INGENIERÍA DEL CONOCIMIENTO

GRADO EN INGENIERÍA DEL SOFTWARE

PRÁCTICA 1: IMPLEMENTACIÓN DEL ALGORITMO A*

UNIVERSIDAD COMPLUTENSE DE MADRID
MANUEL MONFORTE ESCOBAR

1. ALGORITMO A*

Esta práctica consiste en implementar el algoritmo A*. El objetivo del algoritmo es encontrar la mejor ruta, es decir, la ruta con coste mínimo, entre una celda de inicio y una celda de final en un tablero de dimensiones MxN.

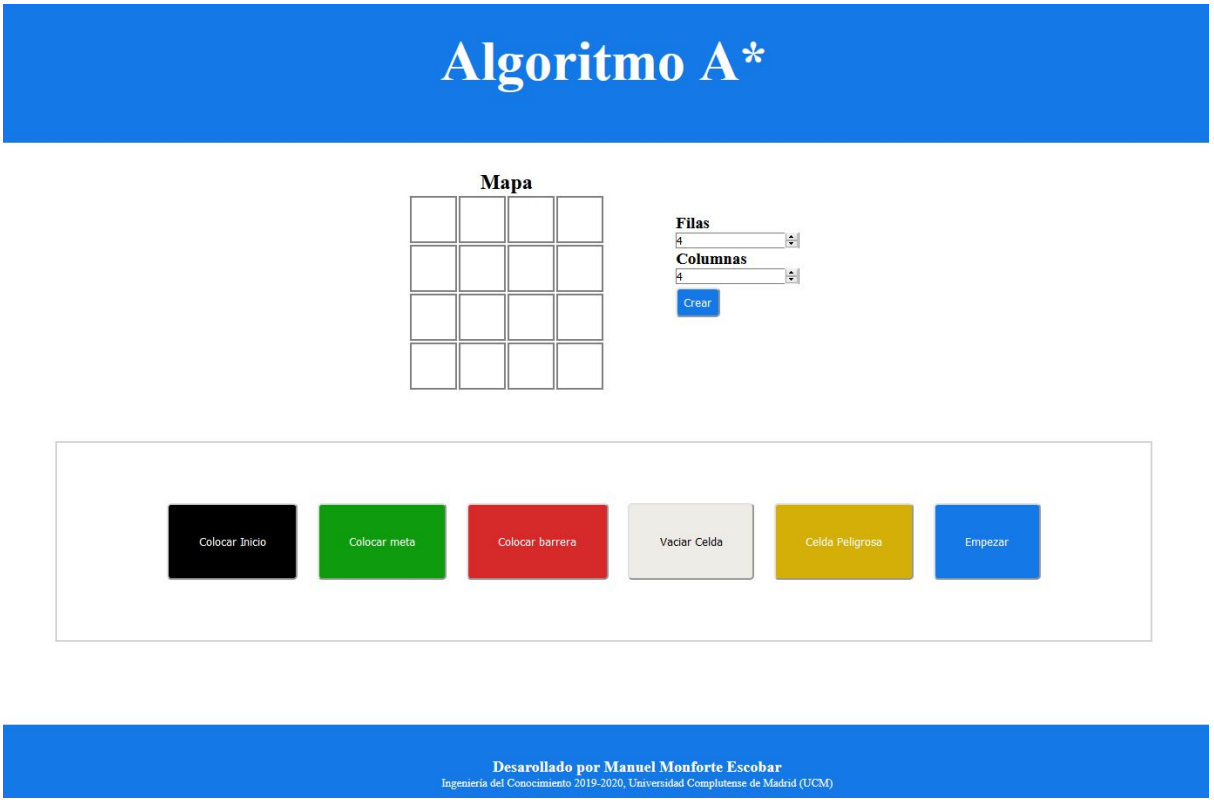
Cada celda puede acceder a sus contiguas de forma diagonal, vertical y horizontal. El coste para acceder a una celda vecina viene determinada por la siguiente función:

$$F(n) = h(x) + g(n)$$

- h(n) :: La distancia heurística del nodo n al nodo final
- g(n) :: Distancia acumulada al nodo n

