Adrià Orozco Lorente 1490952
Arnau Cruz Gargallo 1494996

Functionality: First Implementation of Constructor from Player

Localization: src/main/java/HundirFlota/Player.java, class Player, Player()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testConstructor.

Black box test to see if the constructor is working well for the Player class without added parameters.

Functionality: First Implementation of getName from Player

Localization: src/main/java/HundirFlota/Player.java, class Player, getNombre()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testgetNombre()

Black box test to see if the getter is working well for the Player class comparing exactly the same result.

Functionality: First Implementation of setName from Player

Localization: src/main/java/HundirFlota/Player.java, class Player, setNombre()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testsSetNombre()

Black box test to see if the setter is working well for the Player class comparing exactly the same result.

Functionality: First Implementation of setTurn and getTurn from Player

Localization: src/main/java/HundirFlota/Player.java, class Player, setTurno(), getTurno()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testsetygetTurno()

Black box test to see if the setter and getter of the Turn are working well for the Player class comparing exactly the same result.

Functionality: First Implementation of constructor from Boat class

Localization: src/main/java/HundirFlota/Barco.java, class Barco, Barco()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testconstructor()

Black box test to see if the constructor of the Barco class is working properly comparing exactly the same result.

Functionality: First Implementation of setBoat from Boat class

Localization: src/main/java/HundirFlota/Barco.java, class Barco, setBarco()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testsetbarco()

Black box test to see if the Setter of the Barco class is working properly comparing exactly the same result.

Functionality: First Implementation of getBoat from Boat class

Localization: src/main/java/HundirFlota/Barco.java, class Barco, getBarco()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testgetbarco()

Black box test to see if the Getter of the Barco class is working properly comparing exactly the same result.

Functionality: First Implementation of constructor from Board class

Localization: src/main/java/HundirFlota/Board.java, class Board, Board(), iniciarTablero()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testIniciar()

Black box test to see if the constructor of the Board class is working properly comparing exactly the same result, and the limit values of the Board matrix.

Functionality: First Implementation of constructor from Board class

Localization: src/main/java/HundirFlota/Board.java, class Board, Board(), iniciarTablero()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testIniciar()

Black box test to see if the constructor of the Board class is working properly comparing exactly the same result, and the limit values of the Board matrix.

Functionality: First Implementation of insertBoats from Board class, this function pretend to set all the boats in a board to start the game

Localization: src/main/java/HundirFlota/Board.java, class Board, CrearBarcos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testIniciar()

Black box test to see if the boats are inserted correctly on the board. Simple test that check the number of positions occupied. The test detect a lot of errors at these functionality.

Functionality: First Implementation of showBoard from Board class

Localization: src/main/java/HundirFlota/Board.java, class Board, mostrarTabero()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testMostrarTablero()

Black box test to see if the Board is showed correctly to the comparing exactly the same result.

Functionality: First Implementation of startBoardInfo from Player, this function is destined to create the board that player will see during the game so he can control the information from his previous turns.

Localization: src/main/java/HundirFlota/Player.java, class Player, iniciarTableroInfo()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testIniciarTableroInfo()

Black box test to see if the setter of the information board is working well for the Player class comparing exactly the same result, and added Frontier values to test it.

Functionality: First Implementation of shoot from Player, this function is going to be that one who controlled where the users want to make a shoot to sink a boat.

Localization: src/main/java/HundirFlota/Player.java, class Player, disparar()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testdisparar ()

Black box test to see if the shoot method well for the Player class comparing exactly the same result, and added some Frontier values to test it.

Functionality: First Implementation of ocuppiedPositions from Board, this function count the number of positions that are occupied by ships in a board.

Localization: src/main/java/HundirFlota/Board.java, class Board, posicionesOcupadas()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testPosicionesOcupadas()

Black box test to see if the posicionesOcupadas method was returning correctly the number of positions occupied, for this test we implemented a mock object called mockOb, this mock object implemented some boards so we could test the function in different scenarios.

Functionality: First Implementation of insertBoatfirstpos from Board, this function return true if we can place the first position from our ships, and false if we cannot. It's the second version for insertBoats, that at this point we decide to separate the code to test-it better.

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertBoats1()

Black box test to see if the insertBoatFirstPos method was placing correctly the ships into the board, for this test we used the mock object called mockOb, this mock object implements some boards with boats so we could test the function in different scenarios, we also tested frontier and limit values from the board to check that you can't place the first position from a ship outside the board.

