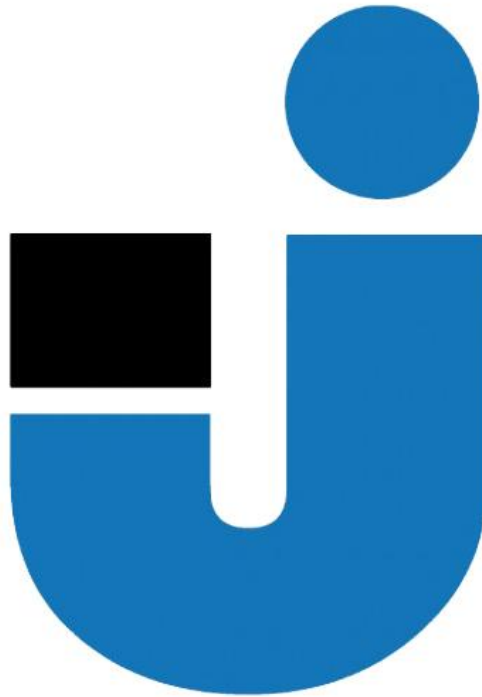


Informe:



Universidad Nacional
ARTURO JAURETCHE

Integrantes:

- Lucas Tortolini.

Profesor: Leonardo Javier Amet .

Materia: Complejidad Temporal, Estructura de Datos y Algoritmo.

Comisión: 5

Índice:

Introducción:.....	3
Desarrollo.....	4
Funcionamiento del juego:.....	9
Dificultades y soluciones:	13
Oportunidades de mejora:	13
Conclusión:.....	13

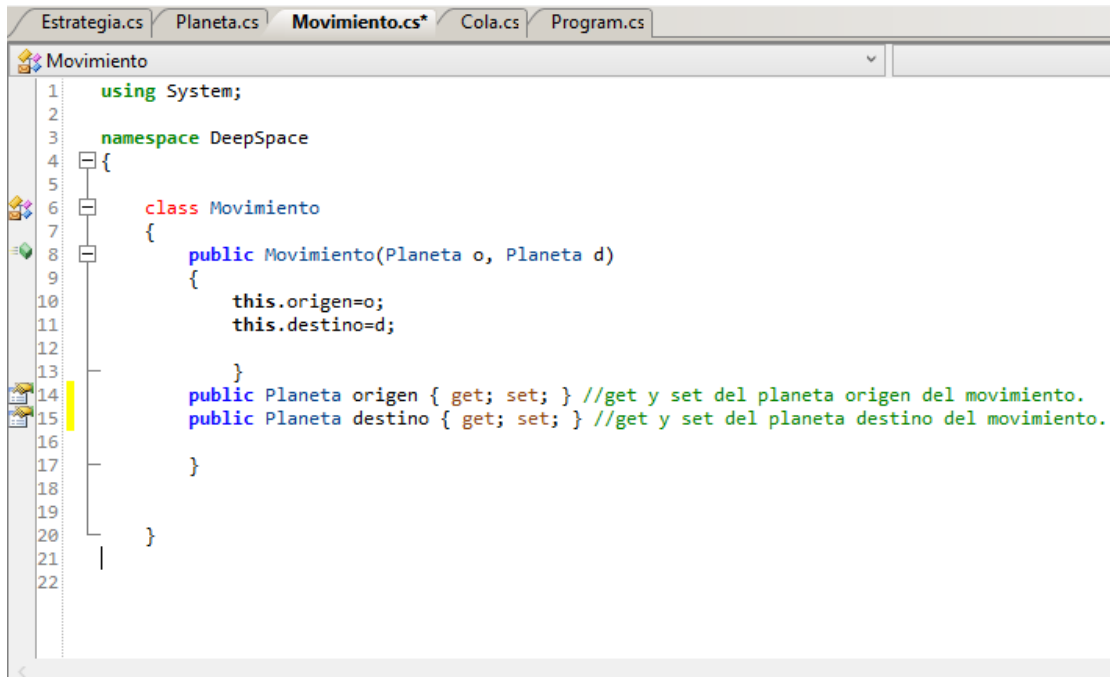
Introducción:

Para este Trabajo Practico Final lo que voy a hacer es un juego de conquista planetas, objetivo derrotar al enemigo. Y para llevar a cabo este juego tenemos que llenar un par de líneas de código.

