  2. Colocar a user\_answer a “”  3. assertThrows(TestException.class, () -> {  question.evaluateAnswer();  }, "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA2 | user\_answer | “” | Throw Exception | Fail |
| correct\_answer | “1” |

**Observações:**

Output: “O utilizador respondeu mas o resultado e: Polymorphism ==> Expected java.lang.Throwable to be thrown, but nothing was thrown.”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_06 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 5 do json  2. Colocar a user\_answer a “a”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA3 | user\_answer | “a” | True | Pass |
| correct\_answer | “a” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_07 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 5 do json  2. Colocar a user\_answer a “A”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA4 | user\_answer | “A” | True | Fail |
| correct\_answer | “a” |

**Observações:**

Output: “O utilizador respondeu A mas o resultado e: a ==> expected: <true> but was: <false>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_08 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 7 do json  2. Colocar a user\_answer a “abcdefghijklmnopqrstvxyz”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA5 | user\_answer | “abcdefghijklmnopqrstvxyz” | True | Pass |
| correct\_answer | “abcdefghijklmnopqrstvxyz” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_09 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 6 do json  2. Colocar a user\_answer a “0”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA6 | user\_answer | “0” | True | Pass |
| correct\_answer | “0” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_10 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 3 do json  2. Colocar a user\_answer a “1”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA7 | user\_answer | “1” | True | Pass |
| correct\_answer | “1” |

**Observações:**

Inicialmente não passou no teste. "O utilizador respondeu 1.0 mas o resultado e: 0.0 ==> expected: <true> but was: <false>”, no json está = "1"

A leitura do ficheiro JSON é que se encontra errada! Adicionando a questão através do método addQuestion() o teste já passou!

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_11 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 8 do json  2. Colocar a user\_answer a “2147483647”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA8 | user\_answer | “2147483647” | True | Pass |
| correct\_answer | “2147483647” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_12 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 2 do json  2. Colocar a user\_answer a “NO”  3. assertTrue(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionYesNo) question).getUser\_answer() + " mas o resultado e: " + ((QuestionYesNo) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA9 | user\_answer | “NO” | True | False |
| correct\_answer | “no” |

**Observações:**

Output: “O utilizador respondeu NO mas o resultado e: no ==> expected: <true> but was: <false>”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_13 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 5 do json  2. Colocar a user\_answer a “b”  3. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA10 | user\_answer | “b” | False | Pass |
| correct\_answer | “a” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_14 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 5 do json  2. Colocar a user\_answer a “B”  3. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA11 | user\_answer | “B” | False | Pass |
| correct\_answer | “a” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_15 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 7 do json  2. Colocar a user\_answer a “abcdefghijklmnopqrstvxy”  3. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionMultipleChoice) question).getUser\_answer() + " mas o resultado e: " + ((QuestionMultipleChoice) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA12 | user\_answer | “abcdefghijklmnopqrstvxy” | False | Pass |
| correct\_answer | “abcdefghijklmnopqrstvxyz” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_16 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 3 do json  2. Colocar a user\_answer a “0”  3. Colocar a correct\_answer a 1  4. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA13 | user\_answer | “0” | False | Pass |
| correct\_answer | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_17 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 8 do json  2. Colocar a user\_answer a “2147483646”  3. Colocar a correct\_answer a “2147483647”  4. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionNumeric) question).getUser\_answer() + " mas o resultado e: " + ((QuestionNumeric) question).getCorrect\_anwser()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA14 | user\_answer | “2147483646” | False | Pass |
| correct\_answer | “2147483647” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | EA\_18 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Ler a questão 2 do json  2. Colocar a user\_answer a “YES”  3. assertFalse(question.evaluateAnswer(), "O utilizador respondeu " + ((QuestionYesNo) question).getUser\_answer() + " mas o resultado e: " + ((QuestionYesNo) question).getCorrect\_answer()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA15 | user\_answer | “YES” | False | Pass |
| correct\_answer | “no” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A1 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Invocar método addQuestion(qn) 2. Invocar o teste assertThrows(QuestionException.class,()-> qn.answer(""); 3. Invocar o teste assertTrue(qn.isDone()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA1. ECP1 | user\_answer | “” | Throws: QuestionException | Fail |

