**T.P Final CTEDyA.**

Nombre: Bruno morales.

Comisión 5.

Detalles de implementación:

El programa se construye de, además de la clase “Estrategia”, de otras 5 clases creadas a partir del contenido visto en la materia, con la implementación de algunos de los recorridos de ArbolGeneral, como PreOrden, o PorNiveles, adaptados a la búsqueda de Planetas.

Para la primera consulta, se crea una “lista” de planetas, se llama a la clase “Recorridos” que contiene los recorridos, en este caso el PreOrden = ConPreorden.

Para la Segunda consulta se llama a la clase “Consultas” al método “cantidadniveles”

Que hace un recorrido por niveles, sumando a variables como por ejemplo “nivel1”, cada vez que la Población del Planeta (arbolAux.getDatoRaiz().population) sea mayor a 10.

La Tercera consulta, es similar a la segunda, pero con la variable que guarda la cantidad de población de un Nivel, y la cantidad de Planetas por otro. Al final se realiza la operación de dividir la Suma Total de planetas, con la Cantidad de Planetas, por niveles.

Para el método “CalcularMovimiento” se realiza la estrategia de chequear a quién pertenece la “raíz”.  
Si es un planeta Neutral, se llama a la clase “CuandoEsNeutral” que realiza un recorrido PreOrden, y guarda en una Lista, el recorrido desde la Raíz hasta el Bot o Planeta de la IA.  
Una vez con el recorrido, para calcular el movimiento, se establece como “Planeta Origen” el ultimo planeta, después se elimina de la lista con un Remove, y se establece el planeta que queda, esta vez último, como el “Planeta Destino”.

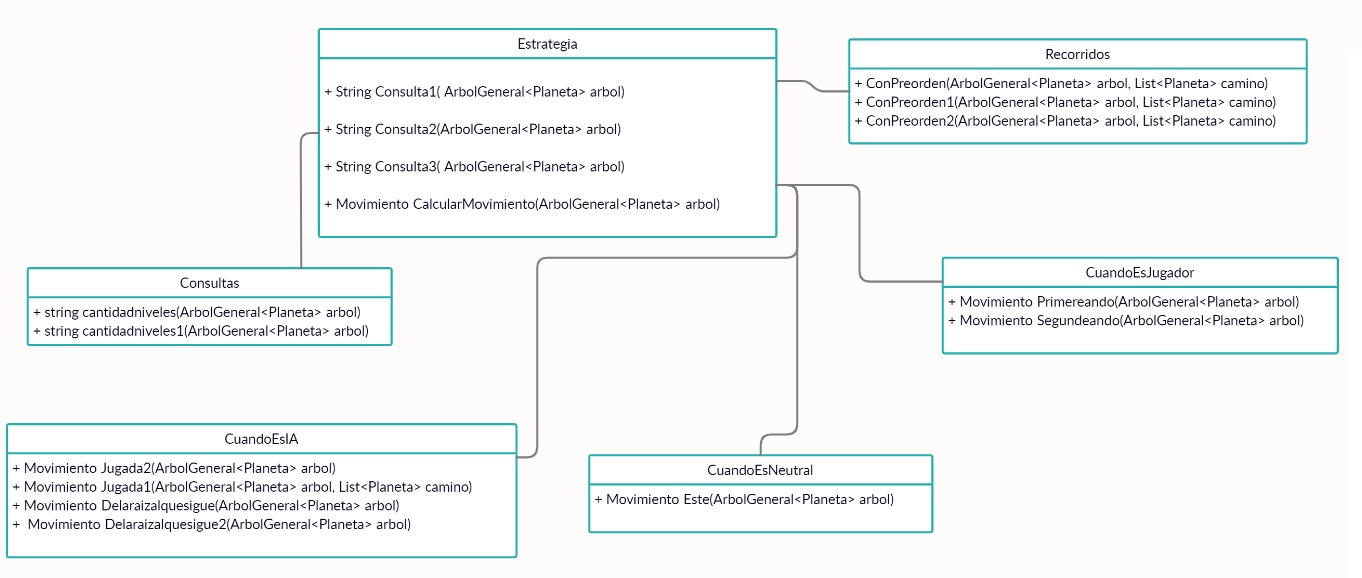
Cuando la Raíz es del jugador, se realiza el mismo procedimiento que cuando es planeta neutral.