Desarrollo:

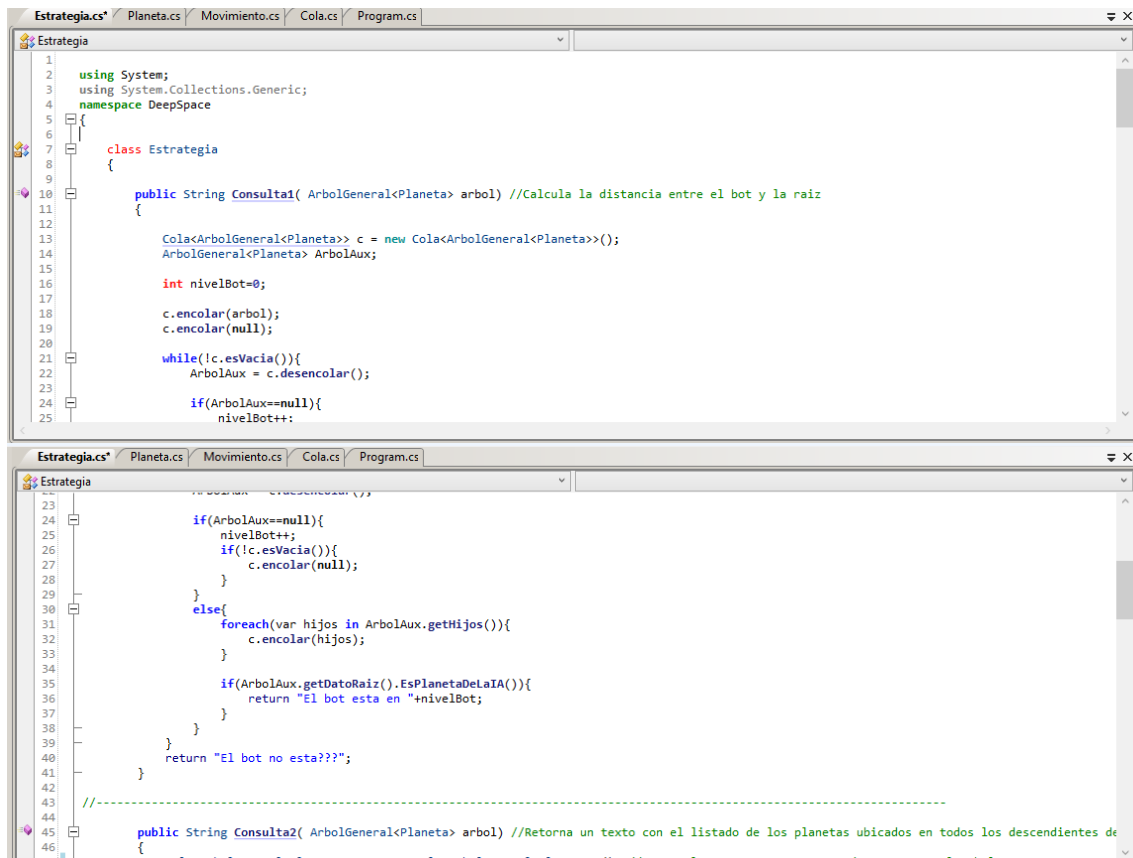
Para empezar lo que hice fue lo siguiente:

Primero hice el constructor de la clase de Movimiento.



```
1 using System;
2
3 namespace DeepSpace
4 {
5
6     class Movimiento
7     {
8         public Movimiento(Planeta o, Planeta d)
9         {
10             this.origen=o;
11             this.destino=d;
12         }
13
14         public Planeta origen { get; set; } //get y set del planeta origen del movimiento.
15         public Planeta destino { get; set; } //get y set del planeta destino del movimiento.
16     }
17
18 }
19
20
21
22
```

