*Instituto Tecnológico de Costa Rica*

*Unidad de Computación*

*BattleShip*

*Daniel Francisco Barrantes Zamora*

*Ryan Antonio Vargas Alvarado*

*Sede San Carlos*

*23 de abril de 2024*

**Introducción:**

En este proyecto tiene como objetivo la recreación del clásico juego de Batalla Naval pero desde un entorno digital adaptado al mundo tecnológico de hoy en día. Este juego se realiza mediante el lenguaje de Python permitiendo explorar y aplicar diferentes conceptos de programación como la gestión de matrices, la interacción con el usuario y el almacenamiento de datos. Con lo cual este proyecto no solo brinda una experiencia de juego al usuario, sino también es una oportunidad para desarrollar y mejorar las habilidades de programación.

**Análisis del problema:**

1. Estructura de datos:

* Es necesario determinar cómo almacenar y administrar la información sobre la ubicación y estado de los barcos en el tablero.

1. Interfaz de usuario:

* Se debe diseñar una interfaz de usuario que permita al jugador interatuar con el juego de manera fácil y efectiva, considerando aspectos visuales y el uso de notificaciones.

1. Almacenamiento de datos:

* Se requiere crear un funcionamiento que permita guardar y cargar datos de la partida en curso, tales como la posición y estado de los barcos y la puntuación de los jugadores.

1. Depuración:

* Es de suma importancia realizar pruebas para garantizar la funcionalidad del juego y evitar posibles errores.

**Solución de problemas:**

1. Diseño de estructura de datos:

* Crear una matriz para cada jugador donde se almacene la información de la posición y estado de los barcos.
* Almacenar el puntaje de los jugadores en una lista para cada jugador.
* Conservar en un listas las imágenes de los barcos y otros.

1. Desarrollo de interfaz de usuario:

* Crear controles tales como, botones y entradas que facilite la interacción del usuario con el juego durante el registro de jugadores, colocación de los barcos, ataques, entre otros.
* Incluir notificaciones visuales que informen al jugador sobre eventos importantes durante la partida.

1. Implementación de almacenamiento de datos:

* Se desarrollan mecanismos que permiten guardar y cargar los datos de la partida mediante archivos de texto

1. Depuración:

* Emplear pruebas para encontrar posibles errores, en busca del buen funcionamiento del programa.

1. Imágenes:

Menú “añadir jugadores”

A screenshot of a video game

Description automatically generated

Menú “dimensiones tablero”

A screenshot of a website

Description automatically generated

Menú “colocar barcos”

A screenshot of a game

Description automatically generated

Pantalla de juego

A screenshot of a computer game

Description automatically generated

**Análisis de resultados:**

Si bien se logró terminar con las condiciones que requería el proyecto, también hay que considerar que el juego es bastante mejorable, puesto que la mayoría de las soluciones que se propusieron a lo largo del periodo de desarrollo aún pueden ser mejor optimizadas y pulidas, simplificando en gran medida la lógica del juego. Eso no quita que el proyecto ya es bastante completo por sí solo, y aún con pequeños errores que pueden darse durante una partida, estos no deberían arruinar la experiencia de juego. A continuación, se mostrará una lista con cada uno de los apartados del juego, enseñando quien realizo cada parte y algunas observaciones generales.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tarea/requerimiento** | **Realizador** | **Estado** | **Observaciones** |
| Registro de jugadores | Daniel Barrantes  (100%) | Completado |  |
| Creación de matriz  Del juego | Daniel Barrantes  (100%) | Completado |  |
| Colocación de la flota de cada jugador | Daniel Barrantes  (100%) | Completado |  |
| Desarrollo del juego | Ryan Vargas A  (100%) | Completado | A la hora de colocar las matrices en la ventana puede existir problemas con tamaños muy grandes. |
| Movimiento de naves | Ryan Vargas A  (100%) | Completado |  |
| Guardado del juego | Ryan Vargas A  (100%) | Completado | La opción de cargar un juego requiere de recordar el nombre del archivo de juego. |
| Documentación interna | Ryan Vargas A  (50%)  Daniel Barrantes  (50%) | Completado |  |
| Uso de control de versiones | Ryan Vargas A  (50%)  Daniel Barrantes  (50%) | Completado |  |
| Documentación externa | Ryan Vargas A  (50%)  Daniel Barrantes  (50%) | Completado |  |

**Conclusiones:**

Código demasiado extenso:Aunque algunas partes se simplificaron en busca de mejorar el rendimiento del juego, no hay que negar que se puede mejorar gran parte de la lógica. Por ejemplo, usando recursividad en las funciones es una excelente manera de simplificar el código en un buen porcentaje.

