KAKURO

DESCRIPCIÓN JUEGOS DE PRUEBA

Judith Almoño Gómez
Álvaro Armada Ruíz
Pau Cuesta Arcos
Pol Vallespí Soro

ÍNDICE

CELL

Creadora

- **Objetivos:** La creación de una celda
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCell
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** Se ha creado la celda
- **Resultado:** Correcto

isWhite

- **Objetivos:** Comprobar si la celda es blanca o no
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCell
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** La celda no es blanca
- **Resultado:** Correcto

setValue

- **Objetivos:** Colocar valor en celda blanca
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCell
- **Stubs: -----**
- **Entrada:** valor
- **Salida:** La celda no es blanca
- **Resultado:** Correcto

BLACKCELL

Creadora sin valor

- **Objetivos:** Crear una celda negra sin valor
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverBlackCell
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** se crea la celda negra sin valores
- **Resultado:** Correcto

Creadora con valor

- **Objetivos:** Creación de una celda negra con valor
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverBlackCell
- **Stubs: -----**
- **Entrada:** suma fila, sumacolumna
- **Salida:** se crea la celda negra con valores
- **Resultado:** Correcto

getRow

- **Objetivos:** Retornar el valor de la fila
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverBlackCell
- **Stubs: -----**
- **Entrada:** valor de la fila
- **Salida:** valor de la fila
- **Resultado:** Correcto

getColumn

- **Objetivos:** Retornar el valor de la columna
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverBlackCell
- **Stubs: -----**
- **Entrada:**
- **Salida:** valor de la columna
- **Resultado:** Correcto

setRow

- **Objetivos:** Colocar la suma de la fila
- **Otros elementos integrados: -----**

- **Drivers construidos:** driverBlackCell
- **Stubs: -----**
- **Entrada:** suma de la fila
- **Salida:** se coloca la suma de la fila en la celda negra
- **Resultado:** Correcto

setColumn

- **Objetivos:** Colocar la suma de la columna
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverBlackCell
- **Stubs: -----**
- **Entrada:** suma de la columna
- **Salida:** se coloca la suma de la columna en la celda negra
- **Resultado:** Correcto

WHITECELL

Creadora sin valor

- **Objetivos:** Crear una celda blanca sin valor
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverWhiteCell
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** Se ha creado una celda blanca sin valor
- **Resultado:** Correcto

Creadora con valor

- **Objetivos:** Crear una celda blanca con valor
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverWhiteCell
- **Stubs: -----**
- **Entrada:** valor
- **Salida:** se crea una celda blanca con valor
- **Resultado:** Correcto

getValue

- **Objetivos:** Obtener valor de la celda
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverWhiteCell
- **Stubs: -----**
- **Entrada:** valor
- **Salida:** Value = *value*
- **Resultado:** Correcto

setValue

- **Objetivos:** Colocar valor de la celda
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverWhiteCell
- **Stubs: -----**
- **Entrada:** valor
- **Salida:** Value = *value,* y se coloca el valor en la celda
- **Resultado:** Correcto

getCorrectValue

- **Objetivos:** Obtener valor correcto de la celda
- **Otros elementos integrados:-----**

- **Drivers construidos:** driverWhiteCell
- **Stubs:-----**
- **Entrada: -----**
- **Salida:** Correct value = 0
- **Resultado:** Correcto

setCorrectValue

- **Objetivos:** Colocar valor correcto de la celda
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverWhiteCell
- **Stubs:-----**
- **Entrada:** valor correcto
- **Salida:** Correct value = *valor correcto* y se coloca el valor correcto de la celda
- **Resultado:** Correcto

isWhite

- **Objetivos:** Comprobar que la celda es blanca
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverWhiteCell
- **Stubs:-----**
- **Entrada:-----**
- **Salida:** La celda es blanca
- **Resultado:** Correcto

KAKURO

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

Creadora1

- **Objetivos:** Crear un kakuro
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** se crea una instáncia de Kakuro
- **Resultado:** Correcto

Creadora2

Prueba 1

- **Descripción:** Kakuro correcto
- **Objetivos:** Crear un kakuro a partir de String
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:** kakuro en formato string
- **Salida:** se ha creado el kakuro
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Carácter inválido
- **Objetivos:** Crear un kakuro a partir de String
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:** kakuro en formato string
- **Salida:** salta la excepción number format
- **Resultado:** Correcto

**Para probar si funciona cuando se le pasa un tamaño que no coincide con el kakuro, lo hemos hecho desde JUnit porque después de intentar leer un kakuro con menos filas de las que se especifican no se puede identificar la siguiente opción.*

toString

- **Objetivos:** Convertir el kakuro a string
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell

- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:-----**
- **Salida:** String con el kakuro
- **Resultado:** Correcto

correctToString

- **Objetivos:** Convertir el kakuro correcto a string
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** string con el kakuro correcto
- **Resultado:** Correcto

getID

- **Objetivos:** Obtener el valor del id
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** ID = identificador
- **Resultado:** Correcto

getDifficulty

- **Objetivos:** Obtener la dificultad del kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** dificultad
- **Salida:** dificulty = dificultad del kakuro
- **Resultado:** Correcto

setDifficulty

- **Objetivos:** Colocar dificultad al kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** dificultad
- **Salida:** dificulty = dificultad del kakuro y se coloca la dificultad
- **Resultado:** Correcto

getRowSize

- **Objetivos:** Obtener el valor de la fila
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** RowSize = tamaño de fila
- **Resultado:** Correcto

getColumnSize

- **Objetivos:** Obtener el valor de la columna
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** ColumnSize = tamaño de la columna
- **Resultado:** Correcto

getBoard

- **Objetivos:** Obtener el tablero
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** Se ha obtenido el tablero y se imprime: tablero
- **Resultado:** Correcto

getCell

- **Objetivos:** Obtener una celda
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y
- **Salida:** Se ha obtenido la celda buscada
- **Resultado:** Correcto

setValue

Prueba 1

- **Descripción:** Celda negra
- **Objetivos:** Colocar valor en celda blanca
- **Otros elementos integrados:** Cell, WhiteCell

- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor
- **Salida:** La celda en la posición x e y es negra y por lo tanto no se ha actualizado
- **Resultado:** Correcto

Prueba 2
- **Descripción:** Celda blanca
- **Objetivos:** Colocar valor en celda blanca
- **Otros elementos integrados:** Cell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:**posición x, posición y, valor
- **Salida:** La celda en la posición x e y es blanca y se ha actualizado correctamente
- **Resultado:** Correcto

checkColumn

Prueba 1
- **Descripción:** Se introduce un valor incorrecto indicando que es incorrecto
- **Objetivos:** Comprobar integridad de la columna con los correctValue
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** fila, columna, valor nuevo, última casilla de la run vertical, suma
- **Salida:** Solución correcta
- **Resultado:** Correcto

Prueba 2
- **Descripción:** Se introduce un valor correcto indicando que es correcto
- **Objetivos:** Comprobar integridad de la columna con los correctValue
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** fila, columna, valor nuevo, última casilla de la run vertical, suma
- **Salida:** Solución correcta
- **Resultado:** Correcto

Prueba 3
- **Descripción:** Se introduce un valor correcto indicando que es correcto
- **Objetivos:** Comprobar integridad de la columna con los correctValue
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell

- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** fila, columna, valor nuevo, última casilla de la run vertical, suma
- **Salida:** Solución correcta
- **Resultado:** Correcto

Prueba 4
- **Descripción:** Se introduce un valor incorrecto indicando que es incorrecto
- **Objetivos:** Comprobar integridad de la columna con los correctValue
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** fila, columna, valor nuevo, última casilla de la run vertical, suma
- **Salida:** Solución correcta
- **Resultado:** Correcto

checkRowValidity

Prueba 1
- **Descripción:** Se introduce un valor incorrecto con las otras celdas del bloque ya llenas
- **Objetivos:** Comprobar que la suma de los valores no supera el número clave
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido no cumple las condiciones de la fila
- **Resultado:** Correcto

Prueba 2
- **Descripción:** Se introduce un valor correcto con las otras celdas del bloque ya llenas
- **Objetivos:** Comprobar que la suma de los valores da la esperada
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido cumple las condiciones de la fila
- **Resultado:** Correcto

Prueba 3
- **Descripción:** Se introduce un valor incorrecto con las otras celdas del bloque ya llenas

- **Objetivos:** Comprobar que falla cuando hay valores repetidos
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido no cumple las condiciones de la fila
- **Resultado:** Correcto

Prueba 4

- **Descripción:** Se introduce un valor incorrecto con alguna de las otras celdas del bloque vacías
- **Objetivos:** Comprobar que, si no supera el número clave, no da error
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido cumple las condiciones de la fila
- **Resultado:** Correcto

checkColumnValidity

Prueba 1

- **Descripción:** Se introduce un valor incorrecto con las otras celdas del bloque ya llenas
- **Objetivos:** Comprobar que la suma de los valores no supera el número clave
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido no cumple las condiciones de la columna
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Se introduce un valor correcto con las otras celdas del bloque ya llenas
- **Objetivos:** Comprobar que la suma de los valores da la esperada
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido cumple las condiciones de la columna
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Se introduce un valor incorrecto con las otras celdas del bloque ya llenas
- **Objetivos:** Comprobar que falla cuando hay valores repetidos
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido no cumple las condiciones de la columna
- **Resultado:** Correcto

Prueba 4

- **Descripción:** Se introduce un valor incorrecto con alguna de las otras celdas del bloque vacías
- **Objetivos:** Comprobar que, si no supera el número clave, no da error
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs: -----**
- **Entrada:** posición x, posición y, valor, kakuro
- **Salida:** El valor introducido cumple las condiciones de la columna
- **Resultado:** Correcto

isFinished

Prueba 1

- **Descripción:** Tablero por empezar
- **Objetivos:** Comprobar que no se ha acabado
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:** String de kakuro a comprobar y string del mismo kakuro terminado
- **Salida:** El kakuro aún no se ha terminado
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Tablero a medio hacer
- **Objetivos:** Comprobar que se ha acabado
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:** String de kakuro a comprobar y string del mismo kakuro terminado
- **Salida:** El kakuro aún no se ha terminado
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Tablero acabado
- **Objetivos:** Comprobar que no se ha acabado
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:** String de kakuro a comprobar y string del mismo kakuro terminado
- **Salida:** El kakuro se ha terminado
- **Resultado:** Correcto

