Universidad Nacional ARTURO JAURETCHE

Complejidad Temporal, Estructura de Datos y Algoritmos

Trabajo Práctico Final

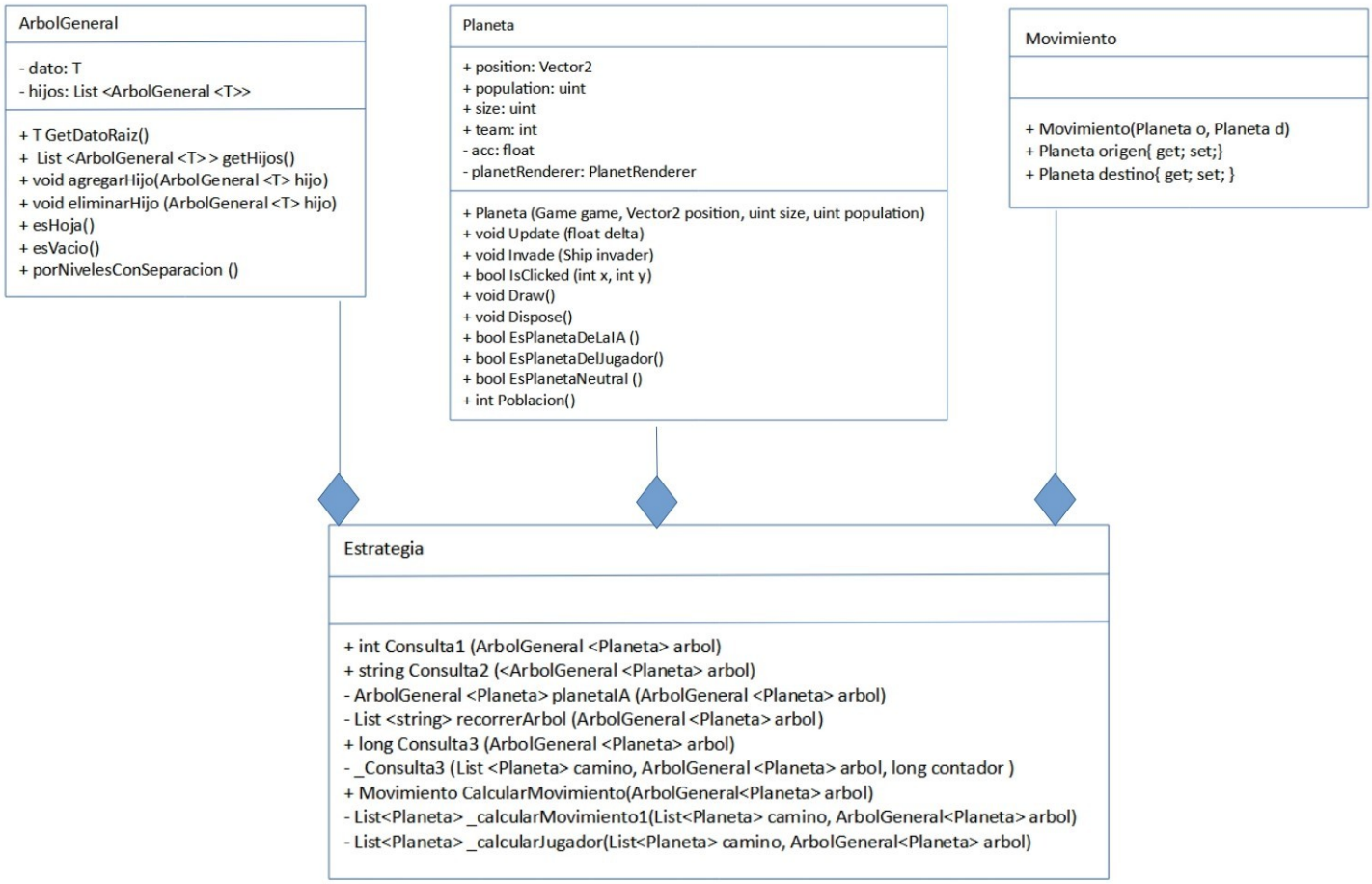
“Conquista Planetaria”

