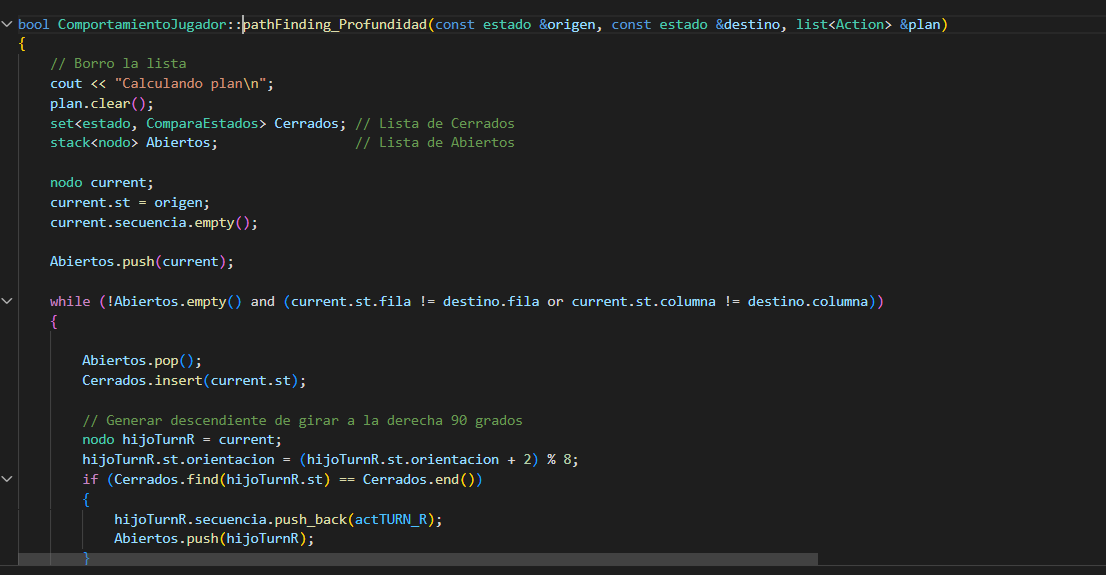
Memoria Practica 2 IA:  
David Martínez Díaz GII-ADE

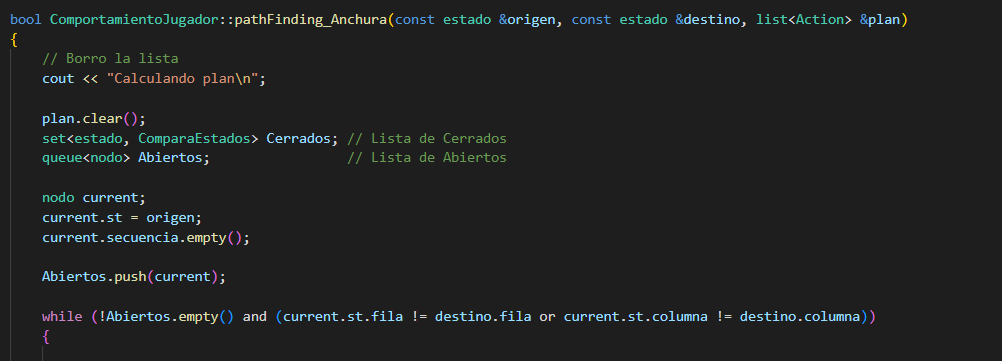
En primer lugar, realizamos en clase el tutorial básico, con esta demostración conseguí adquirir los conocimientos básicos del funcionamiento del path-finding, a través del método de profundidad.

No había que implementar nada en ese nivel:



Lo único importante de este método es la utilización de una pila para los nodos abiertos.

En el nivel 1, había que implementar el path-finding de anchura, donde lo único que había que cambiar era la pila de abiertos por una cola:

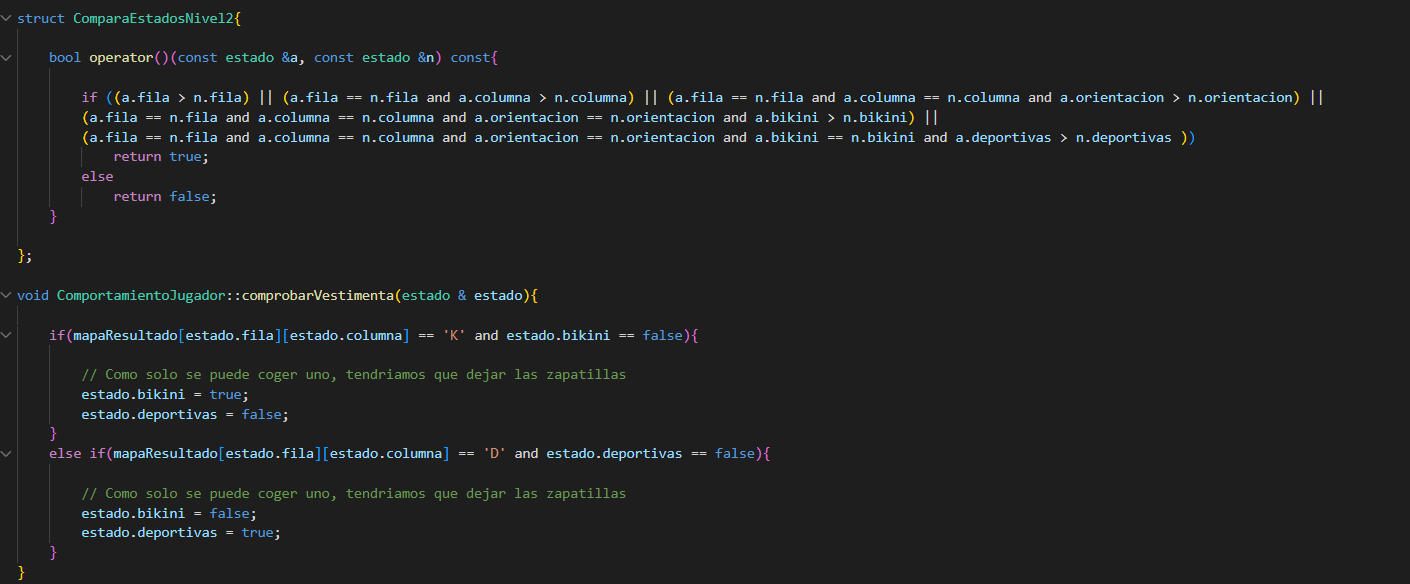


En cuanto al nivel 2, la cosa se empieza a complicar ya que en un primer lugar decidir hacer el Coste Uniforme:

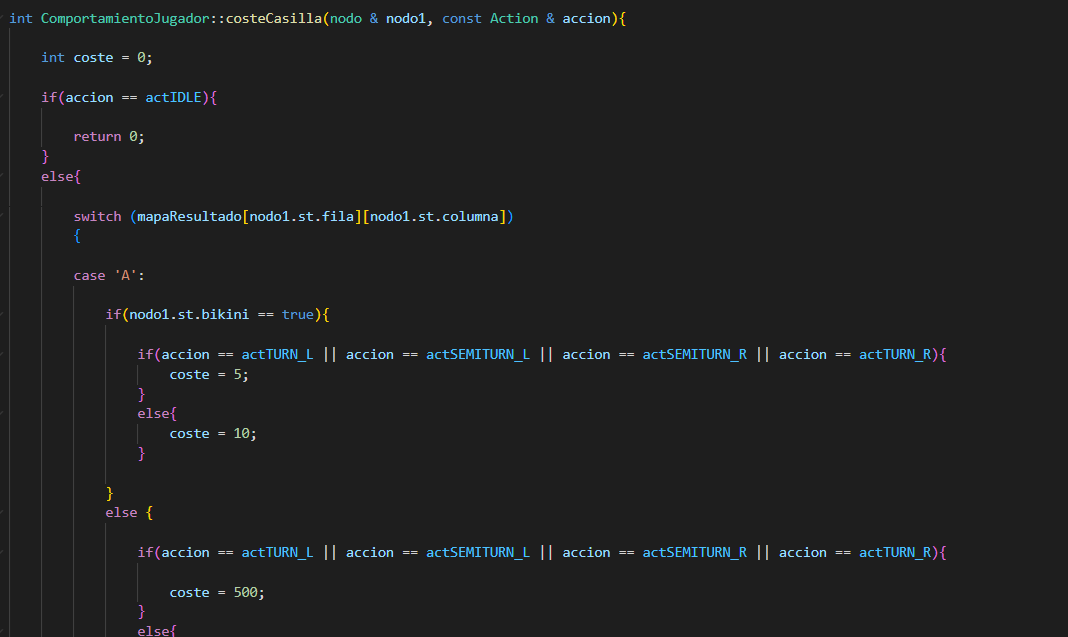


Donde vi que era necesario una cola de prioridad para ordenar los nodos abiertos y así conseguir el coste mas eficiente.