Cuando la Raíz es conquistada por el Bot, Pasa a tener una estrategia de ataque, formando caminos, uno para encontrar al Jugador, y otro para enviarles tropas, desde el planeta más cercano a la Raíz. La estrategia consiste en fortalecer la Raíz, aunque ya haya sido conquistada.

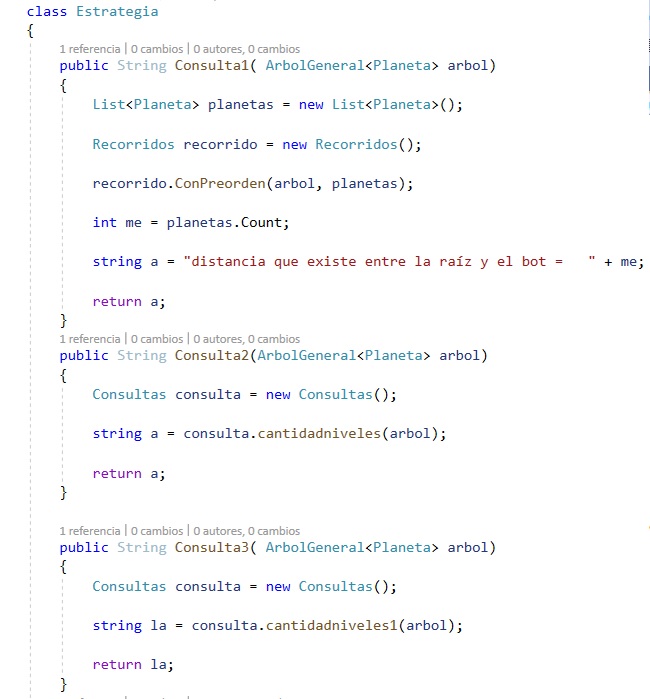
**Problemas encontrados**

El principal problema que encontré fue que el método que realiza el movimiento, se realiza una sola vez, entonces cuando se realizaba el movimiento de planeta origen al planeta destino, se hacía por única vez.   
Lo solucioné con un “While” y con un atributo o variable “Random”, que hace que pueda tomar, un camino u otro, en una velocidad tan rápida que es difícil verla a simple vista.

El Random permitió tomar dos caminos y estrategia, en el mismo método que calcula el movimiento, es decir, mientras el Bot ataca, también envía planetas a la Raíz, para fortalecerse.

**Diagrama de UML :**

**Imagen de las clases :**

****

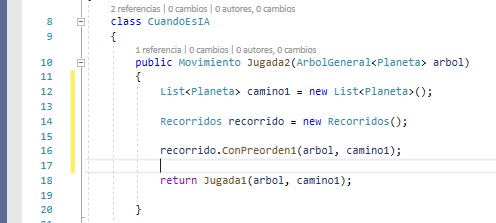
Las consultas, llaman a la clase “Consultas” que van a aparecer abajo.