Profesor: Salina Mauro

Comisión: N.º 4

Alumno: Melgar Luciano

Diagrama de Clases UML



Detalles de implementación

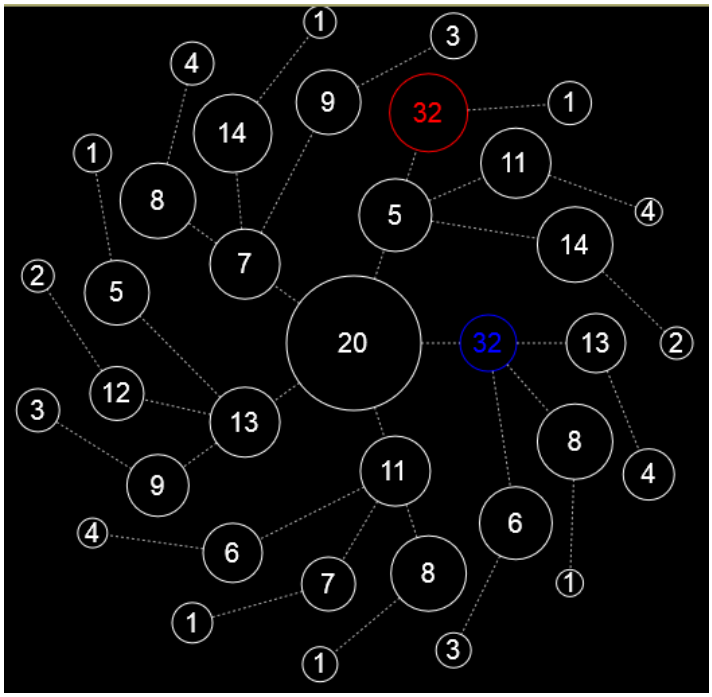
Para la realización de este proyecto se presentaron distintas dificultades, entre las cuales estaba la implementación del método CalcularMovimiento( ), el cual luego de su procesamiento realizaba un movimiento de ataque desde el planeta de la IA (Inteligencia Artificial) hasta el planeta del jugador. Tras repasar virtuales grabadas brindadas por la cátedra, se pudo plantear una forma para resolver dicho algoritmo, el cual consiste en crear un nuevo método privado que genere un camino desde la raíz del árbol hasta el planeta de la IA y otro método que genere un camino desde la raíz del árbol hasta el planeta del jugador.

Luego ambos caminos con sus respectivos orígenes y destinos serian llamados desde el método público CalcularMovimiento retornando los valores Origen y Destino de manera sencilla.

De la misma forma los métodos de Consultas tuvieron su parte dificultosa dentro del proyecto.

En un principio se intento realizar los métodos de Consulta de forma directa, siendo el mismo método el que resuelva toda la consulta, luego con la implementación del método CalcularMovimiento() se tomo la idea de dividir el procesamiento en distintos métodos, uno privado que realizaría una parte del proceso y retornaría un resultado que seria pasado hacia el método público, el cual a partir del retorno del método privado, realizaría un procesamiento más sencillo y directo de la consigna solicitada. De esta forma se consiguió la correcta implementación de todos los métodos, dividiendo las tareas en distintos métodos privados que retornarían un resultado y dicho resultado seria pasado hacia el método público.

Imágenes del sistema codificado



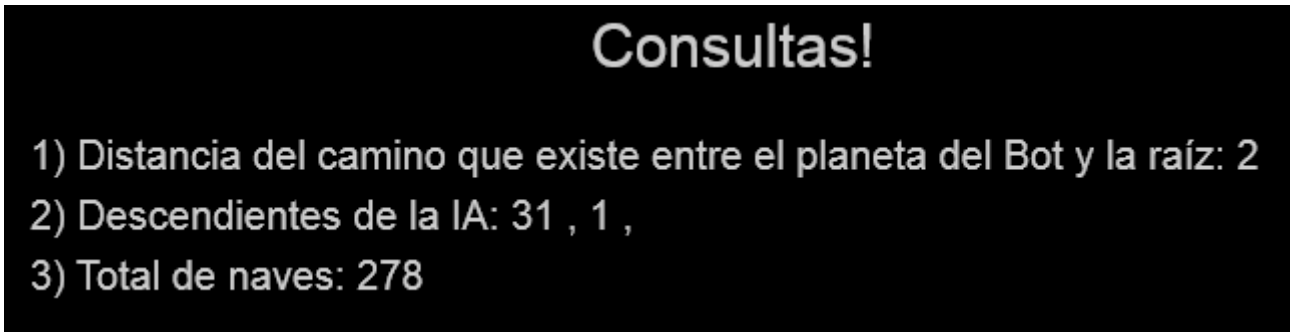
En esta imagen se puede observar el inicio del juego Conquista Planetaria, el cual se compone por un árbol con múltiples descendientes, los cuales a su vez tienen descendientes.

