Juan Edgar Baldelomar Salazar 1423955
Hatim Mezouar El Mejdoubi 1362302

BattleShip

CLASS USER

Funcionalitat: Comprobamos que el método SetNameUser() asigne el nombre que le pasamos de forma correcta y que lo pasa a la función getNameUser(). Utilizamos pruebas de caja negra.

Localització:

- **Arxiu:** User.java
- **Classe:** User
- **mètode desenvolupat:** public void setNameUser(String nameUser)

Test:

- **Arxiu:** UserTest.java
- **Classe:** UserTest
- **Mètode de test associat a la funcionalitat:** public void testGetSetNameUser()

Funcionalitat: Comprobamos que el método SetCoordinates() asigne los valores que le pasamos de forma correcta y que lo pasa a la función getRow(), getCol(), getOrientacion(), getTypeBoat(). Utilizamos pruebas de caja negra con las técnicas de particiones equivalentes y valores límites y frontera. También utilizamos pruebas de caja blanca con las técnicas de Decision Coverage y Condicion Coverage.

Localització:

- **Arxiu:** User.java
- **Classe:** User
- **mètode desenvolupat:** public void setCoordinates(int row, int col, int orientacion, int typeBoat)

Test:

- **Arxiu:** UserTest.java
- **Classe:** UserTest
- **Mètode de test associat a la funcionalitat:** public void testGetSetcoordinates()

CLASS BOARD

Funcionalitat: Comprobamos que se pueda colocar los diferentes barcos en las coordenadas introducidas. Utilizamos pruebas de caja negra con las técnicas de particiones equivalentes y valores límites y frontera**.**

Localització:

- **Arxiu:** Board.java
- **Classe:** Board
- **mètode desenvolupat:** public int setBoat(int initX, int initY, int orientacion, int boatType)

Test:

- **Arxiu:** BoardTest.java

Juan Edgar Baldelomar Salazar 1423955

Hatim Mezouar El Mejdoubi 1362302

- **Classe:** BoarTest
- **Mètode de test associat a la funcionalitat:** public void testSetBoats()

Funcionalitat: Realizamos disparos para comprobar si es agua, fallido o hemos dado un barco. Utilizamos pruebas de caja negra con las técnicas de particiones equivalentes y valores límites y frontera**.**

Localització:

- **Arxiu:** Board.java
- **Classe:** Board
- **mètode desenvolupat:** public boolean shotBoat(int x, int y)

Test:

- **Arxiu:** BoardTest.java
- **Classe:** BoardTest
- **Mètode de test associat a la funcionalitat:** public void testShootBoat()

Funcionalitat: Comprobamos que si se ha ganado la partida. Utilizamos pruebas de caja negra.

Localització:

- **Arxiu:** Board.java
- **Classe:** Board
- **mètode desenvolupat:** public boolean checkWinner()

Test:

- **Arxiu:** BoardTest.java
- **Classe:** Board
- **Mètode de test associat a la funcionalitat:** public void testCheckWinner()

Funcionalitat: Comprobamos que inicializa bien el tablero con los tableros definidos. Utilizamos pruebas de caja negra.

Localització:

- **Arxiu:** Board.java
- **Classe:** Board
- **mètode desenvolupat:**

Test:

- **Arxiu:** BoardTest.java
- **Classe:** Board
- **Mètode de test associat a la funcionalitat:** public void testInitBoard()

CLASS BOAT

Funcionalitat: Comprobamos que los diferentes barcos puedan colocarse en las coordenadas introducidas. Utilizamos pruebas de caja negra con las técnicas de particiones equivalentes y valores límites y frontera**.**

Localització:

- **Arxiu:** Boat.java

Juan Edgar Baldelomar Salazar 1423955
Hatim Mezouar El Mejdoubi 1362302

- **Classe:** Boat
- **mètode desenvolupat:** public boolean checkRule(int x, int y, int orientacion, int[][] board)

Test:

- **Arxiu:** BoatTest.java
- **Classe:** BoatTest
- **Mètode de test associat a la funcionalitat:** public void testCheckRule()

Funcionalitat: Comprobamos que en las coordenadas el espacio esta disponible. Utilizamos pruebas de caja negra.

Localització:

- **Arxiu:** Boat.java
- **Classe:** Boat
- **mètode desenvolupat:** public boolean freeSpace(int x, int y, int orientacion, int[][] board)

Test:

- **Arxiu:** BoatTest.java
- **Classe:** BoatTest
- **Mètode de test associat a la funcionalitat:** public void testFreeSpace()

Funcionalitat: Comprobamos que los diferentes barcos con sus respectivas medidas. Se ha utilizado pruebas de caja negra.