Functionality: First Implementation of checkSpace from Board, this function return true if we have space to place the ship in the board, it checks from the first position counting the length of the boat to the north, east, west, and south, if you can't place the boat in any of this direction this method return false. Also checks if a boat is closely to another one and return false if is parallel to him at the same positions. It's part of the second version of insertBoats()

Localization: src/main/java/HundirFlota/Board.java, class Board, checkSpace()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testcheckSpace()

Black box test to see if the checkSpace method was returning true when we have enough space to place it in the board, and returning false if we haven't enough space, for this test we used the mock object called mockOb, this mock object implements some boards so we could test the function in different scenarios, we also tested some frontier values from the board to check that you can't place a ship outside the board.

Functionality: Second Implementation of checkSpace from Board, we just added some test so we can coverage every scenario.

Localization: src/main/java/HundirFlota/Board.java, class Board, checkSpace()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testcheckSpace()

Black box test to see if the checkSpace method was returning true when we have enough space to place it in the board, and returning false if we haven't enough space correcting previous errors. Added White box tests checking the conditionals structures and the flux of the code. Added a few lines for path coverage

Functionality: Third Implementation of checkSpace from Board, we just added some test so we can coverage every scenario, and also added more boards to the mock object. Also modified the function.

Localization: src/main/java/HundirFlota/Board.java, class Board, checkSpace()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testcheckSpace(),testcheckSpace2()

Black box test to see if the checkSpace method was returning true when we have enough space to place it in the board, checking now for all the types of boats we will have in our game and returning false if we haven't enough space correcting previous errors and added more boards to the mockObject to test more cases.

Added White box tests checking the conditionals structures and the flux of the code, and added path coverage to test it well, the test show us we have to correct some types of conditional structures.

Functionality: Fourth Implementation of checkSpace from Board, we just added some test so we can coverage every scenario, and also added more boards to the mock object. Also finished the function

Localization: src/main/java/HundirFlota/Board.java, class Board, checkSpace()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testcheckSpace(),testcheckSpace2()

testCheckSpace3()

Black box test to see if the checkSpace method was returning true when we have enough space to place it in the board, checking now for all the types of boats we will have in our game and returning false if we haven't enough space correcting previous errors and added more boards to the mockObject to test more cases.

Added White box tests checking the conditionals structures, the decision coverage and the flux of the code, and added path coverage to test it well, At this point this test doesn't show errors.

Functionality: Second Implementation of insertBoatfirstpos from Board, modified the code adding a call to checkSpace to control finally if the boat can be placed there

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertBoats()

Black box test to see if the insertBoatFirstPos method was placing correctly the ships into the board, Adding new integration test to see how this function it works by adding another function result.

Added White box test to full coverage all the code, with decision, path coverage and flux of code.

Functionality: Third Implementation of insertBoatfirstpos from Board, modified the code adding a state to insert the motor boats that occupied only one position

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertBoats()

Black box test to see if the insertBoatFirstPos method was placing correctly the ships into the board, Adding new integration test to see how this function it works by adding another function result.

Added White box test to full coverage all the code, with decision, path coverage and flux of code.

Functionality First Implementation of insertBoatLarstpos from Board, another extension of our first version InsertBoats(). This function will be only called if the result of insertFirsPos() is true, so with this the user can put the boat on the board.

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatLastpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertLastPos()

Black box test to see if the insertBoatLastPos method was placing correctly the ships into the board. The test uses the MockObject previously created to test the board when another boats are putted in too.

Functionality Second Implementation of insertBoatLarstpos from Board, Added all types of boats to be checked and not only the Air craft carriers.

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatLastpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertLastPos()

Black box test to see if the insertBoatLastPos method was placing correctly the ships into the board. The test uses the MockObject previously created to test the board when another boats are putted in too. Coverage the frontier values and the limit values, and the conditions that made the function return false

Discovered some errors at conditional structures thanks to the test

Functionality Thrid Implementation of insertBoatLarstpos from Board, Corrected conditional structures and added extra conditions.

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatLastpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertLastPos()

Black box test to see if the insertBoatLastPos method was placing correctly the ships into the board. The test uses the MockObject previously created to test the board when another boats are putted in too. Coverage the frontier values and the limit values, and the conditions that made the function return false

Added White box tests to, coverage as many code as possible, and passing for all the conditions and decisions

Functionality: Second Implementation of Constructor from Player, Added additional parameters to set the name by constructor

Localization: src/main/java/HundirFlota/Player.java, class Player, Player()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testConstructor.