Rendimiento: El rendimiento del juego deja algo que desear. Cuando se actualiza cada matriz, se produce un ligero retraso en la carga, que aumenta a medida que el tamaño del tablero crece. Además, si se mueve la ventana mientras se está generando la matriz, pueden experimentarse bajones de rendimiento tanto en el programa como en la computadora. A pesar de estos problemas, no afectan de manera significativa la experiencia de juego, por lo que no podemos decir que sea imposible jugar. Sin embargo, es evidente que podríamos haber solucionado este problema optimizando el código desde el principio. Ahora que somos conscientes de esto, lo tendremos en cuenta para futuros proyectos.

Sistema de guardado Nos sorprendió gratamente lo fácil que resultó implementar un sistema de guardado para nuestro juego, sin duda gracias a que ya estábamos organizando nuestros datos en variables y utilizando listas de manera ordenada. Sin embargo, no todo es perfecto, esto debido a un manejo inadecuado del tiempo, nuestro código requiere que recordemos el nombre del archivo de guardado para poder cargarlo posteriormente. Esta solución podría considerarse un tanto rudimentaria.

**Recomendaciones:**

Algunas recomendaciones a partir de la experiencia obtenida a partir de la elaboración de nuestro proyecto, y que podrían enriquecer a alguien su conocimiento:

* El uso de múltiples funciones para elaborar proyectos más grandes es una excelente manera de mantener un orden a la hora de programar, y ayudar a entender a otras personas ajenas a la creación de la lógica.
* Sin duda, implementar librerías a nuestro código es una buena manera de enriquecer la lógica de este, aumentando enormemente la calidad del proyecto.
* Usar múltiples archivos para nuestro juego supuso un problema al mantener variables y moverlos de un archivo a otro, por eso recomendamos usar múltiples archivos para dividir la lógica pura de la interfaz.
* Para proyectos grandes, puede ayudar el primero a hacerse un mapa mental para establecer las ideas principales y la línea que seguirá el proyecto, ya que si se inicia sin una idea clara luego esto puede provocar atascos a la hora de programar

***Referencias bibliográficas:***

Imágenes utilizadas durante la realización del juego:

[*https://www.google.com/url?sa=i&url=https%3A%2F%2Felecodigital.com.uy%2Fsociedad%2Fla-increible-historia-del-tesoro-en-colonia-aseguran-haber-encontrado-el-tesoro-del-pirata-tomas-de-molina-2%2F&psig=AOvVaw1MDFLbWL9FcxPUS-E365hC&ust=1713921798453000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCJDyo7GW14UDFQAAAAAdAAAAABAE*](https://www.google.com/url?sa=i&url=https%3A%2F%2Felecodigital.com.uy%2Fsociedad%2Fla-increible-historia-del-tesoro-en-colonia-aseguran-haber-encontrado-el-tesoro-del-pirata-tomas-de-molina-2%2F&psig=AOvVaw1MDFLbWL9FcxPUS-E365hC&ust=1713921798453000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCJDyo7GW14UDFQAAAAAdAAAAABAE)

[*https://www.freepik.es/icono/mensaje\_4518512*](https://www.freepik.es/icono/mensaje_4518512)

[*https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcQ4RWl1t9t81vkPRwbUgW7\_Qv44gNRnONivr0PzPjY7S5Ly\_\_WX*](https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcQ4RWl1t9t81vkPRwbUgW7_Qv44gNRnONivr0PzPjY7S5Ly__WX)