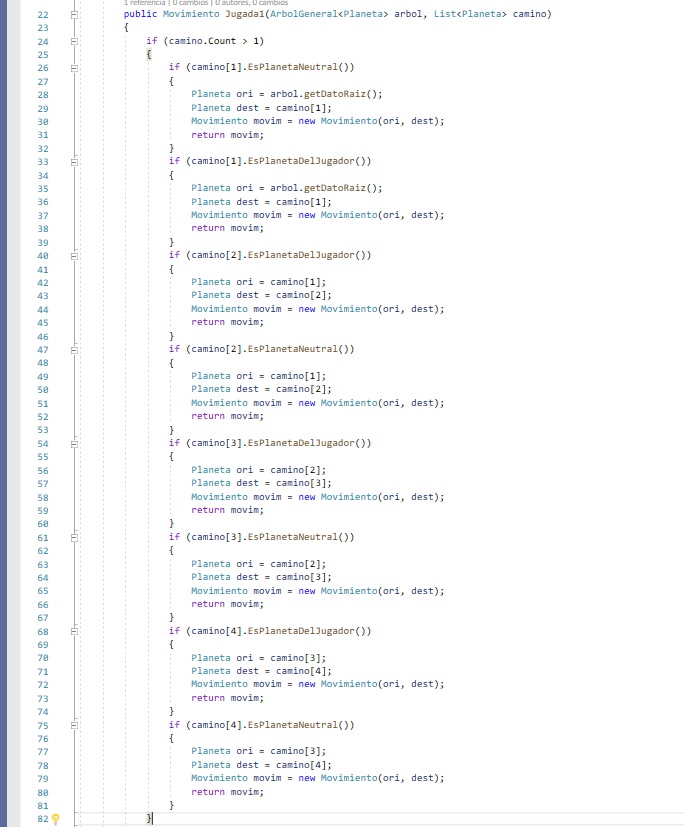
****

Para la consulta 3, es similar a la consulta 2, guardando esta vez otra variable, con la suma de la población de planetas, para dividirlo después por el total de planetas por nivel en el ultimo String que retorna el método.

Del Método CalcularMovimiento, y sus respectivas clases asociadas, dependiendo si la Raíz es “Neutral”, “de la IA” o “del Jugador”.

****

****

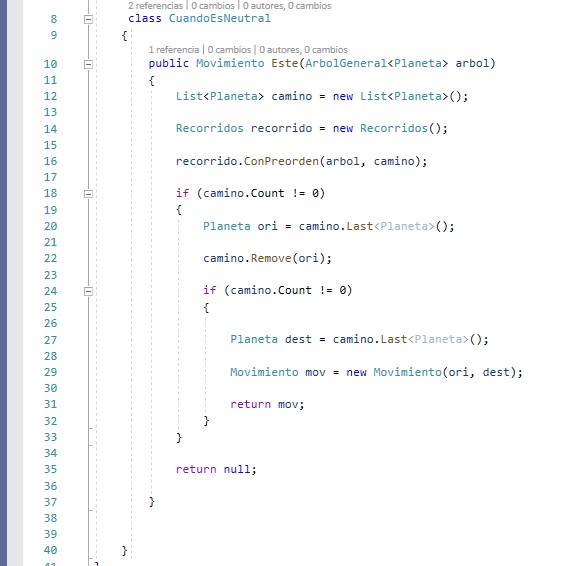
****

****

Hasta acá llega la clase que se llama cuando es “planeta de la IA”, que contiene el método que manda su población a la Raíz, además de Atacar.

Los siguientes movimientos son más sencillos. Realiza un camino y elimina el siempre el ultimo, para saber cuál es el origen y cual es el destino.

****

****

La sugerencia que se puede hacer para mejorarlo, es realizar muchos caminos, con grafos, y guardar un camino anterior, para que se ejecute, para que vayan avanzando todos los planetas en conjunto y no se queden algunos quietos solamente sumando población, aunque sirve que se queden sumando, porque cuando se lo ataca con poca tripulación, este avanza el doble, o el tripe gracias a su población acumulada.

**Una breve conclusión o reflexión de la experiencia adquirida a partir de la realización del trabajo final.**

La conclusión no es del “T.P Final” sino de la materia y de la cursada en sí.  
Se aprendió muchísimo en cuanto a conceptos de estructuras con una claridad a la hora de las explicaciones, que no había notado en ninguna otra materia. Los contenidos de estructuras de datos, se me ocurre que se pueden aplicar a distintas funcionalidades con otros objetivos. En la materia además de abordar los contenidos propios de la materia, también se aprendió a analizar el código, a mirarlo y ver parte por parte cuando ocurren errores, que es fundamental para buscar por nosotros mismos, e incluso buscar definiciones de cómo funcionan palabras, métodos, palabras reservadas o de cosas que no están dentro de la materia pero que sirven un montón para entender como se programa, y como se busca soluciones reales a los problemas de la programación.

Agradezco enormemente la clara explicación del profesor [Leonardo Javier AMET](https://campus.unaj.edu.ar/user/view.php?id=4737&course=2875) en todos los temas de la materia, y por la enseñanza de como analizar, mejorar y solucionar problemas de código.