2. GUIA DE USO

Para simular el algoritmo, primero descomprimir todos los archivos procedentes de la entrega del Campus Virtual en el mismo directorio. Una vez descomprimido, deberemos abrir el archivo index.html con el navegador Mozilla Firefox, navegador con el que se ha probado la práctica
Para facilitar la implementación del algoritmo se ha creado la siguiente interfaz:



Cada botón sirve para asignar un tipo de casilla a una celda:

- **Inicio**: Negro
- **Final** : Verde
- **Barrera**: Rojo
- **Peligro**: Marrón

Se podrá variar el tamaño del tablero introduciendo un número entre 2 y 10 en el apartado de columnas y Filas, después se hará click en el botón crear.

Para crear celdas, primero se ha de seleccionar el botón y después hacer clic en la celda que se desea colocar. Si nos hemos confundido, haremos clic en vaciar celda y volveremos a seleccionar la celda que queremos vaciar. Una vez creado el

tablero daremos a iniciar, debe existir un inicio y un final para que se inicie el algoritmo, en caso de no existir se mostrará una alerta indicándolo.

Se mostrará con color azul claro el camino óptimo encontrado por el algoritmo.

3. IMPLEMENTACIÓN DEL ALGORITMO

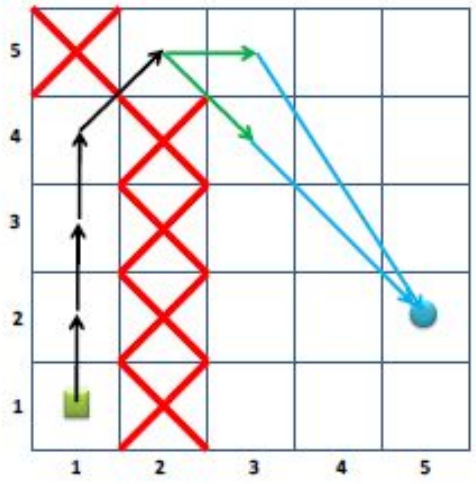
Una vez colocadas las celdas, se guarda la información en una matriz MxN donde cada celda es un nodo que tiene los siguientes atributos:

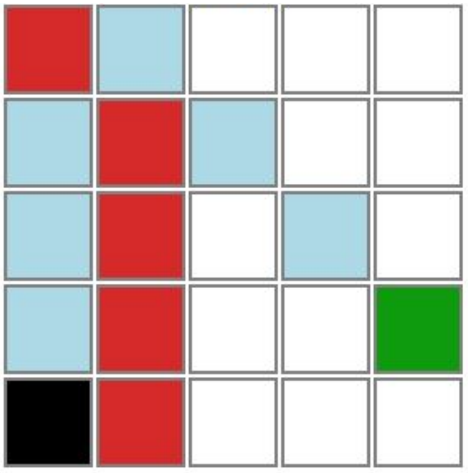
- **i:** Nº de la fila de la matriz
- **j:** Nº de la columna de la matriz
- **f:** Coste de la función de evaluación
- **g:** Distancia acumluada al nodo (i,j)
- **h:** Distancia heurística del nodo (i,j) a la meta
- **parent:** Puntero hacia el padre del nodo (i,j)
- **dangerous:** Penalización por pasar por el nodo (i,j)
- **altura :** Altura del nodo (i,j)

Una vez preparada la matriz de dimensiones MxN se aplica el algoritmo A* para encontrar el camino con menor coste. El algoritmo es el siguiente:

1. Añadimos el nodo inicial a la lista abierta
2. Mientras que la lista abierta no esté vacía y no hayamos encontrado el camino:
    a. Obtenemos el nodo con menor coste F de la lista abierta
    b. Si es el nodo meta
        i. Hemos llegado al final, por lo que recorremos los punteros y construimos el camino óptimo.
    c. Si no es el nodo meta:
        i. Lo añadimos a la lista cerrada
        ii. Lo eliminamos de la lista abierta
        iii. Buscamos los vecinos
        iv. Para cada vecino:
            1. Si no es una barrera, tiene una altura accesible y no está en la lista abierta ni cerrada:
                a. Actualizamos los atributos del nodo
                b. Lo añadimos a la lista abierta
            2. Si no es barrera, tiene una altura accesible y no está en cerrada pero si en abierta:
                a. Calculamos g,h, y f.
                b. Si la f es mejor que el nodo que estaba actualizamos costes y padre.
            3. Volvemos al pto 2
3. Devolvemos el camino

