Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

The game that we are going to test is battleship without GUI, all the functionalities of the game are performed in console.
The github repository of this game is forked to us account in this location:
https://github.com/AdriaOrozco/BattleShipTesting
The original github repository of this game is:
https://github.com/cristianve/BattleShipTesting

TEST CASES:

Test Case ID	TC_Smoke_01		
Description	Set boat in the first position for true values		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	30/11/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok

Test Data Sets

Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	
Y Coordinate	1	1	
Orientation	0 (Right)	(Down)	
Test Case Result		Both Datasets results are correct.	

Test Case ID	TC_Smoke_02		
Description	Set boat in the last positions for true values		
Module	BattleShip.java		
Prepared By	Adrià Orozco	**Date Prepared**	30/11/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok

3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Put the boat in another direction Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	10	10	
Y Coordinate	10	10	
Orientation	2 (Up)	3(Left)	
Test Case Result		Both of Datasets insert a boat in another direction, In the case of Orientation “Up” insert the boat to the left, and in the case of Ortientation “Left” insert the boat upside.	

Test Case ID	TC_Smoke_02		
Description	Set boat in the next limit values (1,10)		
Module	BattleShip.java		
Prepared By	Adrià Orozco	**Date Prepared**	30/11/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Put the boat in another direction Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	
Y Coordinate	10	10	
Orientation	3 (Left)	1(Down)	
Test Case Result		For the first Dataset the result is incorrect according to the fail that the game haves (Orientation left and up has to be interchanged)	

Test Case ID	TC_Smoke_02		
Description	Set boat in the next limit values (1,10)		
Module	BattleShip.java		
Prepared By	Adrià Orozco	**Date Prepared**	30/11/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	
Y Coordinate	10	10	
Orientation	3 (Left)	1(Down)	
Test Case Result		Correct Result for Dataset 2	

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Test Case ID	TC_Smoke_03		
Description	Set boat in the next limit values (10,0)		
Module	BattleShip.java		
Prepared By	Adrià Orozco	**Date Prepared**	30/11/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Put the boat in another direction Ok

Test Data Sets

Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	
Y Coordinate	10	10	
Orientation	2(Up)	0(Right)	
Test Case Result		For the first Dataset the result is incorrect according to the fail that the game haves (Orientation left and up has to be interchanged)	

Test Case ID	TC_Smoke_03		
Description	Set boat in the next limit values (10,0)		
Module	BattleShip.java		
Prepared By	Adrià Orozco	**Date Prepared**	30/11/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	
Y Coordinate	10	10	
Orientation	2(Up)	0(Right)	
Test Case Result		Result for dataset2 is correct	

Test Case ID	TC_Smoke_04		
Description	Set boat in a middle value of the matrix expecting true results (5,5)		
Module	BattleShip.java		
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	5	5
Y Coordinate	5	5	5
Orientation	0(Right)	1(Down)	2(Left)

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Test Case Result	Result for Dataset1 and 2 are correct

Test Case ID	TC_Smoke_04
Description	Set boat in a middle value of the matrix expecting true results (5,5)
Module	BattleShip.java

Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Put the boat in another direction Ok

Test Data Sets

Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	5	5
Y Coordinate	5	5	5
Orientation	0(Right)	1(Down)	2(Left)

Test Case Result	For the third Dataset the result is incorrect according to the fail that the game haves (Orientation left and up has to be interchanged)

Test Case ID	TC_Smoke_05
Description	Set boat on another boat
Module	BattleShip.java

Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**

1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok
6.	Set X coordinate	Program ask for the Y coordinate	Ok
7.	Set Y coordinate	Program ask for the orientation	Ok
8.	Set Orientation	Boat not inserted Program ask for the first X coordinate again	Ok Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	5	
Y Coordinate	5	5	
Orientation	0(Right)	1(Down)	
Test Case Result		Correct Result	

Test Case ID		TC_Smoke_06	
Description		Set the last coordinate of boat on another boat	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok
6.	Set X coordinate	Program ask for the Y coordinate	Ok
7.	Set Y coordinate	Program ask for the orientation	Ok
8.	Set Orientation	Boat not inserted Program ask for the first X coordinate again	Ok Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	3	
Y Coordinate	5	5	
Orientation	0(Right)	1(Down)	
Test Case Result		Correct Result	

Test Case ID		TC_Smoke_07	
Description		Set invalid coordinates	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly Program ask for the X coordinate of the next boat	Ok Ok
6.	Set X coordinate	Program ask again for the X coordinate	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	-1	-9	11
Y Coordinate	5	5	5
Orientation	0(Right)	1(Down)	0(Right)
Test Case Result		Correct Result	

Test Case ID		TC_Smoke_08	
Description		Shoot boat	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**

1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Insert all the boats	Choose a coordinate to shoot	Ok
4.	Set X coordinate	Set Y coordinate	Ok
5.	Set Y coordinate	Result of the Shoot	Ok
6.	Set X coordinate	Set Y coordinate	Ok
7.	Set Y coordinate	Result of the Shoot	Ok
8.			
9.			

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	9	7
Y Coordinate	5	9	7

Test Case Result	Correct Result, Shows the correct response to the interaction with the user. Miss or Hit.

