Universidad Rafael Landívar
Facultad de ingeniería
Laboratorio de Programación sec. 16
Catedrático: Abraham Gutiérrez

Proyecto Final

Walter Francisco Meléndez Aguilar
Carnet:1174722

Guatemala 18 de noviembre de 2022

Definición de la estrategia

Las estrategias utilizadas en el proyecto fueron:

1. Crear un documento (de cualquier tipo) para tener una idea vaga sobre el proyecto.
2. Investigar los datos a usar en páginas recomendadas o designadas.
3. Buscar información sobre algún elemento que se va a usar en el proyecto.

En la parte número uno de la definición de estrategias, se opto por hacer un documento de Google Sheet para poder modelar una versión temprana del programa y sus valores que mostrará cuando el proyecto sea ejecutado.

La búsqueda de datos, se realizó en varios lugares, el primero fue en “Deck Shop”, después en “Royale API” y por último se optó por entrar en el jugo para poder verificar si los datos serán verídicos.

Y por ultimo se usó varios videos de referencia para poder comprender varias herramientas de Forms o crear procesos, como puede ser el caso de verificación de cantidad, la decoración del proyecto, la forma de utilizar los “picture box” o “group box” y entre muchas herramientas.

Validaciones.

Para validar los nombres de los jugadores se usó dos “textbox” para recibir el nombre de jugador y su equipo, una estructura de ciclo “for” con la palabra “. length” más una validación de tipo “if” que siempre va a aceptar las letras en la primera posición y en la quinta en el caso del nombre del equipo, también tiene y un botón para poder hacer su proceso. En la parte de elección de mazos de cada jugador, existe un tablero que muestra las estadísticas de cada mazo y otro que muestra de que tan buena es en la hora de disputar un enfrentamiento, cuyos resultados son la suma o promedio (en el caso del costo promedio) de todas las características de cada carta por mazo. Y por último en la pestaña de enfrentamiento se hace una validación con un botón para poder saber quién es el ganador, pero su proceso es de comparación a cada valor entre sí de los dos mazos de los jugadores y además de un botón llamado “limpiar” para poder reestablecer el tablero de puntos.

Análisis de las cartas de, mazos y sus estrategias

Antes de empezar con la explicación del análisis de las cartas, en mi proyecto hay otras dos características de las cartas que igualmente se necesitaba crear variable y almacenarlas, sus nombres son: “el costo” y su alcance (que es su radio de acción si es un hechizo o el rango de una carta si ataca a larga distancia), también que el proyecto fue realizado en Windows Form, en el leguaje de C#. Primero, para poder crear las cartas se podía usar un constructor o método de “Get” y “Set”, el método que use para crear para las cartas fue el “Get” y “Set” y la razón es por su facilidad de declarar y de asignar sus valores que un constructor. Para crear el método primero se crea una varia de tipo “int” para la vida y daño, tipo “String” para el nombre y tipo “double” para otros dos valores extras “costo”, costo de las cartas y su “alcance”. Después asignar cada valor un método de tipo “void” (o sea que de vuelva nada) para que este sea “el Set” con una pequeña declaración de una variable del mismo tipo pero con un nombre distinto, y por ultimo se escribe dentro del método “this.’Nombre’ = ‘nombre’;” Para crear el “Get”, simplemente se tiene repetir el mismo proceso que para el “Set” pero sin el método “void” porque si va a devolver un valor y en vez de “this.’Nombre’ = ‘nombre’;” adentro del método se puede escribir “ return Nombre” o “return this. Nombre” (al fin ninguno de las dos formas afecta el resultado).

Para crear los mazos primero se necesitó crear una nueva clase que este en el mismo proyecto, después declarar la clase carta 8 veces, debido a que en el mazo son ocho cartas, en el programa las clases fueron declaradas como publicas para que se pudieran utilizar en cualquier otra clase si es que fuera necesario o en el Forms del proyecto. Después de declarar las clases carta se puede hacer dos cosas, uno declara las variables en donde se van almacenar el total de las sumas de las estadísticas de cada carta en cada mazo o declarar la clase mazo en el Form del proyecto y asignar los valores en cada carta de cada mazo, utilizando el método “Set” y “Get” que anteriormente fue mencionado. Después de realizar los dos pasos se procede a crear el método para almacenar la suma de los datos un “Get”, por cada uno de las características por la razón de que este método solo va a servir para obtener un valor y no dar un valor a una variable. La forma a de hacer la suma de los datos según cada característica de cada carta, es escribir la variable a usar para almacenar la suma de la característica es igual a “Nombre de la clase carta. Getcarateristica()”, pero hay un caso especial en una de las características de la carta, en el costo se utilizó la clase “System.Math” en su tipo “Math.Round” para redondear un el resultado. Se usa “Get” debido a que se quiere obtener un valor de una carta y no dar valor a una carta. Por ultimo se agrega una etiqueta en el Forms (en la parte de diseño) para poder designar el valor a mostrar.

Propiedades de las cartas

Cada carta va tener un nombre, una cantidad de vida, un alcance y un costo. Algunas cartas no tienen alcance, vida, daño o pueden tener más una característica extra que les proporcione daño o vida, en ese caso simplemente se optó por sumarle la cantidad extra a la vida o daño que tienen al principio, lo que no pueden tener es otra cosa que modificar su costo, su rango o su nombre. Algunos ejemplo de lo anteriormente dicho es de la "Torre bombardera", su daño base es de 222 y su vida es de 1356, un alcance de 6 y un costo de 4, pero tiene "daño de muerte" que es de 222, entonces esos 222 de daño de muerte se le añaden a al daño base dando resultado a que tiene un daño base de 444 o la carta de "Bola de fuego" que no tiene vida, tiene un daño, tiene un costo y un alcance, pero también tiene una característica que hace daño a las torres de corona, que es el "daño de torres de corona " que es de 207 puntos de daño más los 689 puntos de daño que hace normalmente, da como resultado 896 puntos de daño en total. Por último, en la parte de Forms (de diseño) se agrego una imagen por cada mazo que contiene a cada carta creada en el proyecto.

Referencia

- Programación Windows Forms C# 28 --- PictureBox: https://www.youtube.com/watch?v=cn7xJ7twtRA

- CAMBIAR COLOR A LAS ETIQUETAS EN VISUAL BASIC: https://youtube.com/watch?v=yc-A6_9aQXg

- Cambiar individualmente el color de letras de un label https://es.stackoverflow.com/questions/542873/c-cambiar-individualmente-el-color-de-letras-de-un-label