**Reporte del Sprint #5**

Las principales tareas de esta asignación son:

1. Agrega la función de grabar (record) un juego en un archivo de texto. Se requiere la historia de usuario y los criterios de aceptación tanto de grabación como de reproducción
2. Realización de un ejercicio de revisión de código.
3. Resumir las lecciones aprendidas del Sprint 0 al Sprint 5.

El siguiente es un diseño de GUI de muestra del producto final, donde "Replay" es opcional.

**El trabajo es de caracter individual.**

|  |  |  |
| --- | --- | --- |
| SOS Icon  Description automatically generated Simple game Icon  Description automatically generated General game Board size  8 | | |
| 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  Replay |
| ☒ Record game | Current turn: blue (or red) | New Game |

Figura 1. Sample GUI layout of the final product Diseño de GUI del producto final

**Puntos totales**

1. **Demostración (10 puntos)**

Envía un video de no más de 15 minutos, demostrando claramente que has implementado todas las funciones en la siguiente tabla. En el video, debes explicar lo que se está demostrando. **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.**

|  |  |
| --- | --- |
|  | **Feature** |
| 1 | Se graba un juego simple completo de dos jugadores humanos. |
| 2 | Se graba un juego general completo de dos jugadores humanos |
| 3 | Se graba un juego simple completo de jugadores humano-computadora |
| 4 | Se graba un juego general completo de jugadores humano-computadora |
| 5 | Se graba un juego simple completo de jugadores computadora-computadora |
| 6 | Se graba un juego general completo de jugadores computadora-computadora |

Si has implementado la función de "replay" para obtener crédito adicional, debes incluir tu demostración en el video.

1. **Historias de usuario y criterios de aceptación para los requisitos para los requerimientos Record/Replay (1 punto)**

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

Agrega o elimina filas si es necesario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Nombre de historia de usuario** | **Descripción de historia de usuario** | **Prioridad** | **Esfuerzo estimado (horas)** |
| 20 | Grabar el juego | El jugador al final o en medio de la partida quiere ver las jugadas hechas y el orden de estas | media | 5 horas |
| .. |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID y nombre de la historia de usuario** | **AC**  **ID** | **Descripción del criterio de aceptación** | **Estado (completado, por hacer, en progreso)** |
| Grabar juego | 20.1 | AC 20.1 <Grabar movimiento>  Dado un juego iniciado  Cuando se haga un movimiento  Entonces se guarda el movimiento | completado |
| 20.2 | AC 20.2 <Grabar punto>  Dado un juego iniciado en modo general  Cuando se complete un sos  Entonces se guardara quien hizo el sos | completado |
| … |  |  |

**3. Revisión de código (4 puntos)**

Aplica la revisión del código fuente a una o dos de las clases más importantes (y a otras clases si el tiempo te permite) e informa de los resultados. Además de buscar errores, la revisión debe verificar: (1) si todo el proyecto ha seguido el estándar de codificación de manera consistente, (2) si el proyecto ha seguido los principios de diseño presentados en clase y (3) si hay olores de código que indican la necesidad de refactorización.

Las siguientes listas de verificación proporcionan pautas básicas. Puedes agregar nuevos elementos a cada una de las listas de verificación. Asegúrate de que tus respuestas sean el resultado del ejercicio de revisión del código. Si no hay hallazgos para una entrada, debes proporcionar una explicación. Por ejemplo, si tu respuesta a "¿Se violan las convenciones de nomenclatura?" es no, debes describir una convención de nomenclatura y presentar un ejemplo. No recibirás puntaje por si tus respuestas son simplemente sí o no sin información adicional.

Clases que han sido revisadas:Board , screens

Fecha/hora de duración del ejercicio de revisión del código:7/06/2023 inicio :3:07 am final : 4.06 am

