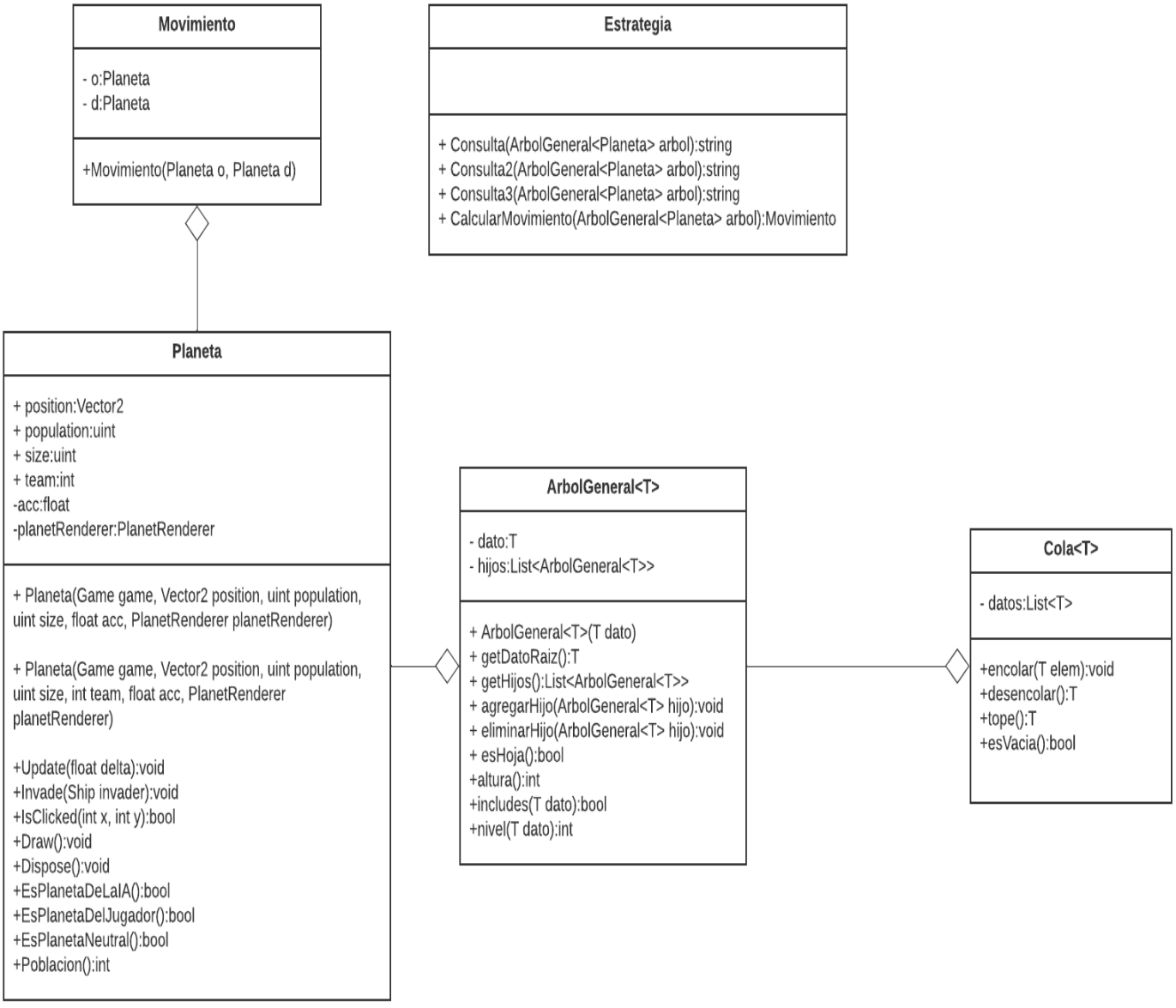
TRABAJO PRÁCTICO FINAL

- MATERIA: COMPLEJIDAD TEMPORAL, ESTRUCTURAS DE DATOS Y ALGORITMOS
- ALUMNO: CLAUDIO BERNAL
- COMISIÓN N°3
- DESCRIPCIÓN BREVE: EN ESTE TRABAJO SE ME PIDE IMPLEMENTAR LA ESTRATEGIA DEL BOT EN UN VIDEOJUEGO UTILIZANDO LO APRENDIDO DURANTE LA CURSADA.

