**Reporte Sprint #2**

Implementen las siguientes características del juego SOS: (1) los componentes básicos para las opciones del juego (tamaño del tablero y modo de juego) y el juego inicial, y (2) la ubicación del S/O para jugadores humanos sin verificar la formación de SOS o determinar el ganador. La siguiente es una interfaz de muestra. Se recomienda enfáticamente la implementación de una GUI. Deben practicar la programación orientada a objetos, haciendo que su código sea fácil de extender. Es importante separar el código de la interfaz de usuario y el código de la lógica del juego en diferentes clases (consulta el ejemplo de TicTacToe). Se requieren pruebas de xUnit.

|  |  |  |
| --- | --- | --- |
| SOS Icon  Description automatically generated Simple game Icon  Description automatically generated General game Board size  8 | | |
| Blue player  Icon  Description automatically generated S  Icon  Description automatically generated O | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | | O |  |  |  |  |  |  |  | |  |  | S | O | S |  |  |  | |  |  |  |  | S |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | S | | Red player  Icon  Description automatically generated S  Icon  Description automatically generated O |
|  | Current turn: blue (or red) |  |

Figura 1. Diseño de GUI de muestra del programa en el Sprint 2

**Entregables:**

1. **Demostración (8 puntos)**

Envíen un video de no más de tres minutos, donde demuestran claramente que implementaron las funciones requeridas y escribieron algunas pruebas unitarias automatizadas. En el video, deben explicar lo que se está demostrando.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Característica** | **métodos** | **Prueba unitaria** |
| 1 | Escoge el tamaño del tablero | SeleccionarTamanioTablero | NewTablero |
| 2 | Escoge el modo del juego | GameSelector | SelectSimpleGameMode  SelectGeneralGameMode |
| 3 | Juego inicial del tamaño de tablero elegido y modo de juego | InitBoard | ShowGameState |
| 4 | Movimientos “S” | MakeMove | MakeBlueMoveS\_SimpleGame() |
| 5 | Movimientos “O” | MakeMove | MakeRedMoveO\_SimpleGame() |
| 6 | Pruebas unitarias automatizadas |  |  |

1. **Resumen del código fuente (2 puntos)**

|  |  |  |
| --- | --- | --- |
| Nombre del archivo de código fuente | ¿Código de producción o de prueba? | # lineas de código |
| Tablero.cs | Producción | 61 |
| Form1.cs | Producción | 180 |
| Juego.cs | Producción | 91 |
| UnitTest1.cs | Prueba | 148 |
| Consola.cs | Prueba | 43 |
| IJuego | Producción | 17 |
| Total | | 540 |

**Deben enviar todo el código fuente para obtener más puntos por esta tarea.**

1. **Código de producción vs Historias de usuario/Criterio de aceptación (4 puntos)**

Actualicen sus historias de usuario y los criterios de aceptación de la asignación anterior y asegúrense de que capturan adecuadamente los requisitos. Resuman cómo se implementa cada uno de los siguientes criterios de aceptación/historia de usuario en tu código de producción (nombre de clase y nombre de método, etc.)

|  |  |
| --- | --- |
| **ID de historia de usuario** | **Nombre de historia de usuario** |
| 1 | Escoge el tamaño del tablero |
| 2 | Escoge el modo de juego de un tablero escogido |
| 3 | Comienza un nuevo juego del tamaño de tablero y del modo de juego elegidos |
| 4 | Hacer un movimiento en un juego simple |
| 6 | Hacer un movimiento en un juego general |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre y ID de la historia usuario** | **AC ID** | **Nombre clase(s)** | **Nombre Método(s)** | **Estatus (completo o no)** | **Notas (opcional)** |
| 1 | 1.1 | GameBoardCanvas | SeleccionarTamanioTablero | Completo |  |
| 2 | 2.1 | GameBoardCanvas | GameSelector | Completo |  |
|  | 2.2 | GameBoardCanvas | GameSelector | Completo |  |
| 3 | 3.1 | GameBoardCanvas | ShowGameStatus | Completo |  |
| 4 | 4.1 | Tablero | MakeMove | Completo |  |
|  | 4.2 | Tablero | MakeMove | Completo |  |
| 5 | 5.1 | Juego | JuegoGanado | Completo |  |
|  | 5.2 | Juego | JuegoGanado | Completo |  |
| 7 | 7.1 | Juego | TableroLleno | Incompleto |  |
|  | 7.2 | Juego | JuegoGanado | Incompleto |  |
|  | 7.3 | Juego | JuegoGanado | Incompleto |  |

1. **Pruebas vs Historias de usuario/Criterio de aceptación (6 puntos)**

Resuman cómo cada uno de los criterios de aceptación/historia de usuario es probado por su código de prueba (nombre de clase y nombre de método) o pruebas realizadas manualmente.

|  |  |
| --- | --- |
| **User Story ID** | **User Story Name** |
| 1 | Escoge el tamaño del tablero |
| 2 | Escoge el modo de juego de un tablero escogido |
| 3 | Comienza un nuevo juego del tamaño de tablero y del modo de juego elegidos |
| 4 | Hacer un movimiento en un juego simple |
| 6 | Hacer un movimiento en un juego general |

