Marc Belmonte Alcázar

Xavier Prats Fernández

Dilluns 10:30

2048 (https://github.com/NIU1633672/2048TQS)

**Funcionalitat**: Inicialitzar el tauler

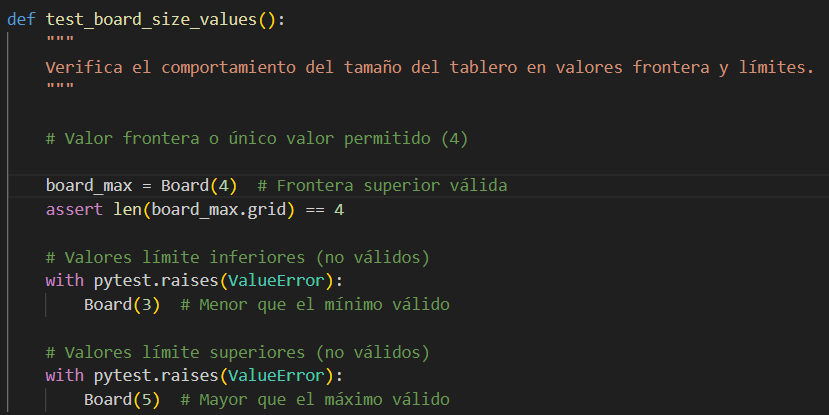
**Localització**: <Arxiu, classe i mètode desenvolupat>

* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: \_init\_

**Test**:

* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat:
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:



* Valors límit i frontera (Límits: 3, 5. Frontera: 4) / Particions equivalents (valor vàlid = 4, no vàlid != 4)  
  

**Funcionalitat**: Reset

**Localització**: <Arxiu, classe i mètode desenvolupat>

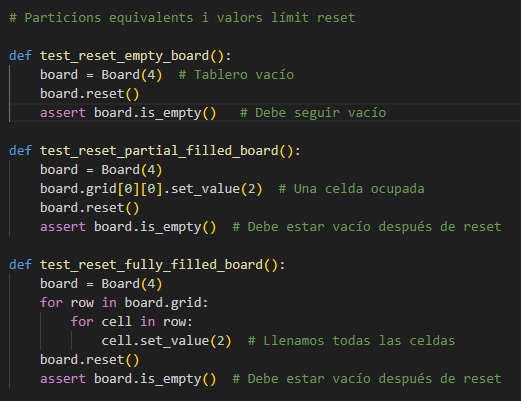
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: reset

**Test**:

* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat:
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:



* Valors límit i frontera / Particions equivalents (tauler ple, amb algunes posicions ocupades, buit)



**Funcionalitat**: Afegir cel·la aleatòria

**Localització**: <Arxiu, classe i mètode desenvolupat>

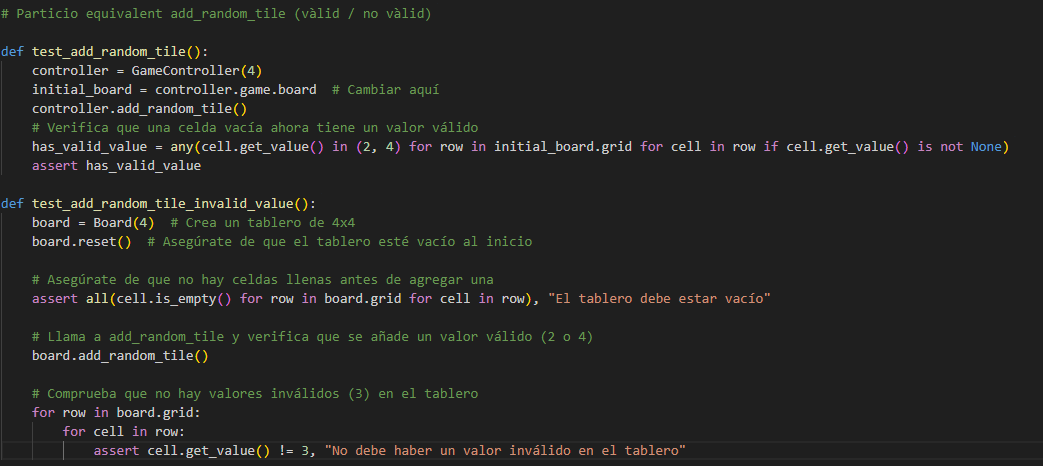
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: add\_random\_tile

**Test**:

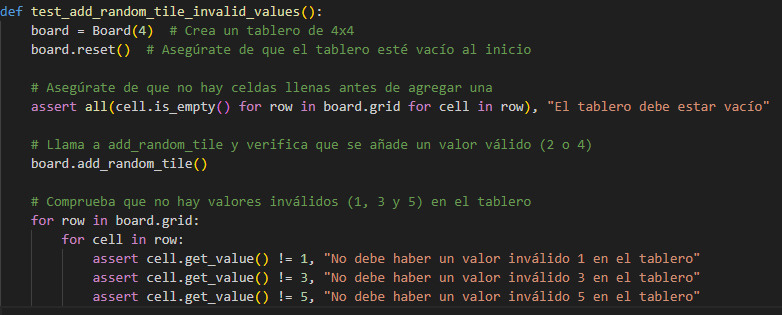
* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat:
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:



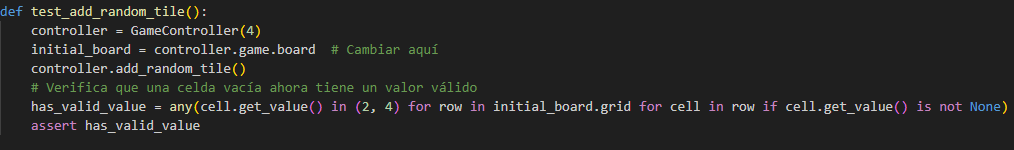
* Particions equivalents (valor vàlid / no vàlid)



* Valor límit superior i inferior (1,3,5)



* Valor frontera (2 o 4)



**Funcionalitat**: Moure a l’esquerra

**Localització**: <Arxiu, classe i mètode desenvolupat>

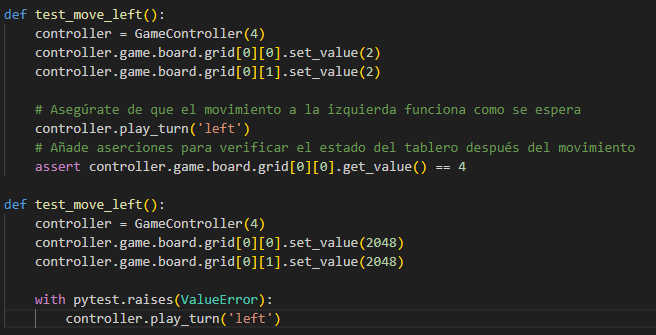
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: move\_left

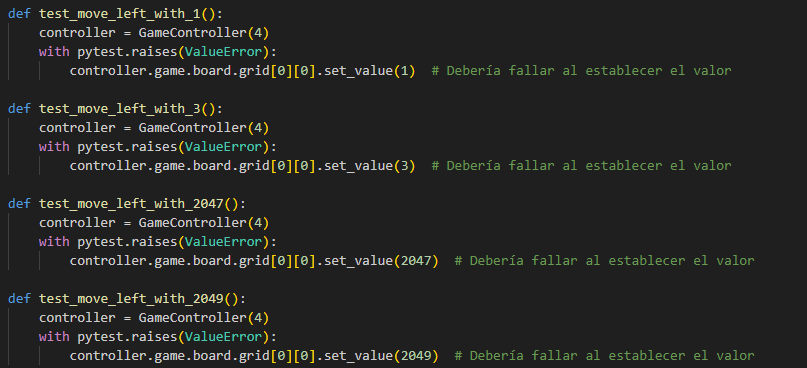
**Test**:

* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat:
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:

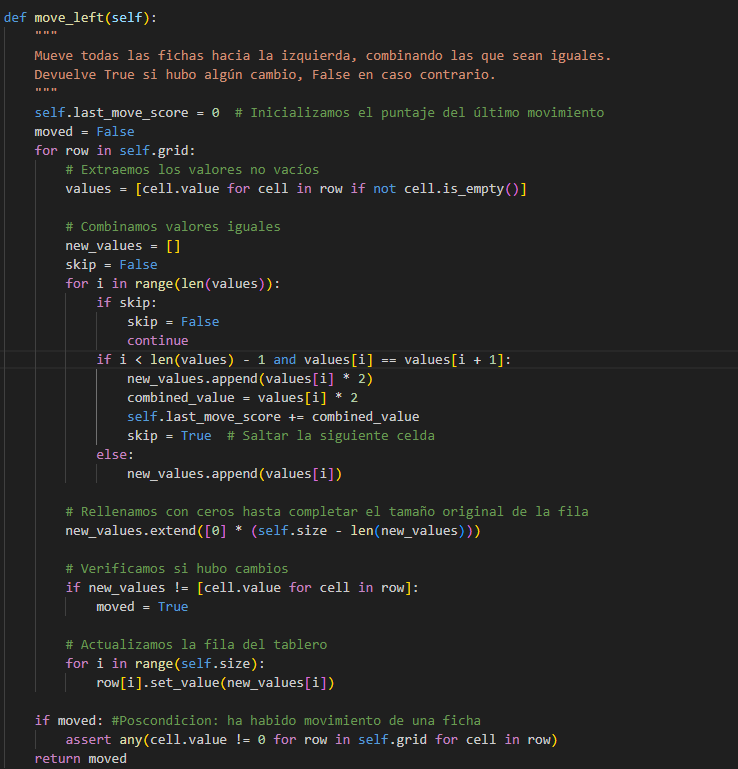
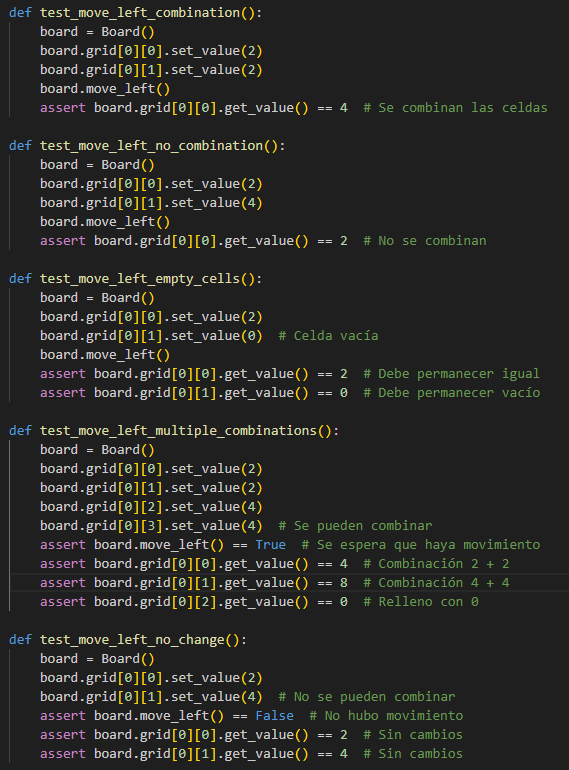


* Valor límit / frontera:





* Particions equivalents: combinació possible / no possible. També són proves de Decision coverage



if i < len(values) - 1 and values[i] == values[i + 1]:  
  
- Ruta 1: Si son iguals, es combinen

- Ruta 2: Si no son iguals, s’afegeixen a new\_values

if new\_values != [cell.value for cell in row]:

- Ruta 1: Si hi ha canvis, s’estableix moved = true

- Ruta 2: Si no hi ha canvis, moved = false

Amb això cobrim:

* combinació de cel·les si o no
* canvis en l’estat del tauler (canvis o sense canvis)
* cel·les buides / plenes

**Funcionalitat**: Moure a la dreta

**Localització**: <Arxiu, classe i mètode desenvolupat>

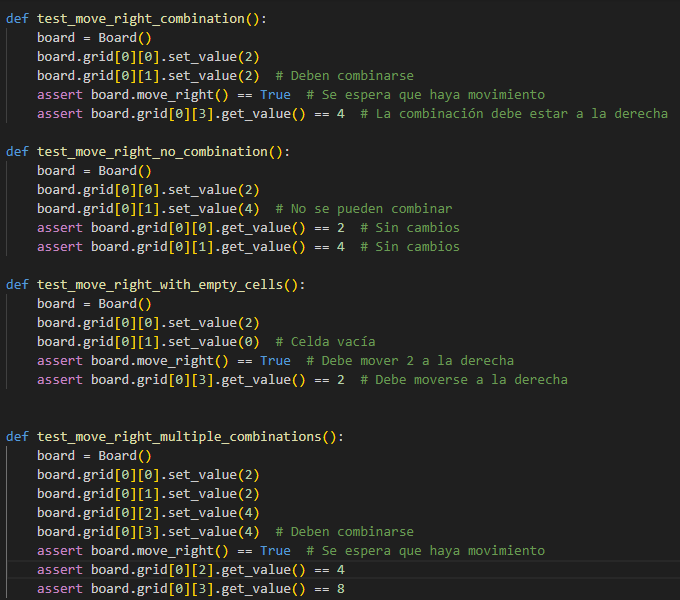
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: move\_right

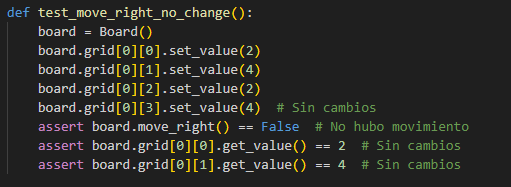
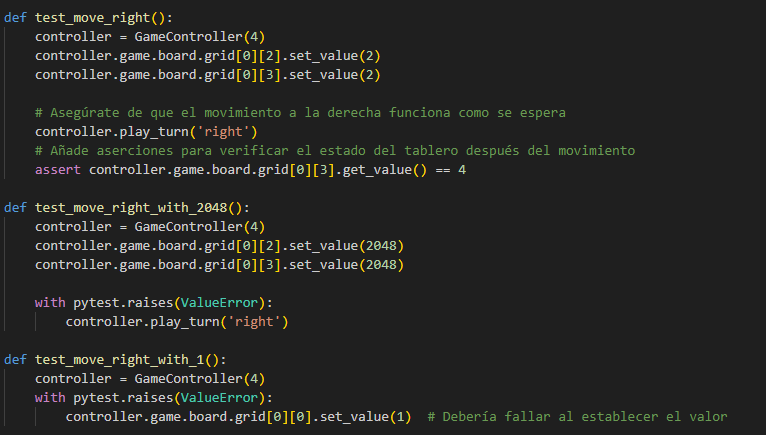
**Test**:

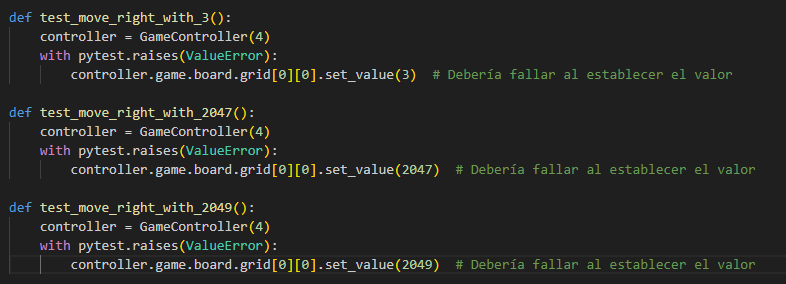
* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat: test\_move\_right
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:



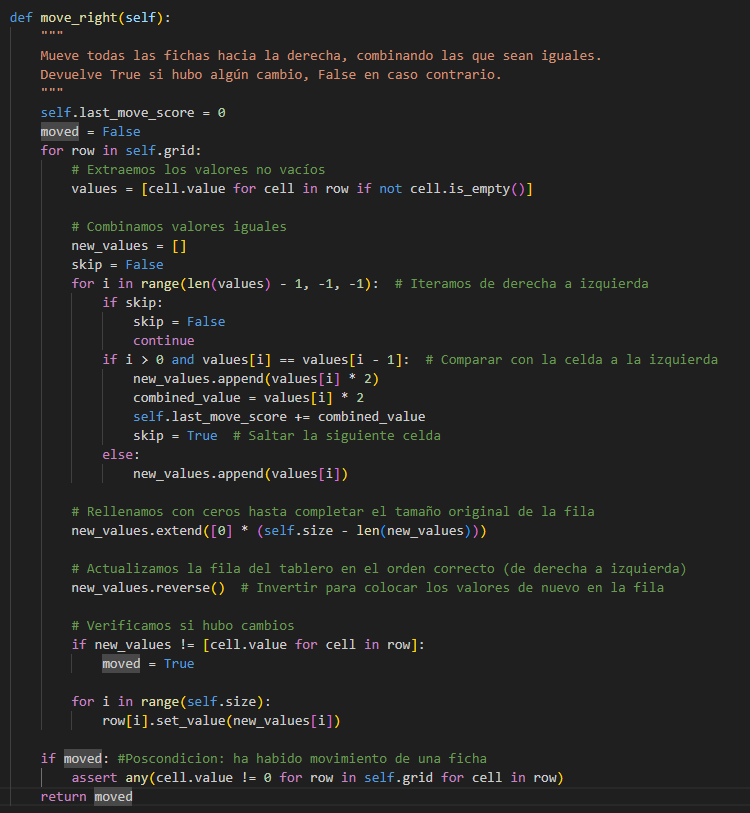
* Particions equivalents (cel·les es poden combinar, no es poden combinar, cel·les buides, cel·les amb un valor i una cel·la buida que es poden moure).