Ejemplo del enunciado de la práctica para el algoritmo implementado:





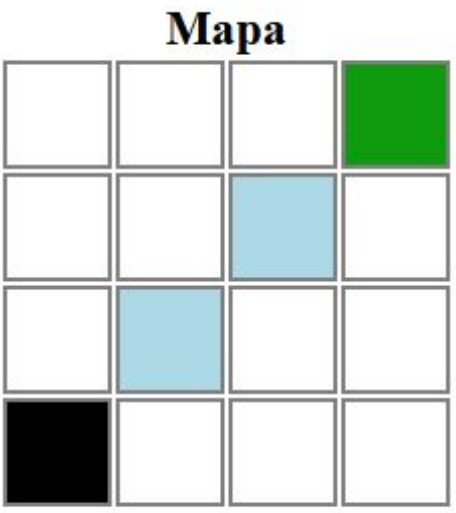
4. AMPLIACIONES REALIZADAS

1. Celdas Peligrosas

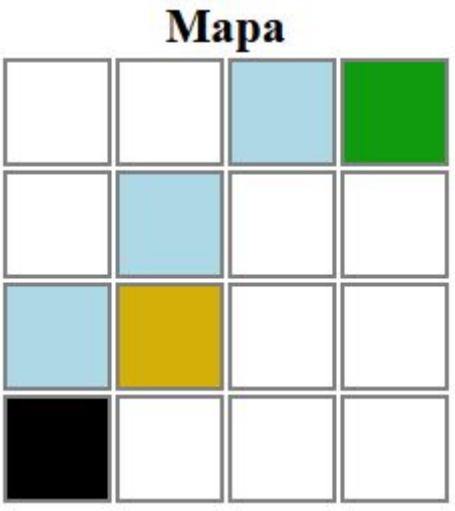
En la interfaz gráfica se ha añadido un botón para crear celdas peligrosas, estas celdas tienen asociadas un coste de una unidad, por tanto la función de evaluación es:

$$F(n) = h(x) + g(n) + 1$$

A continuación se muestran dos capturas para mostrar cómo se comporta el algoritmo cuando hay celdas peligrosas.



Sin celda peligrosa

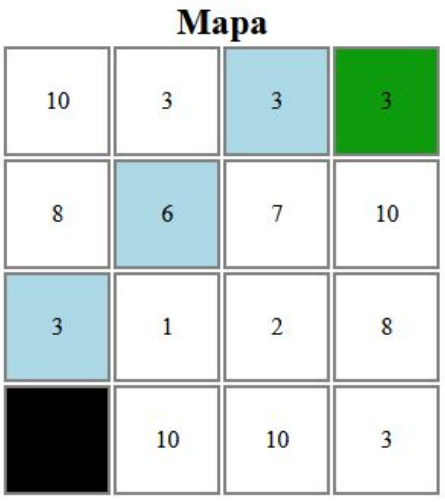


Con celda peligrosa

Cómo observamos el algoritmo evita la celda ya que pasar por ella supone un coste superior en la función de evaluación.