**Observações:**

Com a inserção de uma string vazia, obteve-se a seguinte exceção:

“java.lang.NumberFormatException: empty String”. Seria esperado que retornasse a exceção “QuestionException”, mas a resposta obtida foi “Expected interfaces.exceptions.QuestionException to be thrown, but nothing was thrown”, logo o teste falhou.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A2 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Invocar método addQuestion(qn) 2. Invocar o teste assertThrows(QuestionException.class,()-> qn.answer(null); 3. Invocar o teste assertTrue(qn.isDone()); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA2 | user\_answer | null | Throws: QuestionException | Fail |

**Observações:**

Com a inserção de uma string vazia, obteve-se a seguinte exceção:

“java.lang.NullPointerException”. Seria esperado que retornasse a exceção “QuestionException”, mas a resposta obtida foi “Expected interfaces.exceptions.QuestionException to be thrown, but nothing was thrown”, logo o teste falhou.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A3 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “YesNo”  2. Definir a resposta correta da questão -> qyn.setCorrect\_answer("yes")  3. Invocar método addQuestion(qyn)  4. Invocar o teste assertAll(() -> { assertEquals("yes", qyn.getUser\_answer());  assertTrue(qyn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA3,ECP3 | user\_answer | “YES” | True | Fail |

**Observações:**

Com este teste esperava-se obter o resultado positivo, uma vez que a resposta é a correta. No entanto, este falhou pois o código não está desenvolvido para detetar o facto de a resposta estar em *UpperCase*. O teste retornou o seguinte erro:

“org.opentest4j.AssertionFailedError: expected: <yes> but was: <YES>”.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A4 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “YesNo”  2. Definir a resposta correta da questão -> qyn.setCorrect\_answer("no")  3. Invocar método addQuestion(qyn)  4. Invocar o teste assertAll(() -> { assertEquals("no", qyn.getUser\_answer());  assertTrue(qyn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA4,ECP3 | user\_answer | “NO” | True | Fail |

**Observações:**

Com este teste esperava-se obter o resultado positivo, uma vez que a resposta é a correta. No entanto, este falhou pois o código não está desenvolvido para detetar o facto de a resposta estar em *UpperCase*. O teste retornou o seguinte erro:

“org.opentest4j.AssertionFailedError: expected: <no> but was: <NO>”.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A5 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “YesNo”  2. Definir a resposta correta da questão -> qyn.setCorrect\_answer("yes")  3. Invocar método addQuestion(qyn)  4. Invocar o teste assertAll(() -> { assertEquals("yes", qyn.getUser\_answer());  assertTrue(qyn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA5,ECP3 | user\_answer | “yes” | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A6 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “YesNo”  2. Definir a resposta correta da questão -> qyn.setCorrect\_answer("no")  3. Invocar método addQuestion(qyn)  4. Invocar o teste assertAll(() -> { assertEquals("no", qyn.getUser\_answer());  assertTrue(qyn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA6,ECP3 | user\_answer | “no” | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A7 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “YesNo”  2. Definir a resposta correta da questão -> qyn.setCorrect\_answer("yes")  3. Invocar método addQuestion(qyn)  4. Invocar o teste assertAll(() -> { assertEquals(2, qyn.getUser\_answer());  assertTrue(qyn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA7,ECP4,ECP8 | user\_answer | 2 | Exception | Untested |