Luego estuve haciendo las 3 consultas dentro de la clase de Estrategia,



```
1 using System;
2 using System.Collections.Generic;
3 namespace DeepSpace
4 {
5
6     class Estrategia
7     {
8
9         public String Consultar1( ArbolGeneral<Planeta> arbol) //Calcula la distancia entre el bot y la raiz
10         {
11
12             Cola<ArbolGeneral<Planeta>> c = new Cola<ArbolGeneral<Planeta>>();
13             ArbolGeneral<Planeta> ArbolAux;
14
15             int nivelBot=0;
16
17             c.encolar(arbol);
18             c.encolar(null);
19
20             while(!c.esVacia()){
21                 ArbolAux = c.desencolar();
22                 if(ArbolAux==null){
23                     nivelBot++;
24                 }
25             }
26
27             if(ArbolAux==null){
28                 nivelBot++;
29                 if(!c.esVacia()){
30                     c.encolar(null);
31                 }
32             }
33             else{
34                 foreach(var hijos in ArbolAux.getHijos()){
35                     c.encolar(hijos);
36                 }
37             }
38             if(ArbolAux.getDatoRaiz().EsPlanetaDeLaIA()){
39                 return "El bot esta en "+nivelBot;
40             }
41             return "El bot no esta???";
42         }
43
44         //-----
45         public String Consultar2( ArbolGeneral<Planeta> arbol) //Retorna un texto con el listado de los planetas ubicados en todos los descendientes de
46         {
47             Cola<ArbolGeneral<Planeta>> c = new Cola<ArbolGeneral<Planeta>>();
48             ArbolGeneral<Planeta> ArbolAux;
49
50             c.encolar(arbol);
51             c.encolar(null);
52
53             while(!c.esVacia()){
54                 ArbolAux = c.desencolar();
55                 if(ArbolAux==null){
56                     nivelBot++;
57                 }
58                 if(ArbolAux==null){
59                     nivelBot++;
60                     if(!c.esVacia()){
61                         c.encolar(null);
62                     }
63                 }
64                 else{
65                     foreach(var hijos in ArbolAux.getHijos()){
66                         c.encolar(hijos);
67                     }
68                 }
69                 if(ArbolAux.getDatoRaiz().EsPlanetaDeLaIA()){
70                     return "El bot esta en "+nivelBot;
71                 }
72             }
73             return "El bot no esta???";
74         }
75     }
76 }
```

```

Estrategia.cs | Planeta.cs | Movimiento.cs | Cola.cs | Program.cs
Estrategia
//-----
43
44
45 public String Consulta2( ArbolGeneral<Planeta> arbol)
46 {
47     //Retorna un texto con el listado de los planetas ubicados en todos los descendientes del nodo que contiene al planeta del Bot.
48
49     Cola<ArbolGeneral<Planeta>> c = new Cola<ArbolGeneral<Planeta>>(); //esta cola se va a usar para poder procesar el arbol
50     Cola<ArbolGeneral<Planeta>> cr = new Cola<ArbolGeneral<Planeta>>(); //esta cola es la que tiene el resultado "Cola Return"
51     ArbolGeneral<Planeta> ArbolAux;
52
53     c.encolar(arbol);
54
55     while(!c.esVacia()){
56         ArbolAux = c.desencolar();
57
58         if(ArbolAux.getDatosRaiz().EsPlanetaDeLaIA()){//este if se usa para buscar el bot
59             while(!c.esVacia()){ //cuando encuentro al bot, vacio la cola para poder usarla para procesar
60                 c.desencolar();
61             }
62
63             //ahora tengo que encolar los hijos de esos hijos, y repetirlo hasta que sean hojas
64             while(!ArbolAux.esHoja()){//se ejecuta hasta que no tenga descendientes
65
66                 foreach(var hijos in ArbolAux.getHijos()){
67                     cr.encolar(hijos);
68                     c.encolar(hijos);
69                 }
70                 ArbolAux = c.desencolar();
71             }
72         }
73         else{
74             foreach(var hijos in ArbolAux.getHijos())
75                 c.encolar(hijos);
76         }
77     }
78
79     string texto = "Los planetas descendientes del bot son: \n";
80     while(!cr.esVacia()){
81         ArbolAux = cr.desencolar();
82         texto = texto+ArbolAux.getDatosRaiz().Poblacion().ToString()+"-";
83     }
84     texto = texto + "\n";
85     return texto;
86 }
87
88 //-----
89 public String Consulta3( ArbolGeneral<Planeta> arbol)
90 {
91     Cola<ArbolGeneral<Planeta>> c = new Cola<ArbolGeneral<Planeta>>();
92     ArbolGeneral<Planeta> ArbolAux;
93
94     int nivelBot=0;
95     int[] poblacionXNivel = {0,0,0,0};
96     int[] promedioXNivel = {0,0,0,0};
97
98     int cantPlanetasXNivel=0;
99     c.encolar(arbol);
100     c.encolar(null);
101
102     while(!c.esVacia()){
103         ArbolAux = c.desencolar();
104
105         if(ArbolAux==null){
106             nivelBot++;
107             cantPlanetasXNivel=0;
108             if(!c.esVacia()){
109                 c.encolar(null);
110             }
111         }
112     }
113
114     if(!c.esVacia()){
115         c.encolar(null);
116     }
117     else{
118         foreach(var hijos in ArbolAux.getHijos()){
119             c.encolar(hijos);
120         }
121         cantPlanetasXNivel++;
122         poblacionXNivel[nivelBot]=poblacionXNivel[nivelBot]+ArbolAux.getDatosRaiz().Poblacion();
123         promedioXNivel[nivelBot]=poblacionXNivel[nivelBot]/cantPlanetasXNivel;
124     }
125
126     int poblacionTotal=0;
127     int niv=0;
128     while(niv<=poblacionXNivel.Length-1){//hardco
129         poblacionTotal+=poblacionXNivel[niv];
130         niv++;
131     }
132
133     return "\n\nLa poblacion total es: "+poblacionTotal+"\nEl promedio del primer nivel es: "+promedioXNivel[1]+" total: "+poblacionXNivel[1]+
134 }
135
136 //-----

