Universidad de los Andes > INGENIERÍA

Entrega Nº3:

Bitmonlandia

Integrantes: Leonardo Huilitraro Quezada
Katherine Jara Carrasco
Valeria Rosales Bazalar

Curso: Programación Orientada a Objetos

Carrera: Ingeniería civil

Profesores: Nicolás Gómez

Grupo: 2

Fecha de entrega: Viernes 31 de Mayo del 2019

Modelo de clases

Clase AgregarBitmons

Es una clase que permite añadir 5 bitmons de manera aleatoria en el tablero principal.

Método:

- public List<Bitmon>GetBitmons() . Este método retornara una especie de bitmon.
- public List<Bitmon>AddBitmon(Button[,] matrizBotones, int COLUMNAS, int FILAS, int BITMONS, bool[,] hayBitmon) . Aquí añade los primeros bitmons al mapa (1-Dorvalo,2-Doti,3-Ent,4-Gofue,5-Taplan,6-Wetar).

Clase Bitmon:

Es una clase abstracta que define las características de todos los Bitmons.

Atributos

- protected int tiempoDeVida.
- protecte int puntosDeVida.
- protected int puntoDeAtaque.
- protected string especie.
- protected int cantidadDeHijos.
- protected int posicionX
- protected int posicionY
- protected int direccionMov
- protected bool afin

Métodos:

- public abstract void CambioTerreno(Button [,] matrizBotones) es un método abstracto con parámetros de matrizBotones, que contiene botones del tablero, que permitirá saber si el terreno es modificado por el bitmon en el terreno actual.

- pulic abstract bool Daño: es un método abstracto con parámetros del tipo de bitmon, que muestra el daño que ha recibido el bitmon en el ataque.

- Public abstract int Daño(Bitmon bitmon). Retornará el daño que realizará a un bitmon en específico

- Public int Posicion X() . Retornará la posición x del bitmon

- Pubic int PosicionY(). Retornará la posición y de un bitmon

- Public string Especie() . Retornará el tipo de especie que tiene bitmon

- Public bool Muere(). Nos dirá true si bitmon muere, en caso de que muera el bitmon su tiempo de vida y puntos de vida será igual a 0. False cuando tenga tiempo y puntos de vida.

- Public void Reproducirse(). Modificará la cantidad de hijos en 1 y aumentará el tiempo de vida

- Public void ReducirTiempoDeVida(int a) . Permitirá reducir el tiempo de vida de un bitmon a cantidad
- Public void ReducirPuntosDeVida(int ataque). Modificará sus puntos de vida reduciéndola con el ataque.

- Public int ObtenerAtaque() Retornara los puntos de ataque que reciba determinado bitmon.

- public abstract bool AfinidadTerreno(Button[,] matrizBotones). Nos dirá si el bitmon es afín (true) o no lo es (false) en el terreno actual

- public abstract void Desplazamiento(Button[,] matrizBotones) . Modificará la posiciónX y posicionY del bitmon
- public abstract bool AfinidadBitmons(Bitmon bitmon). True si son afines, false si no

Clase ConfiguracionInicial:

Es una clase en form.
Atributos
- public int cantidadDeBitmons
- public int tiempoDeSimulación

Métodos
- private void ConfiguraciónInicial_Load
- private void button1_CLick(object sender, EventArgs e) . Recibirá la cantidad y tipo de Bitmons. Además del tiempo de simulación (meses)

Clase Dorvalo:

Clase hija de Bitmon.

Métodos
- public override void CambioTerreno(Button[,] matrizBotones) . Este bitmon no modificará el terreno, por lo que está vacío

- public override int Daño(Bitmon bitmon). Dependiendo del Bitmon es la cantidad de daño que retornará

- public override bool AfinidadTerreno: permitirá ver si tiene afinidad y si tiene podrá mantenerse en la celda.

