10-1-2023

Daniel Santiago Rodriguez Sierra

PRÁCTICA TESTING Y DEBUG

Contenido

[Descargar repositorio Git 2](#_Toc125490108)

[Caja Blanca 3](#_Toc125490109)

[Caja Negra 4](#_Toc125490110)

[Debug 5](#_Toc125490111)

[Modificación de líneas 7](#_Toc125490112)

|  |
| --- |
| Descargar repositorio Git  1. **Descargar el código de GIT así como aprendimos en el tema 1. Encontraran el código en** [**https://github.com/aoyshi/Java-Tic-Tac-Toe**](https://github.com/aoyshi/Java-Tic-Tac-Toe) |

-Vamos al link anterior y copiamos la URL

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

* Creamos una carpeta y hacemos un **git init** a esta para inicializar repositorio git y descargar el repositorio. Hacemos un **git remote add origin y el enlace,** seguido de un **git fetch origin**

Texto

Descripción generada automáticamente

* Finalmente, para descargar el repositorio hacemos un **git pull origin master**

Texto

Descripción generada automáticamente

Cambiamos la URL del repositorio

* Para cambiar el URL del repositorio, primero creamos un repositorio en GitHub, copiamos el link del repositorio y hacemos un **git remote set-url origin” y el url de nuestro repositorio”.** Verificamos a donde se harán los push del repositorio y finalmente hacemos un **git push -u origin master**

Texto

Descripción generada automáticamente

|  |
| --- |
| Caja Blanca  1. **Crea un test de caja blanca con la función “main”.** |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
| Caja Negra  1. **Crea un test de caja negra en la función “checkHit”.** |

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

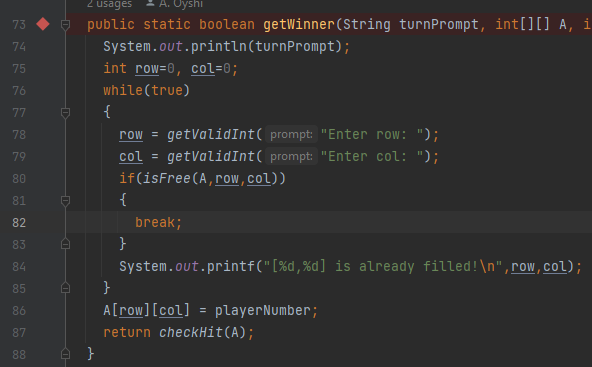
|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

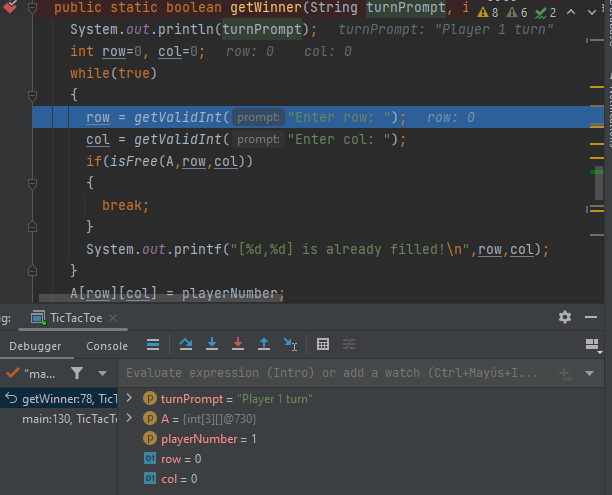
|  |  |
| --- | --- |
|  |  |

|  |
| --- |
|  |

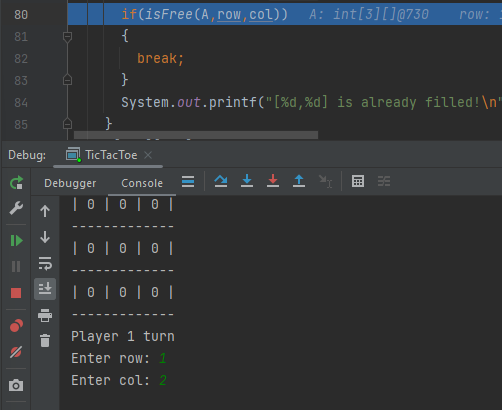
|  |
| --- |
| Debug  1. **Haz un debug exhaustivo en la función “getWinner”. Explica que hace el algoritmo y, sobre todo, las líneas 80 y 86. Documenta el proceso y muestra capturas de pantalla con las correspondientes explicaciones.** |



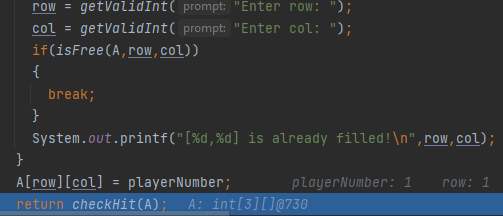
Comienza desde el jugador 1, haciendo que row y col utilicen el método getValid para introducir la fila y la columna para hacer el movimiento y a su vez comprobar que el movimiento sea valido