Test Case ID		TC_Smoke_09	
Description		Sunk boat	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Insert all the boats	Choose a coordinate to shoot	Ok
4.	Set X coordinate	Set Y coordinate	Ok
5.	Set Y coordinate	Result of the Shoot	Ok
6.	Set X coordinate	Set Y coordinate	Ok
7.	Set Y coordinate	Result of the Shoot	Ok
8.	Set X coordinate	Set Y coordinate	Ok
9.	Set Y coordinate	Result of the Shoot (Sunked)	Shows only that the boat is hit

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	9	9
Y Coordinate	5	9	10
Test Case Result		Incorrect Result, the game must show that the boat is sunk, we only know that we hit the boat, but never when we sunked-it	

Test Case ID		TC_Smoke_10	
Description		Invalid coordinates to shoot a boat	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Insert all the boats	Choose a coordinate to shoot	Ok
4.	Set X coordinate	Set again the X coordinate	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	-1	11	19
Y Coordinate	5	10	7
Test Case Result		Correct Result	

Test Case ID		TC_Smoke_11	
Description		Shoot a boat coordinate that is already shooted	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**

1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Insert all the boats	Choose a coordinate to shoot	Ok
4.	Set X coordinate	Set Y coordinate	Ok
5.	Set Y coordinate	Result of the Shoot	Ok
6.	Set X coordinate	Set Y coordinate	Ok
7.	Set Y coordinate	Set Y coordinate again	The game let you shoot again the same position

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	9	9	
Y Coordinate	9	9	
Test Case Result		Incorrect result, the game let you shoot the same coordinate of a boat again and convert the hit shoot to a miss shoot on the board.	

Test Case ID		TC_Smoke_12	
Description		Shoot a coordinate that was a miss shoot before	
Module		BattleShip.java	
Prepared By	Adrià Orozco	**Date Prepared**	03/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	03/12/2020
Tested By	Adrià and Arnau	**Date Tested**	03/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Insert all the boats	Choose a coordinate to shoot	Ok
4.	Set X coordinate	Set Y coordinate	Ok
5.	Set Y coordinate	Result of the Shoot	Ok
6.	Set X coordinate	Set Y coordinate	Ok
7.	Set Y coordinate	Set Y coordinate again	The game let you shoot again the

			same position

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	5	
Y Coordinate	5	5	
Test Case Result		Incorrect result, the game let you shoot the same coordinate	

Test Case ID	TC_Smoke_13		
Description	Invalid Orientation		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	06/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	06/12/2020
Tested By	Adrià and Arnau	**Date Tested**	06/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	5	5
Y Coordinate	5	5	5
Orientation	-4	5	9
Test Case Result		All Datasets results are correct.	

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Test Case ID	TC_Smoke_14		
Description	Invalid Orientation		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets

Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	1
Y Coordinate	1	1	1
Orientation	2(Up)*	3(Left)*	5

Test Case Result	All Datasets results are correct. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up.

Test Case ID	TC_Smoke_15		
Description	Invalid Orientation		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	9	9	10
Y Coordinate	9	9	10
Orientation	2(Up)*	*3(Left)	-1
Test Case Result		All Datasets results are correct. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up.	

Test Case ID	TC_Smoke_16		
Description	Invalid Orientation		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	10	10	1
Y Coordinate	1	1	1
Orientation	3(Left)*	1(Down)	4

Test Case Result	All Datasets results are correct. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up.

Test Case ID	TC_Smoke_17		
Description	Invalid Orientation with a previous boat set		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl. No.	Step Description	Expected Results	Actual Results
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly	Ok
6.	Set X coordinate	Program ask for the Y coordinate	Ok
7.	Set Y coordinate	Program ask for the orientation	Ok
8.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets

Data Type	Data Set 1	Data Set 2	Data Set 3
X Coordinate	5	5	7
Y Coordinate	3	1	4
Orientation	0(Right)	0(Right)	3(Up)*

Test Case Result	First Dataset corresponds to the coordinates of the first boat, the second ones are the invalid orientations that we tried. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up. These Two last datasets are both correct.

Test Case ID	TC_Smoke_18		
Description	Invalid Orientation with a previous boat set		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl.	Step Description	Expected Results	Actual

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

No.			Results
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly	Ok
6.	Set X coordinate	Program ask for the Y coordinate	Ok
7.	Set Y coordinate	Program ask for the orientation	Ok
8.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	5	3	5
Y Coordinate	3	4	9
Orientation	0(Right)	1(Down)	2(Left)*

Test Case Result	First Dataset corresponds to the coordinates of the first boat, the second ones are the invalid orientations that we tried. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up. These Two last datasets are both correct.

Test Case ID	TC_Smoke_19		
Description	Invalid Orientation with a previous boat set		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly	Ok
6.	Set X coordinate	Program ask for the Y coordinate	Ok
7.	Set Y coordinate	Program ask for the orientation	Ok
8.	Set Orientation	Program ask for the X coordinate again	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	6
Y Coordinate	1	4	1
Orientation	1(Down)	2(Left)*	3(Up)*

Test Case Result	First Dataset corresponds to the coordinates of the first boat, the second ones are the invalid orientations that we tried. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up. These Two last datasets are both correct.