```

```
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129 "+poblacionXNivel[1]+"\nEl promedio del segundo nivel es: "+promedioXNivel[2]+" total: "+poblacionXNivel[2]+"\nEl promedio del tercer nivel es: "+pro
130
131
132
133

Estrategia.cs | Planeta.cs | Movimiento.cs | Cola.cs | Program.cs | Consulta2(ArbolGeneral<Planeta> arbol)

109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129 do nivel es: "+promedioXNivel[2]+" total: "+poblacionXNivel[2]+"\nEl promedio del tercer nivel es: "+promedioXNivel[3]+" total: "+poblacionXNivel[3];
130
131
132
133

Estrategia.cs | Planeta.cs | Movimiento.cs | Cola.cs | Program.cs | Consulta2(ArbolGeneral<Planeta> arbol)

119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
    promedioXNivel[nivelBot]=poblacionXNivel[nivelBot]/cantPlanetasXNivel;
    }
    }
    int poblacionTotal=0;
    int niv=0;
    while(niv<=poblacionXNivel.Length-1){//hardco
        poblacionTotal+=poblacionXNivel[niv];
        niv++;
    }
    return "\n\nLa poblacion total es: "+poblacionTotal+"\nEl promedio del primer nivel es: "+promedioXNivel[1]+" total: "+poblacionXNivel[1];
}

//-----

public Movimiento CalcularMovimiento(ArbolGeneral<Planeta> arbol)
{
    return null;
}
}
```

Después tuve que hacer el método `CalcularMovimiento` para que tanto el jugador como el bot se puedan mover.

```

Estrategia.cs Program.cs
Estrategia
137
138 public Movimiento CalcularMovimiento(ArbolGeneral<Planeta> arbol)
139 {
140     List<ArbolGeneral<Planeta>> bot = new List<ArbolGeneral<Planeta>>(); //lista de planetas del bot
141     List<ArbolGeneral<Planeta>> player = new List<ArbolGeneral<Planeta>>(); //lista de planetas del jugador
142
143     __caminoBot(arbol, arbol, bot);
144     __caminoJugador(arbol, arbol, player);
145
146     if(!bot[0].getDatoRaiz().EsPlanetaDeLaIA()){
147
148         Movimiento movimiento = new Movimiento(bot[bot.Count-1].getDatoRaiz(), bot[bot.Count-2].getDatoRaiz());
149
150         return movimiento;
151     }
152     int contador = 0;
153     if(player[0].getDatoRaiz().EsPlanetaDeLaIA()){ //si la raiz es de la IA
154         while(contador < player.Count){
155             if(player[contador].getDatoRaiz().EsPlanetaDeLaIA() && !player[contador+1].getDatoRaiz().EsPlanetaDeLaIA()){
156                 //si el planeta del contador es de la ia y el siguiente no es de la IA
157
158                 Movimiento movimiento = new Movimiento(player[contador].getDatoRaiz(), player[contador+1].getDatoRaiz());
159
160                 return movimiento;
161             }
162             else{
163                 contador++;
164             }
165         }
166     }
167     return null;
168 }
169
170

```

```

Estrategia.cs Program.cs
Estrategia
144 __caminoJugador(arbol, arbol, player);
145
146 if(!bot[0].getDatoRaiz().EsPlanetaDeLaIA()){
147
148     Movimiento movimiento = new Movimiento(bot[bot.Count-1].getDatoRaiz(), bot[bot.Count-2].getDatoRaiz());
149
150     return movimiento;
151 }
152 int contador = 0;
153 if(player[0].getDatoRaiz().EsPlanetaDeLaIA()){ //si la raiz es de la IA
154     while(contador < player.Count){
155         if(player[contador].getDatoRaiz().EsPlanetaDeLaIA() && !player[contador+1].getDatoRaiz().EsPlanetaDeLaIA()){
156             //si el planeta del contador es de la ia y el siguiente no es de la IA
157
158             Movimiento movimiento = new Movimiento(player[contador].getDatoRaiz(), player[contador+1].getDatoRaiz());
159
160             return movimiento;
161         }
162         else{
163             contador++;
164         }
165     }
166 }
167 return null;
168 }
169
170