Diagrama de clases UML

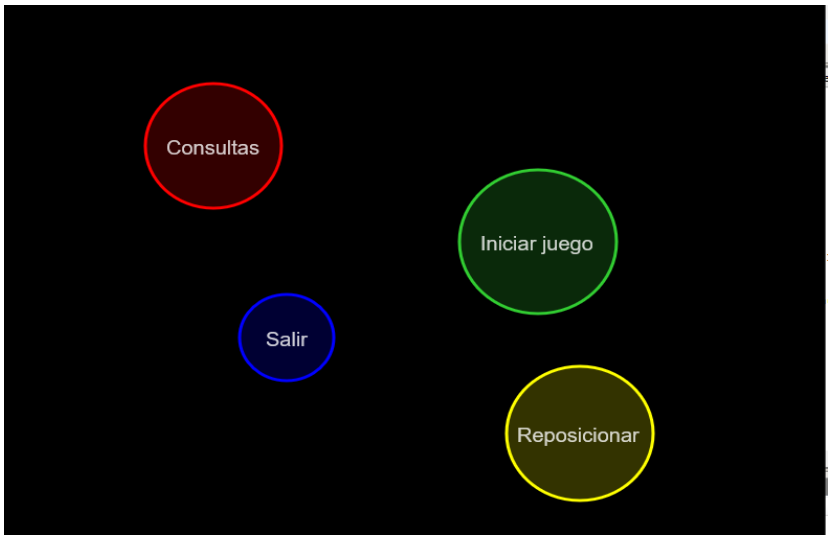


Problemas encontrados y su solución

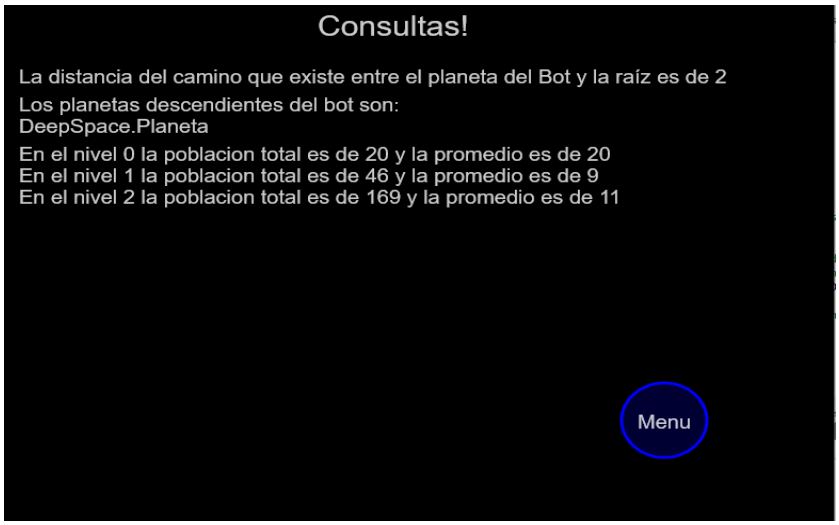
En el método "consulta2()" cuando el planeta del bot era una hoja me devolvía solo el texto "Los planetas descendientes del bot son: ", entonces cuando evalúa si el árbol auxiliar es planeta de la ia dentro le agregue otro condicional, para que retorne otro texto en caso de que este árbol sea hoja.

En el método "consulta3()" solo me devolvía la población total y el promedio del nivel 0, 1 y 2. Esto era porque, una vez que terminaba de recorrer el nivel 3, al hacer la comparación de si el nivel del arbol auxiliar es igual al nivel en el que estoy evaluando los resultados, recién ahí, se guardaba el resultado en la variable que posteriormente se retornará. Pero no entraba ya que la condición no se cumplía porque lo último que se encolo fue null, porque no hay nivel 4 en el arbol. Entonces, lo solucione agregándole a la condición de que, si arbolAuxiliar es null, también entre en el condicional if.

Interfaz gráfica:



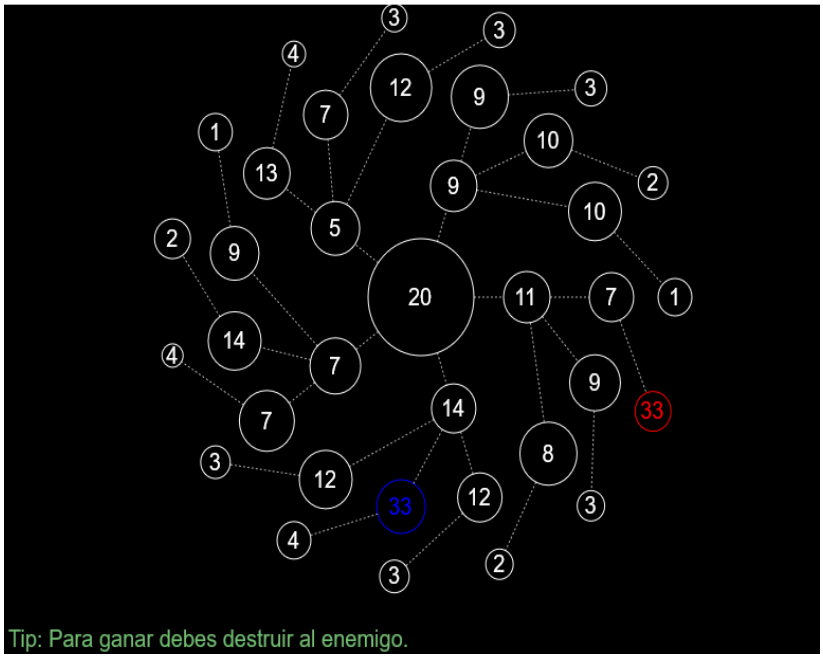
Este es el menú principal, donde se pueden seleccionar las opciones: "consultas", "iniciar juego", "reposicionar" y "salir".