Test Case ID	TC_Smoke_20		
Description	Invalid Orientation with a previous boat set		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Boat inserted correctly	Ok
6.	Set X coordinate	Program ask for the Y coordinate	Ok
7.	Set Y coordinate	Program ask for the orientation	Ok
8.	Set Orientation	Program ask for X coordinate again	Ok

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	5
Y Coordinate	1	2	1
Orientation	1(Down)	1(Down)	1(Down)

Test Case Result	First Dataset corresponds to the coordinates of the first boat, the second ones are the invalid orientations that we tried. *As we said the orientations 3 and 4 are interchanged, so we consider the correct values of these to make this test cases. Assuming that orientation 3 is Up instead of left and orientation 4 is Left instead of Up. These Two last datasets are both correct.

Test Case ID	TC_Smoke_21		
Description	Invalid type of values		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl. No.	Step Description	Expected Results	Actual Results
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Program ask for the orientation again	The game should ask for the orientation again

Test Data Sets

Data Type	Data Set 1	Data Set 2	Data Set 3
X Coordinate	1	1	5
Y Coordinate	1	5	5
Orientation	string	a	up

Test Case Result	We tried to introduce an string instead a numeric value, and the result is that the game doesn't control these exceptions, instead of ask for a new orientation, the game crashes.

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Test Case ID	TC_Smoke_22		
Description	Invalid type of values		
Module	BattleShip		
Prepared By	Adrià Orozco	**Date Prepared**	09/12/2020
Reviewed / Updated	Arnau Cruz	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for the orientation	Ok
5.	Set Orientation	Program ask for the orientation again	The game should ask for the orientation again

Test Data Sets

Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	9	1	5
Y Coordinate	9	1	5
Orientation	5-3	1+1	/n

Test Case Result	We tried to introduce an operation or a reserved keyword instead a numeric value, and the result is that the game doesn't control these exceptions, instead of ask for a new orientation, the game crashes.

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Test Case ID	TC_Smoke_23		
Description	Put correct X coordinate but wrong Y coordinate		
Module	BattleShip		
Prepared By	Arnau Cruz	**Date Prepared**	09/12/2020
Reviewed / Updated	Adria Orozco	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities

Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for Y coordinate again	Ok

Test Data Sets

Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	1	1
Y Coordinate	17	-12	17890

Test Case Result	We tried to introduce a wrong Y coordinate and the result is that game asks for a new Y coordinate, so it's designed properly.

Test Case ID	TC_Smoke_24		
Description	Put 0 before the orientation.		
Module	BattleShip		
Prepared By	Arnau Cruz	**Date Prepared**	09/12/2020
Reviewed / Updated	Adria Orozco	**Date Reviewed**	09/12/2020
Tested By	Adrià and Arnau	**Date Tested**	09/12/2020

Test Activities			
Sl. No.	**Step Description**	**Expected Results**	**Actual Results**
1.	Run the project	Starter menu of battleship game	Ok
2.	Write 1 to start the game	Game started	Ok
3.	Set X coordinate	Program ask for the Y coordinate	Ok
4.	Set Y coordinate	Program ask for orientation	Ok
5.	Set orientation	Program ask for orientation again	The game write the wrong orientation

Test Data Sets			
Data Type	**Data Set 1**	**Data Set 2**	**Data Set 3**
X Coordinate	1	5	9
Y Coordinate	1	5	9
Orientation	01	02	04

Test Case Result	We tried to introduce an orientation with a 0 before, because this can be a problem that user wants to write 0 and by mistake he can add a 1 after 0, so in this case we want that the system ask again for orientation, but when we doing this test the system just takes the last integer since 0 before a number is a “null number”.

EXPLORATORY TESTING:

Attack 1: Apply inputs that force all error messages to occur at least once.

1.1 Value from outside of the options of the menu.
Result: Error controlled, correct information to the user.

1.2 Value from outside the board
Result: Error well controlled, the user can’t put a boat outside the board, but there isn’t informative error to the user as is showed below.

```
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
11
Position X (1-10):
15
Position X (1-10):
-20
Position X (1-10):
-1
Position X (1-10):
```

1.3 Input type

-Non numeric: The program doesn't control if there is an input that isn't a numeric value so the exception is not caught and the program crash.

```
Position X (1-10):
string
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Board.readPosition(Board.java:257)
        at BattelshipTesting.Board.insertPosicion(Board.java:285)
        at BattelshipTesting.Player.locateBoat(Player.java:54)
        at BattelshipTesting.Player.<init>(Player.java:37)
        at BattelshipTesting.Match.<init>(Match.java:29)
        at BattelshipTesting.Menu.getOption(Menu.java:43)
        at BattelshipTesting.Menu.getOption(Menu.java:51)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

-Decimal value: The program doesn't control if there is an input that isn't an enter value so the exception is not caught when the user introduced a floating value and the program crash.