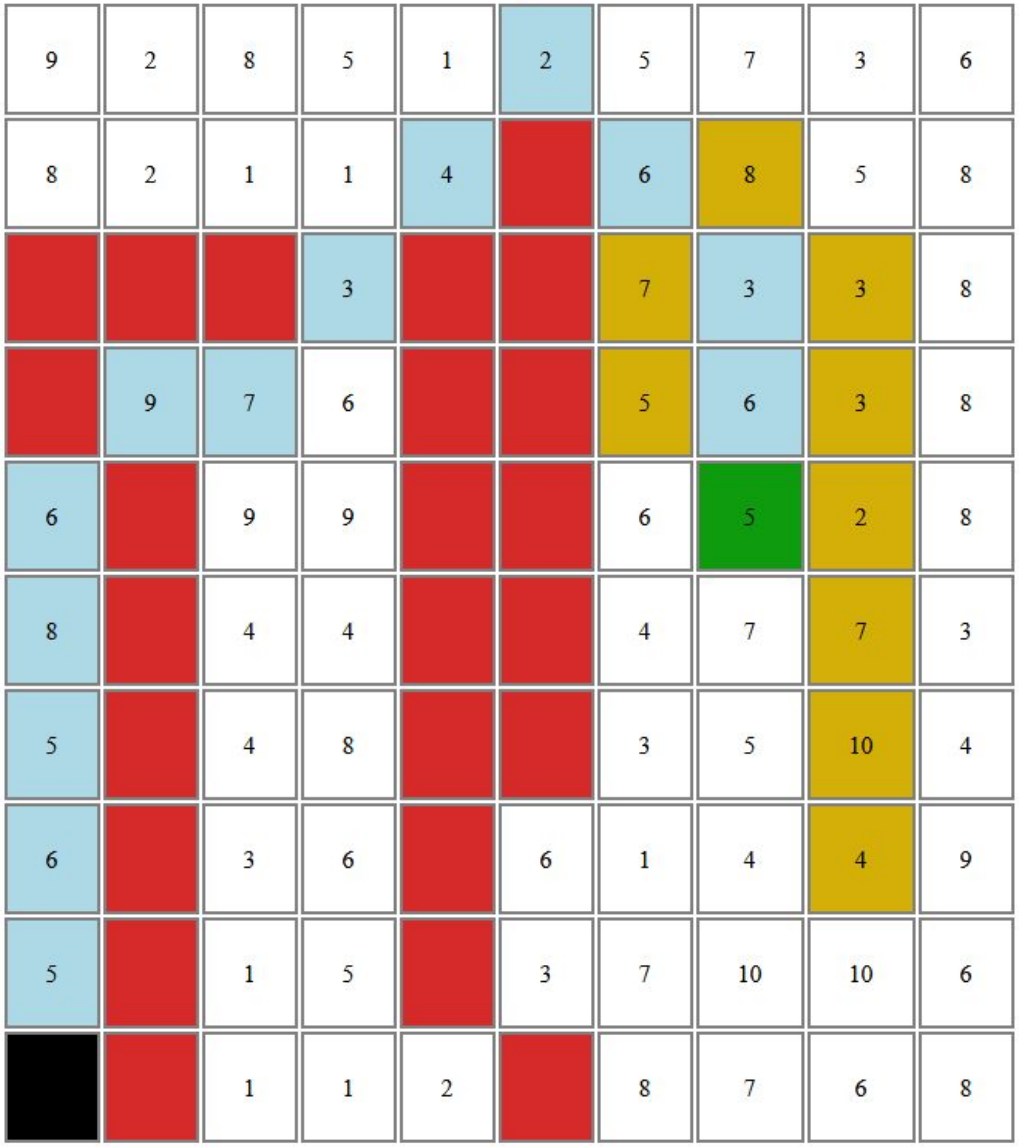
2. Alturas

Para implementar las alturas se ha realizado lo siguiente, cada celda se le asigna una altura aleatoria entre 0-10. Cada vez que se pulse el botón iniciar, se generarán todas las casillas aleatorias. Al inicio y al final se les ha asignado una altura de 3. Por tanto todas aquellas celdas que tengan una altura superior a 5 de diferencia con la anterior serán inaccesibles. A continuación se muestra un ejemplo:



A continuación se muestra un ejemplo con todas las ampliaciones realizadas:

Mapa



Como podemos observar el algoritmo ha seguido el camino óptimo teniendo en cuenta:

- Si pasa por una celda peligrosa existe una penalización, por lo tanto intentará evitarlas en la medida de lo posible.
- Celdas que tengan una altura superior (estricta) a 5 puntos respecto a la celda actual no son accesibles.

5. LENGUAJE UTILIZADO

El lenguaje seleccionado para implementar la interfaz ha sido HTML & CSS, y Javascript para implementar el algoritmo.