Prueba 4

- **Descripción:** Tablero rellenado pero valores incorrectos
- **Objetivos:** Comprobar que no se ha acabado
- **Otros elementos integrados:** Cell, BlackCell, WhiteCell
- **Drivers construidos:** driverKakuro
- **Stubs:-----**
- **Entrada:** String de kakuro a comprobar y string del mismo kakuro terminado
- **Salida:** El kakuro aún se ha terminado
- **Resultado:** Correcto

CTRLDATA

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

getInstance

- **Objetivos:** Obtener la instancia del CtrlData
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlData
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** Se ha obtenido la instancia deseada
- **Resultado:** Correcto

searchKakuro

Prueba 1

- **Descripción:** Dificultad 1
- **Objetivos:** Obtener Kakuro de dificultad 1
- **Otros elementos integrados:------**
- **Drivers construidos:** driverCtrlData
- **Stubs:------**
- **Entrada:** dificultad, tamaño fila, tamaño columna
- **Salida:** Se escribe un kakuro de la dificultad y tamaños especificados
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Dificultad 2
- **Objetivos:** Obtener Kakuro de dificultad 2
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlData
- **Stubs: ------**
- **Entrada:** dificultad, tamaño fila, tamaño columna
- **Salida:** Se escribe un kakuro de la dificultad y tamaños especificados
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Dificultad 3
- **Objetivos:** Obtener Kakuro de dificultad 3
- **Otros elementos integrados:------**
- **Drivers construidos:** driverCtrlData
- **Stubs:------**
- **Entrada:** dificultad, tamaño fila, tamaño columna
- **Salida:** Se escribe un kakuro de la dificultad y tamaños especificados
- **Resultado:** Correcto

Prueba 4

- **Descripción:** No existe Kakuro
- **Objetivos:** Comprobar que no encuentra ningun kakuro
- **Otros elementos integrados:------**
- **Drivers construidos:** driverCtrlData
- **Stubs:------**
- **Entrada:** dificultad, tamaño fila, tamaño columna
- **Salida:** No se ha encontrado un Kakuro que cumpla las condiciones
- **Resultado:** Correcto

getKakuro

Prueba 1

- **Descripción:** Kakuro encontrado dificultad 1
- **Objetivos:** Comprobar que encuentra el kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlData
- **Stubs: -----**
- **Entrada:** ruta y kakuro que debería encontrar
- **Salida:** Se imprimen los dos kakuros
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Kakuro encontrado dificultad 2
- **Objetivos:** Comprobar que encuentra el kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlData
- **Stubs: -----**
- **Entrada:** ruta y kakuro que debería encontrar
- **Salida:** Se imprimen los dos kakuros
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Kakuro encontrado dificultad 3
- **Objetivos:** Comprobar que encuentra el kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlData
- **Stubs: -----**
- **Entrada:** ruta y kakuro que debería encontrar
- **Salida:** Se imprimen los dos kakuros
- **Resultado:** Correcto

Prueba 4

- **Descripción:** No existe
- **Objetivos:** Comprobar que no encuentra el kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlData
- **Stubs: -----**
- **Entrada:** ruta y kakuro que debería encontrar
- **Salida:** No se ha podido obtener un kakuro con esta ruta
- **Resultado:** Correcto

saveKakuro

- **Objetivos:** Comprobar que guarda el kakuro
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlData
- **Stubs: -----**
- **Entrada:** String del kakuro, dificultad, tamaño filas y tamaño columnas
- **Salida:** Se ha generado un fichero con el kakuro guardado y se puede ver el kakuro guardado en la ruta “data/diffX/tamañofilas_tamañocolumnas/” donde X es la dificultad. Será el kakuro con el identificador más grande
- **Resultado:** Correcto

getNumberOfFiles

Prueba 1

- **Descripción:** Obtiene bien el número kakuros de un tamaño de dificultad1
- **Objetivos:** Comprobar que cuenta bien el número de kakuros
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlData
- **Stubs:-----**
- **Entrada:** ruta
- **Salida:** Hay X kakuro(s) con esta dificultad y tamaño
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Obtiene bien el número kakuros de un tamaño de dificultad2
- **Objetivos:** Comprobar que cuenta bien el número de kakuros
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlData
- **Stubs:-----**
- **Entrada:** ruta
- **Salida:** Hay X kakuro(s) con esta dificultad y tamaño
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Obtiene bien el número kakuros de un tamaño de dificultad2
- **Objetivos:** Comprobar que cuenta bien el número de kakuros
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlData
- **Stubs:-----**
- **Entrada:** ruta
- **Salida:** Hay X kakuro(s) con esta dificultad y tamaño
- **Resultado:** Correcto