```
------Main Menu------
1- Play
2- Exit
1.54
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Menu.getOption(Menu.java:39)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

1.4 After discover that the type of the inputs of the user are integers, we made proves with the length of these and discover that the program doesn't caught the exceptions if the value entered is bigger than the length of an integer and the program crashes too.

```
<terminated> Game [Java Application] C:\Users\adri-\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x
------Main Menu------
1- Play
2- Exit
2147483647
Invalid Option!------Main Menu------
1- Play
2- Exit
2147483648
Exception in thread "main" java.util.InputMismatchException: For input string: "2147483648
        at java.base/java.util.Scanner.nextInt(Scanner.java:2264)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Menu.getOption(Menu.java:39)
        at BattelshipTesting.Menu.getOption(Menu.java:51)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

Attack 2: Apply inputs that force the software to establish default values

2.1 We tried to shoot positions already shooted to see if the program controlled it properly and the status of the board change or not.
The first case is to shoot a position where there is a boat and shoot it again.
Result: The first shoot is correctly showed at the board, but in the second shoot there is not informative error message, and there is a mistake on the code that actualize the position of the board like if there isn't a boat there. Showed with screenshots below.
First shoot at position [9,3]:

```
Position X (1-10):
9
Position Y (1-10):
3
Bot Hit!
```

	A	B	C	D	F	G	H	I	J	K
1	#									
2										
3										
4										
5					#					
6										
7										
8										
9			X						#	
10	#	#			#					

Second shoot at the same position:

```
What position do you want to attack?
Position X (1-10):
9
Position Y (1-10):
3
Water!
```

	A	B	C	D	F	G	H	I	J	K
1	#									
2										
3										
4										
5					#					
6										
7										
8										
9			#						#	
10	#	#			#					

```
Machine => RandomIA ATTACK RANDOM!
Water!
```

So now there is no way to know if there was a boat in there or if the boat has got more positions at this row/column.

2.2 On the other hand to shoot a position already shouted don't change anything at the board, but there isn't an informative error to the user.

Attack 3: Explore allowable character sets and potentially special meaning values in string fields

3.1 Reserved keywords
First off all we prove to insert an "\n", as the type of the values entered is integer the program crashes on input a no numerical value:

```
Position Y (1-10):
\n
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Board.readPosition(Board.java:266)
        at BattelshipTesting.Board.attack(Board.java:376)
        at BattelshipTesting.Player.attack(Player.java:79)
        at BattelshipTesting.Match.startMatch(Match.java:42)
        at BattelshipTesting.Match.<init>(Match.java:31)
        at BattelshipTesting.Menu.getOption(Menu.java:43)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

This happens with all the other options of special values like “+ or –“:

```
------Main Menu------
1- Play
2- Exit
1+1
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Menu.getOption(Menu.java:39)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

```
------Main Menu------
1- Play
2- Exit
+
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Menu.getOption(Menu.java:39)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

3.2 Special characters.

In case of NULL(^%) and EOF(^Z) the program behavior it’s the same:

```
------Main Menu------
1- Play
2- Exit
NULL(^%)
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Menu.getOption(Menu.java:39)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)

1- Play
2- Exit
EOF(^Z)
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at BattelshipTesting.ManagerIO.inInt(ManagerIO.java:40)
        at BattelshipTesting.Menu.getOption(Menu.java:39)
        at BattelshipTesting.Menu.<init>(Menu.java:18)
        at BattelshipTesting.Game.main(Game.java:21)
```

Attack 4: Overflow input buffers in string fields or parameters.

We can't do this attack because this attack is supposed to overflow input buffers in string fields, but in this game there isn't any input string.

Attack 5: Find inputs that may interact and test combinations of their values.

We are going to do this attack where we want to place the second coord from the ship so we will combine the first position with the second one and let's see what the result is.

5.1We put a correct value for X and then an incorrect value for Y and vice versa:

The program reacts well and ask again for the value Y since the number that we wrote isn't in the specified values:

```
------Main Menu------
1- Play
2- Exit
1
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
2
Position Y (1-10):
14
Position Y (1-10):
```

The program reacts well and ask again for the value Y since the number that we wrote isn't in the specified values:

```
------Main Menu------
1- Play
2- Exit
1
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
13
Position X (1-10):
```

The same for negative values:

```
1- Play
2- Exit
1
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
-3
Position X (1-10):
```

5.2 We put correct coords and then we will put a wrong orientation:

The program reacts good and ask for a new position:

```
1- Play
2- Exit
1
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
2
Position Y (1-10):
9
Orientation:
 0-Right
 1-Down
 2-Up
 3-Left

0
Position Position [x=2, y=9, orientation=Right]
It is not correct!
Enter one again!
Position X (1-10):
```

Attack 6: Add a lot of 0 before the value:

We are going to do this attack where we want to put the coordinates to place the boat. We will add a lot of 0 before the value desired.

