Batalla Naval/espacial

**Introducción**

En Batalla Naval Espacial, el juego se juega en una cuadrícula o tablero, que representa el espacio estelar. La cuadrícula se compone de filas y columnas numeradas que forman casillas. el jugador tiene su propia cuadrícula, una para ocultar sus naves y otra para marcar los intentos de ataque.

Preparación:

El jugador acuerda el tamaño de la cuadrícula, que suele ser una cuadrícula de 10x10 casillas, pero en el caso de este juego el tamaño de la matriz es escogida por el user.

El jugador y la “pc” se turnan para realizar ataques. El jugador elige una casilla en la cuadrícula del oponente para atacar, anunciando las coordenadas (por ejemplo, “5,5”). El oponente revela si el ataque fue un "Hit" o "Miss".

Si el ataque impacta en una casilla ocupada por una parte de una nave del oponente, se anuncia como un "Hit". El jugador marca la casilla como "H" en su cuadrícula de ataques. Si el ataque no impacta en ninguna nave del oponente, se anuncia como "Miss". El jugador marca la casilla como "M" en su cuadrícula de ataques.

El user y la “pc” continúan alternando turnos hasta que todas las naves de un jugador hayan sido hundidas.

**Funcionamiento del juego**

El funcionamiento del juego de Batalla Espacial se basa en la utilización de 4 funciones clave que gestionan la creación y el desarrollo de la batalla en la galaxia.

llenadoMatriz: Esta función es responsable de crear la matriz de juego utilizando parámetros proporcionados por el usuario. Requiere el nombre de la matriz, la cantidad de filas, la cantidad de columnas y el número de naves, además de una variable llamada "respuesta". El número de naves se calcula como (filas \* columnas) / 10.

La función se encarga de llenar la matriz con naves en posiciones aleatorias según la "respuesta" proporcionada.

hitormiss: La función "hitormiss" verifica si en la coordenada proporcionada por el jugador existe un "0", que representa la presencia de una nave en el tablero. Si se encuentra un "0", la función devuelve un valor verdadero ("hit"), indicando que el ataque fue un impacto en una nave. Si no se encuentra un "0", devuelve un valor falso ("miss"). Esta función es el primer paso antes de cambiar "hit" por "h" y "miss" por "m" en el tablero.

actualizaMatriz: Al recibir el resultado de la función "hitormiss", "actualizaMatriz" se encarga de cambiar el valor en el tablero. Si el resultado es verdadero ("hit"), se cambia la casilla por "h". Si es falso ("miss"), se cambia la casilla por "m". Aunque su tarea sea aparentemente simple, es fundamental para determinar la cantidad de naves que aún permanecen en el tablero.

NumeroNaves: La función "NumeroNaves" recorre la matriz y cuenta la cantidad de "0" o naves que aún no han sido hundidas. Si el recuento es igual a 0, se determina que todas las naves en la matriz han sido derribadas.

**Dificultades**

1.Cantidad de paramatros para las funciones: uno de los principales obstáculos fue la gestión de múltiples parámetros al solicitar datos de entrada al usuario. Al requerir el nombre de la matriz, la cantidad de filas y columnas, y luego la fila y columna específicas de una coordenada proporcionada por el usuario, la tarea podría haberse vuelto muy confusa y propensa a errores.

Sin embargo, el uso de funciones se convirtió en un aliado. A través de estas funciones, se logró simplificar y optimizar el código, reduciendo la complejidad y facilitando la administración de parámetros. Esto permitió mantener el hilo de la lógica del juego de manera más eficiente, a pesar de la cantidad de datos requeridos.

2.Programación y Pruebas Repetidas: Una dificultad fundamental en el desarrollo fue el proceso de programación y las pruebas repetidas. Sentarse a codificar, revisar una y otra vez el código y realizar pruebas exhaustivas es un desafío común en el desarrollo de juegos. No todos tienen la paciencia y la destreza para superar estas etapas.

Dados 10,000

El jugador tiene la oportunidad de lanzar los seis dados en su turno. Después de cada lanzamiento, el jugador tiene la opción de guardar los dados que les otorgan puntos y volver a lanzar los restantes para intentar acumular aún más puntos. Sin embargo, cada tirada implica cierto riesgo, ya que, si se obtiene una combinación inválida en una tirada, pierdes todos los puntos acumulados en ese turno y tu turno termina.

Sistema de Puntos

En Dados 10,000, ganas puntos al apartar dados con valores especiales en tus lanzamientos:

* Unos: Cada dado con el valor "uno" que saques vale 100 puntos. Por ejemplo, si obtienes dos unos en una tirada, ganas 200 puntos.
* Cincos: Cada dado con el valor "cinco" que obtengas vale 50 puntos. Por lo tanto, si sacas dos cincos, ganas 100 puntos.
* Tres Iguales: Si obtienes tres dados con el mismo valor (excepto tres unos, donde se ganan 1000 puntos), ganas puntos igual al valor de esos dados multiplicado por 100. Por ejemplo, si obtienes tres dados con el valor "cuatro", ganas 400 puntos.