Esto es lo que se muestra al seleccionar la opción "Consultas" del menú principal.

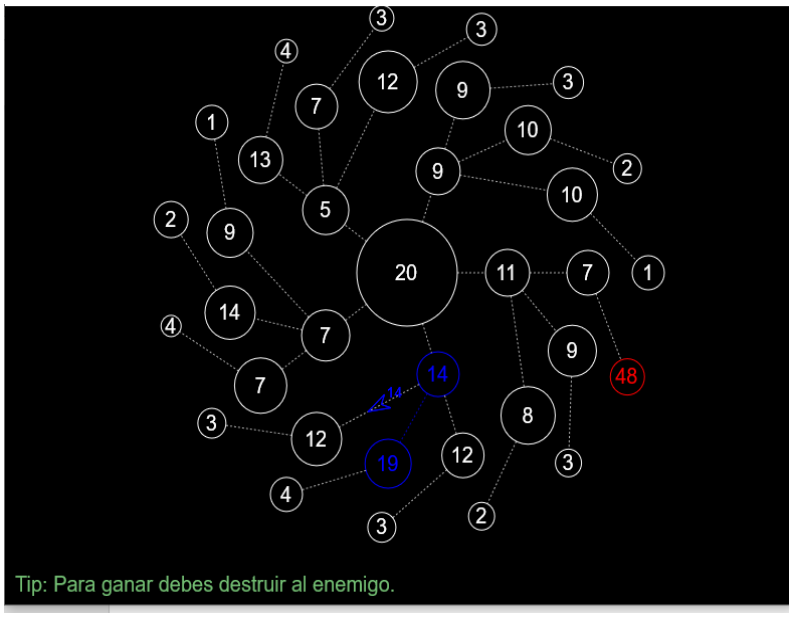
Estos son los strings que retornan los métodos "Consulta1()", que es la distancia del camino entre el planeta del bot y la raíz, "Consulta2(), los planetas descendientes del bot, y "Consulta3()" la población total y la promedio por cada nivel del arbol.

Esto se muestra al elegir la opción “Reposicionar” que lo que hace, como lo dice en pantalla, es reposicionar aleatoriamente el planeta del bot y el del jugador.



Así es como se ve el juego, se inicia al seleccionar la opción “iniciar juego”.

El jugador puede conquistar los planetas (nodos) que sean hijos o del nodo que sea hijo el planeta del jugador. Esto lo hace clickeando su planeta(nodo rojo) y el planeta que desea conquistar. En este caso, solamente puede conquistar el planeta con población de 7.



Como puede apreciarse, el bot(nodo azul), que tenía una población de 33, conquistó el planeta neutral que tenia población de 14 y, este envió sus tropas a el que tiene una población de 12.

Esto funciona gracias al método "CalcularMovimiento()" de la clase "Estrategia". Que lo que hace, es retornar una instancia de la clase "Movimiento", si la población del bot es mayor o igual al doble de la población de los planetas adyacentes.

Posibles mejoras

Este videojuego podría mejorarse si se divide en niveles, aumentando la dificultad a medida que van pasando los niveles, de modo que, por ejemplo, por cada nivel se aumente la velocidad a la que ataca a otros planetas o que ataque a mas de uno a la vez. También podría mejorarse estéticamente, en vez de que se muestre una flechita moviéndose de un nodo a otro podría mostrarse alguna animación con naves, y en vez de que los planetas sean circunferencias, podrían verse como planetas. Además, podría ser agregada una banda sonora diferente por cada nivel.

Conclusión

Con este trabajo pude afianzar varios conceptos de la materia, entre ellos: el uso del TAD "cola" y del árbol general para realizar los métodos de la clase estrategia. Además, también me sirvió para repasar lo aprendido anteriormente en la carrera como, por ejemplo, el uso de clases, objetos, recursión, entre otros temas. En fin, este Trabajo y la cursada me ayudó mucho a repasar bastantes temas y también a utilizar el razonamiento lógico para la resolución de los problemas de código.