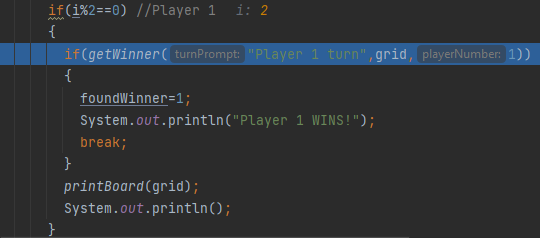
Aquí nos pide que pongamos la fila y la columna y una vez que verifica las filas y las columnas que hemos puesto, verifica si están en el rango del juego (3 x 3) y una vez verificado pasa al **if**, en el que nos envía a método **isFree** para comprobar que en la matriz la fila y la columna está o no ocupada, en el caso de estar ocupada, muestra un mensaje que dice que ya está ocupada y vuelve a pedir otra vez la fila y la columna, además en caso de no estar ocupada continua



Después el valor de la fila y la columna escrito se le asigna al jugador que estaba en su turno y después al hacer el return CheckHit(A) comprueba si hay 3 números iguales tanto fila, columna y diagonal (para saber si hay un ganador)

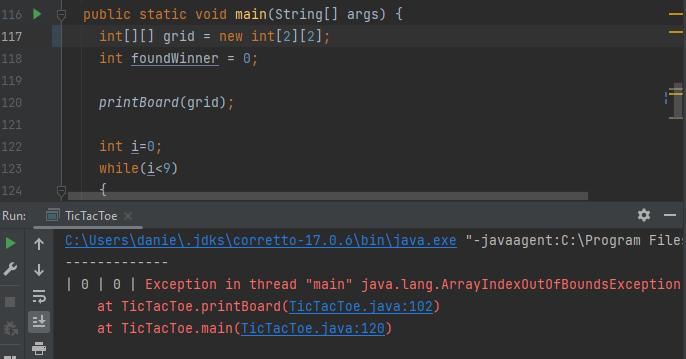


Posteriormente se va al main entrando al bucle y quitando una jugada más de las nueve que hay y a su vez señalando el turno del siguiente jugador y así sucesivamente hasta que haya un ganador o finalicen las 9 jugadas

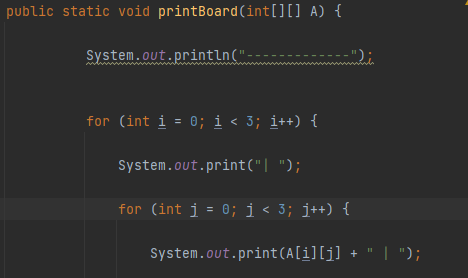


|  |
| --- |
| Modificación de líneas  1. **Modifica la línea 117 y pon int[] [] grid = new int [2][2];. ¿Hay algún error? Si es así, utiliza el debug para encontrar el por qué. Prueba el mismo poniendo int [][] grid = new int [4][4]; y explica por qué hay (o no ) error** |

Este es el error que nos ha aparecido poniendo **int[] [] grid = new int [2][2];**

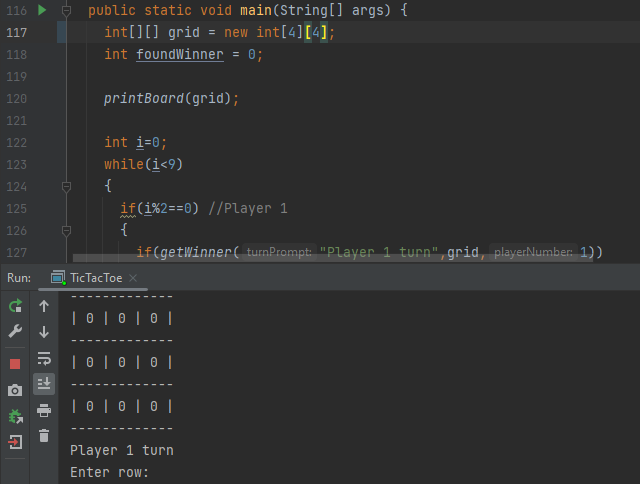
****

Como podemos ver, el error se encuentra en la línea 102, en la que podemos ver que es impreso el tablero del juego si tiene mínimo 3 filas que reconocer, ya en los **for** señala la cantidad de filas y columnas mínimo o sino aparecerá que esta fuera de los límites, en este caso para que nos imprima el tablero debemos cambiar los “3” señalados en la siguiente imagen, por “2”



Esto me ha pasado poniendo **int [] [] grid = new int [4][4];**

**-**como podemos ver, no ha habido ningún problema debido a que en los **for** señalados anteriormente indica que necesita al menos 3 filas y 3 columnas para ser impresos, por lo que, al imprimirlo, solo nos mostrara 3 filas y 3 columnas del array bidimensional y el resto quedara fuera

****