```

Y por ultimo estos códigos lo que hace es preguntar si ese nodo pertenece a la IA o al jugador, si es verdadero pone al caminoHallado verdadero, si no salta y recorre todos los hijos.

```

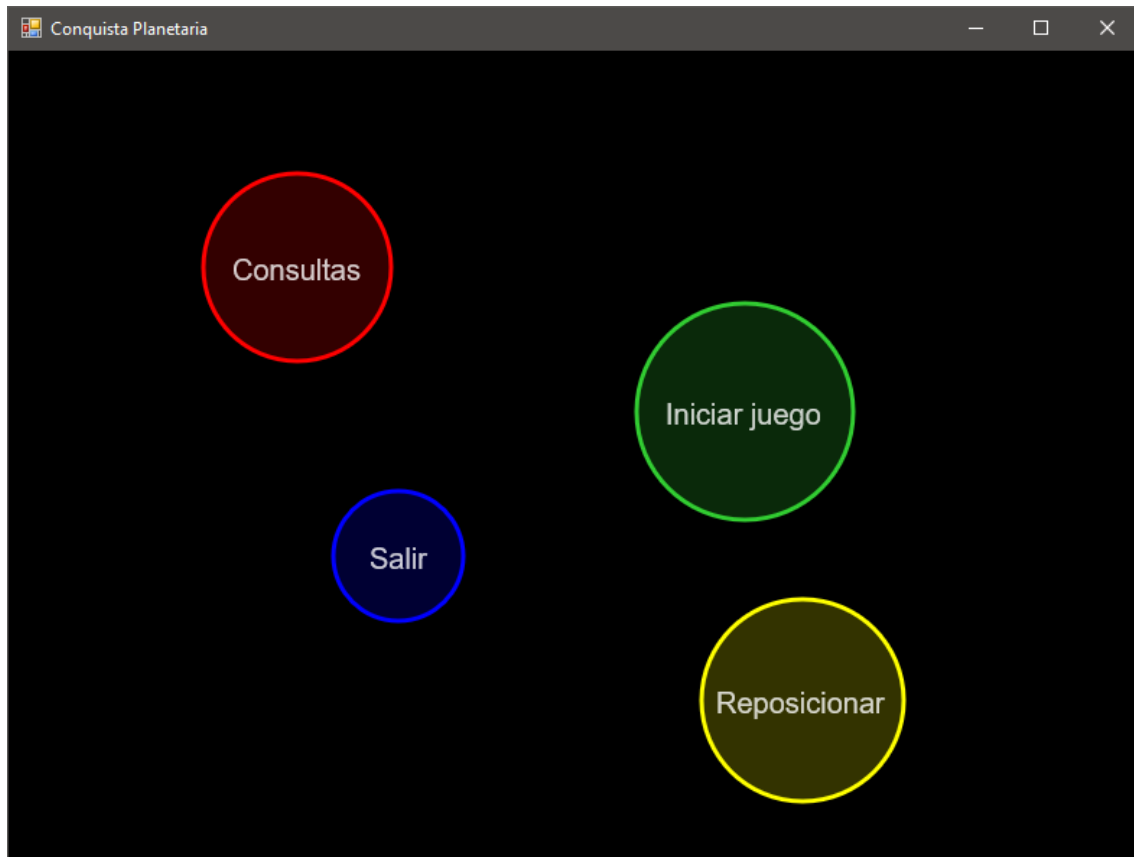
Estrategia.cs Program.cs
Estrategia
174
175 private bool __caminoBot(ArbolGeneral<Planeta> arbol, ArbolGeneral<Planeta> origen, List<ArbolGeneral<Planeta>> camino){
176     bool caminoHallado = false;
177
178     camino.Add(origen);
179
180
181     if(origen.getDatoRaiz().EsPlanetaDeLaIA()){
182         caminoHallado = true;
183     }
184
185     else{
186         foreach(var hijo in origen.getHijos()){
187             caminoHallado = __caminoBot(arbol, hijo, camino);
188
189             if(caminoHallado){
190                 break;
191             }
192
193             camino.RemoveAt(camino.Count - 1);
194         }
195     }
196     return caminoHallado;
197 }
198
199