**Observações:**

Não foi possível efetuar o teste, uma vez que os inputs pretendidos não são aceites pelos métodos, logo está protegido pela linguagem.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A8 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “QuestionNumeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(Double.toString(-Double.MAX\_VALUE-1));  4. Invocar o teste assertAll(() -> { assertEquals(-Double.MAX\_VALUE-1, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA8,ECP2.ECP5 | user\_answer | Double.toString(-Double.MAX\_VALUE-1) | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A9 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “QuestionNumeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(Double.toString(-Double.MAX\_VALUE));  4. Invocar o teste assertAll(() -> { assertEquals(-Double.MAX\_VALUE, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA9,ECP5 | user\_answer | Double.toString(-Double.MAX\_VALUE) | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A10 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “QuestionNumeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(Double.toString(-Double.MAX\_VALUE+1));  4. Invocar o teste assertAll(() -> { assertEquals(-Double.MAX\_VALUE+1, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA10,ECP5 | user\_answer | Double.toString (-Double.MAX\_VALUE+1) | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A11 | **Dependências** | - | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 | - | - | - |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “QuestionNumeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(Double.toString(Double.MAX\_VALUE-1));  4. Invocar o teste assertAll(() -> { assertEquals(Double.MAX\_VALUE-1, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA11,ECP5 | user\_answer | Double.toString(Double.MAX\_VALUE-1) | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A12 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “QuestionNumeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(Double.toString(Double.MAX\_VALUE));  4. Invocar o teste assertAll(() -> { assertEquals(Double.MAX\_VALUE, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA12,ECP5 | user\_answer | Double.toString(Double.MAX\_VALUE) | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A13 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “QuestionNumeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(Double.toString(Double.MAX\_VALUE+1));  4. Invocar o teste assertAll(() -> { assertEquals(Double.MAX\_VALUE+1, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA13,ECP2,ECP5 | user\_answer | Double.toString(Double.MAX\_VALUE+1) | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A14 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “Numeric”  2. Invocar método addQuestion(qn)  3. Invocar método qn.answer(“palavra”));  4. Invocar o teste assertAll(() -> { assertEquals(“palavra”, qn.getUser\_answer());  assertTrue(qn.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA14,ECP6 | user\_answer | “palavra” | Exception | Untested |

**Observações:**

Não foi possível efetuar o teste, uma vez que os inputs pretendidos não são aceites pelos métodos, logo está protegido pela linguagem.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A15 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “MultipleChoice”  2. Invocar método addQuestion(qmc)  3. Invocar método qmc.answer("c");  3. Invocar o teste assertAll(() -> { assertEquals(“c”, qmc.getUser\_answer());  assertTrue(qmc.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA15,ECP7 | user\_answer | “c” | True | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | A16 | **Dependências** |  | |
|  |  |  |  | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** |
| 18-11-2019 | 17h30 |  |  |  |
|  | | | | |
| **Pré-condições / estado inicial** | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | |
| 1. Instanciar uma questão de “MultipleChoice”  2. Definir a resposta correta da questão -> qmc.setCorrect\_answer("c")  3. Invocar método addQuestion(qyn)  4. Invocar o teste assertAll(() -> { assertEquals(“C”, qmc.getUser\_answer());  assertTrue(qmc.isDone());  }); | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** |
| BVA16,ECP7 | user\_answer | “C” | True | Fail |

**Observações:**

Com este teste esperava-se obter o resultado positivo, uma vez que a resposta é a correta. No entanto, este falhou pois o código não está desenvolvido para detetar o facto de a resposta estar em *UpperCase*. O teste retornou o seguinte erro:

“org.opentest4j.AssertionFailedError: expected: <c> but was: <C>”.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | IC\_01 | **Dependências** | isDone | | |
|  |  |  |  | |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | |
| 22-11-2019 | 12h00 |  | Fail | Protegido pela linguagem | |
|  | | | | |  |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1. Instanciar Test  2. Invocar isComplete | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | |
| BVA-1 | question | isComplete(1,2); | “Build failed with an exception” | Untested | |