**Funcionamiento del Juego**

Dados 10,000 se basa en un conjunto de 5 funciones clave que son esenciales para su dinámica:

Lanza Dados: Esta función se encarga de asignar un valor entre 1 y 6 a cada posición del vector de dados, determinando qué valores tienes en mano para acumular puntos.

Mostrar Dados: Aquí puedes observar tus dados lanzados, los dados que has decidido apartar y el puntaje acumulado en tu turno. El vector "Dados en Mano" almacena los dados que eliges guardar antes de calcular puntos, mientras que "Dados Apartados" guarda los dados en mano una vez lanzas los demás dados diferentes de los que están en mano. Esta estructura optimiza el cálculo de puntos.

Fn Contadora: Esta función tiene una tarea simple pero crucial: contar. Utiliza dos vectores como parámetros, uno para contar y otro para almacenar los contados. Bajo cada índice, guarda la cantidad de un número específico. Por ejemplo, en el índice 1, se almacena la cantidad de unos.

Cálculo Score: Como su nombre indica, esta función calcula el puntaje del jugador. Recibe como parámetro el vector contador y examina cada índice para determinar los puntos que se sumarán al puntaje del turno. Es el cierre de cada jugada.

Validos: Con dos parámetros, contados y trabas, esta función verifica si tus dados lanzados contienen dados de puntaje. En Dados 10,000, perderás puntos si en un lanzamiento no obtienes dados de puntaje, es decir, ni 1, ni 5, ni un trío de dados iguales. "Validos" es esencial para mantener la tensión del juego y asegurar que los puntos se sumen adecuadamente.

Ya explicadas las funciones, el juego inicia su partida con 3 vectores esenciales. Cada uno de estos va a guardar el valor de los dados, pero bajo distintas situaciones.

Vector Dados: Contiene valores aleatorios entre 1 y 6, representando los dados lanzados en cada turno.

Vector DadosMano: Este vector permite al jugador elegir cuáles de los dados desea apartar y banquear por puntos. Los valores seleccionados entran en la función contadora y, más tarde, en la función de cálculo de puntos.

Vector DadosApartados: Aquí se almacenan los valores de DadosMano una vez que se utiliza la opción "Lanzar Dados". Estos valores son inamovibles y sirven para permitir al jugador ver qué dados banqueará por puntos.

En la primera tirada de dados, el juego presenta un menú con tres opciones:

Apartar los Dados: Esta opción permite al jugador elegir cuáles de los dados desea apartar. Estos dados se banquearán por puntos al final del turno.

Lanzar Dados: Al seleccionar esta función, el vector de Dados recibe valores aleatorios entre 1 y 6 en cada posición diferente de 0. Las posiciones con valor 0 son los dados que el jugador ha elegido apartar o mejor dicho se encuentran en el vector DadosMano y/o DadosApartados.

Banquear Puntos: Como su nombre indica, esta opción permite al jugador banquear los puntos acumulados en su turno. Estos puntos se suman al puntaje global del jugador y se acercan al objetivo de 10,000 puntos para ganar.

**Dificultades**

El desarrollo de Dados 10,000 presentó algunas dificultades que se tuvieron que superar para crear un juego fluido:

1. Función Validos: Uno de los mayores retos fue la creación de la función "Validos". Esta función se encarga de verificar si los dados lanzados contienen dados de puntaje, y es esencial para mantener la eficiencia del juego.

2. Interdependencia de Vectores: Los vectores en el juego, como "DadosMano", "DadosApartados" y "Dados", están estrechamente interconectados. Los valores en "DadosMano" influyen en el cálculo de puntos, y "DadosApartados" asegura que el jugador pueda ver qué dados banqueará por puntos. Garantizar que estos vectores funcionaran correctamente y en armonía fue una tarea que requirió mucha atención.

3.Programación y Pruebas Repetidas: Una dificultad fundamental en el desarrollo fue el proceso de programación y las pruebas repetidas. Sentarse a codificar, revisar una y otra vez el código y realizar pruebas exhaustivas es un desafío común en el desarrollo de juegos. No todos tienen la paciencia y la destreza para superar estas etapas.

El desarrollo de Dados 10,000 no solo requirió habilidades técnicas, sino también perseverancia y dedicación para resolver problemas y alcanzar un resultado satisfactorio. Programar no es tarea fácil y no es para todos, pero el esfuerzo puede superar cualquier dificultad.