6.1 We put 80 0 before the value:

The program takes the last value as integer.

```
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
00000000000000000000000000000000000000000000000000000000000000000000000000000001
Position Y (1-10):
00000000000000000000000000000000000000000000000000000000000000000000000000000001
Orientation:
 0-Right
 1-Down
 2-Up
 3-Left

0
|
    | A | B | C | D | F | G | H | I | J | K |
+---+---+---+---+---+---+---+---+---+---+---+
| 1 | B | B | B | B |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 2 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 3 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 4 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 5 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 6 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 7 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 8 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 9 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 10|   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
Position (X, Y) head of the ship it occupies 3 squares (1/1)
Position X (1-10):
```

6.2 We put 01 in orientation, so we can create confusion between 0 “Right”, 1 “Down”.

The program takes the last value as integer “1” so it takes “Down” orientation.

```
------Starting Game------Player: HumanPlayer
Position your boats!
Position (X, Y) head of the ship it occupies 4 squares (1/1)
Position X (1-10):
1
Position Y (1-10):
1
Orientation:
 0-Right
 1-Down
 2-Up
 3-Left

01
|
    | A | B | C | D | F | G | H | I | J | K |
+---+---+---+---+---+---+---+---+---+---+---+
| 1 | B |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 2 | B |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 3 | B |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 4 | B |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 5 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 6 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 7 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 8 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 9 |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
| 10|   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+
```

Attack 7: Repeat the same input or series of inputs numerous times

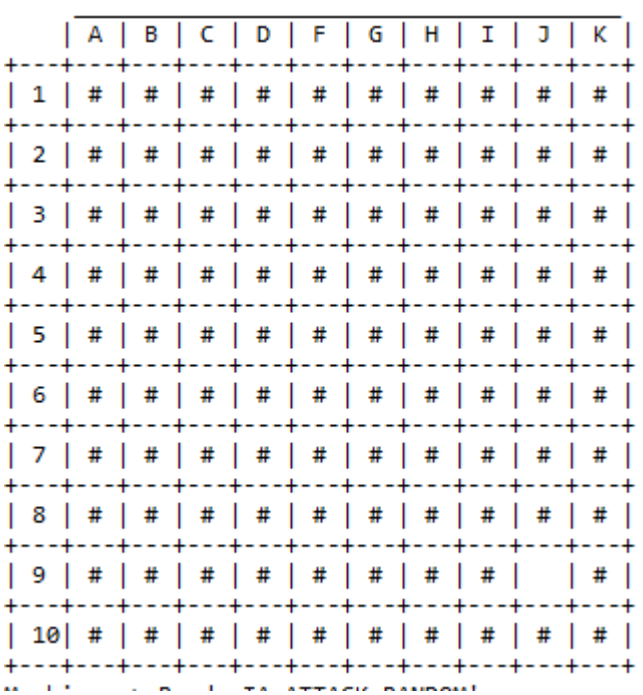
We tried to input the same number a lot of times to check if there is a maximum number of intents, but the result of the test is failed, the program controlled well the buffer of entries and ever ask again for a new coordinate to put on the boat.

Attack 8: Force data structures (known from source code or guessed) to store too many or too few values

In this attack we will try to shoot all the positions from the board including repeated positions to see if the shoots of the user are stored in some structure and we can overload it. (We will try to not sunk all the boats in order to shoot as many positions as possible)

```
    | A | B | C | D | F | G | H | I | J | K |
+---+---+---+---+---+---+---+---+---+---+---+
| 1 | # | # | # | X | # | # | # | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 2 | # | X | # | X | # | X | X | X | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 3 | # | # | # | # | # | # | # | # | # | X |
+---+---+---+---+---+---+---+---+---+---+---+
| 4 | # | X | X | # | X | # | # | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 5 | # | # | # | # | # | # | # | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 6 | # | # | # | # | X | X | X | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 7 | # | # | # | # | # | # | # | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 8 | # | # | # | X | # | # | # | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
| 9 | # | X | # | X | # | # | X | X |   | X |
+---+---+---+---+---+---+---+---+---+---+---+
| 10| # | # | # | # | # | # | # | # | # | # |
+---+---+---+---+---+---+---+---+---+---+---+
```

After arrive to this situation the program responds properly and let us to shoot again without any problems, to continue with the test we won't sunk the last boat, instead of we will shoot the positions marked with an X to see if our test can be successful.



After shoot all the positions marked with an x, we shoot to all the positions marked with an # and this is the result of the board, to verify we are not closely to make the test successful we spam some positions a lot of times, but the program responds well and quickly, now it's time to see if the game is over shooting the last position. (The positions where there was a boat now change their state, that's an error that we documented in another tests.)

As we can see the game it's over and the test fails.

Note: Due to the simplicity of this code we cannot perform more attacks, this program only accept integers as entry values and don't caught the exceptions in case other type is entered. Also the decisions that the user could taka are few, he only can put a board on the board, and shoot a position.

AUTOMATED TESTING:

In our case we have to test a game that works in console, so in order to do automated test, we made a mockObject that creates a simple GUI that contains two textAreas and buttons to simulate the user entries. Then we substitute all the user entries and all the print sentences to do it through the GUI without alter any piece of code.
Doing that we achieve to open the interface in jubula a record some executions, that are stored on an export file on our folder "JubulaTest->BattleShip".
The records that we made are:
-4 records for complete games.

These 4 records finished without errors.
-To do that we dispose the boats in the board introducing some false values that force message errors or user to repeat some entries.
In the first record we complete a game playing as a human will do-it winning the game. We remember that the opponent set random boats in every execution so a lot of times the game won't be the same.
To fix that we made the three next records, that force to the user to shoot all the positions of the board in different directions every record (Horizontal, vertical, diagonal).
-The next two records are destined to show how the game could crash when the user entry is a number bigger than an Integer length or what happens if instead an Integer value the user entry for example an String.