Prueba 4

- **Descripción:** No hay kakuros
- **Objetivos:** Comprobar que detecta que no hay
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlData
- **Stubs:-----**
- **Entrada:** ruta
- **Salida:** No hay ningún kakuro con estas condiciones
- **Resultado:** Correcto

CTRLDOMAIN

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

StartNewGame

Prueba 1:

- **Descripción:** Empezar partida de Kakuro guardado en ficheros
- **Objetivos:** Comprobar que si existe un kakuro con las características indicadas en la carpeta data, inicia la partida con ese y no genera uno nuevo.
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlDomain
- **Stubs:-----**
- **Entrada:** dificultad de un kakuro, número de filas y número de columnas de un kakuro que sabemos que tenemos guardado
- **Salida:** -----
- **Resultado:** Correcto, se ha empezado la partida con el kakuro que queríamos

Prueba 2:

- **Descripción:** Empezar partida de Kakuro generado en ese momento
- **Objetivos:** Comprobar que si no existe un kakuro con las características indicadas en la carpeta data, genera un kakuro con esas características y empieza la partida.
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlDomain
- **Stubs:-----**
- **Entrada:** dificultad de un kakuro, número de filas y número de columnas de un kakuro que no tenemos guardado en data
- **Salida:** -----
- **Resultado:** Correcto, ha generado un kakuro de las características indicadas y se ha iniciado la partida

checkCoord

prueba 1:

- **Descripción:** Comprobar que las coordenadas son correctas
- **Objetivos:** Comprobar si detecta correctamente una coordenada de fila incorrecta
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlDomain
- **Stubs:-----**
- **Entrada:** una ruta a un kakuro de data, una fila incorrecta y una columna correcta
- **Salida:** -----
- **Resultado:** Correcto, ha detectado que la fila es incorrecta

prueba 2:

- **Descripción:** Comprobar que las coordenadas son correctas
- **Objetivos:** Comprobar si detecta correctamente una coordenada de columna incorrecta
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlDomain
- **Stubs:-----**
- **Entrada:** una ruta a un kakuro de data, una fila correcta y una columna incorrecta
- **Salida:** -----
- **Resultado:** Correcto, ha detectado que la columna es incorrecta

prueba 3:

- **Descripción:** Comprobar que las coordenadas son correctas
- **Objetivos:** Comprobar si detecta correctamente que las dos coordenadas son correctas
- **Otros elementos integrados:-----**
- **Drivers construidos:** driverCtrlDomain
- **Stubs:-----**
- **Entrada:** una ruta a un kakuro de data, una fila correcta y una columna correcta
- **Salida:** -----
- **Resultado:** Correcto

CTRLPLAY

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

startGame

- **Objetivos:** Empezar una partida
- **Otros elementos integrados:** Kakuro
- **Drivers construidos:** driverCtrlPlay
- **Stubs: -----**
- **Entrada:** kakuro
- **Salida:** Se ha empezado una partida
- **Resultado:** Correcto

helpMyValue

Prueba 1

- **Descripción:** Aún no hay valor
- **Objetivos:** Comprobar que no hay valor y por lo tanto, no hay ayuda
- **Otros elementos integrados:** Kakuro, Cell, WhiteCell
- **Drivers construidos:** driverCtrlPlay
- **Stubs:-----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** La celda no tiene un valor asignado
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Celda negra sin valor
- **Objetivos:** Comprobar que la celda es negra y por lo tanto, no hay ayuda
- **Otros elementos integrados:** Kakuro, Cell, WhiteCell
- **Drivers construidos:** driverCtrlPlay
- **Stubs:-----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** La celda no es blanca
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Celda negra sin valor
- **Objetivos:** Comprobar que la celda es negra con valor y por lo tanto, no hay ayuda
- **Otros elementos integrados:** Kakuro, Cell, WhiteCell
- **Drivers construidos:** driverCtrlPlay
- **Stubs: -----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** La celda no es blanca

- **Resultado:** Correcto

Prueba 4
- **Descripción:** Celda blanca con valor correcto
- **Objetivos:** Comprobar que la celda es blanca y que el valor es correcto
- **Otros elementos integrados:** Kakuro, Cell, WhiteCell
- **Drivers construidos:** driverCtrlPlay
- **Stubs:-----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** El valor es correcto
- **Resultado:** Correcto

Prueba 5
- **Descripción:** Celda blanca con valor incorrecto
- **Objetivos:** Comprobar que la celda es blanca y que el valor no es correcto
- **Otros elementos integrados:** Kakuro, Cell, WhiteCell
- **Drivers construidos:** driverCtrlPlay
- **Stubs:-----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** El valor no es correcto
- **Resultado:** Correcto

helpCorrectNumber

Prueba 1
- **Descripción:** La celda es blanca
- **Objetivos:** Colocar el valor correcto en la celda blanca
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlPlay
- **Stubs: -----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** El valor de la celda se ha cambiado correctamente
- **Resultado:** Correcto