En el método CalcularMovimiento() se pedía que se retorne un movimiento de ataque que vaya desde el nodo de la IA hasta el nodo del Jugador.

Para llevar a cabo este método lo que se hizo fue, primero crear un método privado, el cual recorrería los distintos caminos del árbol hasta llegar al nodo IA. Estos caminos se almacenarían en una lista, la cual se usaría para determinar el camino correspondiente desde la Raíz del árbol hasta el nodo IA, en caso de llegar a una hoja y no encontrar al nodo IA, se volvería hacia atrás y se descartarían de la lista los nodos inútiles, de esta forma, cuando se encuentre a la IA, el listado solo contendría el camino desde Raíz hasta IA.

Luego se realizo el mismo procedimiento en otro método, pero este nuevo método buscaría el camino desde la Raíz del árbol hasta el nodo del Jugador. De esta forma con dos métodos privados conseguimos el camino de nodos, desde la Raíz hasta la IA y desde la Raíz hasta el Jugador.

Con ambos caminos definidos, solo nos queda realizar el movimiento especificando donde se iniciará y hacia donde llegará.



Para el método Consulta1() se pedía calcular y retornar en un texto la distancia del camino entre el planeta de la IA hasta la Raíz del árbol. Para realizar este método se reutilizo el método que calcula un camino desde la Raíz hasta la IA y almacena cada elemento en una lista, utilizado en CalcularMovimiento(), luego se contó la cantidad de elementos en dicha lista, se le resto 1 y finalmente se lo retorno como string.

Para la Consulta2() se solicitaba un listado en forma de texto de los descendientes del Bot.

Para realizar esta tarea se utilizaron dos métodos privados, el primero recorre el árbol en preorden y retorna el nodo IA. El segundo toma ese nodo IA retornado y lo recorre en forma preorden. Se crea un listado de los nodos descendientes de la IA y se retornan los elementos en forma de string.

Para la Consulta3() se pidió que se calcule y retorne en un texto la población total.

Para esta consulta se utilizo un método privado el cual recorre todo el árbol de forma recursiva y almacena en una lista cada elemento y almacena en un contador su población, luego de finalizar la recursión se retorna el contador.

Ideas o sugerencias

Una idea que podría dificultar la tarea del jugador seria que el nodo IA se refuerce conquistando a sus descendientes, esto podría lograrse generando un movimiento recorriendo sus hijos.

Más dificultoso seria si el nodo IA una vez llegado a la raíz, conquista sus demás hijos, generando nuevos caminos para conquistar planetas. Esto podría realizarse con un método el cual si la raíz pertenece a la IA ataque hacia sus descendientes.

Conclusión

Luego de la finalización del Trabajo Práctico Final “Conquista Planetaria”, puedo asegurar que aprendí de forma grafica todo lo relacionado a árboles, como los distintos recorridos y sus distintas disposiciones como se vio en la cátedra.

También me ayudo con la diferenciación entre las Clases, entendí como funcionaba cada atributo y método dentro de ellas.

En cuanto al código, este trabajo me planteo problemáticas que debían resolverse de una forma distinta, la cual me hizo entender que no todos los problemas deben

resolverse en una sola función o método. Entendiendo esto, separe mi código en distintas funcionalidades, las cuales se podrían reutilizar en problemáticas similares y también nos daba un código más limpio y fácil de entender.

En conclusión este proyecto significa un gran avance en cuanto a mi conocimiento en programación, ya que me brindo bases solidas de estructuras de datos, recorridos y una forma de dividir las funcionalidades del código para optimizar el software.