CALCULATOR AUTOMATED TESTING:
This test is located in this folder: JubulaTest->Calculator".
Description about every test Suite:
<u>TestSuite: Arrel:</u>

TestCase Arrel:

Operation: sqrt(16)
Expected result: 4.0
Result obtained: 4.0
Test result: Ok.

TestCase Quadrat:

Operation: 4^2
Expected result: 16.0
Result obtained: 16.0
Test result: Ok.

<u>TestSuite: Arrel2:</u>

TestCase Arrel2:

Operation: sqrt(64)
Expected result: 8.0
Result obtained: 8.0
Test result: Ok.

TestCase Quadrat2:

Operation: 8^2
Expected result: 64.0
Result obtained: 64.0
Test result: Ok.

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

TestSuite: ExpLog:

TestCase Log:

Operation: ln(8)
Expected result: 2.0794415416798357
Result obtained: 2.0794415416798357
Test result: Ok.

TestCase Exp:

Operation: e(2.0794415416798357)
Expected result: 8.0
Result obtained: 8.000020899932483
Test result: Wrong, the operation is correct but we need to round to 8.0.

TestSuite: ExpLog2:

TestCase Log2:

Operation: ln(1)
Expected result: 0.0
Result obtained: 0.0
Test result: Ok.

TestCase Exp2:

Operation: e(0)
Expected result: 1.0
Result obtained: 1.0
Test result: Ok.

TestSuite: SinAsin:

TestCase Sin:

Operation: sin(90)
Expected result: 1.0
Result obtained: 1.0
Test result: Ok.

TestCase Asin:

Operation: asin(1)
Expected result: 90
Result obtained: 0.01745240643728351
Test result: Wrong, the operation is wrong.

TestSuite: SinAsin2:

TestCase Sin2:

Operation: sin(30)
Expected result: 0.5
Result obtained: 0.4999999999999994
Test result: Wrong, the operation is correct but we need to round to 0.5.

TestCase Asin2:

Operation: asin(0.5)
Expected result: 30.0
Result obtained: 0.008726535498373935
Test result: Wrong, the operation is wrong.

TestSuite: CosAcos:

TestCase Cos:

Operation: cos(0)
Expected result: 1.0
Result obtained: 1.0
Test result: Ok.

TestCase Acos:

Operation: acos(1)

Expected result: 0
Result obtained: 0.9998476951563913
Test result: Wrong, the operation is wrong.

TestSuite: CosAcos2:

TestCase Cos2:

Operation: cos(60)
Expected result: 0.5
Result obtained: 0.5000000000000001
Test result: Wrong, the operation is correct but we need to round to 0.5.

TestCase Acos2:

Operation: acos(0.5)
Expected result: 60.0
Result obtained: 0.9999619230641713
Test result: Wrong, the operation is wrong.

TestSuite: TanAtan:

TestCase Tan:

Operation: tan(45)
Expected result: 1.0
Result obtained: 0.9999999999999999
Test result: Wrong, the operation is correct but we need to round to 1.0.

TestCase Atan:

Operation: atan(1)
Expected result: 45.0
Result obtained: 0.017455064928217585
Test result: Wrong, the operation is wrong.

TestSuite: TanAtan2:

TestCase Tan2:

Operation: tan(0)
Expected result: 0.0
Result obtained: 0.0
Test result: Ok.

TestCase Atan2:

Operation: atan(0)
Expected result: 0.0
Result obtained: 0.0
Test result: Ok.

Conclusions:

Sqrt works properly.
Power works properly.
Log works but need to round the number properly.
Exp works but need to round the number properly.
Sin works but need to round the number properly.
Cos works but need to round the number properly.
Tan works but need to round the number properly.
Asin doesn’t work properly.
Acos doesn’t work properly.
Atan doesn’t work properly.

RTF:

RTF: This is the RTF from the code that we tested for the second practice.

Inspection Summary Report

Inspection Identification:

Project: BattleShip
Inspection ID: 02
Meeting Date: 07/12/2020

Work Product Description:

	Inspectors	Signature	Preparation Time	
Author:	Cristian Vega Sánchez	1426805	1	hours
Moderator:	Ismael Pajuelo Berjano	1456941	1	hours
Recorder:	Adrià Orozco Lorente	1490952	1	hours
Reader:	Arnau Cruz Gargallo	1494996	1 h 30	min
Inspector:	Arnau Cruz Gargallo	1494996	1 h 30	min
Inspector:	Adrià Orozco Lorente	1490952	1	hours
Inspector:	Ismael Pajuelo Berjano	1456941	1	hours
Inspector:	Ruben Campuzano	1528709	30	min
Inspector	Javier Martinez Abril	1497512	30	min
Inspector:	Alex Cruz Gargallo	1494995	30	min

Inspection Data

Pages or Lines of Code: 920 Meeting Time: 1h 40min

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Planned for Inspection: All code

Total Planning Effort: 5h 30min

Actually Inspected: All code

Total Overview Effort: 7h 10min
Total Preparation Effort: 5h 30min
Actual Rework Effort: 4 labor hours