Black box test to see if the constructor is working well for the Player class with added parameters.

Functionality: Second Implementation of Boat Constructor, We decided to create in other way the boats in order to test the class better and to interact better with other classes, now this class will be created with a list of boats to check if the new boat is correct.+

Localization: src/main/java/HundirFlota/Boat.java, class Boat, Barco()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testCreateShip

Black box test to see if the boats are created correctly. Teste every possible condition about to create any boat and added White box test to full coverage the method.

Functionality: Implementation of isHit(), This method will check on the list of boats if any position where he is positioned is hit by a player shoot.

Localization: src/main/java/HundirFlota/Boat.java, class Boat, isHit()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testifHit()

Black box test to see if the boats are returning the correct values on check it. Teste every possible condition about to shoot any boat and added White box test to full coverage the method.

Created a new MockObject to test the Boat functions to simulate there are boats already positioned and created. With this we can check all the possible conditions.

Functionality: Implementation of isSunk(), The same for isHit, but in this case we will check if all the positions of the boat are shooted so that does it means the boat is sunk.

Localization: src/main/java/HundirFlota/Boat.java, class Boat, isSunk()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testisSunk()

Black box test to see if the boats are returning the correct values on check it. Teste every possible condition about to create any boat and added White box test to full coverage the method.

Used the previous MockObject to test the Boat functions to simulate there are boats already positioned and created. With this we can check all the possible conditions.

Functionality: Second Implementation of isSunk(), Added Boolean to left the boat as sunk

Localization: src/main/java/HundirFlota/Boat.java, class Boat, isSunk()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testisSunk()

Added White box test to full coverage the method.

Functionality: Implementation of checkTypeOfBoat(), This method is only to check the types of boats by their length. This method is integrated in the Getter previously designed so we can reuse the same test for it and test it with integrations testing in the methods is Sunk and isHit

Localization: src/main/java/HundirFlota/Boat.java, class Boat, checkTypeofBoat()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testisSunk(), testisHit()

Black box test to see if the boats are returning the correct values on check it. Teste every possible condition about the types of boats.

Functionality: Implementation of getPositions(), This method is only to check the positions where a boat is. Only returns the initial List. Tested in the method isHit seeing the positions of the boats we will thest.

Localization: src/main/java/HundirFlota/Boat.java, class Boat, getPositions()

Test: src/test/java/HundirFlota/BarcoTest.java, BarcoTest, testisHit()

Black box test to see if the boats are returning the correct values on check it.

Functionality: Added new lines to all the test on Player classes to check the new constructor is going well.

Localization: src/main/java/HundirFlota/Player.java, class Player, Player()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testConstructor., testGetNombre(), testDisparar(), testGetTurno, testSetNombre()

Black box test to see if the constructor is working well for the all the Player Test cases.

Functionality Fourth Implementation of insertBoatLarstpos from Board, added sentences to put the boats already tested on thes positions

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatLastpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertLastPos()

Black box test to see if the insertBoatLastPos method was placing correctly the ships into the board. The test uses the MockObject previously created to test the board when other boats are putted in too. Coverage the frontier values and the limit values, and the conditions that made the function return false

Added White box tests to, coverage as many code as possible, and passing for all the conditions and decisions, thanks to this change in the next versions of the tests we can invocate this function with the integration of the boats and no need mockObjects

Functionality: Fourth Implementation of insertBoatfirstpos from Board, modified the code adding a state to insert the motor boats that occupied only one position, now added at class board to.

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertBoats()

Black box test to see if the insertBoatFirstPos method was placing correctly the ships into the board, Adding new integration test to see how this function it works by adding another function result.

Added White box test to full coverage all the code, with decision, path coverage and flux of code, thanks to this change in the next versions of the tests we can invocate this function with the integration of the boats and no need mockObjects

Functionality: Added more tests to insertBoatfirstpos from Board

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertBoats()

Added test to see how it works to put a motor boat on the board

Functionality: Added more tests to CheckSpace from Board

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testInsertBoats1()

Added test to see how it works to put a motor boat on the board and the space it returns.

