Mgtr. Herwing Alexis Rodríguez Franco | Sección #7 (Laboratorio)
Facultad de Ingeniería: Introducción a la programación
Universidad Rafael Landívar

PROYECTO

Clash Royale League de SUPERCELL está por iniciar, y los jugadores: ***iAmJP de Team Queso y Mohamed Light de Golden Wing***, necesitan preparase para las World Finals del 2022, ambos jugadores han definido los 4 DECKS que utilizarán, pero necesitan de tu ayuda para seleccionar 2 de ellos. Deberás **construir un programa en C#** el cual determine cual es el mejor DECK a utilizar en base a *ELIXIR, PUNTOS DE VIDA, DAÑO*.

Hog 2.6 Cycle  https://link.clashroyale.com/deck/en?deck=26000010;26000014;26000021;26000030;26000038;27000000;28000000;28000011	**GIANT MORTAR**  https://link.clashroyale.com/deck/en?deck=26000003;26000019;27000002;26000018;26000016;26000001;26000015;26000012
XBOW 2.9  https://link.clashroyale.com/deck/en?deck=26000001;26000010;26000030;26000038;27000006;27000008;28000000;28000011	**SPELL BAIT**  https://link.clashroyale.com/deck/en?deck=26000026;26000041;28000004;26000000;26000030;28000011;27000003;28000003

Mgtr. Herwing Alexis Rodríguez Franco | Sección #7 (Laboratorio)
Facultad de Ingeniería: Introducción a la programación
Universidad Rafael Landívar

Instrucciones:

- **Introducción a la programación del juego**
  - Crear un algoritmo paso a paso desde cero para elección de mazos (Deck) en Clash Royale y simulación de juego 1 vs 1.
  - Construir una lógica programática de juego mediante operadores lógicos y operadores de comparación, elija 2 mazos (Deck) con 8 cartas cada una y realice una comparación mediante el daño, puntos de vida, velocidad, tipo de carta, calidad, el resultado deberá desplegar cual mazo es más efectivo y cuál de los dos ganaría, debe indicar cual sería su comparación para determinar la victoria.
    - Deberá mostrar las cartas de cada mazo.
    - Deberá indicar la comparación y la lógica que utilizo para determinar al ganador.

- **Verificación y construcción de mazos (Decks)**
  - Crear una cuenta en: https://royaleapi.com/ y agregar los 4 Decks a su cuenta para determinar el *coste medio de elixir (promedio de elixir) y el ciclo de 4 cartas.*
  - Verifique cual es el porcentaje de ataque, defensa, sinergia y balance de cada mazo construyéndolo en el siguiente enlace. https://www.deckshop.pro/es/deck-builder/clan-wars/build?n=1
    - Crear un documento en Google Sheet colocando la información de cada uno de sus mazos obtenida en RoyaleAPI y DECKSHOP PRO, deberá ser creativo en la manera que mostrará las estadísticas individuales de cada carta, por ejemplo: Daño de área, puntos de vida, velocidad de ataque, Objetivos, velocidad, alcance, etc., calidad y tipo de carta, así mismo deberá realizar un comparativo de cada uno de los mazos.
    - Ordenar las cartas por puntos de daño, de mayor a menor.
    - Identificar cuáles son las cartas con menor punteo de daño.

Mgtr. Herwing Alexis Rodríguez Franco | Sección #7 (Laboratorio)
Facultad de Ingeniería: Introducción a la programación
Universidad Rafael Landívar

- **Programación C#**
- Deberá ingresar al programa el nombre de los jugadores: ***iAmJP y Mohamed Light.***
- Deberá ingresar al programa el nombre del equipo al que pertenecen los jugadores: ***Team Queso y Golden Wing.***
- Construir la abreviatura del nickname a utilizar en competencia, ej. iAmJP de Team Queso, debe ser**: TQ – iAmJP40**, y Mohamed Light de Golden Wing, deberá ser: **GW – MohamedLight60**
    - Para la abreviatura del nickname deberá utilizar indexOf(), substring() y length().
    - TQ lo obtendrá mediante substring().
    - iAmJP y Mohamed Light lo obtendrá mediante substring().
    - El número 4 de iAmJP y 6 de Mohamed lo obtendrá con length() -1.
    - El número 0 se obtendrá del indexOf de la primera letra del nombre
    - Almacenar en un String nuevo el nombre del nickname de cada jugador, ejemplo: nickNameJ1 = **TQ – iAmJP40**, nickNameJ2 **GW – MohamedLight60** *(Podrá utilizar la cantidad de variables que considere necesarias y posteriormente puede concatenar los resultados hasta obtener el nombre del nickname).*
- Deberá ingresar los 4 DECKS al sistema, es decir: *nombre de la carta, elixir, puntos de vida y daño en cada una*, recuerde que el DECK únicamente puede tener 8 cartas.
    - Crear una clase llamada Deck
    - Declarar las siguientes variables: *nombre del jugador, equipo, nombre de la carta, tipo de carta, elixir, puntos de vida y daño*.
    - Crear un constructor para la clase, recuerde que el nombre y equipo lo solicito al inicio, pero las debe incluir en el constructor, las variables elixir, puntos de vida y daño se deben inicializar en 0.
    - Crear un método denominado addCard, en donde solicitará puntos de vida, elixir y daño.
    - Deberá ir almacenado los puntos de vida, elixir y daño en una variable independiente, es decir: totalPuntosVida,

totalPuntosDaño, queda a discreción del estudiante si estos los agrega en el método addCard o bien crea uno nuevo.