Localització:

- **Arxiu:** Boat.java
- **Classe:** Boat
- **mètode desenvolupat:** getsize()

Test:

- **Arxiu:** BoatTest.java
- **Classe:** BoatTest
- **Mètode de test associat a la funcionalitat:** public void testInitBoats()

DECISION COVERAGE

Funcionalitat: Realizamos el decision coverage al método SetCoordinates() utilizando los valores que hemos introducido en un array del mockUser para que el test nos devuelva un True y después con otros valores un false.

Localització:

- **Arxiu:** User.java
- **Classe:** User
- **mètode desenvolupat:** public boolean setCoordinates(int row, int col, int orientacion, int typeBoat)

Test:

- **Arxiu:** UserTest.java
- **Classe:** UserTest
- **Mètode de test associat a la funcionalitat:** public void decisionTest()

Juan Edgar Baldelomar Salazar 1423955
Hatim Mezouar El Mejdoubi 1362302

CONDITION COVERAGE

Funcionalitat: Realizamos el condition coverage para el método setCoordinates() utilizando el array del MockUser para las diferentes pruebas. En total se ha realizado 8 pruebas que en cada parámetro se ha obtenido el True y el False.

Localització:

- **Arxiu:** User.java
- **Classe:** User
- **mètode desenvolupat:** public boolean setCoordinates(int row, int col, int orientacion, int typeBoat)

Test:

- **Arxiu:** UserTest.java
- **Classe:** UserTest
- **Mètode de test associat a la funcionalitat:** public void conditionCovTest()

UserTest.java	100.0 %	631	0	631
UserTest	100.0 %	631	0	631
conditionCovTest()	100.0 %	201	0	201
decisionTest()	100.0 %	51	0	51
setUp()	100.0 %	15	0	15
testGetSetcoordinates()	100.0 %	313	0	313
testGetSetNameUser()	100.0 %	48	0	48
Battleship	0.0 %	0	12	12

MOCKOBJECT

Funcionalitat: Se ha realizado un MockBoard que almacena diferentes matrices con barcos para hacer comprobaciones de los diferentes métodos que lo requieran. Como por ejemplo; `public void testShootBoat()`, `public void testCheckWinner()`, `public void testCheckRule()` y `public void testFreeSpace()`.

Localització:

- **Arxiu:** MockBoard.java
- **Classe:** MockBoard
- **mètode desenvolupat:**

  public int[][] getBoardOneAlive()

  public int[][] getShootBoard()

  public int[][] getB1()

  public int[][] getBoardTestFreeSpace()

  public int[][] getTestSubmarineDiagonal()

  public int[][] getTestSubmarine()

  public int[][] getTestBigBoatsDiagonal()

  public int[][] getTestBigBoats()

  public int[][] getTestMediumBoatsDiagonal()

Juan Edgar Baldelomar Salazar 1423955

Hatim Mezouar El Mejdoubi 1362302

public int[][] getTestMediumBoats()

public int[][] getTestBoardLittleBoats()

Test:

- **Arxiu:** BoardTest.java y BoatTest.java
- **Classe:** BoardTest y BoatTest
- **Mètode de test associat a la funcionalitat:**

  public void testShootBoat()

  public void testCheckWinner()

  public void testSetBoats()

MOCKOBJECT

Funcionalitat: Se ha realizado un MockUser que almacena diferentes arrays con diferentes 4 parámetros para hacer comprobaciones con el método setCoordinates().

Localització:

- **Arxiu:** MockUser.java
- **Classe:** MockUser
- **mètode desenvolupat:**

  public int[] decisionTestTrue()

  public int[] decisionTestFalse()

  public int[] conditionTestFirstParamTrue()

  public int[] conditionTestFirstParamFalse()

  public int[] conditionTestSecondParamTrue()

  public int[] conditionTestSecondParamFalse()

  public int[] conditionTestThirdParamTrue()

  public int[] conditionTestThirdParamFalse()

  public int[] conditionTestFourParamTrue()

  public int[] conditionTestFourParamFalse()

Test:

- **Arxiu:** UserTest.java
- **Classe:** UserTest
- **Mètode de test associat a la funcionalitat:**

  public void decisionTest()

  public void conditionCovTest()