- Pubic override void Desplazamiento Button[,] matrizBotones) . Este método permitirá mover al bitmon a las celdas que le sea posible

- Public override bool AfinidadBitmons(Bitmon bitmon) . True con quienes tenga afinidad, false con quienes no

Clase Doti:

Es una clase hijo de Bitmon.
Métodos

- public override void CambioTerreno(Button[,] matrizBotones) .
- public override int Daño(Bitmon bitmon). Dependiendo del Bitmon es la cantidad de daño que retornará

- public override bool AfinidadTerreno: permitirá ver si tiene afinidad y si tiene podrá mantenerse en la celda.

- Pubic override void Desplazamiento Button[,] matrizBotones) . Este método permitirá mover al bitmon a las celdas que le sea posible

- Public override bool AfinidadBitmons(Bitmon bitmon) . True con quienes tenga afinidad, false con quienes no

Clase Ent

Es una clase hijo de Bitmon.
Métodos

- public override void CambioTerreno(Button[,] matrizBotones) . Este bitmon no modificará el terreno, por lo que está vacío
- public override int Daño(Bitmon bitmon). No tiene daño

- public override bool AfinidadTerreno: permitirá ver si tiene afinidad y si tiene podrá mantenerse en la celda.

- Pubic override void Desplazamiento Button[,] matrizBotones) . No se mueve
- Public override bool AfinidadBitmons(Bitmon bitmon) . True con quienes tenga afinidad, false con quienes no

Clase Gofue:

Es una clase hijo de Bitmon.

Métodos

- public override void CambioTerreno(Button[,] matrizBotones) .
- public override int Daño(Bitmon bitmon). Dependiendo del Bitmon es la cantidad de daño que retornará
- public override bool AfinidadTerreno: permitirá ver si tiene afinidad y si tiene podrá mantenerse en la celda.
- Pubic override void Desplazamiento Button[,] matrizBotones) . Este método permitirá mover al bitmon a las celdas que le sea posible.
- Public override bool AfinidadBitmons(Bitmon bitmon) . True con quienes tenga afinidad, false con quienes no

Clase Wetar:

Es una clase hijo de Bitmon.

Métodos

- public override void CambioTerreno(Button[,] matrizBotones) . Este bitmon no modificará el terreno, por lo que está vacío
- public override int Daño(Bitmon bitmon). Dependiendo del Bitmon es la cantidad de daño que retornará
- public override bool AfinidadTerreno: permitirá ver si tiene afinidad y si tiene podrá mantenerse en la celda.
- Pubic override void Desplazamiento Button[,] matrizBotones) . Este método permitirá mover al bitmon a las celdas que le sea posible
- Public override bool AfinidadBitmons(Bitmon bitmon) . True con quienes tenga afinidad, false con quienes no

Clase Taplan:

Es una clase hijo de Bitmon.

Métodos

- CambioTerreno, determinará si Taplan puede cambiar de terreno.

- Daño con el parámetro bitmon, el cual muestra si recibe daño de Doti y Wetar, retornará los nuevos puntos de ataque. Y si recibe daño de otros bitmons retornara puntos de ataque distintos.

- AfinidadTerreno: permitirá ver si tiene afinidad y si tiene podrá mantenerse en la celda, retornara si hay afinidad o no.4- Desplazamiento, este método permitirá mover al bitmon a las celdas que sean seguras para el tipo de especie.

Clase ListaDeBitmons:

Es una clase form que permite determinar la cantidad de bitmons que hay en el tablero.
Atributo:

- Lista<string> listaBitmons.
- Int bitActual
- Int cantidadDeBitmon

Clase Terreno:

Es una clase que introduce 5 distintos tipos de terreno aleatoriamente.

Método

- Public void Configuración(Button[,] matrizBotones, bool [,] terreno, int CELDAS, int COLUMNAS, int FILAS). Modificará el mapa, dejándolo con terrenos distintos en cada celda

Cambios con respecto a la Entrega 2:

- Eliminamos clase Mapa, Zona,Simulación
- Creamos dos clases para configurar el tablero de bitmons. Lista de Bitmons y Configuración inicial.
- Agregamos la clase AgregarBirmons y Terreno
- La simulación fue hecha en form1