Prueba 2
- **Descripción:** La celda es negra sin valor
- **Objetivos:** Comprobar que la celda es negra y que no puede hacerse la ayuda
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlPlay
- **Stubs: -----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** No es una celda blanca
- **Resultado:** Correcto

Prueba 3

- **Descripción:** La celda es negra con valor
- **Objetivos:** Comprobar que la celda es negra con valor y que no puede hacerse la ayuda
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlPlay
- **Stubs: -----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** No es una celda blanca
- **Resultado:** Correcto

Prueba 4

- **Descripción:** La celda es blanca con un valor ya asignado
- **Objetivos:** Comprobar que la celda es blanca y sobreescribe el valor por el correcto
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlPlay
- **Stubs: -----**
- **Entrada:** kakuro, kakuroSolución, posx, posy
- **Salida:** El valor de la celda se ha cambiado correctamente
- **Resultado:** Correcto

CTRLVALIDATE

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

Creadora

- **Descripción:** Crear la clase CtrlValidate
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -----**
- **Entrada: -----**
- **Salida:** Se ha creado.
- **Resultado:** Correcto

validate

Prueba 1

- **Descripción:** Solución múltiple 3x3
- **Objetivos:** Comprobar que si tiene solución múltiple es inválido
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -------**
- **Entrada:** Kakuro
- **Salida:** El kakuro no es valido
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Solución múltiple 6x6
- **Objetivos:** Comprobar que si tiene solución múltiple es inválido
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -------**
- **Entrada:** Kakuro
- **Salida:** El kakuro no es valido
- **Resultado:** Correcto
-

Prueba 3

- **Descripción:** Sin solución
- **Objetivos:** Comprobar que si no tiene solución es inválido
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -------**
- **Entrada:** Kakuro
- **Salida:** El kakuro no es valido
- **Resultado:** Correcto

Prueba 4

- **Descripción:** Solución única
- **Objetivos:** Comprobar que si tiene solución única guarda el valor 1.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -------**
- **Entrada:** Kakuro
- **Salida:** El kakuro es válido
- **Resultado:** Correcto

setDifficulty

Prueba 1

- **Descripción:** Fácil
- **Objetivos:** Comprueba que un kakuro con un 57% de blancas, 29 casillas con valor trivial, un rating de 1, una media de 3 de longitud de “runs” y la “run” más larga de 6 tenga dificultad fácil.
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -----**
- **Entrada:** Kakuro
- **Salida:** El kakuro es de dificultad fácil
- **Resultado:** Correcto

Prueba 2

- **Descripción:** Hard
- **Objetivos:** Comprueba que un kakuro con un 64% de blancas, 17 casillas con valor trivial, un rating de 2.12, una media de 3.08 de longitud de “runs” y la “run” más larga de 7 tenga dificultad difícil.
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -----**
- **Entrada:** Kakuro
- **Salida:** El kakuro es de dificultad difícil
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Medium
- **Objetivos:** Comprueba que un kakuro con un 60% de blancas, 26 casillas con valor trivial, un rating de 1.43, una media de 2.7 de longitud de “runs” y la “run” mas larga de 7 tenga dificultad dificil.
- **Otros elementos integrados: -----**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: -----**

- **Entrada:** Kakuro
- **Salida:** El kakuro es de dificultad mediana
- **Resultado:** Correcto

isUnique

Prueba 1

- **Descripción:** No hay 1 en el array
- **Objetivos:** Comprobar que si no hay 1 en el array retorna falso
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** int a []
- **Salida:** No hay 1's entre los números introducidos o bien hay mas de un 1
- **Resultado:** Correcto
-

Prueba 2

- **Descripción:** Hay un único 1 en el array
- **Objetivos:** Comprobar que si hay un único 1 en el array retorna verdadero
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** int a []
- **Salida:** Hay exactamente un 1 entre los números introducidos
- **Resultado:** Correcto

Prueba 3

- **Descripción:** Hay más de un 1 en el array
- **Objetivos:** Comprobar que si hay más de un 1 en el array retorna falso
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** int a []
- **Salida:** No hay 1's entre los números introducidos o bien hay mas de un 1
- **Resultado:** Correcto

howManyNumbers

Prueba 1

- **Descripción:** No hay 1
- **Objetivos:** Comprobar que retorna 0 si no hay 1.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** int a []

- **Salida:** El array tiene 0 unos.
- **Resultado:** Correcto

Prueba 2
- **Descripción:** Hay tres 1
- **Objetivos:** Comprobar que retorna el número de 1 si hay más de un 1.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** int a []
- **Salida:** El array tiene 0 unos.
- **Resultado:** Correcto

validatePosSums

Prueba 1
- **Descripción:** Fila, valor único y actualización.
- **Objetivos:** Comprobar que si tratamos por fila encuentra valor único y actualiza las demás casillas
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** tempBoard, posCombs, length, row, i, j
- **Salida:** Se actualiza el tablero y encuentra un valor único.
- **Resultado:** Correcto

