# HEURÍSTICA

La heurística del juego está implementada en la función evaluacionNodo, la cual analizará el valor de los distintos nodos que recorra el algoritmo y les dará un valor en función de si benefician a jugadorMax o a jugadorMin. En nuestro caso, para analizar estos nodos, utilizamos la siguiente mecánica: primero recorremos todo el tablero para ver si los másters de ambos jugadores siguen en pie. Si lo están, guardamos sus posiciones. Una vez analizado el tablero, comprobamos si alguno de los másters ha sido eliminado, dando una puntuación alta positiva si el máster que falta es el de jugadorMin y alta negativa si es el de jugadorMax, ya que se trata de un estado final. En caso de que ambos másters sigan en pie, se comprueba si alguno de ellos ha alcanzado la casilla de salida del máster contrario, dando una puntuación alta positiva si el máster que ha llegado es el de jugadorMax y alta negativa si es el de jugadorMin, ya que también se trata de un estado final.

En caso de que ambos másters sigan en pie y ninguno haya alcanzado la casilla de salida del otro, procederemos a evaluar el nodo en función de las piezas eliminadas y la posición de las piezas que aún quedan. Es importante mencionar que la función aplica se encarga de almacenar al final del nuevo nodo las piezas eliminadas tanto por jugadorMax como por jugadorMin. Para evaluar las posiciones de las piezas, hemos asignado una puntuación a cada casilla. Hemos considerado que las tres primeras filas desde donde se sale sean cada vez de más valor, ya que nos acercamos al contrincante, pudiendo eliminar sus piezas. Sin embargo, las dos últimas filas las hemos considerado peligrosas, ya que las piezas están muy expuestas. Por ello, tienen una puntuación mucho más baja.

Después de esto, recorreremos el tablero y sumaremos el valor de las casillas en las que se encuentran nuestras piezas. Si el número de piezas comidas por jugadorMax es mayor que el de jugadorMin, devolveremos un valor positivo, y en caso contrario, uno negativo. Además, sumaremos la puntuación de las posiciones de las piezas. Esto nos permitirá que, por ejemplo, si hay dos estados en los que jugadorMax ha eliminado una pieza de jugadorMin y jugadorMin no ha eliminado ninguna, el algoritmo tenga en cuenta, además de que se ha eliminado una pieza del contrincante, las posiciones de las piezas para futuros movimientos.

En caso de que jugadorMax y jugadorMin hayan comido las mismas piezas, únicamente tendremos en cuenta la suma del valor de las posiciones. Hay que tener en cuenta que, como la llamada al algoritmo minimax es de profundidad 1, solo se tendrá en cuenta un movimiento de jugadorMax y otro de jugadorMin. Por ello, como mucho podrán comer una pieza cada uno para cada estado.

Es importante mencionar que se actuará de manera distinta dependiendo de si el jugadorMax tiene id 0 o 1, ya que su posición de inicio en el tablero será distinta.