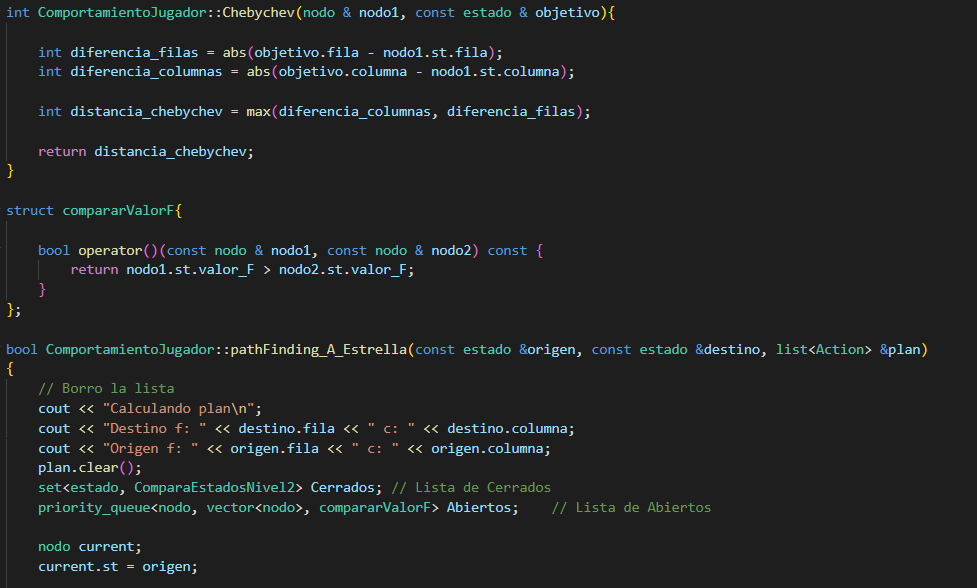
Sin embargo, este algoritmo me daba una serie de resultados poco eficientes y lentos por los que decidí pasarme al algoritmo A estrella:



Me hice un par de métodos de comprobación para poder realizar la búsqueda de nodos de la manera correcta y obtener un coste coherente:

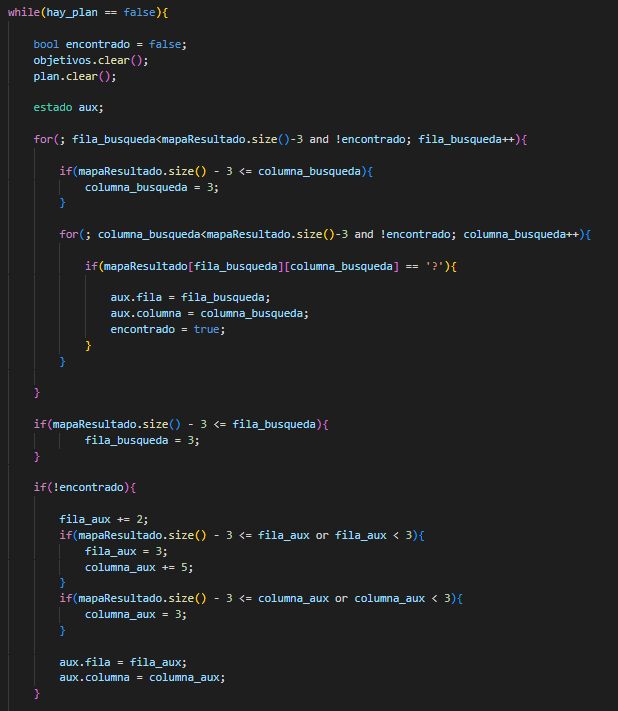


Utilice la heurística de chebychev para ordenar los nodos, ya que la distancia de Manhattan no me daría el camino mas eficiente y por tanto no sería el correcto:

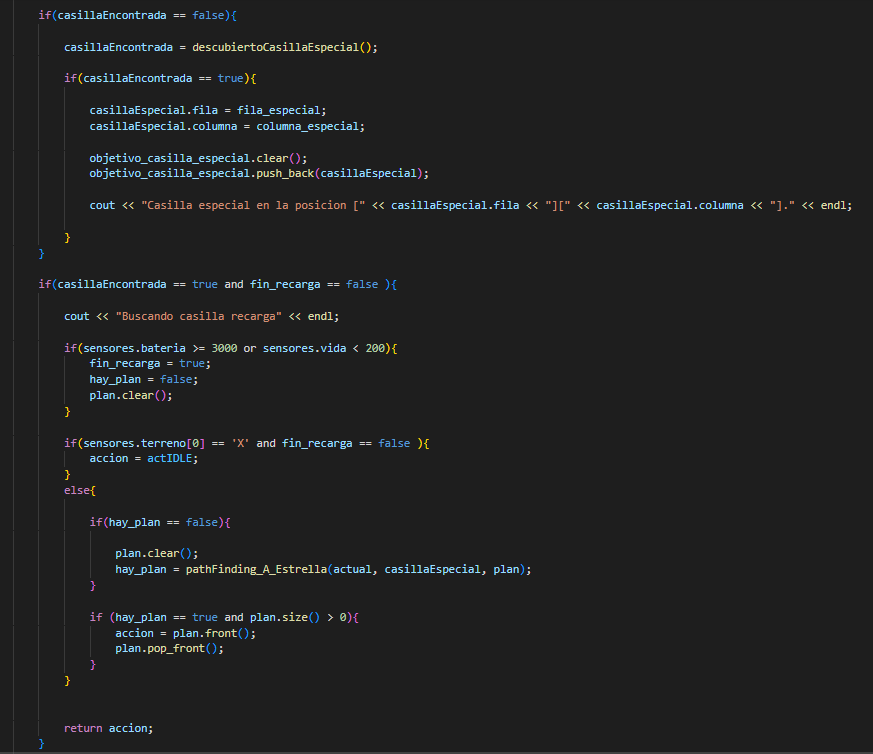


Para el nivel 3, donde se trata otra vez el tema de exploración, simplemente diseñe un código que buscase objetivos que no hayan sido descubiertos todavía, y que cuando este terminase realizase una búsqueda uniforme recorriendo el mapa de arriba hacia abajo.

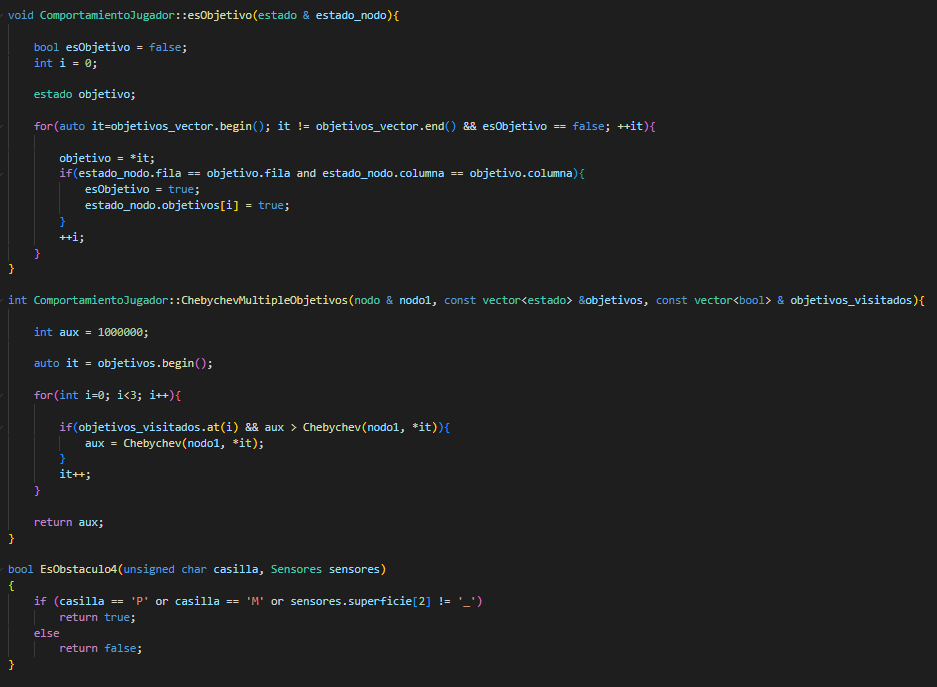
También tuve que tener en cuenta que si cogía los elementos del borde provocaría fallos por lo que limite el rango del mapa para la búsqueda:



Por otro lado, añadí un algoritmo de búsqueda para la casilla de recarga por lo que cuando tuviese poca batería iría a recargar directamente si conoce su ubicación exacta:

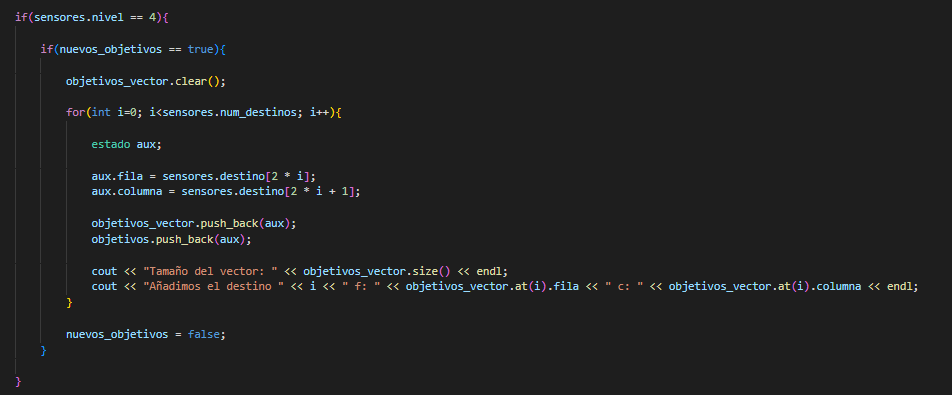


Para el nivel 4, me he creado una serie de métodos que tuvieran en cuenta múltiples objetivos y la existencia tanto de lobos como de aldeanos:

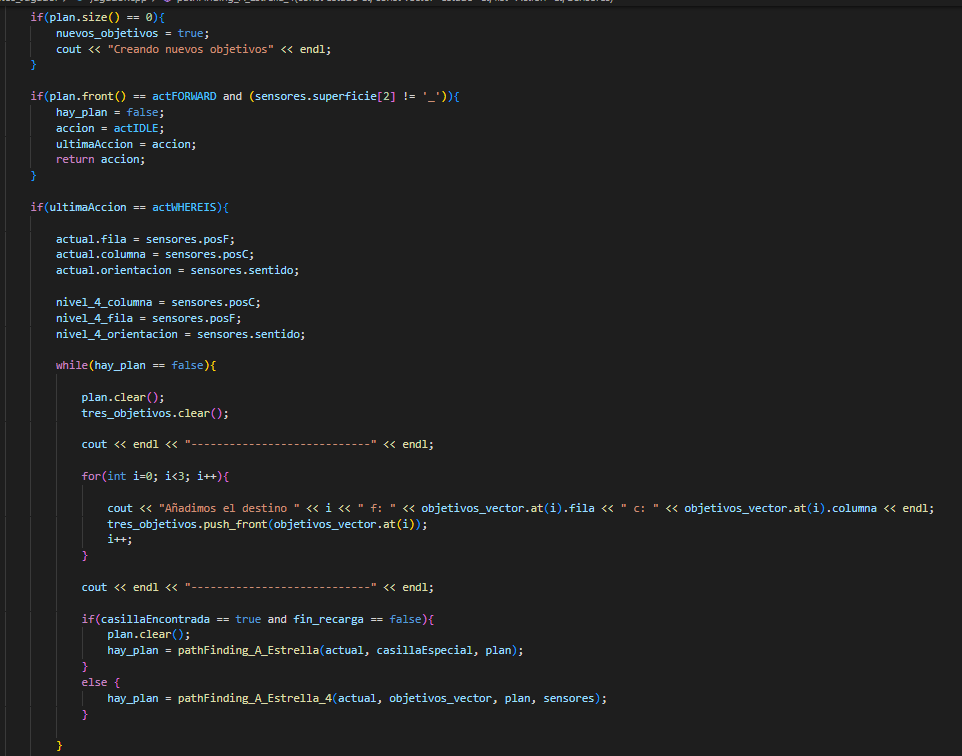


También me he vuelto a crear un A estrella, pero esta vez orientado a múltiples objetivos, creándome un vector de objetivos dentro del nodo para poder ir sabiendo cuando he pasado por uno o no.

En el método think, voy añadiendo los objetivos de 3 en 3, y los añado a un vector para luego pasárselo al path-finding.



Y poco a poco, cuando termino un recorrido compruebo que los objetivos han sido alcanzados y si ese es el caso añado unos nuevos automáticamente:



Por último, comentar que también he tenido en cuenta la casilla de recarga, por lo que cuando tenga poca batería ira a por ella, aunque no he implementado la logística cuando te empuja un lobo.