

- raiz : NodoDiccionarioABB<Clave, Valor>*

+ insertar(clave : Clave, valor: Valor)

+ eliminar(clave: Clave)

+ contiene(clave: Clave): bool

+ obtenerValor(clave: Clave): Valor

+ imprimirEnOrden()

+ estaVacio() : bool

- insertar(nodo: NodoDiccionarioABB<Clave, Valor>*, clave: Clave, valor: Valor): NodoDiccionarioABB<Clave, Valor>*

- eliminar(nodo: NodoDiccionarioABB<Clave, Valor>*, clave: Clave): NodoDiccionarioABB<Clave, Valor>*

- eliminarCasoHoja(nodo: NodoDiccionarioABB<Clave, Valor>*): NodoDiccionarioABB<Clave, Valor>*

- eliminarCasoUnHijo(nodo : NodoDiccionarioABB<Clave, Valor>*) : NodoDiccionarioABB<Clave, Valor>*

- eliminarCasoDosHijos(nodo: NodoDiccionarioABB<Clave, Valor>*): NodoDiccionarioABB<Clave, Valor>*

- sucesor(nodo: NodoDiccionarioABB<Clave, Valor>*): NodoDiccionarioABB<Clave, Valor>*

- buscarMenor(nodo: NodoDiccionarioABB<Clave, Valor>*): NodoDiccionarioABB<Clave, Valor>*

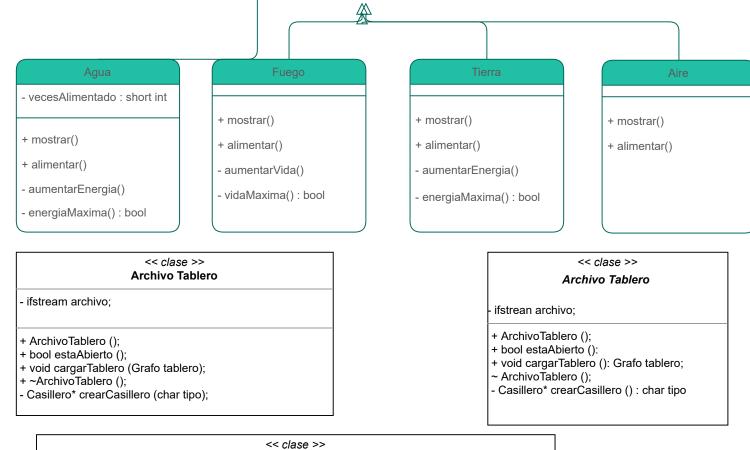
- buscarMayor(nodo: NodoDiccionarioABB<Clave, Valor>*): NodoDiccionarioABB<Clave, Valor>*

- buscar(nodo : NodoDiccionarioABB<Clave, Valor>*, clave : Clave) : NodoDiccionarioABB<Clave, Valor>*

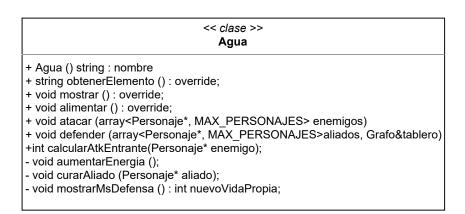
- imprimirEnOrden(nodo : NodoDiccionarioABB<Clave, Valor>*)

eliminarTodo(nodo: NodoDiccionarioABB<Clave, Valor>*)

NodoDiccionarioABB Personaje - clave : Clave # nombre : string - valor : Valor # escudo : short int - izquierda: NodoDiccionarioABB<Clave, Valor>* # vida : int - derecha: NodoDiccionarioABB<Clave, Valor>* # energia : int - padre : NodoDiccionarioABB<Clave, Valor>* + mostrar() + esHoja(): bool + alimentar() + soloHijolzquierdo(): bool # _mostrarAtributos() + soloHijoDerecho(): bool



- ifstream archivoEntrada; - ofstream archivoSalida; + ArchivoPartida() + bool hayPartidaGuardada (); + void cargarPartida (); Juego & juego; + void eliminarArchivo (); + ~ArhcivoPartida (); - void cargarPersonajes (Grafo& tablero, Jugador &jugador) - Personaje* crearPersonaje () string, string, string, string, array<int,2> posicion - void guardarPersonajes() array<Personaje* MAX_PERSONAJES> personajes, int cantPersonajes



<< clase >> Aire + Aire () string : nombre + string obtenerElemento (): override; + void mostrar (): override; + void alimentar (): override; + void atacar (array<Personaje*, MAX_PERSONAJES> enemigos) + void defender (array<Personaje*, MAX_PERSONAJES>aliados, Grafo&tablero) +int calcularAtkEntrante(Personaje* enemigo); void aumentarEnergia ();

<< clase >>

Tierra

- + Tierra() string : nombre
- + string obtenerElemento () : override;
- + void mostrar (): override;
- + void alimentar (): override;
- + void atacar (array<Personaje*, MAX_PERSONAJES> enemigos)
- + void defender (array<Personaje*, MAX_PERSONAJES>aliados, Grafo&tablero)
- + int calcularAtkEntrante(Personaje* enemigo);
- void aumentarEnergia ();
- bool energiaMaxima ();

<< clase >>

Archivo Personaje

- ifstream archivo;
- + ArchivoPersonajes ();
- + bool estaAbierto ();
- + void procesarArchivo (DiccionarioABB<string, Personaje*>& diccionario)
- + ~ArchivoPersonaje ();
- Personaje* crearPersonaje () string, string, string, string);

<< clase >>

Menu principal

- opcion: int
- + void iniciar()
- + void pausar();
- + void interfazPrincipal(Juego& juego, string, Personaje*>&diccionario)
- void limpiarPantalla();
- void mostrarOpciones();
- void pedirOpcion ();
- void AgregarPersonaje (DiccionarioABB<string, Personaje*> diccionario);
- void eliminarPersonaje (DiccionarioABB<string, Personaje*> diccionario);
- void mostrarPersonaje (DiccionarioABB<string, Personaje*> diccionario);
- void detallePersonaje (DiccionarioABB<string, Personaje*> diccionario);
- void pedirElemento (); string elemento;
- void pedirNomre (); string nombre;;
- void validarElemento (); string elemento;
- void pasarAMinuscula () string : cadena;
- Personaje* crearPersonaje(string, string);
- void cargarPersonajes (jugador1, jugador2, DiccionarioABB<string, Personaje*>diccionario);
- void menuJuego (jugador1. jugador2, DiccionarioABB<string, Personaje*>diccionario);
- void mostrarJuego ();
- void seleccionarPersonaje (jugador1, jugador2, DiccionarioABB<string, Personaje*>diccionario);