|  |  |  |  |
| --- | --- | --- | --- |
| **Checklist** | **Items Checklist** | **Conclusiones** | |
| Estándares de codificación | Convenciones de nombres | Cada clase y metodo siguen una convencion en sus nombres | |
| Convención de ordenación de argumentos de método | Hay orden en los argumentos pero , en los metodos de insert\_piece : deberia tener orden alfabetico por que deberia colocarse primero col y despues row | |
| Comentarios significativos y válidos. | Cada metodo tiene un comentario con su funcionalidad pero podria adicionar comentarios dentro de los metodos para que sean mas claros | |
| Estilo consistente de bloques de código | Al ser python el lenguaje que uso , la identacion es necesaria por lo que los bloques de codigo son consistentes , aun asi podria haber malentendidos en el metodo one\_cell\_clicked | |
| Indentación consistente | si | |
| … |  | |
| Principio de diseño | Clase o método no bien modularizado | La clase containers2 tiene muchos metodos que se podria mejorar refactorizando y colocnadolos en la clase board  El metodo on\_cell\_clicked es el metodo que mas podria mejorar | |
| Visibilidad adecuada de cada variable, método y clase. | No,Python no distingue entre metodos privados o no privados pero como Buena practica se deberia poner en el nombre \_private | |
| Alguna clase con pobre abstracción | La clase board , esta podria derivarse de una clase cell para un major manejo de las Casillas ,como en el programa de tictactoe | |
| Diseño por contrato ( pre/postcondiciones) |  | |
| ¿Se viola el Principio Abierto-Cerrado? | Si , ya que algunos metodos fueron configurados para que puedan funcionar con la disposicion en la que estan | |
| ¿Se viola el Principio de Responsabilidad Única [[1]](#footnote-1)? | Si , el metodo one\_cell\_clicked viola este principio | |
| Smells código | Números mágicos | no | |
| Variable global /clase innecesaria | La clase board podria mejorarse con una clase cell , como en el juego tictactoe | |
| Código duplicado | Si , en la clase board Podemos ver el complete\_sos\_general y complete\_sos\_simple los cuales buscan un sos en un modo de juego general y simple,respectivamente,lo cual se podria implementar en un solo metodo, | |
| Métodos largos | On\_cell\_clicked | |
| Larga lista de parámetros | no | |
| Expresión demasiado compleja | no | |
| Switch o if-then-else que necesita ser reemplazado con polimorfismo |  | |
| Nombre de método o variable cuya intención no está clara | Creo que todo puede interpretarse de forma clasra | |
| ¿Algún método similar en otras clases? | El metodo win\_or\_tie\_g en conteiners2 y win\_or\_tie\_general en board , el que se encuentra en board nos retorna si se formo un punto mientras que el de screens hace que se register y se muestre en la interfaz grafica , analogamente con win\_or\_tie\_s y win\_or\_tie\_simple | |
| … |  | |
| **Errores** | **Fragmento de código con errores** | **¿Cuál es el error?** | **¿Por qué es un error?** |
|  |  |  |
|  |  |  |
|  |  |  |

**4. Resumen de todo el código (1 points)**

|  |  |  |
| --- | --- | --- |
| Nombre del archivo de código fuente | Código de producción o prueba? | # lineas de código |
| Board | Codigo de produccion | 191 |
| Screens | Codigo de produccion | 480 |
| main | Codigo de produccion | 5 |
| Manager | Codigo de produccion | 33 |
| Test\_board | prueba | 167 |
| Total de líneas de código | | 876 |

**No recibirás puntaje por esta tarea a menos que envíes tu código fuente completo.**

5. Resume las lecciones aprendidas de todo el proyecto respondiendo las siguientes preguntas desde la perspectiva de los procesos de desarrollo, codificación, diseño, refactorización y prueba (**4 puntos**):

* ¿Qué ganaste personalmente con el proyecto?

Gane experiencia , pero mas que eso , conocimientos de metodologias que se usan en el campo laboral de la carrera y que podria implementar en futuros proyectos .Conocimiento en testing y mas importante en la refactorizacion , en como no debo cargar un metodo o funcion con muchas funcionalidades porque tiende a fallar , a poder comentar con mayor presicion para que otras personas puedan entender con mayor facilidad mi codigo .

* ¿Qué hace bien tu proyecto y qué podría hacer mejor tu proyecto?

Lo que hace bien es lo que se le pidio que hiciera , es decir que todo lo que se pidio en anteriores sprint se logro pero lo que se podria mejorar seria la logica detras de esto por ejemplo :la forma en encontrar un sos , mi programa recorre todo el tablero , pero no es necesario ya que podria solo fijarse en el ultimo movimiento hecho y a partir de eso buscar si se formo un sos , otra cosa que se podria mejorar es el enfentramiento contra el computador ,actualmente funciona colocando una pieza en algun lugar aleatorio del tablero , para mejorar esto busque la forma de implementar una IA lo cual me hizo investigar sobre aprendizaje por refuerzo , para que el computador pueda ver la mejor jugada posible en el tablero , pero por motivos de tiempo y que no se mucho sobre el tema no pude implementar .

Tambien podria mejorar la refactorizacion de algunos metodos e implementar un Sistema de celdas como el juego de tictactoe .

* ¿Cómo podrías mejorar tu proceso de desarrollo si desarrollas un juego similar desde cero?

Podria mejorar en el proceso de crear pruebas , gracias al rgr ahora se como podria implemetar ,de una mejor manera, un pruebas . Tambien el no cargar mis metodos y major refactorizalos , ahora que se metodologias de trabajo podria yo similar un cliente y avanzar un juego en sprints parte por parte asi teniendo un enfoque de que mejorar en cada entrega personal que haga.

Requisito mínimo para (5): Una página completa a espacio simple, tamaño de fuente no mayor a 12 puntos.

1. Revisa: [Violation solution for single responsibility principle](https://softwareengineering.stackexchange.com/questions/342051/violation-solution-for-single-responsibility-principle) [↑](#footnote-ref-1)