```

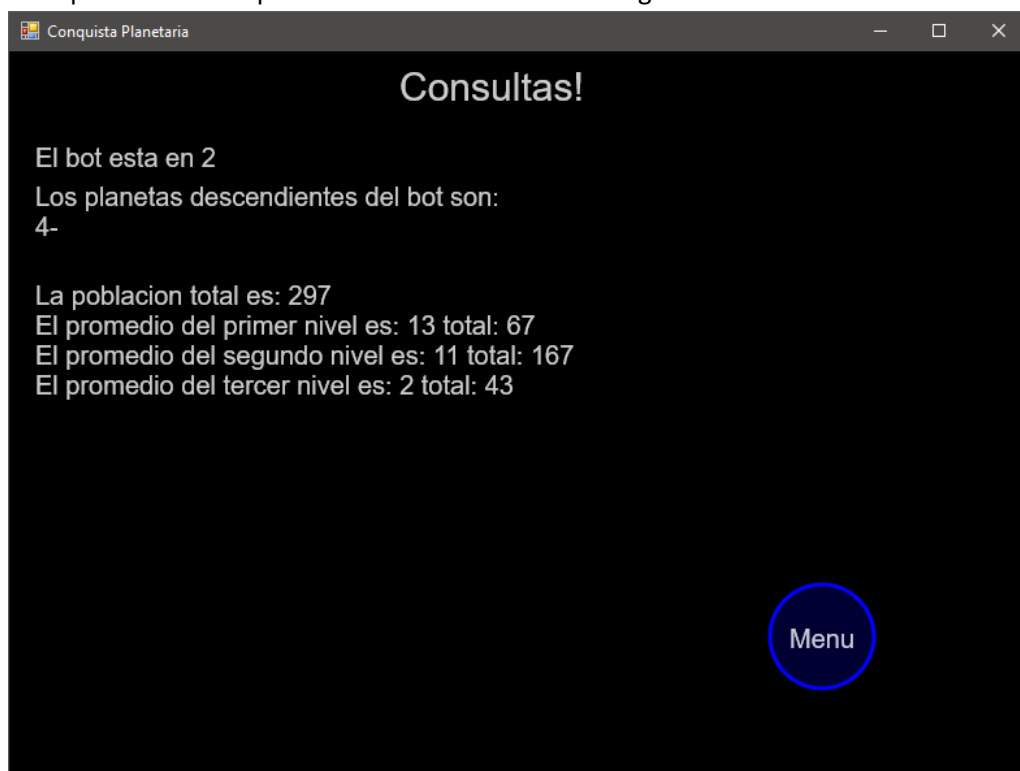
```
Estrategia.cs Program.cs
199
200 private bool __caminoJugador(ArbolGeneral<Planeta> arbol, ArbolGeneral<Planeta> origen, List<ArbolGeneral<Planeta>> camino){
201     bool caminoHallado = false;
202
203     camino.Add(origen);
204
205     if(origen.getDatoRaiz().EsPlanetaDelJugador()){
206         caminoHallado=true;
207     }
208
209     else{
210         foreach(var hijo in origen.getHijos()){
211             caminoHallado = __caminoJugador(arbol,hijo,camino);
212
213             if(caminoHallado){
214                 break;
215             }
216
217             camino.RemoveAt(camino.Count - 1);
218         }
219     }
220     return caminoHallado;
221 }
222
223
224
225 }
```


Funcionamiento del juego:

El programa funciona con un menú principal donde hay 4 opciones "Iniciar Juego", "Consultas", "Reposicionar" y "Salir".