Functionality: Modification on how to create players, Create the method crearJugadores() to start the game with this players

Localization: src/main/java/HundirFlota/Player.java, class Player, crearJugadores()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, testSetPlayers()

Black box test with integration on class game who will contain the players to initiate the game. Added coverage too.

Functionality: Second version to create players, method crearJugadores() to start the game with this players

Localization: src/main/java/HundirFlota/Player.java, class Player, crearJugadores()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testconstructor()

Black box test without integration on class Player to see if the constructor and the method to create players work as expected. Added coverage too and modified all the test of the class PlayerTest to check the new changes didn't affect.

Functionality: Second version of set the turn on players on setTurno()

Localization: src/main/java/HundirFlota/Player.java, class Player, crearJugadores()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, testSetPlayers()

Black box test with integration on class game who will contain the players and the turn to initiate the game. Added coverage to.

Functionality: Third version to create players, method crearJugadores() and constructor to start the game with this players, now a player will have associated a board

Localization: src/main/java/HundirFlota/Player.java, class Player, Player()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testconstructor()

Modified all the test of the check the new changes are working well

Functionality: Changes in class Game, Create the method getPlayers() to see the players that are playing

Localization: src/main/java/HundirFlota/Player.java, class Game, getPlayers()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, testConstructor()

Black box test with integration of class Player to initiate the game.

Functionality: Changes in class Game, The constructor will create an array of two players to limit the number of players on a game.

Localization: src/main/java/HundirFlota/Player.java, class Game, Game()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, testConstructor()

Black box test with integration of class Player to initiate the game. Checking also the players we have.

Functionality: Create method getBoard() of every player on this class

Localization: src/main/java/HundirFlota/Player.java, class Player, getTablero()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testGetTablero()

Black box test with integration on class Board to see if the board created is correct and to check it with the real board where the method is instantiated.

Functionality: Modified tests to check both constructors from player are valid

Localization: src/main/java/HundirFlota/Player.java, class Player,Player()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, all the tests

Functionality: Second version of getBoard() method.

Localization: src/main/java/HundirFlota/Player.java, class Player, getTablero()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, modified testGetTablero to check that new version is working properly.

Functionality: Second version of getPlayers() method, we just changed how to represent the players in a board so we had to change whole function.

Localization: src/main/java/HundirFlota/Game.java, class Game, getPlayers()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, modified testSetPlayers to check that new version of getPlayers is working properly.

Functionality: Second version of getPlayers() method, we just changed how to represent the players in a board so we had to change whole function.

Localization: src/main/java/HundirFlota/Game.java, class Game, getPlayers()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, modified testSetPlayers to check that new version of getPlayers is working properly.

Functionality: First version of insertAirCraft() method, it's a method inside the game class(**view part**), this method will ask to the player where he wants to put every AirCraft (type of ship). The function will just call some functions that we had developed before.

Localization: src/main/java/HundirFlota/Game.java, class Game, insertAirCraft()

Test: We didn't make any method to test this functions, because this method was calling functions that are already tested on previous tests, and this method is called in the view where it will interact with the user.

Functionality: First version of insertVessel() method, it's a method inside the game class (**view part**), this method will ask to the player where he wants to put every Vessel (type of ship). The function will just call some functions that we had developed before.

Localization: src/main/java/HundirFlota/Game.java, class Game, insertVessel()

Test: We didn't make any method to test this functions, because this method was calling functions that are already tested on previous tests, and this method is called in the view where it will interact with the user. We did only exploratory testing for this method.

Functionality: First version of insertMotorBoat() method, it's a method inside the game class (**view part)**, this method will ask to the player where he wants to put every MotorBoat (type of ship). The function will just call some functions that we had developed before.

Localization: src/main/java/HundirFlota/Game.java, class Game, insertMotorBoat()

Test: We didn't make any method to test this functions, because this method was calling functions that are already tested on previous tests, and this method is called in the view where it will interact with the user. We did only exploratory testing for this method.

Functionality: First version of play() method, it's a method that for the **view part**, this method will just call all the functions that we developed before.

Localization: src/main/java/HundirFlota/Game.java, class Game, play()

Test: We didn't make any method to test this functions, because this method was calling functions that are already tested on previous tests, and this method is called in the view where it will interact with the user. We did only exploratory testing for this method.

Functionality: Second version of shoot from Player, we added new condition to check if the player is shooting outside the board.

