PRÁCTICA 1
IMPLEMENTACIÓN DEL ALGORITMO A*

Montserrat Sacie Alcázar

Asignatura: Ingeniería del Conocimiento

Ingeniería del Software

UCM, 2020

Índice

1. Introducción

El objetivo de la práctica 1 es implementar el Algoritmo A* que permita encontrar la ruta óptima en un tablero entre un punto de partida y una meta o destino, simulando la ruta que debería escoger por ejemplo una barca en el mar para llegar a una isla.

En la memoria se explican:

- En **el apartado 2,** los **detalles de la implementación del algoritmo**, así como las funcionalidades voluntarias escogidas para ser implementadas y la **distribución de clases de código del proyecto** implementado en Eclipse.
- El **apartado 3** explica los **detalles de la interfaz gráfica para ejecutar la práctica**. En la entrega se añade un video explicativo de ejecución del programa para mayor claridad.
- Por último, en el **apartado 4** hemos añadido **6 ejemplos de tableros para ejecutar en la corrección** de dicha práctica, con la situación de partida (tablero inicial) y la solución obtenida. 1 ejemplo es el que aparece en el enunciado y los otros 5 ejemplos son diferentes (uno de ellos sin solución).

2. Detalles de Implementación del Algoritmo A*

El Algoritmo A* se ha implementado en el lenguaje de programación **Java** y usando el entorno de desarrollo **Eclipse**.

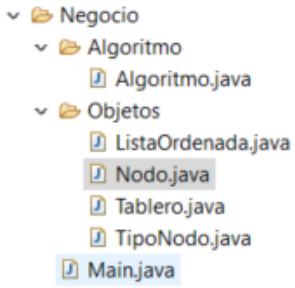
Las **funcionalidades básicas** implementadas han sido las necesarias para la aplicación del Algoritmo A* simplificado, explicadas en el enunciado de la práctica que se nos facilitó. Además, se han implementado las **funcionalidades voluntarias o ampliaciones:**

a) Realizar un recorrido por distintas posiciones predeterminadas denominadas "**way points**" que deben estar incluidas en el camino solución.

d) Implementación de **celdas peligrosas** que pueden atravesarse asumiendo un riesgo, añadiendo un factor de corrección en la evaluación heurística.

Estructura del Proyecto

Negocio: clases con la implementación de la lógica del programa



- **Main.java**: clase principal a ejecutar por el usuario en Eclipse para lanzar el programa. Esta invoca a la ventana principal de la interfaz gráfica.
- **TipoNodo.java:** enumerado que contiene los tipos de nodo que se pueden colocar en el tablero:
    - INICIO
    - META
    - PROHIBIDO (nodo prohibido por donde no se puede pasar)
    - PELIGROSO (Nodo que supone + 1 en la función de evaluación o coste f)
    - WAY_POINT (Nodo por el que tiene que pasar el camino solución sí o sí)
    - CAMINO (tras la resolución del algoritmo los nodos que pertenecen al camino solución pasan a ser de tipo Camino. En caso de ser way points, no le cambiamos el tipo, ya que sabemos que pertenecen al camino y necesitamos saber cuales son Way Points para su representación en la interfaz gráfica.)
    - CAMINO_PELIGROSO (tipo que se usa para marcar los nodos peligrosos que pertenecen al camino solución, útil para la representación gráfica. Los nodos de tipo Camino_Peligroso vuelven a ser de tipo Peligroso tras la ejecución y pintado del camino)
    - LIBRE (Cuando un nodo se crea tiene tipo Libre por defecto)
- **Nodo.java**: objeto Nodo, localizado en una fila y columna del tablero. El atributo g representa el coste o distancia al nodo Inicio y el h la distancia heurística al nodo Meta. Obtenemos el coste f = (g + h) invocando a calculaF().

