# FUNCIONAMIENTO DEL JUEGO

Se ha realizado kernels para hacer los movimientos hacia arriba y hacia la izquierda. Para hacer otro de los movimientos se hace un “flip” de la matriz generando su simétrica respecto al eje vertical u horizontal según corresponda.

Se rellena la matriz de decisiones que indica las casillas de unión que se generarán. Estas quedan en una posición tal que nos permita calcular d forma sencilla cuales deberán borrarse para poder hacer el movimiento. Con otro kernel se calcula cuales serán estas. Finalmente, otro más realiza los cambios y pone a 0 las casillas que daban borrarse y al valor de la nueva ficha las que de hayan unido.

Finalmente se cuentan los 0 por fila o casilla desde cada casilla al final de la matriz.

Un último kernel mueve las finchas tanto como se indique para concluir el movimiento.

* Se lleva un recuento de los puntos conseguidos por cada vida.
* Cuando acaba la partida se muestra la suma de todos esos puntos, así como la suma de todos los conseguidos a lo largo del uso de jugo (se hace uso de un archivo de texto)
* En todo momento se permite **cargar pulsando c** o **guardar pulsando g** una matriz para jugar con ella.
* También se permite cambiar de modo automático a **manual pulsando m** o **terminar la partida antes de tiempo pulsando e.**
* El tamaño del texto en la consola se ajusta en función del tamaño de la matriz.

**Se detecta cuando un movimiento no produce cambios en la matriz y no se deja hacerlo al usuario.**

**Si no quedan movimientos por hacer se detecta también.**

**Se leen las características de la gráfica para decidir el tamaño de la tesela de modo que se gasten los menos hilos posibles.**

# IA

Hay dos modos de IA, una IA aleatoria que realiza movimientos dando favoritismo a dos de ellos de modo que busca concentrar estos en una posición concreta.

La otra versión evalúa los 4 movimientos posibles y realiza aquél que tenga más puntuación basada en la siguiente heurística.

Nunca realizar un movimiento que acabe la partida si hay otero que no lo hace.

Preferir movimientos que cambien muchas posiciones de sitio.

Preferir movimientos que proporcionen más puntuación.

# Límites y dificultades

La mayor dificultad técnica a la hora de hacer la práctica ha venido del paso de global a bloques. En global toda la matriz está siempre en un único bloque lo que permite la sincronización de hilos. No obstante, al pasar a bloques donde habrá hilos en varios de ellos esto ya no es posible. Para solventarlo nos hemos dado cuenta de la utilidad de tener unas matrices desde las que solo se va a leer y otras en las que solo se va a escribir como parámetros de cada kernel. Leer y escribir en la misma matriz puede dar problemas debido a la sincronización de hilos.

Otra opción para resolver esto ha sido realizar múltiples llamadas a los kernell como en el caso de las sumas por reducción. Un parámetro de kernel es un índice que indica los valores que se deben mirar para hacer la suma.

# GPU

Regla general, para favorecer el control, evitar que los hilos escriban unos encima de otros o haya problemas del tipo que un hilo escriba antes de que los otros lean se evitará en la medida de los posible leer de las mismas matrices en las que se va a escribir.

Si se lee de una matriz no se escribirá en ella, se utilizará otra para almacenar la salida.

Se pone a continuación los resultados parciales de cada kernel en un movimiento hacia abajo junto con una breve explicación del kernel en general.

**Datos de la partida.**

A screenshot of a cell phone

Description automatically generated

**MATRIZ INICIAL**

A close up of a green screen

Description automatically generated

**FLIP**🡪da la vuelta a una matriz

Tiene la versión vertical y horizontal.

A close up of a green screen

Description automatically generated

**TAKEDECISIONS**🡪crea la matriz de decisiones en la cual se indican las nuevas celdas (resultado de la unión de dos) que se generarán al hacer un movimiento.

Deja el valor que se obtendrá tras añadir dos elementos en la posición del elemento que se va a añadir

Sirve también para saber los puntos que se obtiene al hacer el movimiento y contar los movimientos realizados. Se pondrá el valor a obtener en elementos que sumen con otro ocupan un lugar impar contando solo los ocupados por elementos iguales desde el primero que no es igual a ellos

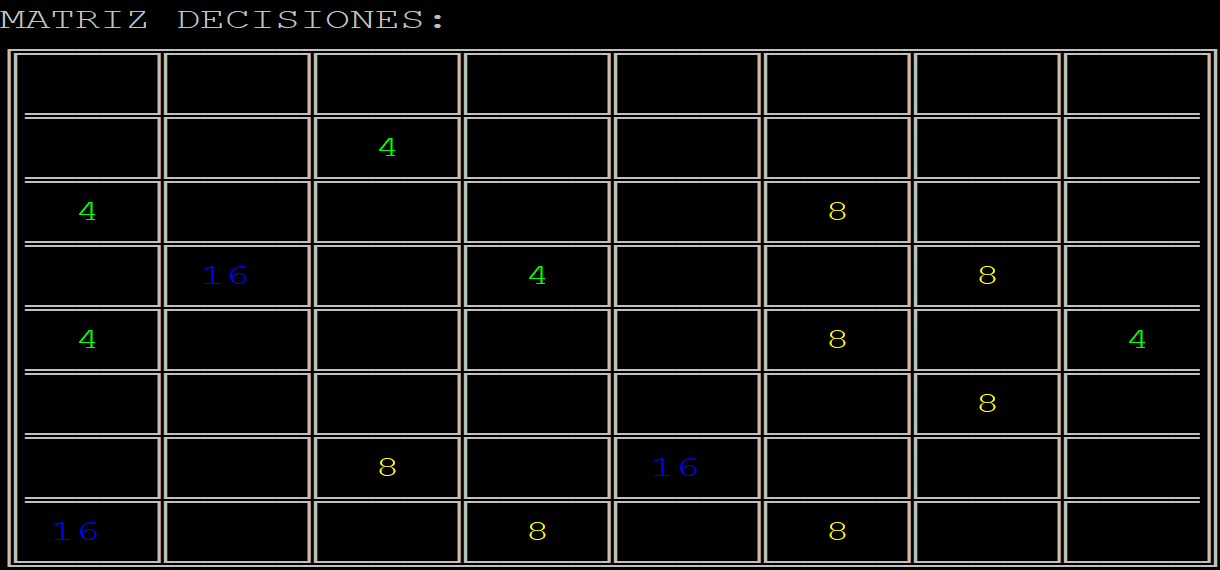
Ej:

Tablero de entrada: 2222 400404

Matriz de decisiones: 0404 000800

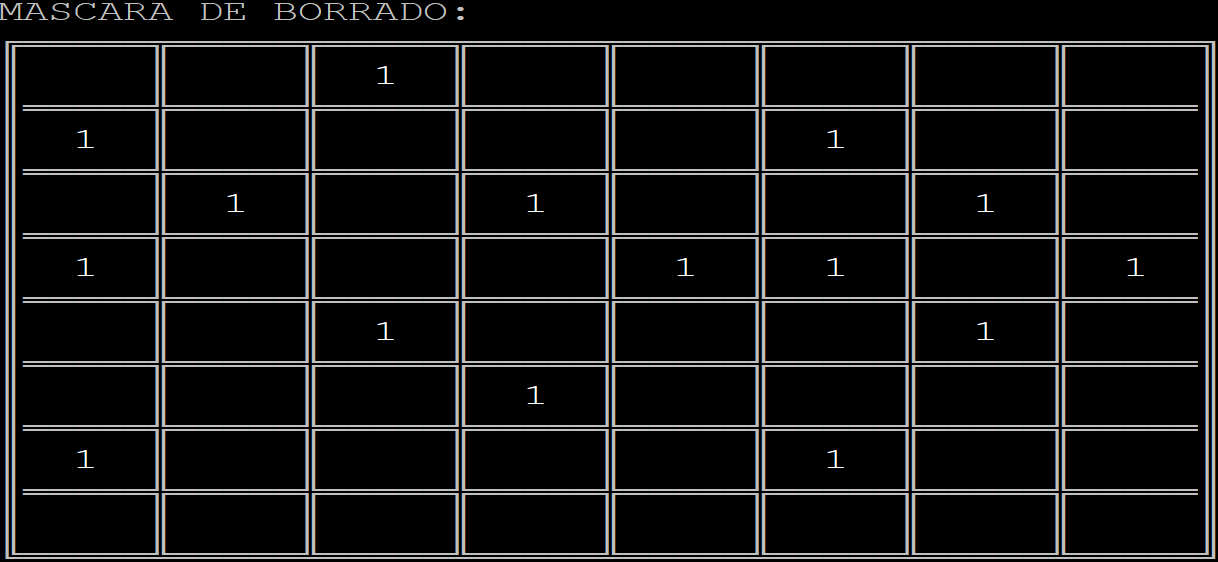
Se pone la nueva ficha en las posiciones impares cada vez que se encuentra un número par de valores sumables

Tiene versión vertical y horizontal.



**CREATEDELETER**🡪crea una matriz con un 1 en las posiciones del tablero que haya que poner a 0 para poder realizar un movimiento

Tiene versión vertical y horizontal.



**MOVE**🡪mueve las casillas del tablero tanto como indique la matriz de salto.

Tiene versión horizontal y vertical.

A close up of a green screen

Description automatically generated

**ZEROCOUNT**🡪cuenta la cantidad de 0s que hay desde cada casilla el final de su fila o columna.

Tiene versión vertical y horizontal.



(Los elementos que estaban a 0 no se desplazan)

**MATRIZ FINAL SIN HABER ROTADO**

**A close up of a green screen

Description automatically generated**

**MATRIZ FINAL YA ROTADA**

**A close up of a green screen

Description automatically generated**

**MATRIZ FINAL CON LOS ELEMENTOS ALEATORIOS AÑADIDOS**

**A close up of a green screen

Description automatically generated**

**SUMLEFT🡪** Genera una matriz con un 1 en cada posición en la que se pueda hacer un movimiento

Sumando el resultado de todos los unos sabremos si se pueden hacer movimientos o no.

Cada hilo mira a sus cuatro elementos de los lados y al suyo.

Se utiliza para saber cuando ya no quedan movimientos y por tanto terminar la partida o quitar una vida según corresponda.

A picture containing weapon

Description automatically generated

**SUM**🡪tiene distintas versiones: suma la cantidad de 0s, suma la cantidad de valores distintos de 0 o suma toda la matriz.

Suma 0s: se utiliza para saber si hay huecos en la matriz.

Suma distintos de 0: se utiliza para la heurística de la IA, para valorar los buenos que son sus movimientos. Y para saber si una matriz ha cambiado o no.

A picture containing person

Description automatically generated

**CPYMATRIX**🡪 copia una matriz en otra.

**HASCHANGED**🡪pone un 1 en una matriz por cada valor que difiera en las proporcionadas.

**SETVALUE**🡪Pone en todos los elementos de la matriz el valor indicado.

# CPU

Funciones principales.

**PRINTTABLERO**🡪Imprime el tablero en color creando una celdilla alrededor de los números los cuales quedan centrados gracias a una clase auxiliar de control de buffer.

**ADDRANDOM**🡪Pone números aleatorios en los huecos del tablero, pone tantos como se le indiquen.