Localization: src/main/java/HundirFlota/Player.java, class Player, disparar()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testdisparar ()

We added some new scenarios (shooting outside the boats) using frontier values to the black box test that we already had. We did only exploratory testing for this method.

Functionality: First version of getWinner().

Localization: src/main/java/HundirFlota/Game.java, class Game, getWinner()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, testconstructor ()

Black box test to check that the function is returning the correct winner.

Functionality: First version of getTurn().

Localization: src/main/java/HundirFlota/Game.java, class Game, getTurn()

Test: src/test/java/HundirFlota/GameTest.java, GameTest, testconstructor ()

Black box test to check that the function is returning the correct turn.

Functionality: First version of getBarcos(), this should return an arrayList of barcos.

Localization: src/main/java/HundirFlota/Board.java, class Board, getBarcos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testGetBarcos ()

Black box test to check that the function is returning the correct list of ships, we used MockOb2. The test adds some boats from the mock object inside a board, and then we call the function getBarcos to check that arraylist was returning exactly those ships.

Functionality: Second version of getBarcos(), corrected an error from method.

Localization: src/main/java/HundirFlota/Board.java, class Board, getBarcos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testGetBarcos ()

Black box test to check that the function is returning the correct list of ships, we used MockOb2. The test adds some boats from the mock object inside a board, and then we call the function getBarcos to check that arraylist was returning exactly those ships.

Functionality: First version of getBarco().

Localization: src/main/java/HundirFlota/Board.java, class Board, getBarco()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testGetBarco ()

Black box test to check that the function is returning the correct ship, the test insert some ships inside a board, and then we call the function getBarco() with some correct and some incorrect ships to check that function is working properly.

Functionality: Third version of shoot from Player, we corrected the new condition that we added on second version of method.

Localization: src/main/java/HundirFlota/Player.java, class Player, disparar()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testdisparar ()

We added more test to check that shoot function is working properly.

Functionality: Second version of showBoard, we added cords so it's easy for the user to see where he wants to shoot.

Localization: src/main/java/HundirFlota/Board.java, class Board, showBoard()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, showBoard ()

We changed the test since we had to add cords to the board so it fits with the new version of showBoard.

Functionality: First version of isEveryBoatSunk, this function check if every ship from a player are sunk. If every ship is sunk it returns true, if there is any ship alive it return false, this method will be used to check if someone won the game.

Localization: src/main/java/HundirFlota/Board.java, class Board, isEveryBoatSunk()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testisEveryBoatSunk() Black box test. On this test we inserted some boats inside the board from a player, and then we sunk them.

Functionality: Second version of isEveryBoatSunk

Localization: src/main/java/HundirFlota/Board.java, class Board, isEveryBoatSunk()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testisEveryBoatSunk().

Added test for all types of boats in different positions and sunked one by one. After every sunk we call the function testisEveryBoatSunk(), that should return true unless we already sunk every boat.

Functionality: First version of numberOfBoatsAlive, this function return the number of boats alive into an array of 3 positions (aircrafts, vessels and motor boats).

Localization: src/main/java/HundirFlota/Board.java, class Board, numberOfBoatsAlive()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, numberOfBoatsAlive() Black box test. On this test we inserted some boats inside the board from a player, and then we sunk them one by one. After every sunk we call the function numberOfBoatsAlive() , and that should return the correct number of boats alive from every type of boat.

Functionality: Second version of numberOfBoatsAlive, this function return the number of boats alive into an array of 3 positions (aircrafts, vessels and motor boats).

Localization: src/main/java/HundirFlota/Board.java, class Board, numberOfBoatsAlive()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, numberOfBoatsAlive()

Inserted more boats to test all the conditions and full coverage.

Functionality: Fifth version of insertBoatLastPosition, modified one error that we had and we noticed during tests, we weren't checking properly what happens if you try to place the first position from the boat and the last position from the boat in the same cords.

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatLastPosition()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, insertBoatLastPosition() Black box test. During the white box and black box test for this method we realized that there was a mistake on the code.

Functionality: Fifth version of shoot method, added that when you shoot a boat and this boat sunk it appears a message that you made a boat sunk.

Localization: src/main/java/HundirFlota/Board.java, class Board, disparar()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, insertBoatLastPosition() Black box test. Added test to check that the method is working perfectly, adding a call to this function making a boat sunk, and check that message is appearing properly.