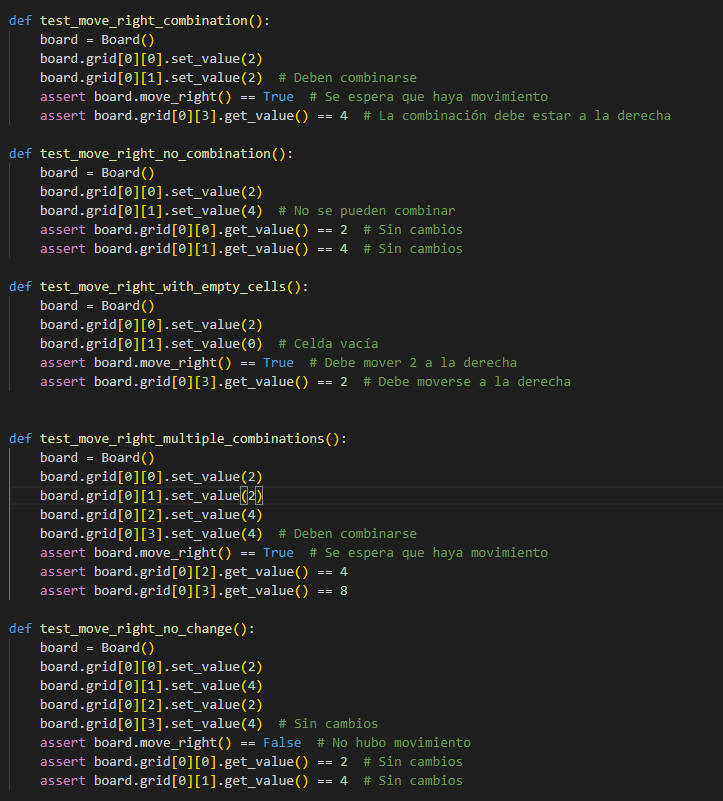


Valors límit i frontera:  
  




* Decision coverage:





* test\_move\_right\_combination: Cobreix el cas on es combinen valors (verdader per la condició de combinació).
* test\_move\_right\_no\_combination: Cobreix el cas on no hi ha combinació (fals per la condició de combinació).
* test\_move\_right\_with\_empty\_cells(): Cobreix el moviment d’una cel·la una cel·la buida.
* test\_move\_right\_multiple\_combinations: Cobreix múltiples combinacions, assegurant que s’executen correctament múltiples decisions en una sola fila.
* test\_move\_right\_no\_change: Cobreix el cas on no hi ha cap canvi, cosa crucial per a la decisió de veure si s’ha mogut alguna cosa.

**Funcionalitat**: Moure cap amunt

**Localització**: <Arxiu, classe i mètode desenvolupat>

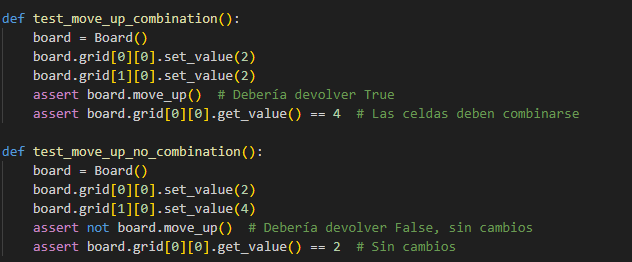
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: move\_up

**Test**:

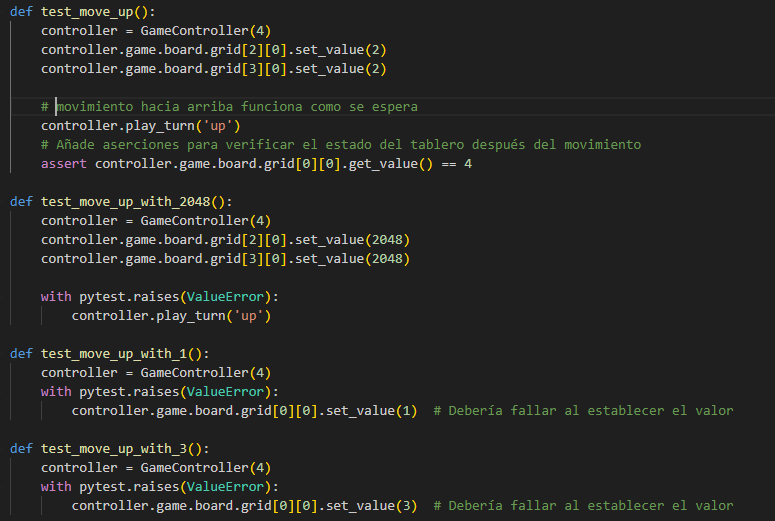
* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat: test\_move\_up
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:

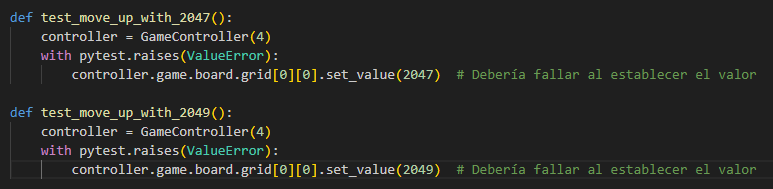


* + Particions equivalents (combinació possible o no)



* Valors límit i frontera





**Funcionalitat**: Moure cap avall

**Localització**: <Arxiu, classe i mètode desenvolupat>

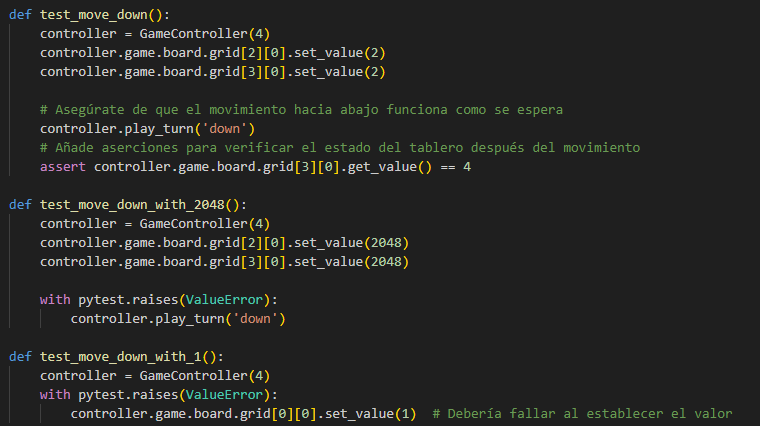
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: move\_down

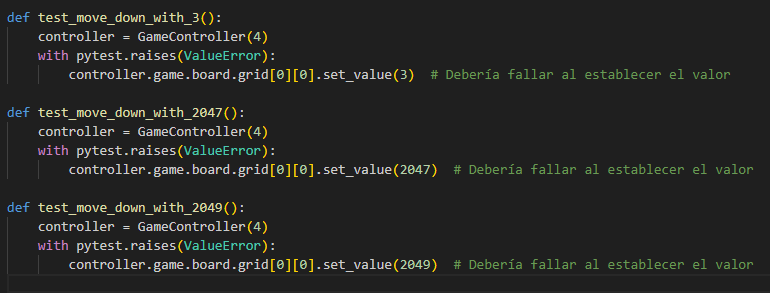
**Test**:

* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat: Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:

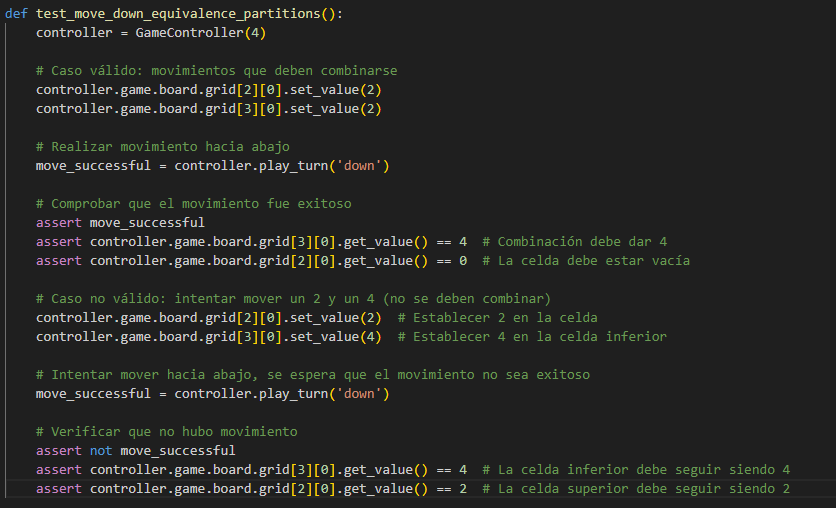


* + Valors límit i frontera:





* Particions equivalents (moviment possible / no possible):



**Funcionalitat**: Tauler ple

**Localització**: <Arxiu, classe i mètode desenvolupat>

* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: is\_full

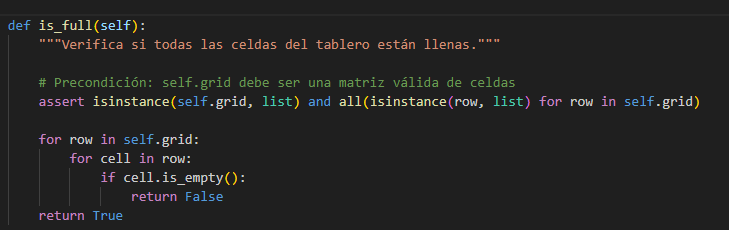
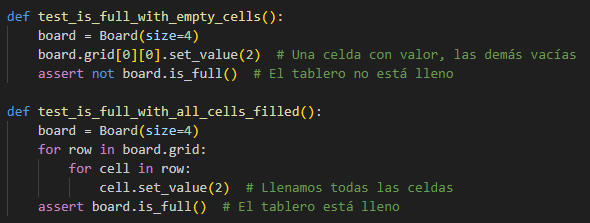
**Test**:

* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat:
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:



* + Decision coverage i condition coverage:

És decision coverage perquè verifiquem que l’if sigui true i false així com la condició sigui true o false



**Funcionalitat**: Hi ha moviments

**Localització**: <Arxiu, classe i mètode desenvolupat>

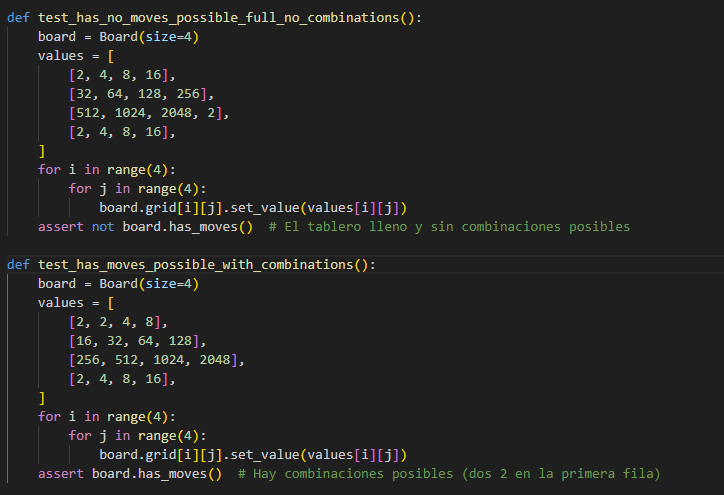
* Arxiu: src/model/board.py
* Classe: Board
* Mètode desenvolupat: has\_moves

**Test**:

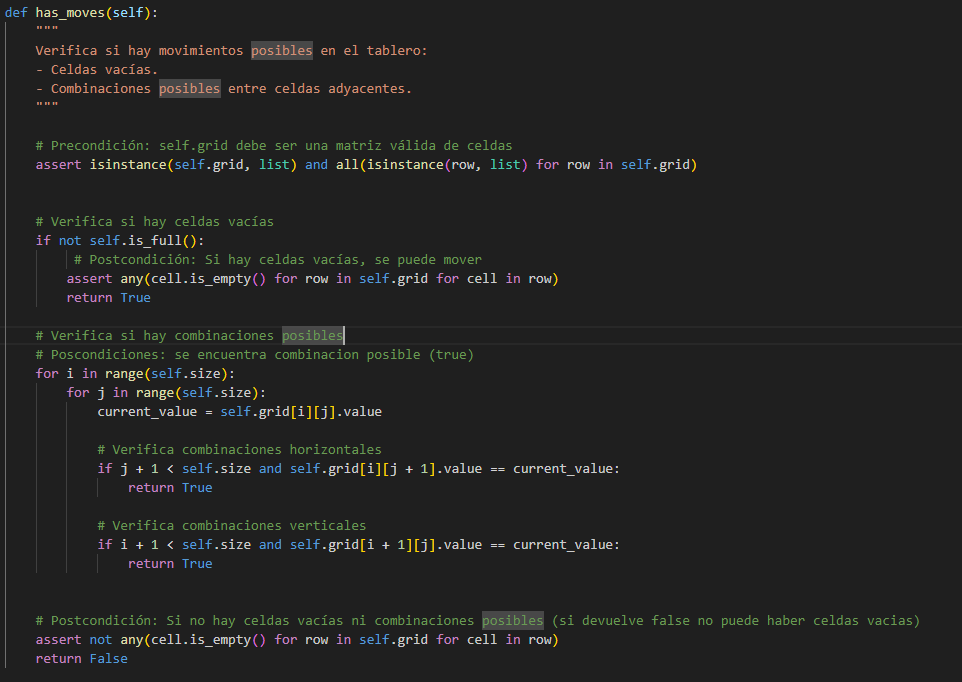
* Arxiu: tests/test\_board.py
* Mètode de test associat a la funcionalitat: has\_moves
* Caixa negra / caixa blanca
* Tècniques utilitzades:
  + Statement coverage:

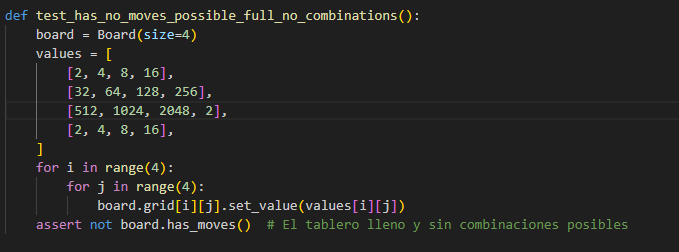


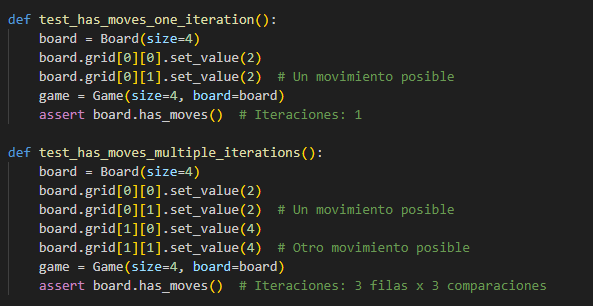
* Particions equivalents i valors limit / frontera (hi ha moviments o no hi ha moviments)



* Loop testing



0 iteraciones: 



**Funcionalitat**: Init

**Localització**:

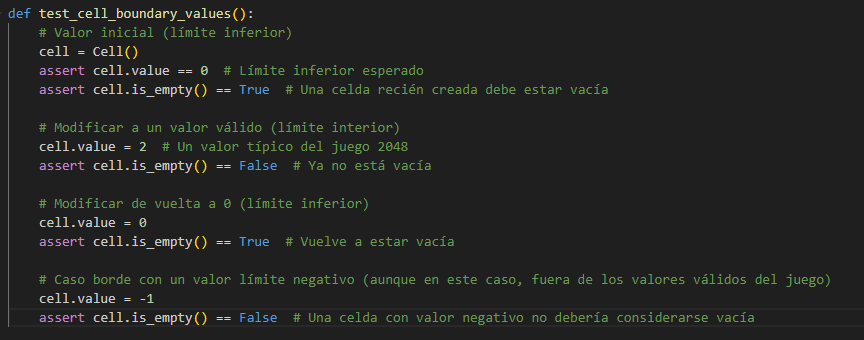
* Arxiu: src/model/cell.py
* Classe: Cell
* Mètode desenvolupat: Init

