**Reporte Sprint #4**

Implementa todas las características que permiten a un jugador (humano o computadora) jugar un juego SOS simple o general contra otro jugador (humano o computadora). Las características mínimas incluyen elegir humano o computadora para jugadores rojos y/o azules, elegir el modo de juego (simple o general), elegir el tamaño del tablero, configurar un nuevo juego, hacer un movimiento (en un juego simple o general) y determinar si un juego simple o general ha terminado. El componente informático debe ser capaz de jugar juegos simples y generales completos. Se te anima a considerar estrategias básicas para ganar juegos simples o generales. No se requiere un juego óptimo. El siguiente es un diseño de GUI de muestra. Debes usar una jerarquía de clases para lidiar con los requisitos del oponente de la computadora. Si tu código actual aún no ha considerado la jerarquía de clases, es hora de refactorizar tu código.

| SOS Icon  Description automatically generated Simple game Icon  Description automatically generated General game Board size | | |
| --- | --- | --- |
| Blue player  Icon                            Description automatically generated Human  Icon  Description automatically generated S  Icon  Description automatically generated O  Icon                            Description automatically generated Computer | Chart, line chart  Description automatically generated | Red player  Icon  Description automatically generated Human  Icon  Description automatically generated S  Icon  Description automatically generated O  Icon  Description automatically generated Computer |
|  | Current turn: blue (or red) |  |

Figura 1. Diseño de GUI de muestra del programa de trabajo para Sprint 3

**Puntos totales: 20**

1. **Demostración (10 puntos)**

Envía un video de no más de cinco minutos, que demuestre claramente que implementaste las funciones del oponente de la computadora y escribiste algunas pruebas de unidad automatizadas.

1. Un juego completamente simple donde el jugador azul es un humano, el jugador rojo es la computadora y hay un ganador.
2. Un juego general completo donde el jugador azul es la computadora, el jugador rojo es un humano y hay un ganador.
3. Un juego simple completo donde ambos lados son jugados por la computadora
4. Un juego general completo donde ambos lados son jugados por la computadora.
5. Algunas pruebas unitarias automatizadas para el oponente de la computadora.

En el video, debes explicar lo que se está demostrando.

1. **Historias de usuario para los requisitos del oponente de la computadora (1 punto)**

**Plantilla de historia de usuario**: Como <rol>, quiero <objetivo> [tal que <beneficio>]

| **ID** | **Nombre de historia de usuario** | **Descripción de historia de usuario** | **Prioridad** | **Esfuerzo estimado (horas)** |
| --- | --- | --- | --- | --- |
| 8 | movimiento computador | Como Computador quiero que mis movimientos sean más lógicos para garantizar mi victoria | 1 | 4 |
| .9. | Botón reinicio | Como Jugador, quiero un botón que reinicie el juego para no tener que cerrar el programa cada vez que termine un juego. | 2 | 3 |
| 10 | Contador SOS | Como jugador quiero un conteo de SOS formados para visualizar y verificar mi victoria o derrota. | 2 | 2 |

1. **Criterio de aceptación (AC) para los requisitos del oponente de la computadora (1 punto)**

| **ID y nombre de la historia de usuario** | **AC**  **ID** | **Descripción del criterio de aceptación** | **Estado (completado, por hacer, en progreso)** |
| --- | --- | --- | --- |
| (8) movimiento computador | 8.1 | AC 8.1 <descripción del escenario>  Dado que el computador está configurado para escribir de manera aleatoria  Cuando es su turno  Entonces hacer más lógico sus movimientos | por hacer |
| 8.2 | AC 8.2 <descripción del escenario>  Dado que el computador aún no tiene análisis del tablero  Cuando es su turno  Entonces facilitar su razonamiento | por hacer |
| (9)Botón reinicio | 9.1 | AC 9.1 <descripción del escenario>  Dado que el jugador  Cuando el juego terminó  Entonces habilitar un botón de reinicio | por hacer |
| (10)Contador SOS | 10.1 | Dado que el jugador forma un SOS  Cuando el juego empezó  Entonces mostrar la cantidad de SOS formados por cada jugador | por hacer |

1. **Resumen de todo el código fuente (1 punto)**

| Nombre del archivo de código fuente | Código de producción o de prueba | # lineas de ódigo |
| --- | --- | --- |
|  |  |  |
|  |  |  |
| Total | |  |

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

1. **Código de producción vs Nuevas historias de usuario/Criterios de aceptación (2 puntos)**

Resume cómo se implementa cada una de las nuevas historias de usuario/criterios de aceptación en tu código de producción (nombre de clase y nombre de método, etc.)

| **ID y nombre de la historia de usuario** | **AC ID** | **Nombre(s) clases** | **Nombres(s) métodos** | **Estado (completo o no)** | **Notas (opcional)** |
| --- | --- | --- | --- | --- | --- |
| 8 | 8.1 |  |  |  |  |
|  | 8.2 |  |  |  |  |
|  | … |  |  |  |  |

1. **Pruebas vs Nuevas historias de usuario/Criterio de aceptación (2 puntos)**

Resume cómo tu código de prueba (nombre de clase y nombre de método) o las pruebas realizadas manualmente prueban cada una de las nuevas historias de usuario/criterios de aceptación.

6.1 Pruebas automatizadas que corresponden directamente a algunos criterios de aceptación.

| **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 |  |  |  |
|  | 1.2 |  |  |  |
|  | … |  |  |  |
| 2 | 2.1 |  |  |  |
|  | … |  |  |  |

6.2 Pruebas manuales que corresponden directamente a algunos criterios de aceptación.

| **Nombre y ID de la historia usuario** | **AC**  **ID** | **Entrada de caso de prueba** | **Salida esperada** | **Notas** |
| --- | --- | --- | --- | --- |
| 1 | 1.1 |  |  |  |
|  | 1.2 |  |  |  |
|  | … |  |  |  |
| 2 | 2.1 |  |  |  |
|  | … |  |  |  |

6.3 Otras pruebas automatizadas o manuales que no correspondan a los criterios de aceptación

| **Número** | **Entrada de prueba** | **Salida esperada** | **Nombre de clase del código de prueba** | **Nombre del método del código de prueba** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

1. Presenta el diagrama de clases de tu código de producción y describe cómo la jerarquía de clases en su diseño trata con los requisitos del oponente de la computadora (3 puntos)?