Product Appraisal

ACCEPTED
___ as is
X conditionally upon verification

NOT ACCEPTED
___ reinspect following rework
___ inspection not completed

Verifier: Juan Manuel Vicente, Keyao li

Projected Rework Completion Date: 14/12/2020

Inspection Issue Log

Project:	**Origin:**	**R**equirements, **D**esign, **C**onstruction, **T**esting
Inspection ID:	**Type:**	**M**issing, **W**rong, **E**xtra, **U**sability, **P**erformance, **S**tyle, **C**larity, **Q**uestion
Meeting Date: **Recorder:**	**Severity:**	**M**ajor, **m**inor

Defects Found: 20, Major 4, Minor 16.

#	Origin	Type	Severity	Location	Description
1	D, C	S	m	Board.java	Some print sentences are allocated on functions from the model classes.
2	C	S	m	Board.java	Repeated conditionals into some functions that can be encapsulated into another procedures

3	C	S	m	Board.java	The conditions of boatProtection() and isValidOrientation() can be replaced for the boat length instead of hard-code to reuse the code in case that new boats with different length are inserted.
5	D	S	m	Board.java	Change the MAGIC NUMBERS that control the frontier values and others to Constants.
6	D	S	m	Board.java	This class is too big and do so many things, it could be better to assign some responsibilities to another classes.
7	D	S	m	Board.java	Add comments to isValidOrientation() function to see correctly all the options from the switch case.
8	D	S	m	Board, ManagerIO.java	Some variables are not defined with a proper name.
9	T	M	m	Board.java	At line 59 on the method showBoard() we can see that the test not cover all the sentences.
10	D	S	m	Board.java	At isValidOrientation() and insertBoat() methods there isn't no default case for switch case.
11	D	S	m	Board.java	At showBoard() method there is no default este for nested conditions.
12	C	W	M	ManagerIO.java	The game crash when the user insert a string instead of an integer.
13	C	W	M	ManagerIO.java	The game crash when the user insert a longer value than the maximum length of an integer.
14	C	W	M	ManagerIO.java	The scanner not use try catch sentences to check the input of the user.
15	C	W	M	Board.java	The sentences of the switch case to control the orientation of the boat are interchanged, (Left and Up), so when the user tries to insert a boat in one of this orientations the result is incorrect.
16	R	M	m	Board.java	The user can't know when a boat is sunked.
17	D	U	m	Board.java	The information of the shoots could be showed under the information board.
18	D	U	m	Board.java	The board isn't showed at the start of the game, so the user can't know where he'll put the first boat.
19	D	U	m	Board.java	The information of the columns are showed with letters, and the user only can introduce numbers.
20	D	U	m	Board.java	When the user shoot a position already shouted where there was a boat in there, the previous mark disappear showing that there isn't a boat there.

Generic Checklist for Code Reviews

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

Structure

- Does the code completely and correctly implement the design?
- Does the code conform to any pertinent coding standards?
- Is the code well-structured, consistent in style, and consistently formatted?
- Are there any uncalled or unneeded procedures or any unreachable code?
- Are there any leftover stubs or test routines in the code?
- Can any code be replaced by calls to external reusable components or library functions?
- Are there any blocks of repeated code that could be condensed into a single procedure?
- Is storage use efficient?
- Are symbolics used rather than “magic number” constants or string constants?
- Are any modules excessively complex and should be restructured or split into multiple routines?

Documentation

- Is the code clearly and adequately documented with an easy-to-maintain commenting style?
- Are all comments consistent with the code?

Variables

- Are all variables properly defined with meaningful, consistent, and clear names?
- Do all assigned variables have proper type consistency or casting?
- Are there any redundant or unused variables?

Arithmetic Operations

- Does the code avoid comparing floating-point numbers for equality?
- Does the code systematically prevent rounding errors?
- Does the code avoid additions and subtractions on numbers with greatly different magnitudes?
- Are divisors tested for zero or noise?

Loops and Branches

- Are all loops, branches, and logic constructs complete, correct, and properly nested?
- Are the most common cases tested first in IF- -ELSEIF chains?
- Are all cases covered in an IF- -ELSEIF or CASE block, including ELSE or DEFAULT clauses?
- Does every case statement have a default?
- Are loop termination conditions obvious and invariably achievable?
- Are indexes or subscripts properly initialized, just prior to the loop?
- Can any statements that are enclosed within loops be placed outside the loops?
- Does the code in the loop avoid manipulating the index variable or using it upon exit from the loop?

Defensive Programming

- Are indexes, pointers, and subscripts tested against array, record, or file bounds?
- Are imported data and input arguments tested for validity and completeness?
- Are all output variables assigned?
- Are the correct data operated on in each statement?
- Is every memory allocation deallocated?
- Are timeouts or error traps used for external device accesses?
- Are files checked for existence before attempting to access them?
- Are all files and devices are left in the correct state upon program termination?