**Test**:

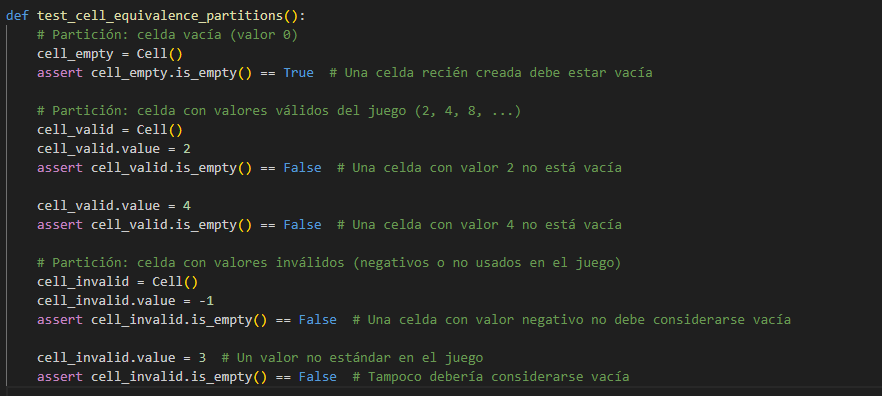
* Arxiu: tests/test\_cell.py
* Mètode de test associat a la funcionalitat: Caixa negra / caixa blanca
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

Statement coverage:  
  


* Valors límit i frontera



* Particions equivalents:



**Funcionalitat**: is\_empty

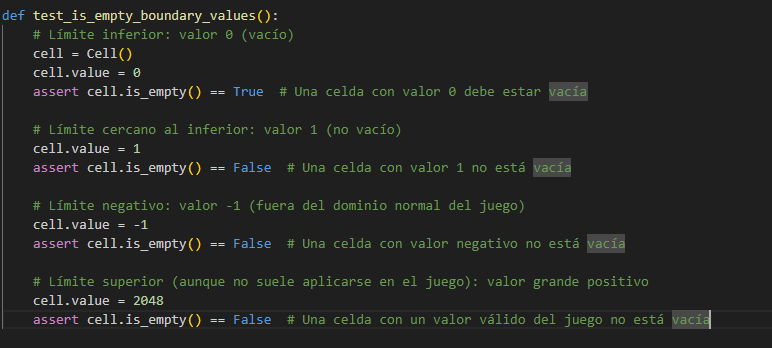
**Localització**:

* Arxiu: src/model/board.py
* Classe: Cell
* Mètode desenvolupat: is\_empty

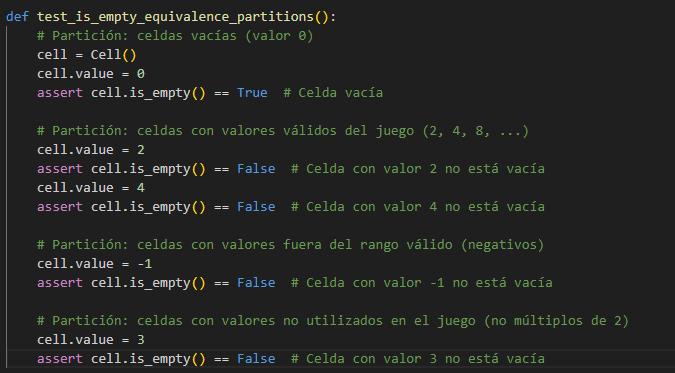
**Test**:

* Arxiu: tests/test\_cell.py
* Mètode de test associat a la funcionalitat: Caixa negra / caixa blanca
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

Statement coverage:  
  


Valors limit i frontera:  
  


Particions equivalents:



**Funcionalitat**: set\_value

**Localització**:

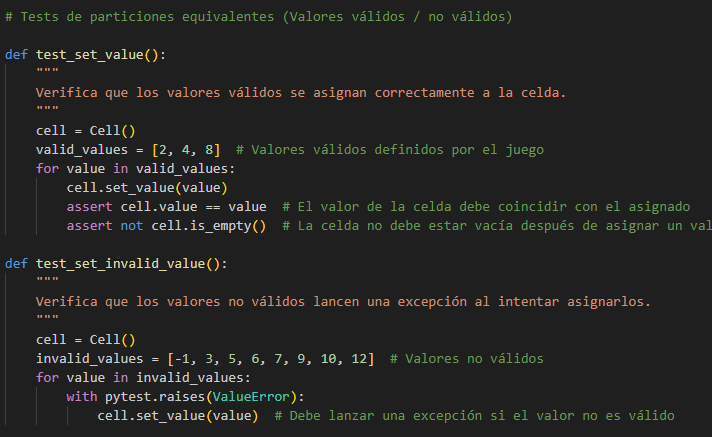
* Arxiu: src/model/board.py
* Classe: Cell
* Mètode desenvolupat: set\_value

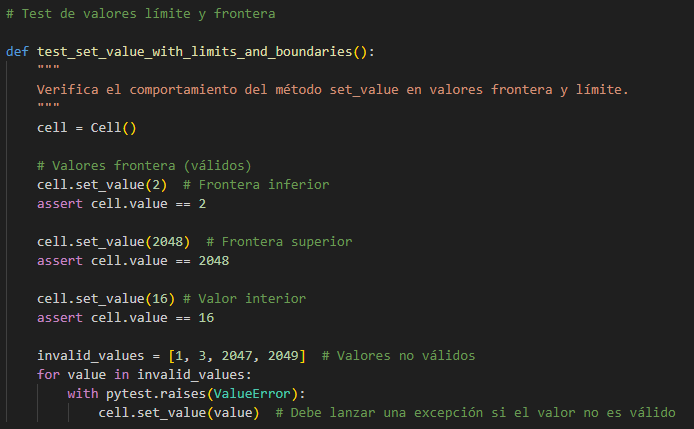
**Test**:

* Arxiu: tests/test\_cell.py
* Mètode de test associat a la funcionalitat: Caixa negra
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

Statement coverage:



Particions equivalents:  


Valors límit i frontera:  
  


**Funcionalitat**: reset

**Localització**:

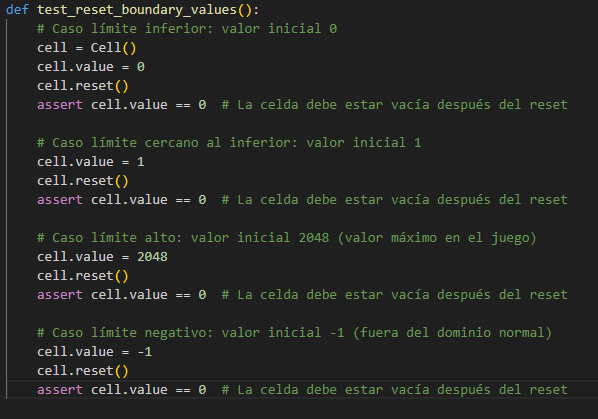
* Arxiu: src/model/board.py
* Classe: Cell
* Mètode desenvolupat: reset

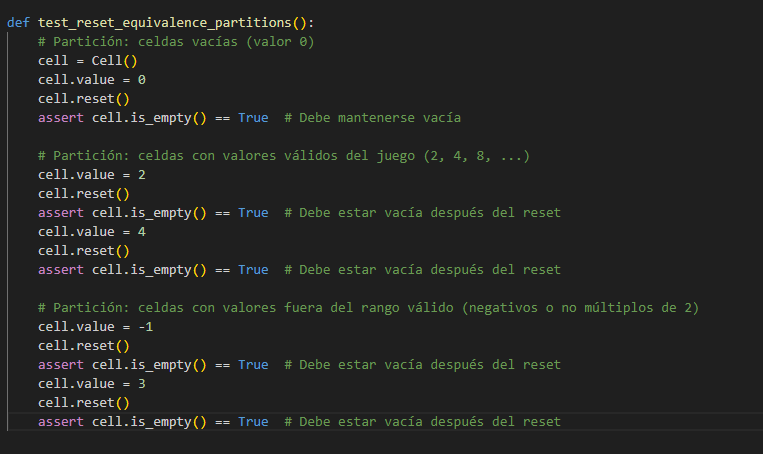
**Test**:

* Arxiu: tests/test\_cell.py
* Mètode de test associat a la funcionalitat: Caixa negra
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

Statement coverage:



Valors limit i frontera:  
  
  
  
Particions equivalents:



**Funcionalitat**: init

**Localització**:

* Arxiu: src/model/game.py
* Classe: Game
* Mètode desenvolupat: init

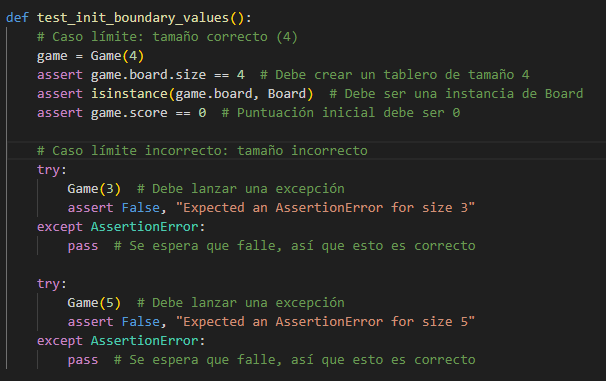
**Test**:

* Arxiu: tests/test\_game.py
* Mètode de test associat a la funcionalitat: Caixa negra
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

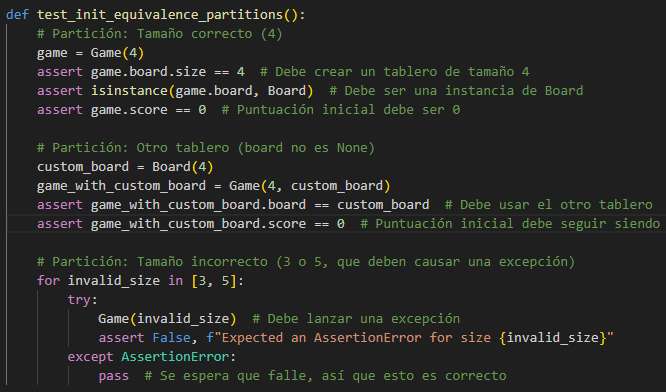
Statement coverage:



Valor limit i frontera:



Particions equivalents:



**Funcionalitat**: play\_turn

**Localització**:

* Arxiu: src/model/game.py
* Classe: Game
* Mètode desenvolupat: play\_turn

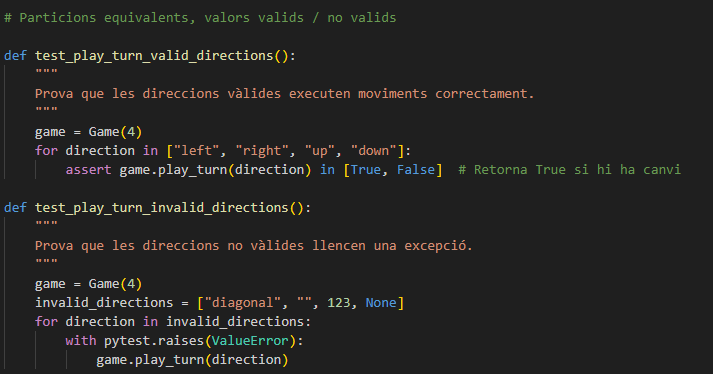
**Test**:

* Arxiu: tests/test\_game.py
* Mètode de test associat a la funcionalitat: Caixa negra
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

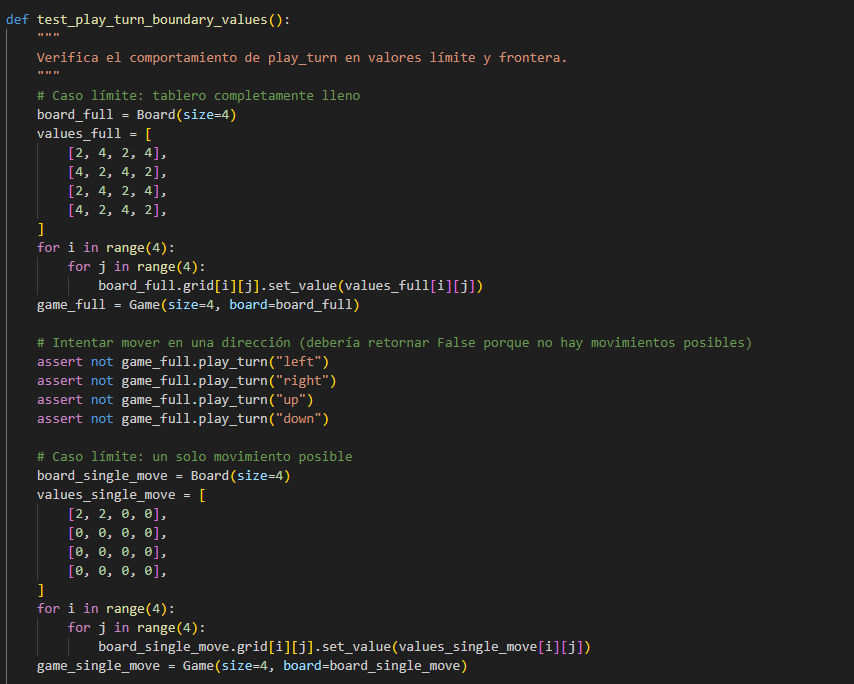
Statement coverage:

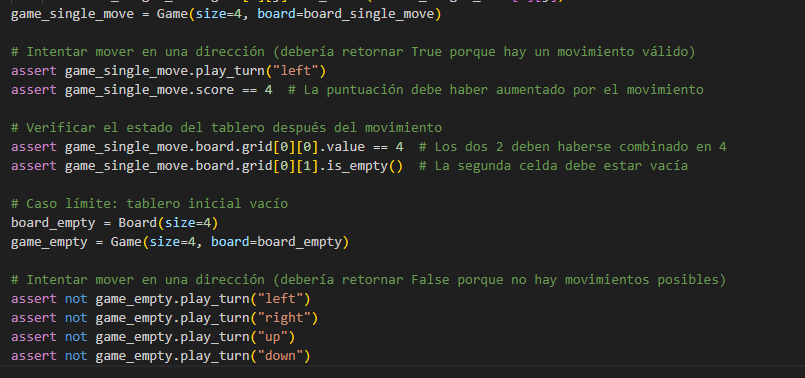


Particions equivalents

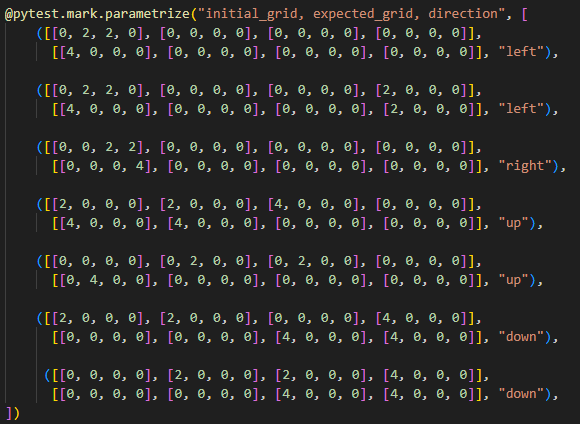


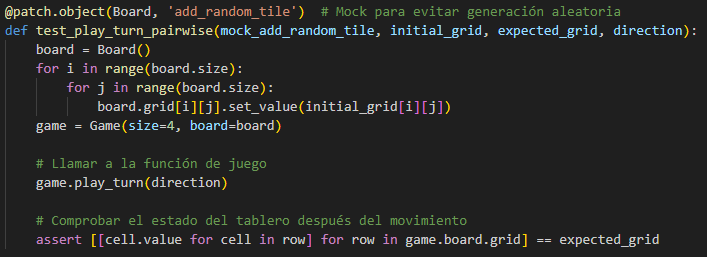
Valors limit i frontera:



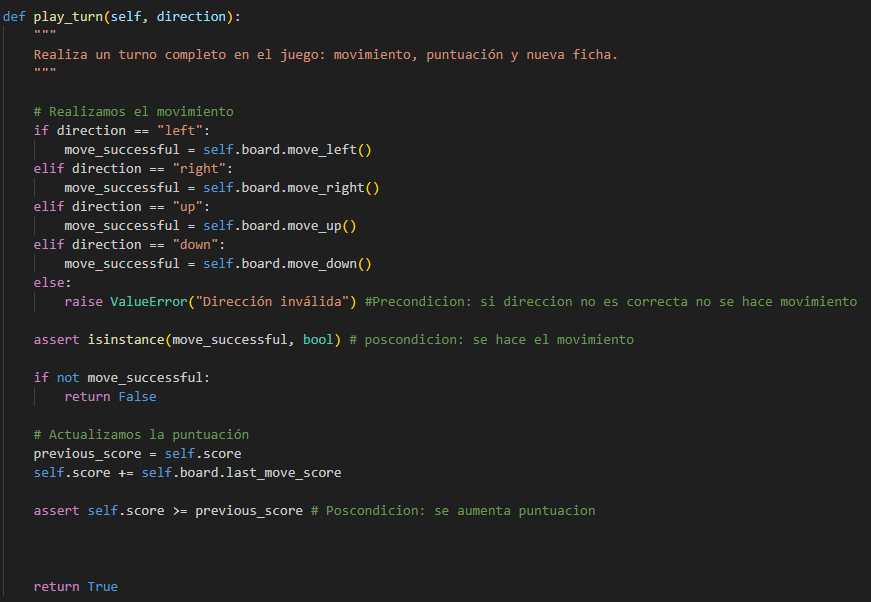


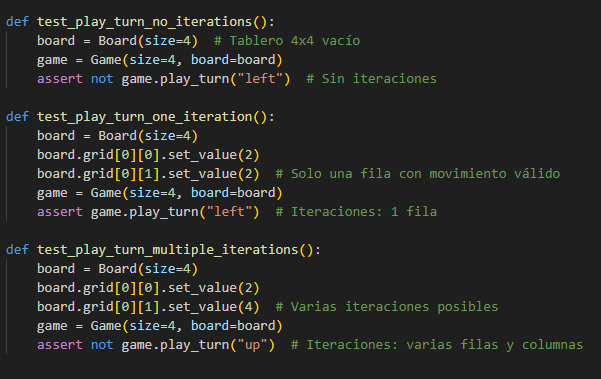
Pairwise testing

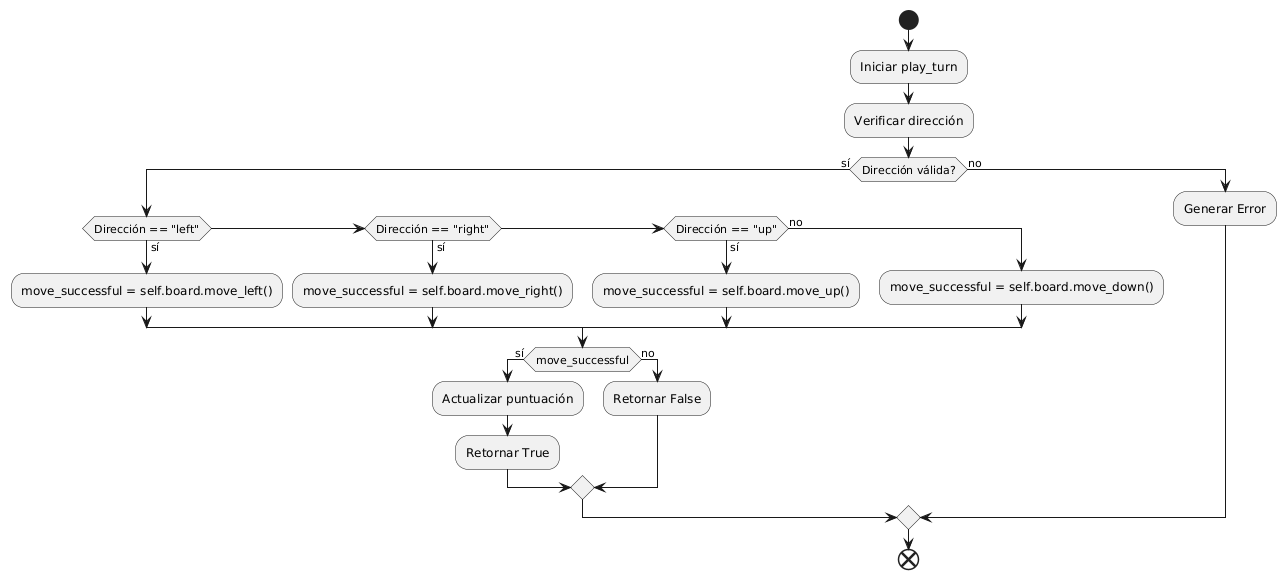


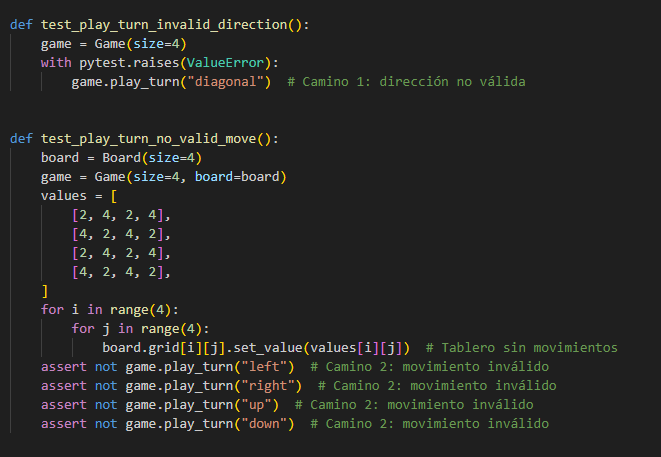


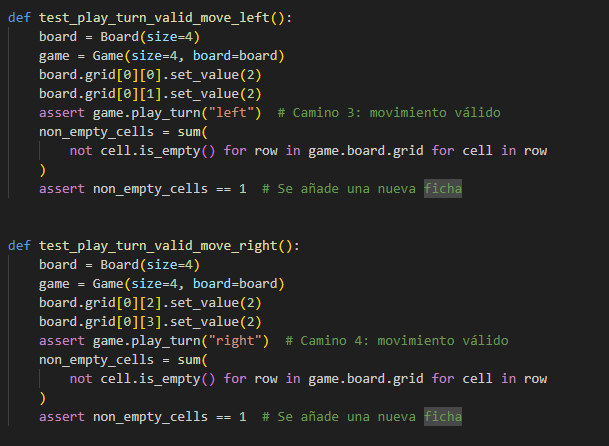
Loop testing:

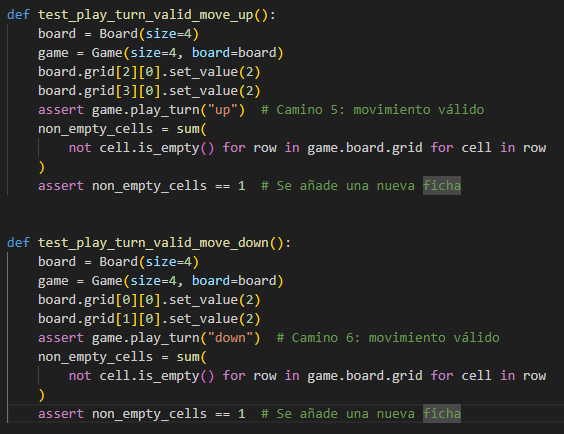




Path coverage:  
  








**Funcionalitat**: is\_game\_over

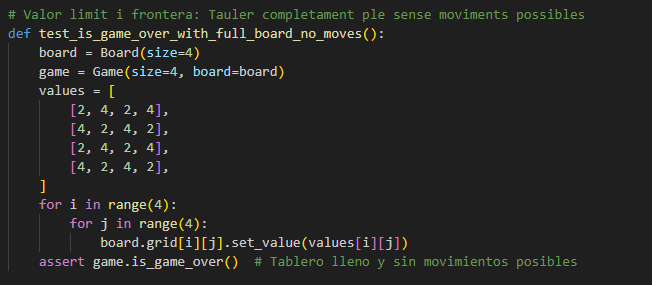
**Localització**:

* Arxiu: src/model/game.py
* Classe: Game
* Mètode desenvolupat: is\_game\_over

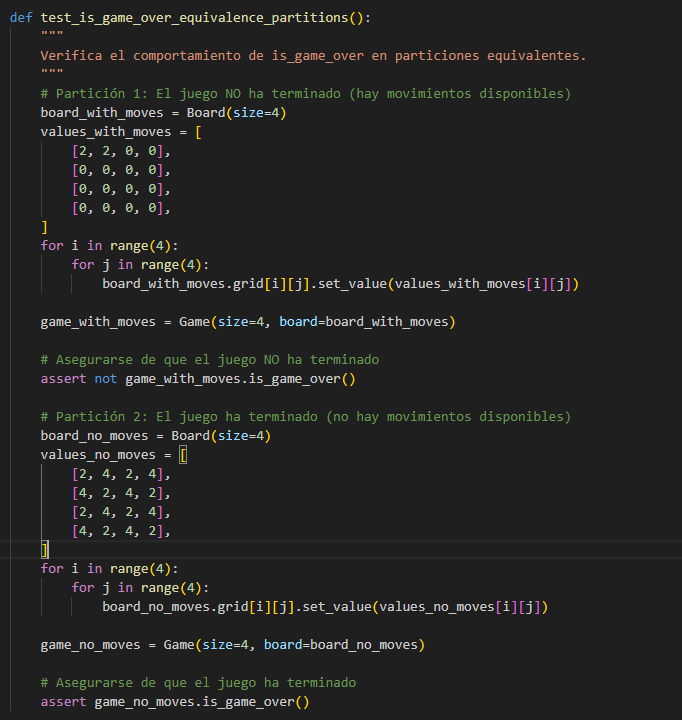
**Test**:

* Arxiu: tests/test\_game.py
* Mètode de test associat a la funcionalitat: Caixa negra, caixa blanca
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera

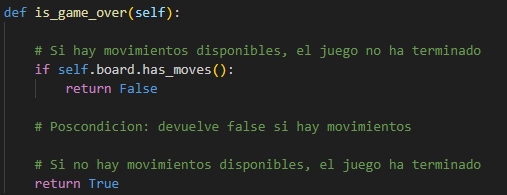
Statement coverage  
  

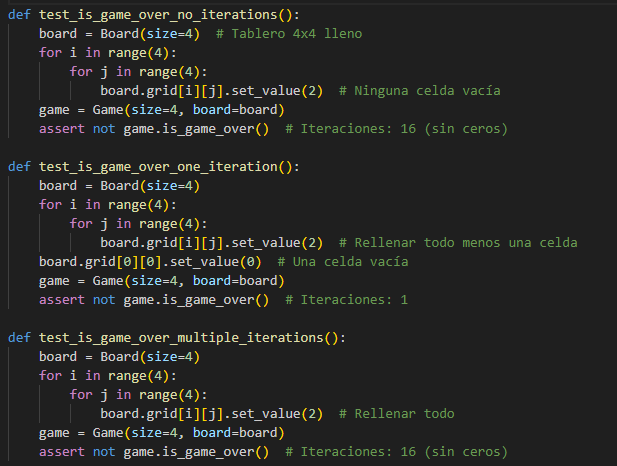

Valors límit i frontera  
  


Partició equivalent:

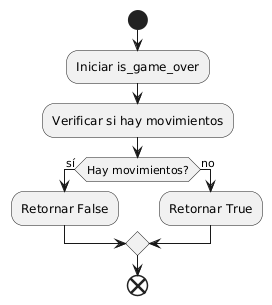


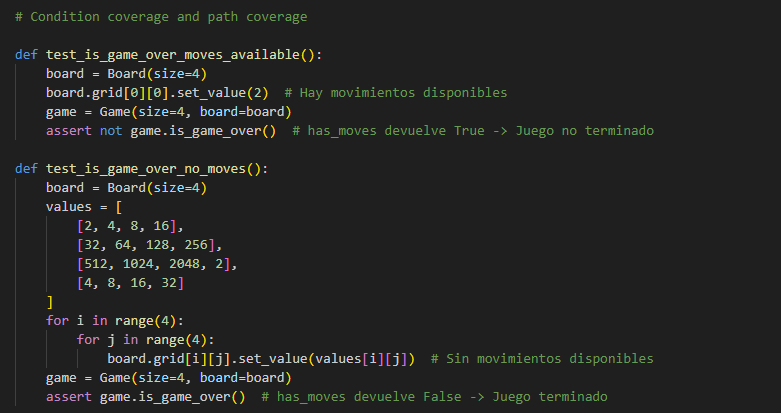
Loop testing:





Condition coverage and path coverage:





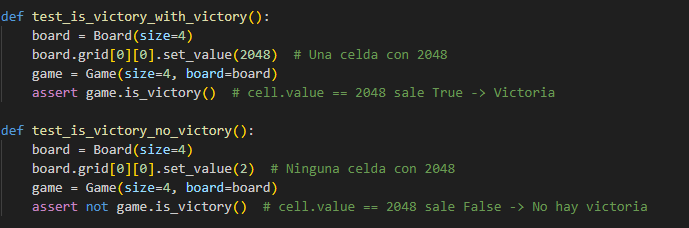
**Funcionalitat**: is\_victory

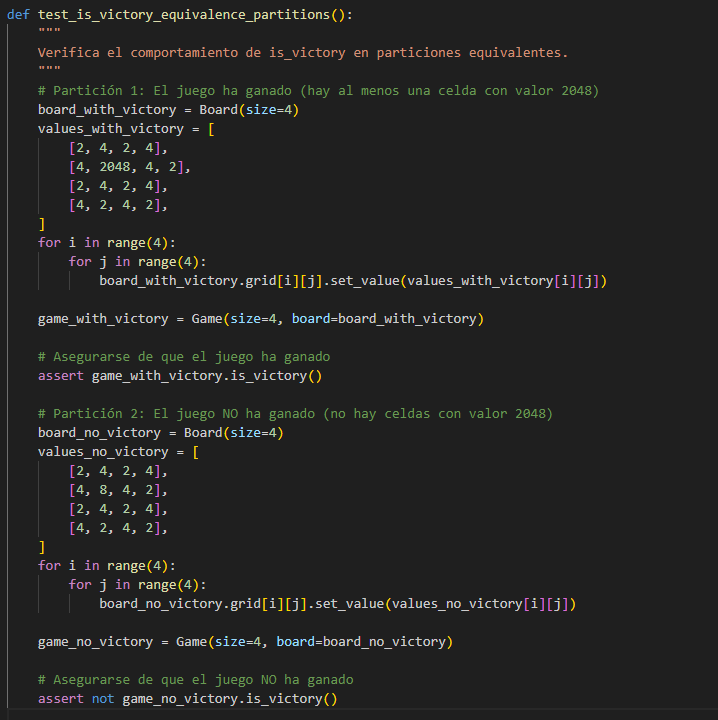
**Localització**:

* Arxiu: src/model/game.py
* Classe: Game
* Mètode desenvolupat: is\_victory

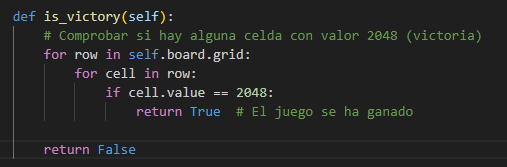
**Test**:

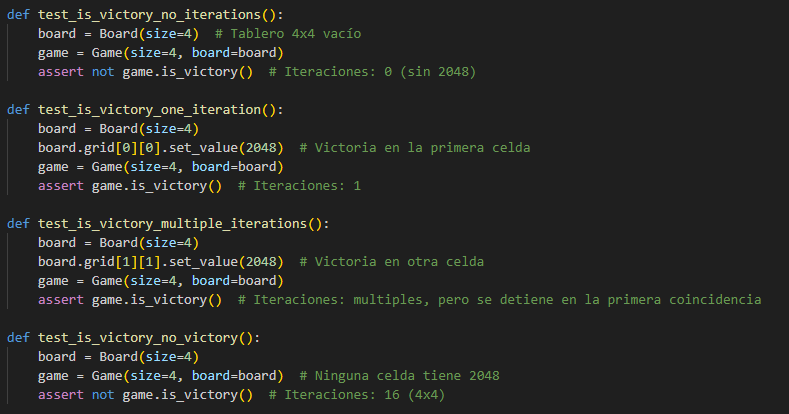
* Arxiu: tests/test\_game.py
* Mètode de test associat a la funcionalitat: Caixa negra, caixa blanca
* Tècniques utilitzades: Statement coverage, Particions equivalents, Valors limit i frontera, Loop testing

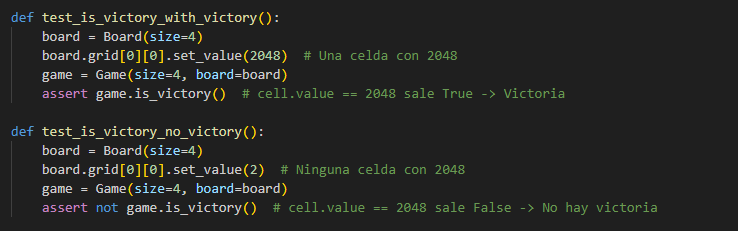
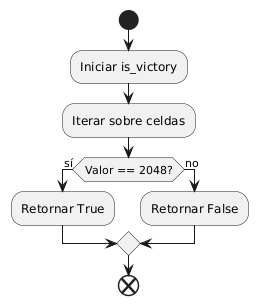
Valors límit i frontera i particions equivalents  
  




Loop testing:





Condition coverage and path coverage:  


**Funcionalitat**: Mock View

Aquesta funcionalitat comprova que el controlador interactua correctament amb la vista simulada.

**Localització**:

* Arxiu: src/mocks/mock\_view.py
* Classe: MockView

**Test**:

* Arxiu: tests/test\_mock.py

**Funcionalitat**: Mock Controller

Aquesta funcionalitat comprova que el controlador simulat interactua correctament amb la vista, el cell i el board.

**Localització**:

* Arxiu: src/mocks/mock\_controller.py
* Classe: MockController

**Test**:

* Arxiu: tests/test\_mock.py

**Funcionalitat**: Mock Cell

Aquesta funcionalitat comprova el correcte funcionament de la classe Cell mitjançant Cell i CellGenerator simulades.

**Localització**:

* Arxiu: src/mocks/mock\_cell.py
* Classe: MockCell, MockCellGenerator

**Test**:

* Arxiu: tests/test\_mock.py

**Funcionalitat**: Mock Board

Aquesta funcionalitat comprova que les cel·les, el controlador i la view interactuen correctament amb el board simulat.

**Localització**:

* Arxiu: src/mocks/mock\_board.py
* Classe: MockBoard

**Test**:

* Arxiu: tests/test\_mock.py

