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.

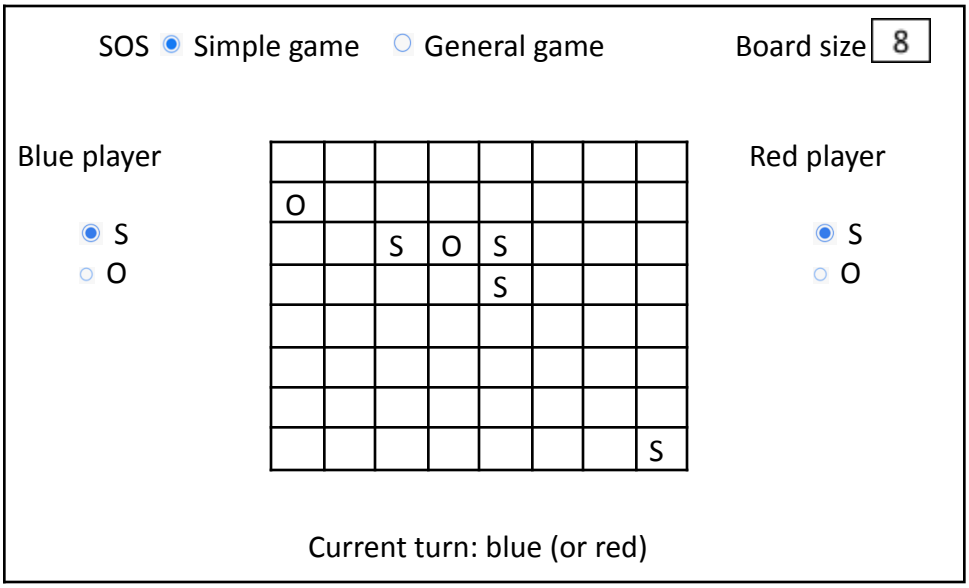

Figura 1. Diseño de GUI de muestra del programa en el Sprint 2

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		
1	Escoge el tamaño del tablero		
2	Escoge el modo del juego		
3	Juego inicial del tamaño de tablero elegido y modo de juego		
4	Movimientos “S”		
5	Movimientos “O”		
6	Pruebas unitarias automatizadas		
…			

2. Resumen del código fuente (2 puntos)

Nombre del archivo de código fuente	¿Código de producción o de prueba?	# lineas de código
test_board.py	prueba	154
board.py	codigo de produccion	113
main.py	codigo de produccion	5
Manager.py	codigo de produccion	33
screens.py	codigo de produccion	274
	Total	579

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

3. 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	Nombre de la historia de usuario	Descripción de historia de usuario	Prioridad	Esfuerzo estimado (horas)
1	Creación de tablero simple	Como usuario necesito ingresar un tamaño de tablero cuadrado antes de iniciar el juego.	media	1 hora
2	nsertar ficha en modo simpleI	Como jugador necesito poder insertar ‘S’ o ‘O’ en los casilleros del tablero cuadrado.	media	4 horas
3	Completar un SOS en modo simple	Como jugador necesito que el juego en modo simple detécte que se forme la palabra SOS	alta	7 horas
4	Termina juego en modo simple	Como jugador necesito que el juego termine y muestre quien es el ganador..	media	1 hora

Nombre y ID de la historia usuario	AC ID	Nombre clase(s)	Nombre Método(s)	Estatus (completo o no)	Notas (opcional)
1 Creación de tablero simple	1.1	Board	create_board	completo	
2 Insertar ficha en modo simple	2.1	Board	insert_piece	completo	
3 Completar un SOS en modo simple	…	Board	-	en proceso	
4.Termina el juego en modo simple		Board	-	en proceso	

4. 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	Crea un tablero vacío con un tamaño válido
2	Hace un movimiento en un juego simple
3	Completa un SOS
4	Termina el juego en modo simple

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 Creación de tablero simple	1.1	TestBoard	test_create_board	Se ingresa un tamaño valido del tablero y se crea un tablero vacío entrada=board(board_size esperada=None
2 Insertar ficha en modo simple	AC 2.1 <Inserta pieza válida en una posición válida>	TestBoard	teste_insert_piece_valid()	Se inserta una pieza valida en una casilla valida entrada=insert.piece(0,0,,'O') esperada='O'
	2.2 <Inserta pieza válida en una posición invalida>	TestBoard	test_insert_piece_invalid_coordinates	Se prueba si se inserta una pieza valida en una posicion invalida entrada=insert_piece(n+1,n+1'S' esperada=None
	2.3 <Inserta pieza válida invalida en una posición valida>	TestBoard	teste_insert_piece_invalid_piece_type test_insert_piece_invalid_piece_value	Se prueba si se inserta una pieza invalida en una posicion valida entrada=board.insert_piece(4) esperada='La pieza debe ser de tipo strign entrada=board.insert_piece('W') esperada='Pieza no valida'
	2.4<Inserta pieza válida en una posición ocupada>	TestBoard	teste_insert_piece_invalid_position_busy	Se prueba si se inserta una pieza valida en una posicion ocuapada insert_piece(0,0,'S') entrada=insert_piece(0,0,'S) esperada='Casilla Ocupada'

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**
11 Creación de tablero simple	1.1	entrada=board(board_size	esperada=None	
2Insertar ficha en modo simple	2.1 Inserta pieza válida en una posición válida	entrada=insert.piece(0,0,,'O')	esperada='O'	
	2.2.<Inserta pieza válida en una posición invalida>	entrada=insert_piece(n+1,n+1'S'	esperada=None	
	2.3 <Inserta pieza válida invalida en una posición valida>	entrada=board.insert_piece('W')	esperada='Pieza no valida'	
	2.4<Inserta pieza válida en una posición ocupada>	entrada=insert_piece(0,0,'S)	esperada='Casilla Ocupada'	

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**