Functionality: Second version of insertAirCraft() method, we notified that there was an error that wasn't setting the values of correctValue and correctValueTwo false after collocate a boat.

Localization: src/main/java/HundirFlota/Game.java, class Game, insertAirCraft()

Test: This method has no test.

Functionality: Second version of insertVessel() method, we notified that there was an error that wasn't setting the values of correctValue and correctValueTwo false after collocate a Vessel.

Localization: src/main/java/HundirFlota/Game.java, class Game, insertVessel()

Test: This method has no test.

Functionality: Second version of insertMotorBoat() method, we notified that there was an error that wasn't setting the values of correctValue and correctValueTwo false after collocate a motor boat.

Localization: src/main/java/HundirFlota/Game.java, class Game, insertMotorBoat()

Test: This method has no test.

Functionality: Sixth version of shoot from Player, we corrected there was a mistake on the tests, we need to call getTablero from player, instead of calling a board.

Localization: src/main/java/HundirFlota/Player.java, class Player, disparar()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testdisparar ()

We added more test to check that shoot function is working properly.

Functionality: Seventh version of shoot method, added coverage from the player class to test the integration with the isSunk() Method

Localization: src/main/java/HundirFlota/Board.java, class Board, disparar()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testdisparar()

Functionality: Added more tests to CheckSpace from Board

Localization: src/main/java/HundirFlota/Board.java, class Board, insertBoatfirstpos()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testCheckSpace3()

Added test to see to check if the motor boats and the vessels are working well on all the possible Conditions.

Functionality: Another version of checkTypeOfBoat(), Changed the name of the boats

Localization: src/main/java/HundirFlota/Boat.java, class Boat, checkTypeofBoat()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testcreateShip()

Black box test to see if the boats are returning the correct values on check it with the new Names.

Functionality: Added loop test on startBoardInfo

Localization: src/main/java/HundirFlota/Player.java, class Player, iniciarTableroInfo()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, iniciarTableroInfo()

Loop test for a nested loop checking the internal and the external loop comparing with the complete matrix and some inside values of these.

Functionality: Added loop test on Initialize Board

Localization: src/main/java/HundirFlota/Board.java, class Board, Board(), iniciarTablero()

Test: src/test/java/HundirFlota/BoardTest.java, BoardTest, testIniciar()

Loop test for a nested loop checking the internal and the external loop comparing with the complete matrix.

Functionality: First verision of getBoardInfo, this method returns a string of tableroInfo so the user can see where he had shot already.

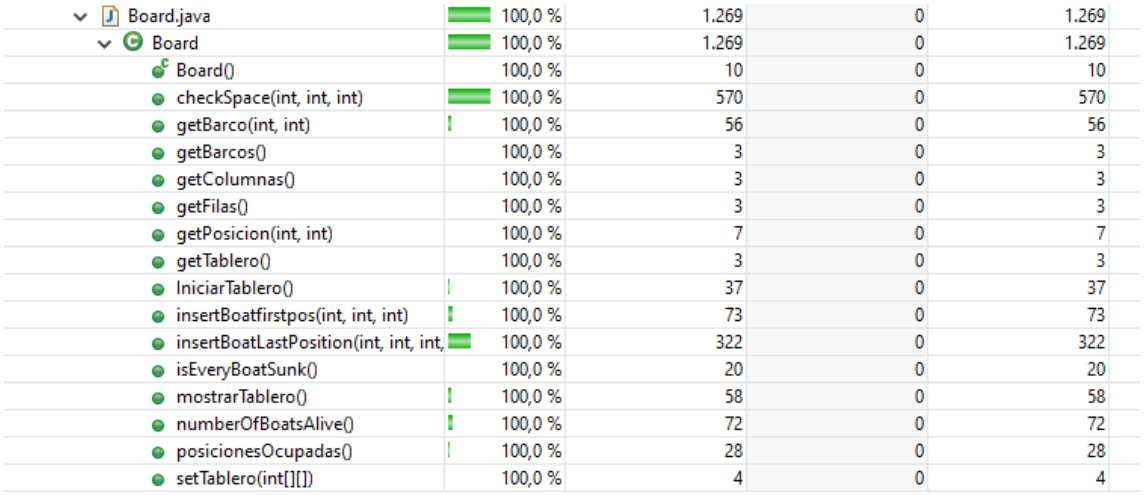
Localization: src/main/java/HundirFlota/Player.java, class Player, mostrarTableroInfo()

Test: src/test/java/HundirFlota/PlayerTest.java, PlayerTest, testmostrarTableroInfo()

Black box test, this test starts a board, and then check that the board info is started properly, later we shoot on different places, where there isn't any boat, then we call an assertEqual, to compare with a board that we changed manually the squares. The test includes limit values for the matrix and integration test with the insertion of new boats into the board and shoots from the player.