**Observações:**

O método não receber argumentos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | IC\_02 | **Dependências** | isDone | | |
|  |  |  |  | |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | |
| 22-11-2019 | 12h00 |  | Pass | - | |
|  | | | | |  |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1 Instanciar Test;  2 Instanciar uma questão;  3 Invocar método addquestion adicionando a questão instanciada na posicao 10 vezes  4 Invocar isComplete | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | |
| ECP-2, ECP-3, BVA-2 | question | questions=10&&done=true(0) | False | Pass | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | IC\_03 | **Dependências** | isDone | | |
|  |  |  |  | |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | |
| 22-11-2019 | 12h10 |  | Pass | - | |
|  | | | | |  |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1 Instanciar Test;  2 Instanciar uma questão;  3 Invocar método addquestion e adicionando 10 questões  4 Invocar setDone para colocar uma questão respondida  5 Invocar isComplete | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | |
| ECP-2, ECP-3, BVA-3 | question | questions=10&&done=true(1) | False | Pass | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | IC\_04 | **Dependências** | isDone | | |
|  |  |  |  | |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | |
| 22-11-2019 | 12h10 |  | Pass | - | |
|  | | | | |  |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1 Instanciar Test;  2 Instanciar uma questão;  3 Invocar método addquestion e adicionando 10 questões  4 Invocar setDone para colocar 5 questões respondida  5 Invocar isComplete | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | |
| ECP-2, ECP-3, BVA-4 | question | questions=10&&done=true(5) | False | Pass | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | IC\_05 | **Dependências** | IsDone | | |
|  | | | | | |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | |
| 22-11-2019 | 12h10 |  | Pass | - | |
|  | | | | |  |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1 Instanciar Test;  2 Instanciar uma questão;  3 Invocar método addquestion e adicionando 10 questões  4 Invocar setDone para colocar 9 questões respondida  5 Invocar isComplete | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | |
| ECP-2, ECP-3, BVA-5 | question | questions=10&&done=true(9) | False | Pass | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | IC\_03 | **Dependências** | isDone | | |
|  |  |  |  | |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | |
| 22-11-2019 | 12h10 |  | Pass | - | |
|  | | | | |  |
| **Pré-condições / estado inicial** | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | |
| 1 Instanciar Test;  2 Instanciar uma questão;  3 Invocar método addquestion e adicionando 10 questões  4 Invocar setDone para colocar 10 questões respondidas  5 Invocar isComplete | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | |
| ECP-1, ECP-4, BVA-6 | question | questions=10&&done=true(10) | True | Pass | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_01 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Fail | Protegido pela linguagem | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-1 | question | question=-1 | *“Build failed with an exception”* | Untested | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_02 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h10 |  | Fail | Protegido pela linguagem | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1 Instanciar Test;  2 Instanciar uma questão;  3 Invocar método addquestion e tentativa de adicionar a questão instanciada na posicao 101  4 Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-2 | question | question=101 | *ArrayIndexOutOfBoundsException* | Fail | | |

**Observações:**

**Da para adicionar no max. 100 questões**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_03 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Fail | Protegido pela linguagem | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-2, BVA-3 | question | numberQuestions(1,2); | “Build failed with an exception” | Untested | | |

**Observações:**

**Não da para enviar argumentos como parametros.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_04 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Pass | - | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1. Instanciar Test  2. Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-4 | question | question=0 | 0 | Pass | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_05 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Pass | - | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1 Instanciar Test  2 Instanciar uma questão  3 Invocar metodo addQuestion adicionado a questão uma vez  4 Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-5 | question | question=1 | 1 | Pass | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_06 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Pass | - | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1 Instanciar Test  2 Instanciar uma questão  3 Invocar metodo addQuestion adicionado a questão 50 vezes  4 Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-6 | question | question=50 | 50 | Pass | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_07 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Pass | - | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1 Instanciar Test  2 Instanciar uma questão  3 Invocar metodo addQuestion adicionado a questão 99 vezes  4 Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-7 | question | question=99 | 99 | Pass | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | NQ\_08 | **Dependências** | - | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | |
| 22-11-2019 | 12h00 |  | Pass | - | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | |
| 1 Instanciar Test  2 Instanciar uma questão  3 Invocar metodo addQuestion adicionado a questão 100 vezes  4 Invocar numberQuestions | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | |
| ECP-1, BVA-8 | question | question=100 | 100 | Pass | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_01 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | Protegido pela linguagem | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-2, ECP-3, BVA-1 | - | correctAnswersPercentage(1,2); | *“Build failed with an exception”* | Untested | | | |