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

This is the RTF from our code of the first practice

Inspection Summary Report

Inspection Identification:

Project: HundirFlota
Inspection ID: 01
Meeting Date: 06/12/2020

Work Product Description:

	Inspectors	Signature	Preparation Time	
Author:	Arnau Cruz Gargallo	1494996	1	hours
Moderator:	Carlos Fernandez	1428230	30	min
Recorder:	Adrià Orozco Lorente	1490952	1	hours
Reader:	Zakaria el Haddad	1462424	30	min
Inspector:	Carlos Fernandez	1428230	30	min
Inspector:	Adrià Orozco Lorente	1490952	1	hours
Inspector:	Zakaria el Haddad	1462424	30	min

Inspection Data

Pages or Lines of Code: 886 Meeting Time: 1hour 10min

Planned for Inspection: Board.java, Barco.java, Player.java, Main.java

Total Planning Effort: 3 labor hours

Actually Inspected: Board.java, Barco.java, Player.java, Main.java

Total Overview Effort: 4 hours 10min
Total Preparation Effort: 3 labor hours
Actual Rework Effort: 2 labor hours

Product Appraisal

ACCEPTED NOT ACCEPTED

Adrià Orozco Lorente 1490952 and Arnau cruz Gargallo 1494996

___ as is
X conditionally upon verification
___ reinspect following rework
___ inspection not completed

Verifier: Juan Manuel Vicente, Keyao li

Projected Rework Completion Date: 13/12/2020

Inspection Issue Log

Project:	**Origin:**	**R**equirements, **D**esign, **C**onstruction, **T**esting
Inspection ID:	**Type:**	**M**issing, **W**rong, **E**xtra, **U**sability, **P**erformance, **S**tyle, **C**larity, **Q**uestion
Meeting Date: **Recorder:**	**Severity:**	**M**ajor, **m**inor

Defects Found: 14, Major 4, Minor 10.

#	Origin	Type	Severity	Location	Description
1	D, C	S	m	Board.java	Some print sentences are allocated on functions from the model classes.
2	D	S	m	Game, Board.java	Some variables not use the lowerCaseCammel notation.
3	C	S	m	Board.java, Game.java	Repeated conditionals into some functions that can be encapsulated into another procedures
4	C	S	m	Board.java	The conditions of checkSpace() can be replaced for the boat length instead of hard-coded to reuse the code in case that new boats with different length are inserted.
5	D	S	m	Board, Barco,java	Change the MAGIC NUMBERS that control the frontier values and others to Constants.
6	C	S	m	Board.java	Encapsulate some conditions from checkSpace() and insertLastPos(), into another functions in order to have functions with less lines of code.
7	D	S	m	All classes	Remove redundant comments and add description comments.

8	D	S	m	Main.java	Unused import sentences.
9	R	W	M	Player.java	The game crash when the user shoot a position above 9.
10	C	W	M	Player.java	The game crash when the user insert a string instead of an integer.
11	C	W	M	Game.java	The game crash when the user insert a longer value than the maximum length of an integer.
12	C	W	M	Game.java	The scanner not use try catch sentences to check the input of the user.
13	D	U	m	Game.java	The length of the boat must be showed under the board information.
14	D	U	m	Game.java	It could be better to ask for a direction instead of the last position of the boat.

Generic Checklist for Code Reviews

Structure

- Does the code completely and correctly implement the design?
- Does the code conform to any pertinent coding standards?
- Is the code well-structured, consistent in style, and consistently formatted?
- Are there any uncalled or unneeded procedures or any unreachable code?
- Are there any leftover stubs or test routines in the code?
- Can any code be replaced by calls to external reusable components or library functions?
- Are there any blocks of repeated code that could be condensed into a single procedure?
- Is storage use efficient?
- Are symbolics used rather than “magic number” constants or string constants?
- Are any modules excessively complex and should be restructured or split into multiple routines?

Documentation

- Is the code clearly and adequately documented with an easy-to-maintain commenting style?
- Are all comments consistent with the code?

Variables

- Are all variables properly defined with meaningful, consistent, and clear names?
- Do all assigned variables have proper type consistency or casting?
- Are there any redundant or unused variables?

Arithmetic Operations

- Does the code avoid comparing floating-point numbers for equality?
- Does the code systematically prevent rounding errors?
- Does the code avoid additions and subtractions on numbers with greatly different magnitudes?

- Are divisors tested for zero or noise?

Loops and Branches

- Are all loops, branches, and logic constructs complete, correct, and properly nested?
- Are the most common cases tested first in IF- -ELSEIF chains?
- Are all cases covered in an IF- -ELSEIF or CASE block, including ELSE or DEFAULT clauses?
- Does every case statement have a default?
- Are loop termination conditions obvious and invariably achievable?
- Are indexes or subscripts properly initialized, just prior to the loop?
- Can any statements that are enclosed within loops be placed outside the loops?
- Does the code in the loop avoid manipulating the index variable or using it upon exit from the loop?

Defensive Programming

- Are indexes, pointers, and subscripts tested against array, record, or file bounds?
- Are imported data and input arguments tested for validity and completeness?
- Are all output variables assigned?
- Are the correct data operated on in each statement?
- Is every memory allocation deallocated?
- Are timeouts or error traps used for external device accesses?
- Are files checked for existence before attempting to access them?
- Are all files and devices are left in the correct state upon program termination?