4.1 Pruebas automatizadas que corresponden directamente a los criterios de aceptación de las historias de usuario anteriores

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nombre y ID de la historia usuario** | **AC**  **ID** | **Nombre Clase (s) del código de prueba** | **Nombre método(s) del código Prueba** | **Descripción de los casos de prueba (entrada & salida esperada)** |
| 1 | 1.1 | TestEmptyBoard | NewTablero | Debe verificar que el tablero esté vacío y sea de tamaño n |
| 2 | 2.1 | TestSelectorModeGame | selectSimpleGameMode | Debe verificar que la variable "juego" se inicialice como un juego simple cuando se selecciona esa opción |
|  | 2.2 | TestSelectorModeGame | selectGeneralGameMode | Debe verificar que la variable "juego" se inicialice como un juego general cuando se selecciona esa opción. |
| 3 | 3.1 | TestShowGameState | ShowGameState | Debe verificar que el estado del juego se muestre después de que se haya seleccionado el tamaño y el modo de juego |
| 4 | 4.1 | TestMakeMove | MakeBlueMoveS\_SimpleGame | Debe verificar que se marque el movimiento S en una casilla válida para el jugador Azul y que el turno se ceda al siguiente jugador. (juego simple) |
|  | 4.2 | TestMakeMove | MakeRedMoveO\_SimpleGame | Debe verificar que se marque el movimiento O en una casilla válida para el jugador Rojo y que el turno se ceda al siguiente jugador (juego simple) |
| 5 | 5.1 | TestGameVictory | VictoryBluePlayerWithS | Debe verificar que el juego termine cuando el jugador Azul forme SOS con un movimiento válido de S y se declare como ganador. (juego general) |
|  | 5.2 | TestGameVictory | VictoryRedPlayerWithO | Debe verificar que el juego termine cuando el jugador Rojo forme SOS con un movimiento válido de O y se declare como ganador. (juego general) |
| 7 | 7.1 | TestEndGame | CheckFullBoard | Debe verificar que el juego termine cuando se llene el tablero con movimientos válidos. |
|  | 7.2 | TestEndGame | JuegoGanado | Debe verificar que se declare ganador al jugador Azul cuando tenga más SOS formados que el jugador Rojo |
|  | 7.3 | TestEndGame | JuegoGanado | Debe verificar que se declare ganador al jugador Rojo cuando tenga más SOS formados que el jugador Azul. |

4.2 Pruebas manuales que corresponden directamente a los criterios de aceptación de las historias de usuario anteriores

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre y ID de la historia usuario | AC  ID | Entrada de caso de prueba | Salida esperada | Notas |
| 1 | 1.1 | Tablero.GetCell(row,colum) para cada celda | 0 para cada celda | Tamaño del tablero: 7x7 Un tablero vacío de 7x7 se crea correctamente. Verificando que cada celda este Vacía |
| 2 | 2.1 | juegoSimple.tipoDeJuego | “SIMPLE” | J. Simple: La variable tipoDeJuego se inicializa como un juego simple. |
|  | 2.2 | juegoGeneral.tipoDeJuego | “GENERAL” | J. General La variable tipoDeJuego se inicializa como un juego general. |
| 3 | 3.1 | Juego.estadoDeJuego | “JUGANDO” | El juego inicia y muestra correctamente el estado de juego. |
| 4 | 4.1 | Tablero.GetCell(1,1)  Tablero.jugador | “S”  “Rojo” | Juego en curso, jugador Azul, movimiento S en (1,1) El movimiento S se marca correctamente en la casilla (1,1) y se cede el turno. |
|  | 4.2 | Tablero.GetCell(2,2)  Tablero.jugador | “O”  “Azul” | Juego en curso, jugador Rojo, movimiento O en (2,2) El movimiento O se marca correctamente en la casilla (2,2) y se cede el turno. |
| 5 | 5.1 | Juego.Ganado() | “true” | Juego en curso sin SOS, turno de Azul, movimiento S en (1,1) El juego termina y Azul gana al hacer un SOS con S en (1,1). |
|  | 5.2 | Juego.Ganado() | “true” | Juego en curso sin SOS, turno de Rojo, movimiento O en (2,2) El juego termina y Rojo gana al hacer un SOS con O en (2,2). |
| 7 | 7.1 |  |  | Juego en curso, tablero lleno por movimientos válidos El juego termina correctamente al llenar todo el tablero con movimientos válidos. |
|  | 7.2 |  |  | Juego terminado, Azul tiene más SOS hechos que Rojo Azul gana correctamente al tener más SOS hechos que Rojo. |
|  | 7.3 |  |  | Juego terminado, Rojo tiene más SOS hechos que Azul Rojo gana correctamente al tener más SOS hechos que Azul. |

4.3 Otras pruebas automatizadas o manuales que no corresponden a los criterios de aceptación de las historias de usuario anteriores

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Entrada prueba** | **Resultado esperado** | **Nombre de clase del código de prueba** | **Nombre del método del código de prueba** |
| 8 | Tablero.Ficha | ‘S’ | TestBlueStartGame | BlueStartGame |
|  |  |  |  |  |