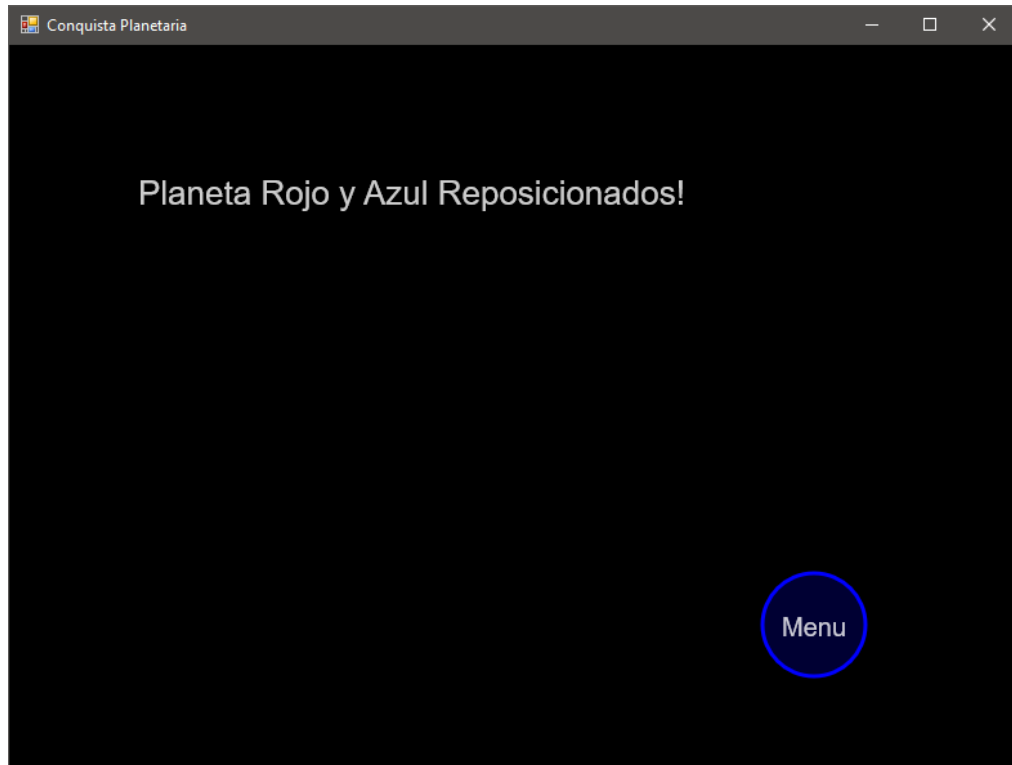


1. Si cliqueamos en la opción de "Consultas" saldrá lo siguiente:



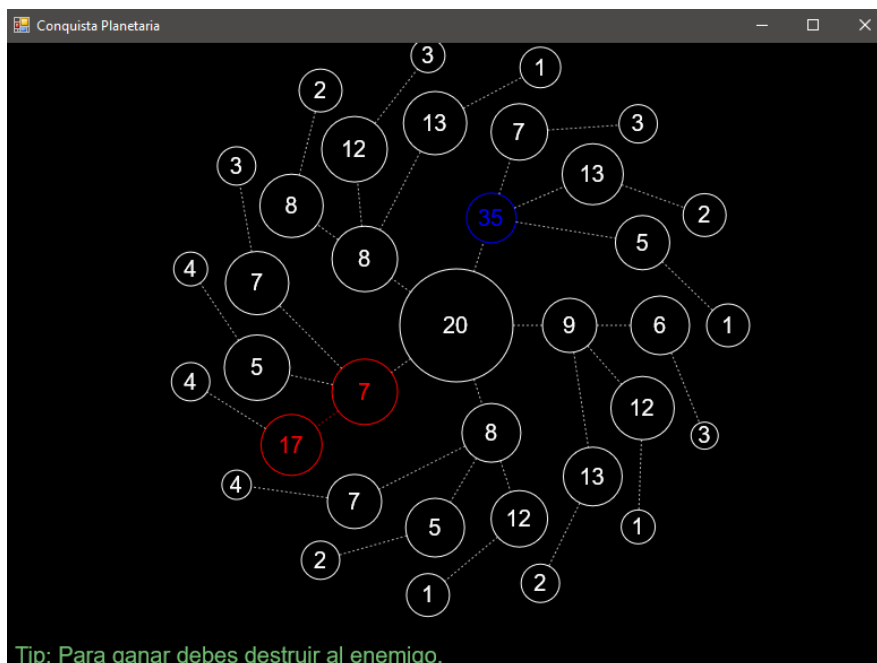
-Y para volver al menú principal cliqueamos en “Menú”.

2. Si cliqueamos en la opción de “Reposicionar” saldrá lo siguiente:



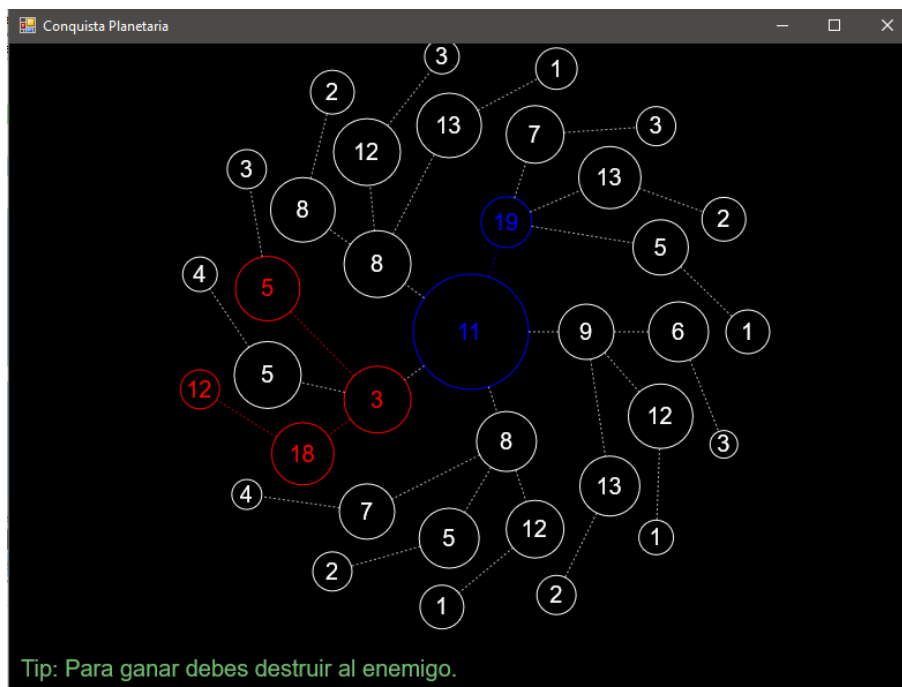
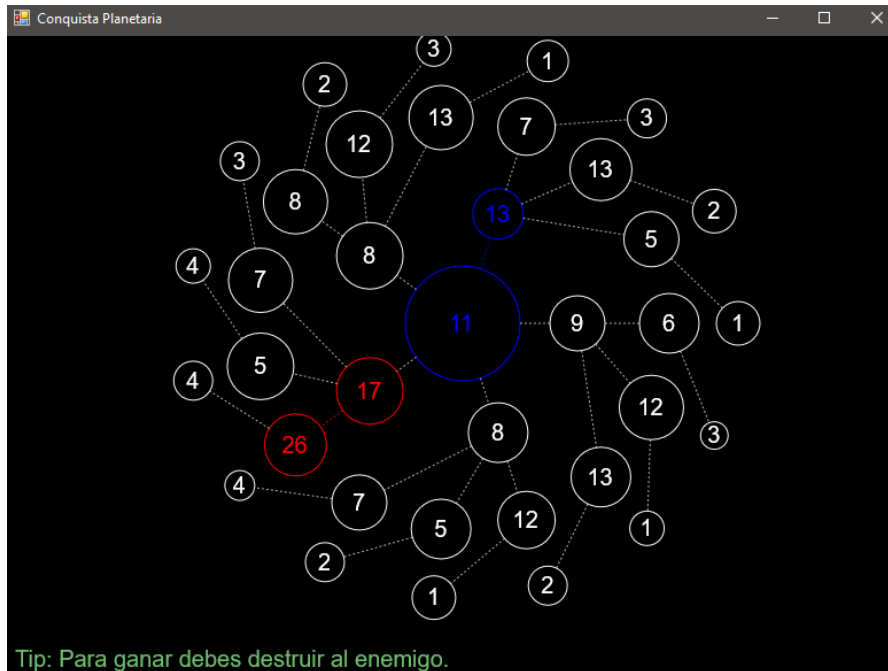
-Y para volver al menú principal cliqueamos en “Menú”.