Prueba 2
- **Descripción:** Columna y actualización.
- **Objetivos:** Comprobar que si tratamos por columna actualiza las casillas
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** tempBoard, posCombs, length, row, i, j
- **Salida:** Se actualiza el tablero.
- **Resultado:** Correcto

computePosSums

Prueba 1
- **Descripción:** 3 con 2 casillas.
- **Objetivos:** Comprobar que produce las combinaciones correctas
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** x, n, no

- **Salida:** 9 números donde 1 indica que aparece y 0 indica que no aparece.
- **Resultado:** Correcto

Prueba 2

- **Descripción:** 28 con 9 casillas.
- **Objetivos:** Comprobar que no existen combinaciones
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** x, n, no
- **Salida:** 9 ceros.
- **Resultado:** Correcto

Prueba 3

- **Descripción:** 3 con 2 casillas y un 1 fijado.
- **Objetivos:** Comprobar que produce las combinaciones correctas si tenemos un número fijado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** x, n, no
- **Salida:** 9 números donde 1 indica que aparece y 0 indica que no aparece.
- **Resultado:** Correcto
-

Prueba 4

- **Descripción:** 45 con 9 casillas y un 8 fijado.
- **Objetivos:** Comprobar que produce las combinaciones correctas si tenemos un número fijado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** x, n, no
- **Salida:** 9 números donde 1 indica que aparece y 0 indica que no aparece.
- **Resultado:** Correcto

checkForNewUniques

Prueba 1

- **Descripción:** Actualiza fila y encuentra valor único
- **Objetivos:** Comprobar que actualiza la fila y encuentra el valor trivial
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** tempBoard

- **Salida:** Un valor único y el tablero actualizado
- **Resultado:** Correcto

Prueba 2
- **Descripción:** Actualiza la columna y encuentra valor único
- **Objetivos:** Comprobar que actualiza la columna y encuentra el valor trivial
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlValidate
- **Stubs: ------**
- **Entrada:** tempBoard
- **Salida:** Un valor único y el tablero actualizado
- **Resultado:** Correcto

CTRLRESOLVE

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

resolve

Prueba 1:
- **Descripción:** Comprobar que la función resolver retorna lo esperado
- **Objetivos:** Comprobar que retorna true si el kakuro tiene solución y que encuentra la solución.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlResolve
- **Stubs: ------**
- **Entrada:** un kakuro para resolver
- **Salida:** true
- **Resultado:** Correcto. Ha encontrado la solución.

Prueba 2:
- **Descripción:** Comprobar que la función resolver retorna lo esperado
- **Objetivos:** Comprobar que retorna false si el kakuro no tiene solución.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlResolve
- **Stubs: ------**
- **Entrada:** un kakuro para resolver
- **Salida:** false
- **Resultado:** Correcto.

CTRLGENERATE

Suponemos que las clases utilizadas aquí se han comprobado anteriormente.

setKakuro

Prueba 1:

- **Descripción:** Comprobación del set
- **Objetivos:** Comprobar que no hay ningún error en la asignación de kakuros.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un kakuro
- **Salida: -----**
- **Resultado:** Correcto.

countWhiteCellsV

Prueba 1:

- **Descripción:** La siguiente celda es negra
- **Objetivos:** Comprobar que cuenta bien el número de casillas situadas antes que la casilla dada
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un número de fila, un número de columna
- **Salida:** el número de casillas blancas de la misma run antes que esta. En este caso 0.
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** La siguiente celda es blanca
- **Objetivos:** Comprobar que cuenta bien el número de casillas situadas antes que la casilla dada
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un número de fila, un número de columna
- **Salida:** el número de casillas blancas de la misma run antes que esta. En este caso 7.
- **Resultado:** Correcto.
-

countWhiteCellsH

Prueba 1:

- **Descripción:** La siguiente celda es negra
- **Objetivos:** Comprobar que cuenta bien el número de casillas situadas antes que la casilla dada
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un número de fila, un número de columna
- **Salida:** el número de casillas blancas de la misma run antes que esta. En este caso 0.
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** La siguiente celda es blanca
- **Objetivos:** Comprobar que cuenta bien el número de casillas situadas antes que la casilla dada
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un número de fila, un número de columna
- **Salida:** el número de casillas blancas de la misma run antes que esta. En este caso 7.
- **Resultado:** Correcto.

ninceCellsRow

Prueba 1:

- **Descripción:** Tablero sin celdas negras en el interior
- **Objetivos:** Comprobar que cuenta bien el número de casillas blancas en cada run
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un kakuro, una matriz de 3 dimensiones donde guardaremos los valores calculados
- **Salida:** -------
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Tablero con celdas blancas y negras repartidas
- **Objetivos:** Comprobar que cuenta bien el número de casillas blancas en cada run
- **Otros elementos integrados: ------**

- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un kakuro, una matriz de 3 dimensiones donde guardaremos los valores calculados
- **Salida:** -------
- **Resultado:** Correcto.

ninceCellsCol

Prueba 1:

- **Descripción:** Tablero sin celdas negras en el interior
- **Objetivos:** Comprobar que cuenta bien el número de casillas blancas en cada run
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un kakuro, una matriz de 3 dimensiones donde guardaremos los valores calculados
- **Salida:** -------
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Tablero con celdas blancas y negras repartidas
- **Objetivos:** Comprobar que cuenta bien el número de casillas blancas en cada run
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un kakuro, una matriz de 3 dimensiones donde guardaremos los valores calculados
- **Salida:** -------
- **Resultado:** Correcto.

computePosSums

Prueba 1:

- **Descripción:** Calcular una combinación única con un valor fijado
- **Objetivos:** Comprobar que encuentra correctamente los valores con los que se puede sumar este valor
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**

- **Entrada:** el valor a sumar (x) , el número de valores posibles (n) y un valor a fijar (no)
- **Salida:** los números que pertenecen a alguna combinación de n valores que sume x
- **Resultado:** Correcto.

Prueba 2:
- **Descripción:** Calcular una combinación única sin un valor fijado
- **Objetivos:** Comprobar que encuentra correctamente los valores con los que se puede sumar este valor
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** el valor a sumar (x) , el número de valores posibles (n) y un valor a fijar (no)
- **Salida:** los números que pertenecen a alguna combinación de n valores que sume x
- **Resultado:** Correcto.

AllZero

Prueba 1:
- **Descripción:** Devuelve true si todos los elementos del vector son cero
- **Objetivos:** Comprobar que detecta correctamente un vector con todos los elementos a cero.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros de 9 posiciones.
- **Salida:** true si todos son cero
- **Resultado:** Correcto.

Prueba 2:
- **Descripción:** Devuelve false si hay algún elemento del vector a 1
- **Objetivos:** Comprobar que detecta correctamente un vector que no todos sus elementos son cero.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros de 9 posiciones, con algún elemento igual a 1.
- **Salida:** false
- **Resultado:** Correcto.

intersection3

Prueba 1:

- **Descripción:** Devuelve true si todos los elementos del segundo vector son cero
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones con 0 en todas las posiciones
- **Salida:** true
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Devuelve true si los dos vectores tienen algún elemento en común
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones, con 1's y 0's y algún 1 en la misma posición que en el vector a.
- **Salida:** true
- **Resultado:** Correcto.

Prueba 3:

- **Descripción:** Devuelve false si no se da el primer caso y no tienen ningún elemento en común
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones, con 1's y 0's y sin ningún 1 en la misma posición que en el vector a.
- **Salida:** false
- **Resultado:** Correcto.

intersection2

Prueba 1:

- **Descripción:** Devuelve true si los dos vectores tienen algún 1 en la misma posición
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones, con 1's y 0's y algún 1 en la misma posición que en el vector a.
- **Salida:** true
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Devuelve false si los vectores no tienen ningún 1 en alguna posición en común
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones, con 1's y 0's y sin ningún 1 en la misma posición que en el vector a.
- **Salida:** false
- **Resultado:** Correcto.

intersection

Prueba 1:

- **Descripción:** Devuelve la posición del único elemento en común de dos vectores.
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones, con 1's y 0's y solo un 1 en la misma posición que en el vector a.
- **Salida:** la posición del elemento en común, en nuestro caso 0.
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Devuelve -1 si los vectores tienen más de un 1 en común
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 1's o 0's y un vector de enteros b de 9 posiciones, con 1's y 0's y com más de un 1 en la misma posición que en el vector a.
- **Salida:** -1
- **Resultado:** Correcto.

Prueba 3:

- **Descripción:** Devuelve -1 si no tienen ningún 1 en común
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 0's y un vector de enteros b de 9 posiciones, con 1's y 0's.
- **Salida:** -1
- **Resultado:** Correcto.

Prueba 4:

- **Descripción:** Devuelve -1 si no hay ningún 1 en los dos vectores
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con 0's y un vector de enteros b de 9 posiciones con 0's.
- **Salida:** -1
- **Resultado:** Correcto.

isUnique

Prueba 1:

- **Descripción:** Devuelve true si solo hay un elemento del vector igual a 1
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros de 9 posiciones, con 0s y un 1.
- **Salida:** true
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Devuelve false si hay más de un elemento con un 1
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con más de un 1
- **Salida:** false
- **Resultado:** Correcto.

Prueba 3:

- **Descripción:** Devuelve false si no hay elementos con un 1
- **Objetivos:** Comprobar que detecta correctamente este caso y devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un vector de enteros a de 9 posiciones, con todo 0s
- **Salida:** false
- **Resultado:** Correcto.

fillBoard:

- **Descripción:** Comprueba si se puede generar un kakuro único con un tablero
- **Objetivos:** Comprobar que dado un tablero posible, genera un kakuro único
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero con el que sabemos que puede generar un kakuro único
- **Salida:** true

- **Resultado:** Correcto.

howManyWhites

Prueba 1:

- **Descripción:** Contar el número de celdas blancas de un tablero con celdas blancas
- **Objetivos:** Comprobar que devuelve correctamente el número de casillas blancas del tablero
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero con celdas blancas y negras
- **Salida:** número de celdas blancas, en nuestro caso 44
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Contar el número de celdas blancas de un tablero sin celdas blancas
- **Objetivos:** Comprobar que devuelve correctamente el número de casillas blancas del tablero
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero sin celdas blancas
- **Salida:** 0
- **Resultado:** Correcto.

generate

- **Descripción:** Generar un kakuro válido dado un tamaño y dificultad
- **Objetivos:** Comprobar que no da error.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tamaño nxn y un entero que representa la dificultad
- **Salida:** -----
- **Resultado:** Correcto.

firstColRow

Prueba 1:

- **Descripción:** Generar la primera (sin contar la superior, i = 0, que siempre es negra) y última fila del tablero para que sean simétricas
- **Objetivos:** Comprobar que no hay errores
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tamaño nxn y un entero que representa la dificultad
- **Salida:** ------
- **Resultado:** Correcto.

Prueba 2:

- **Descripción:** Generar la primera y última fila del tablero para que sean simétricas
- **Objetivos:** Comprobar que no hay errores cuando no existe la primera fila.
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tamaño nxn y un entero que representa la dificultad
- **Salida:** ------
- **Resultado:** Correcto.

randomCells

- **Descripción:** Generar las casillas interiores del tablero
- **Objetivos:** Comprobar que no hay errores
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tamaño nxn y un entero que representa la dificultad
- **Salida:** ------
- **Resultado:** Correcto.

CheckBoard

Prueba 1:

- **Descripción:** Eliminar las runs de una celda blanca situadas en el extremo derecho del tablero
- **Objetivos:** Comprobar que le eliminan correctamente las runs de una celda blanca
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**

- **Entrada:** un tablero que contiene una run de una celda blanca solitaria en la última columna, en nuestro caso la celda está en la posición (1,8)
- **Salida:** ------
- **Resultado:** Correcto. Se han modificado las celdas necesarias para que no pase.

Prueba 2:

- **Descripción:** Eliminar las runs de una celda blanca situadas en la esquina inferior derecha del tablero
- **Objetivos:** Comprobar que le eliminan correctamente las runs de una celda blanca
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero que contiene una run de una celda blanca solitaria en la última columna y última fila, en nuestro caso la celda está en la posición (8,5)
- **Salida:** ------
- **Resultado:** Correcto. Se han modificado las celdas necesarias para que no pase.

Prueba 3:

- **Descripción:** Eliminar las runs de una celda blanca situadas en la esquina inferior izquierda del tablero
- **Objetivos:** Comprobar que le eliminan correctamente las runs de una celda blanca
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero que contiene una run de una celda blanca solitaria, en nuestro caso la celda está en la posición (8,2)
- **Salida:** ------
- **Resultado:** Correcto. Se han modificado las celdas necesarias para que no pase.

Prueba 4:

- **Descripción:** Eliminar las runs de una celda blanca situadas en mitad del tablero
- **Objetivos:** Comprobar que le eliminan correctamente las runs de una celda blanca
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero que contiene una run de una celda blanca solitaria, en nuestro caso la celda está en la posición (4,5)

- **Salida:** ------
- **Resultado:** Correcto. Se han modificado las celdas necesarias para que no pase.

Prueba 5:
- **Descripción:** No se modifica un tablero que no contiene runs de una celda blanca
- **Objetivos:** Comprobar que le eliminan correctamente las runs de una celda blanca
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero válido
- **Salida:** ------
- **Resultado:** Correcto. No se ha modificado el tablero

DFS

Prueba 1:
- **Descripción:** Contar correctamente las celdas blancas alcanzables desde una celda dada
- **Objetivos:** Comprobar que se cuentan correctamente las celdas alcanzables
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero con celdas blancas y negras y la posición de una celda blanca.
- **Salida:** ------
- **Resultado:** Correcto. Se han contado las celdas blancas alcanzables desde la celda dada, en nuestro caso 44.

Prueba 2:
- **Descripción:** Si indicamos como inicial una celda negra, nos devuelve 0
- **Objetivos:** Comprobar que se cuentan correctamente las celdas alcanzables
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero con celdas blancas y negras y la posición de una celda negra.
- **Salida:** ------
- **Resultado:** Correcto.

connexBoard

Prueba 1:

- **Descripción:** El tablero es conexo
- **Objetivos:** Comprobar que devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero conexo
- **Salida:** true
- **Resultado:** Correcto. Al ser conexo, el número de celdas blancas alcanzables desde cualquier celda blanca es el número total de celdas blancas del tablero.

Prueba 2:

- **Descripción:** El tablero no es conexo
- **Objetivos:** Comprobar que devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero no conexo
- **Salida:** false
- **Resultado:** Correcto.

Prueba 3:

- **Descripción:** El tablero no contiene celdas blancas
- **Objetivos:** Comprobar que devuelve lo esperado
- **Otros elementos integrados: ------**
- **Drivers construidos:** driverCtrlGenerate
- **Stubs: ------**
- **Entrada:** un tablero sin celdas blancas
- **Salida:** true
- **Resultado:** Correcto.