- Crear un método independiente para imprimirPuntosVida, imprimirPuntosDaño, es decir, se le solicita el promedio de puntos de vida y daño de todas las cartas agregadas en el Deck.
- Crear una condición (if-else) en donde se evalué si el total de puntos de vida del deck1 es mayor al deck2, y deck 1 es mayor a deck3, y así sucesivamente, recuerde que debe evaluar los 4 deck, el jugador: ***iAmJP*** elegirá su deck en base al que tenga mayor cantidad de puntos de vida.
- Crear una condición (if-else) en donde se evalué si el total de puntos de daño del deck1 es mayor al deck2, y deck 1 es mayor a deck3, y así sucesivamente, recuerde que debe evaluar los 4 deck, el jugador: ***Mohamed Light*** elegirá su deck en base al que tenga mayor cantidad de puntos de daño.
- Al evaluar los decks por medio de la condición (if-else), deberá indicar el deck elegido por el jugador, es decir el deck para puntos de vida y el deck para puntos de daño.
- Crear un método toString el cual retorne lo siguiente:
    - El jugador: "iAmJP" jugara en la CRL con un deck que contiene un total de # puntos de daño.
    - El jugador: "Mohamed Light" jugara en la CRL con un deck que contiene un total de # puntos de vida.
    - En el main deberá crear un objeto para cada jugador, deberá identificarlos como: jugador1 y jugador2 correspondientemente.
    - Colocar el título en el main de cada DECK, ej. DECK #1…
    - Después de colocar el título del Deck, deberá agregar las 8 cartas con sus puntos de vida, daño y elixir correspondiente.
    - Al finalizar el ingreso de las cartas por Deck debe imprimir el promedio de puntos de vida y promedio de puntos de daño de manera independiente.
    - Imprimir el método toString

- El jugador iAmJP al visualizar su programa en C# y su hoja de cálculo (análisis de datos en Google Sheet) decide cambiar dos de las cartas del deck elegido al identificar las del punteo menor.
- Crear un método que le permita modificar dos cartas, las cartas las deberá elegir usted mismo, y las deberá agregar nuevamente al programa, recuerde que la prioridad serán los puntos de daño.
- A pesar que cada jugador tiene una prioridad especifica para crear su Deck, es decir: puntos de daño y vida respectivamente, se solicita que por medio de una condición (if) identifique que deck es mayor en puntos de daño (únicamente debe realizarlo con los 2 decks elegidos) y deberá imprimir que jugador ganará la partida si se llegasen a enfrentar.

- A la plataforma deberá agregar un documento PDF con el enlace de su proyecto en formato el cual deberá incluir el código de su programa (completo) para solo compilarlo, Google Sheet y solo código en formato de texto.

- **Validación de datos y gestión de errores**
    - Ingresar al sitio: https://codesandbox.io/s/clash-royale-engine-wg0pq
    - Elija los 2 mazos ganadores según la resolución de su código y agréguelos al motor de Clash Royale
    - Elegir la siguiente configuración para el motor: *Card data versión: Current, Randomize deck order: YES, Max Elixir: 10, Elixir Regeneration: 1*
    - Grabar un video simulando una partida de juego

Mgtr. Herwing Alexis Rodríguez Franco | Sección #7 (Laboratorio)
Facultad de Ingeniería: Introducción a la programación
Universidad Rafael Landívar

RÚBRICA DE EVALUACIÓN

Evaluación de programación (80%)		
	Puntaje máximo	**Puntaje obtenido**
Código de proyecto	**16**	
Validación de datos y gestión de errores	**16**	
Interfaz	**16**	
Desarrollo de algoritmo y lógica de programación	**16**	
Funcionalidad	**16**	
Evaluación de documentación (10%)		
Presentación	**2**	
Calidad del informe	**2**	
Estructura y formato	**2**	
Ortografía y gramática	**2**	
Redacción y coherencia	**2**	
Evaluación de exposición → VIDEO (10%)		
Calidad de la exposición	**2.5**	
Capacidad síntesis	**2.5**	
Demostración del código y validación de datos	**2.5**	
Dominio de explicación del código	**2.5**	
TOTAL	**100**	