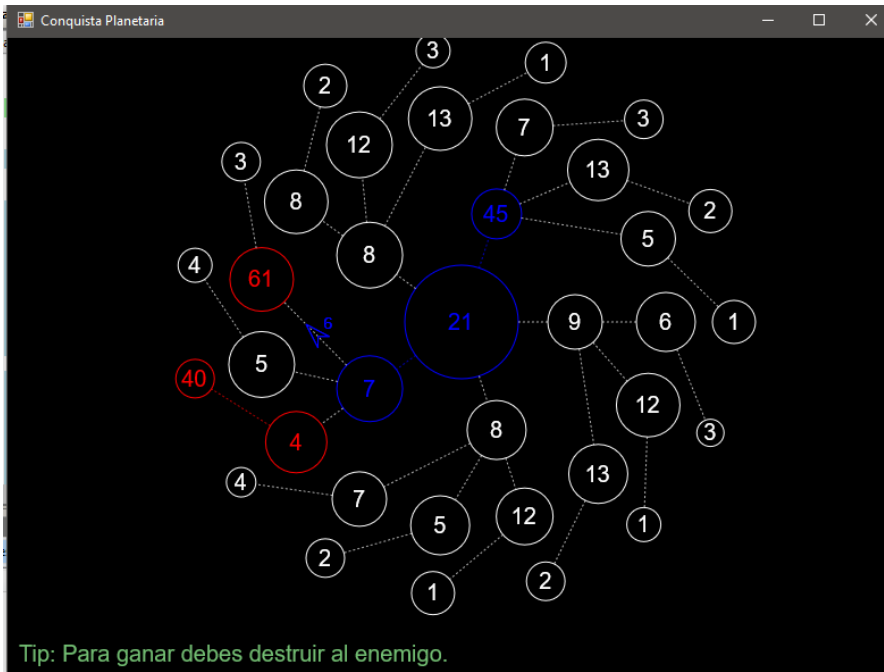
3. Si cliqueamos en la opción de “Salir” finalizara el programa.
4. Si cliqueamos en la opción de “Iniciar Juego” empezara el juego.



Nota: El bot es el azul y el Jugador el rojo.

Lo movimientos tanto del jugador como el del bot van en valor medio y al planeta que decidas mover se le restara esa mitad, es decir si un planeta tiene el valor 40 y quiere pasarse a otro planeta (conquistado o no conquistado) se moverá con un valor 20 y el planeta que selecciono para moverse tendrá el valor 20 y tanto el jugador como el bot se le suma 1 punto por cada segundo. Si decides conquistar el planeta con el valor 20 y mandas a que ataque con 15, el planeta atacado tendrá el valor 5 pero seguirá neutro.





Dificultades y soluciones:

La dificultad que tuve fue el de hacer que el bot se mueva y ataque al jugador. Pero lo pude solucionar, viendo videos en Youtube.

Oportunidades de mejora:

Se puede seguir mejorando el código, pero por falta de tiempo no llegó.

Conclusión:

Llegue a la conclusión de se puede seguir haciendo más pruebas, en hacer mas modificaciones al programa para mejorar el funcionamiento.