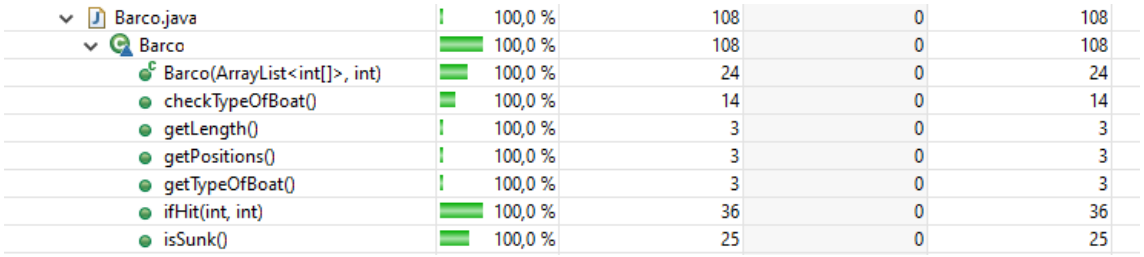
There are the screenshots of the statement coverage for all the methods

Board class:



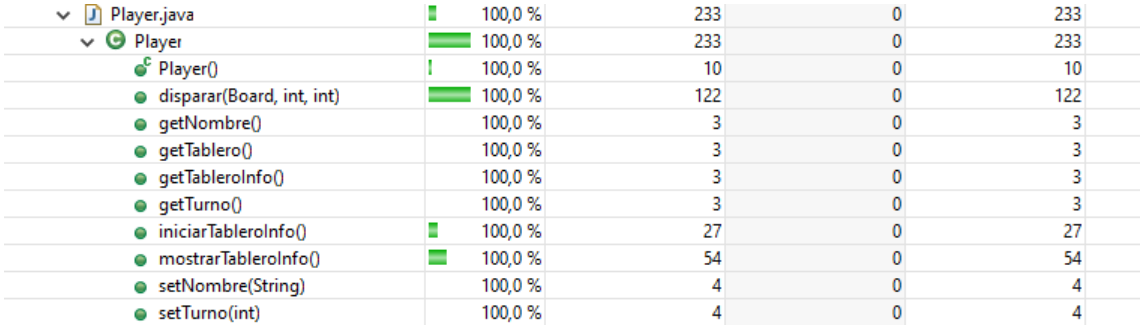
Board.java	100,0 %	1.269	0	1.269
Board	100,0 %	1.269	0	1.269
Board()	100,0 %	10	0	10
checkSpace(int, int, int)	100,0 %	570	0	570
getBarco(int, int)	100,0 %	56	0	56
getBarcos()	100,0 %	3	0	3
getColumnas()	100,0 %	3	0	3
getFilas()	100,0 %	3	0	3
getPosicion(int, int)	100,0 %	7	0	7
getTablero()	100,0 %	3	0	3
IniciarTablero()	100,0 %	37	0	37
insertBoatfirstpos(int, int, int)	100,0 %	73	0	73
insertBoatLastPosition(int, int, int,	100,0 %	322	0	322
isEveryBoatSunk()	100,0 %	20	0	20
mostrarTablero()	100,0 %	58	0	58
numberOfBoatsAlive()	100,0 %	72	0	72
posicionesOcupadas()	100,0 %	28	0	28
setTablero(int[][])	100,0 %	4	0	4

“Barco”, Boat class:



Barco.java	100,0 %	108	0	108
Barco	100,0 %	108	0	108
Barco(ArrayList<int[]>, int)	100,0 %	24	0	24
checkTypeOfBoat()	100,0 %	14	0	14
getLength()	100,0 %	3	0	3
getPositions()	100,0 %	3	0	3
getTypeOfBoat()	100,0 %	3	0	3
ifHit(int, int)	100,0 %	36	0	36
isSunk()	100,0 %	25	0	25

Player class:



Player.java	100,0 %	233	0	233
Player	100,0 %	233	0	233
Player()	100,0 %	10	0	10
disparar(Board, int, int)	100,0 %	122	0	122
getNombre()	100,0 %	3	0	3
getTablero()	100,0 %	3	0	3
getTableroInfo()	100,0 %	3	0	3
getTurno()	100,0 %	3	0	3
iniciarTableroInfo()	100,0 %	27	0	27
mostrarTableroInfo()	100,0 %	54	0	54
setNombre(String)	100,0 %	4	0	4
setTurno(int)	100,0 %	4	0	4

Some screenshots from decision, condition and path coverage.

```java
public boolean insertBoatfirstpos(int fila, int columna, int BoatLeng) {
    if(fila<0 || fila>9 || columna<0 || columna>9) {
        return false;
    }else if(this.getPosicion(fila, columna)!=0) {
            return false;
    }else {
        //Call to checkSpace
        boolean FreeSpace=checkSpace(fila, columna, BoatLeng);
        if(FreeSpace==true && BoatLeng==1) {
            Tablero[fila][columna]=1;
            ArrayList<int[]> positions = new ArrayList<int[]>();
            int[] pos = new int[3];
            pos[0]=fila;
            pos[1]=columna;
            pos[2]=0;
            positions.add(pos);
            Barco barco = new Barco(positions,BoatLeng);
            barcos.add(barco);

        }
        return FreeSpace;
    }
```

```java
public boolean insertBoatLastPosition(int filaIn, int columnaIn, int filaFi, int columnaFi, int BoatLeng) {
    ArrayList<int[]> positions = new ArrayList<int[]>();
    if(filaFi<0 || filaFi>9 || columnaFi<0 || columnaFi>9 || columnaIn==columnaFi && filaIn==filaFi) {
        return false;
    }else if(this.getPosicion(filaFi, columnaFi)!=0) {
            return false;
    }else {
        if(filaIn!=filaFi && columnaFi!=columnaIn) {
            return false;
        }
        else {
            //Insert right
            if(columnaFi>columnaIn) {
                if(columnaFi-columnaIn!=BoatLeng-1) {
                    return false;
                }else {
                    boolean occuped=false;
                    for(int i=columnaIn; i<=columnaFi; i++) {
                        if(Tablero[filaIn][i]==1) {
                            occuped=true;
                        }
                    }
                    if(occuped==true) {
                        return false;
                    }else {
```

```
391                     }
392                     //insert left
393                     if(columnaFi<columnaIn) {
394                         if(columnaIn-columnaFi!=BoatLeng-1) {
395                             return false;
396                         }else {
397                             boolean occuped=false;
398                             for(int i=columnaFi; i<=columnaIn; i++) {
399                                 if(Tablero[filaIn][i]==1) {
400                                     occuped=true;
401                                 }
402                             }
403                             if(occuped==true) {
404                                 return false;
405                             }else {
406                                 for(int i=columnaFi; i<=columnaIn; i++) {
407                                     Tablero[filaIn][i]=1;
408                                     int[] pos = new int[3];
409                                     pos[0]=filaIn;
410                                     pos[1]=i;
411                                     pos[2]=0;
412                                     positions.add(pos);
413                                 }
414                             }
415
```

```
    public boolean disparar(Board tablero,int fila,int columna) {
        boolean hit=false;

        if(fila>tablero.getFilas() || columna>tablero.getColumnas() || fila<0 || columna<0){
            hit=false;
            System.out.println("WRONG COORDS: "+fila+" "+columna);

        }else if(this.tableroInfo[fila][columna]=="[X]") {
            System.out.println("MISSED SHOOT, TAKE CARE BECAUSE YOU ALREADY SHOT THIS POSTION ON A PREVIOUS TURN");
            return false;
        }
        else if(tablero.getPosicion(fila, columna) == 1) {
            hit=true;
            System.out.println("HIT SHOOT on: "+fila+" "+columna);
            this.tableroInfo[fila][columna]="[X]";
            tablero.getBarco(fila, columna).ifHit(fila, columna);
            if(tablero.getBarco(fila, columna).isSunk()) {
                System.out.println("YOU SUNK ONE "+tablero.getBarco(fila, columna).getTypeOfBoat());
            }

        }else {
            hit=false;
            System.out.println("MISSED SHOOT on "+fila+" "+columna);
            this.tableroInfo[fila][columna]="[O]";
        }
        return hit;
    }

}
```