**Universidad de Costa Rica**

**Escuela de Ciencias de la Computación e Informática (ECCI)**

**CI-0112 Programación I**

**Grupo: 002**

**Documento de decisiones: Proyecto 1.**

**Docente:**

**Luis Campos Duarte**

**Estudiantes:**

**Josué Rodríguez Aguilar I C4J023**

**Kevin Calderon Martinez I C4D511**

**I Ciclo**

**Abril 2025**

**a. Acuerdos de Pareja:** Describir los acuerdos alcanzados entre los miembros del equipo, como la división del trabajo y las responsabilidades.

El acuerdo logrado entre ambos estudiantes fue el de, principalmente, la subdivisión mutua de los juegos. A Kevin el ahorcado, y a Josué la batalla naval. Ambos lograron terminar su respectiva clase, y luego Kevin junto a Josué por llamada se encargaron de trabajar el Main de la Interfaz. Josué se encargó de crear el repositorio del proyecto en GitHub y trabajar el README, y ahí se fueron subiendo los avances del proyecto. El doxygen del Man se hizo en conjunto esfuerzo, mientras que la clase ahorcado y batalla naval lo trabajaron Kevin y Josué respectivamente.

**b. Decisiones de Diseño:** Explicar cómo se diseñaron las clases, cómo se estructuraron los métodos y cualquier decisión significativa tomada durante el diseño.

**Batalla Naval:**

Con respecto a las decisiones de diseño de batalla naval, este juego (clase) se sustentó en la utilidad de arrays multidimensionales (matrices) para representar los tableros de los jugadores y sus respectivas vistas del tablero enemigo. Donde se diseñaron varios métodos, por destacar el colocarBarcs que colocaba los barcos de los jugadoras por medio de string y su sustitución a partir de scanners, luego el metodo disparar que básicamente repetía la acción de disparar sobre una casilla y retornaba si este disparo impactó o fallo, luego el método de iniciar juego que básicamente era el cuerpo de la clase que creó el sistema de turnos y empleo los métodos creados para que cada jugador pudiera disparar y si acertaba repitiera su turno, si fallaba sería turno del otro jugador así repetitivamente hasta que alguno de los jugadores se quedará sin barcos y se decidiera el ganador.

**Ahorcado:**

El ahorcado consiste principalmente en un método que se ha denominado como dosRondas, que consiste en que dos jugadores (j1 y j2) traten de adivinar una palabra secreta dada por su adversario, con 6 intentos para cada jugador (Primero adivina j2, luego j1). Y que se compare la cantidad de intentos restantes con la que quedó cada jugador, para definir al ganador.

**c. Decisiones de Implementación:** Detallar las decisiones específicas de implementación, como la elección de estructuras de datos, algoritmos utilizados, etc.

Con respecto a la clase batalla naval, se decidió utilizar strings pues representan mediante un sistema de simbología declarado al inicio del juego el agua(-), los impactos en los barcos(X), los disparos fallados (O) y los barcos (■). También se emplearon arreglos multidimensionales para representar los tableros de los jugadores y su vista del oponente, en conjunto a varios ints para declarar variables como filas y columnas por utilizar, contadores de Barcos, barcos de cada jugador y demás. Luego se utilizaron varios métodos, donde se inicializar los tableros con agua para poder manipularlos con facilidad más adelante, el colocar barcos de los jugadores mediante scanners, el poder disparar a los barcos por medio de un proceso reiterado de sustitución y verificación del contenido de las casillas en los tableros, el método iniciar juego que verifica si impacto o falló el disparo y alternaba turnos mediante booleans, mostrar tableros por turnos con booleano y el método de contar barcos para únicamente mostrar los barcos restantes de los jugadores.

Para el ahorcado, se decidió usar Strings que definieron la palabra secreta, las letras buenas y letras malas, un int con los intentos restantes y un array de tipo char[] que guardase el avance de la palabra secreta mientras se adivinaba en la ronda. Primero, un método donde se verifique si las letras van en la palabra secreta o no, que sume o reste las letrasBuenas/Malas y que reste intentos. Dos booleanos que definan si el jugador ya adivinó la palabra (Ganó) o si se quedó sin intentos (Perdió). El super método de las dos rondas, que mediante un auxiliar guarde los intentos restantes de cada jugador, y que luego mediante un método declare quién ganó, o si fue un rotundo empate.

Cabe aclarar que no se menciona mucho sobre la clase interfaz pues está únicamente ejecuta el programa y permite al usuario determinar qué desea jugar y si quiere volver a jugar o no por lo que las decisiones de dicha clase no fueron determinantes solo se utilizo como método de acceso para las clases de ahorcado y batalla naval.

**d. Puntos de Mejora:** Identificar áreas que podrían mejorarse o ampliarse en futuras versiones del proyecto.

La comunicación entre ambos integrantes fue clara y concisa; no hubo problemas con respecto a la organización del proyecto, la codificación de las clases, la redacción de los doxygen y el documento de las decisiones.

Hay aspectos por mejorar para la clase Ahorcado; principalmente con la declaración poco necesaria de los setters que no se usaron en el Main, únicamente los getters en el mismo método del dosRondas de la clase Ahorcado. Obviando eso, el código fue bastante eficiente, y la mecánica de las dos rondas con la variable auxiliar que guarde los intentos restantes de cada jugador, bastante única.

Por último, con respecto a la clase de batalla naval lo que mejoraría sería la implementación de barcos con mayor tamaño y tableros con tamaños mayores y matrices no cuadradas pues en toda la implementacion del codigo, si el tamaño del tablero no es cuadrado ya no funcionara, y poco más la verdad que es muy eficiente la clase y cumple con lo requerido.