**Observações:**

**Não da para enviar argumentos por parametro**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_02 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h10 |  | Fail | Expected TestException to be thrown, but nothing was thrown. | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1 Instanciar Test;  2. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-2, ECP-3 BVA-2 | question=null | isComplete=false | TestException | Fail | | | |

**Observações:**

O teste não tem todas as questões respondidas.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_03 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Pass | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-3 | isComplete=true | CorrectAnswer=0 && numberQuestions=100 | 0.0 | Pass | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_04 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-4 | isComplete=true | CorrectAnswer=1 && numberQuestions=100 | 0.01 | Fail | | | |

**Observações:**

O retorno devia ser 0.01 e o resultado esperado foi 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_05 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-5 | isComplete=true | CorrectAnswer=50 && numberQuestions=100 | 0.5 | Fail | | | |

**Observações:**

O retorno devia ser 0.5 e o resultado esperado foi 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_06 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-6 | isComplete=true | CorrectAnswer=99 && numberQuestions=100 | 0.99 | Fail | | | |

**Observações:**

O retorno devia ser 0.99 e o resultado esperado foi 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_07 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Pass | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método correctAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-7 | isComplete=true | CorrectAnswer=100 && numberQuestions=100 | 1 | Pass | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_01 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | Protegido pela linguagem | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-2, ECP-3, BVA-1 | - | incorrectAnswersPercentage(1,2); | *“Build failed with an exception”* | Untested | | | |

**Observações:**

O método não recebe argumentos com parametro

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_02 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h10 |  | Fail | Expected TestException to be thrown, but nothing was thrown. | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1 Instanciar Test;  2. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-2, ECP-3 BVA-2 | question=null | isComplete=false | TestException | Fail | | | |

**Observações:**

O teste não tem nenhuma questão respondida

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_03 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Pass | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-3 | isComplete=true | CorrectAnswer=0 && numberQuestions=100 | 0.0 | Pass | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_04 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-4 | isComplete=true | CorrectAnswer=1 && numberQuestions=100 | 0.01 | Fail | | | |

**Observações:**

O retorno devia ser 0.01 e o resultado esperado foi 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_05 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-5 | isComplete=true | CorrectAnswer=50 && numberQuestions=100 | 0.5 | Fail | | | |

**Observações:**

O retorno devia ser 0.5 e o resultado esperado foi 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_06 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Fail | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-6 | isComplete=true | CorrectAnswer=99 && numberQuestions=100 | 0.99 | Fail | | | |

**Observações:**

O retorno devia ser 0.99 e o resultado esperado foi 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | CAP\_07 | **Dependências** | - | | | | |
|  |  |  |  | |  |  |
| **Data de Execução** | **Hora de Execução** | ***Tester*** | **Pass/Fail** | **Resumo da Falha** | | | |
| 23-11-2019 | 12h00 |  | Pass | - | | | |
|  | | | | |  |  |
| **Pré-condições / estado inicial** | | | | | | | |
| Existência de uma coleção de perguntas previamente instanciada | | | | | | | |
| **Procedimento / Passos do *user* para esta função** | | | | | | | |
| 1. Instanciar Test  2. Instanciar Questão  3. Adicionar questões no teste  4. Invocar método isDone para colocar algumas questões respondidas  3. Invocar método incorrectAnswersPercentage | | | | | | | |
| **Requisito/Use Case/ Funcionalidade** | **Inputs** | **Valores do Inputs** | **Resultados Esperados** | **Pass/Fail/Untested** | | | |
| ECP-1, ECP-4, BVA-7 | isComplete=true | CorrectAnswer=100 && numberQuestions=100 | 1 | Pass | | | |