- **ListaOrdenada.java**: Implementación de métodos para tratar la lista ABIERTA del algoritmo A*. Contiene un ArrayList<Nodo> como atributo. Se añaden los nodos sin orden y se saca el nodo de menor coste f.

- **Tablero.java:** Contiene un Array de Arrays de Nodos que representa el tablero (Nodo[][]). Además, se guardan el nodo Inicio y

Meta como atributos de tipo nodo y un ArrayList de WayPoints, para acceder a ellos rápidamente sin recorrer el tablero.

- **Algoritmo.java:** Incluye la implementación propiamente del Algoritmo A*. El método execute() calcula la ruta más corta entre un nodo Incio y nodo Meta. El método executeConWayPoints() permite resolver el algoritmo con la presencia de Way Points, detallada a continuación.

Presentación

Para la implementación de la interfaz gráfica se ha utlizado la herramienta *WindowBuilder* en Eclipse.

- **VistaPrincipal.java:** JFrame que contiene los paneles del tablero y de configuración.
- **PanelTablero.java:** Panel que contiene un Array de arrays de JButtons para la implementación de las casillas del tablero, así como los métodos para su actualización o ActionListeners.
- **PanelConfig.java:** JPanel que contiene los botones para generar un tablero de m filas y n columnas (Por defecto genera un tablero de 5x5), así como las casillas de distinto tipo para colocar en el tablero y los botones “Comenzar” y “Reset”
- **Controlador.java:** clase que contiene una instancia de la clase Tablero.java, para comunicar Negocio con Presentación.
- **Auxiliar.java:** Contiene métodos invocados para ajustar las imágenes a las casillas del tablero.
- **Fondo.java**: JPanel cuya única utilidad es colocar una imagen de fondo. Sobre él se incluirán los paneles Tablero y Config en la vista Principal.

Las imágenes se usan para representar las diversas casillas de inicio, meta,etc.

Implementación de Celdas Peligrosas

En la función de evaluación de un nodo "costeF()" se consulta si es de tipo Peligroso y en ese caso se le suma +1 al coste f que devuelve la función calculado

Implementación de Celdas Way Points

Para obtener un camino desde un nodo Inicio a un nodo Meta que pase por cada nodo "Way Point" colocado, el algoritmo A* (implementado en el método execute() de Algoritmo.java) se ejecuta varias veces para el cálculo de los caminos:

- INICIO > WAYPOINT_1 (De los nodos way Point, se selecciona el más cercano a Inicio)
- WAYPOINT_1 > WAYPOINT_2 (De los nodos way Points que quedan por recorrer, se escoge el más cercano a WAYPOINT_1)
  …
- WAYPOINT_ N > META

El algoritmo en el método execute() se ejecuta sobre un tablero auxiliar (atributo tableroAux en la clase Algoritmo.java) varias veces, actualizando en él el nodo Inicio y Meta para el cálculo de los subcaminos anteriores listados.

Por otro lado, la solución se va construyendo sobre el atributo tablero (instancia de la clase Tablero.java presente en la clase Algoritmo.java).

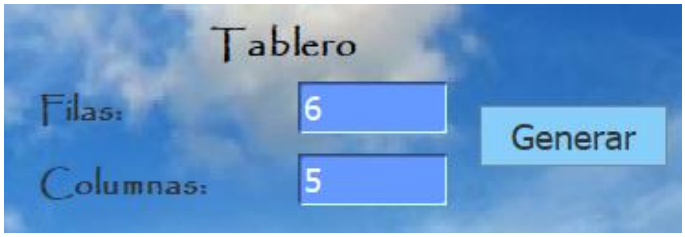
3. Manual de Usuario

Importar el proyecto A_Estrella en Eclipse

Arrancar la aplicación pulsando Run  colocados sobre 

Nos aparece:

Para **generar un tablero** de otro tamaño escribir el número de filas y columnas que deseamos que tenga el tablero, por ejemplo 6 x 5 y pulsamos el botón Generar:

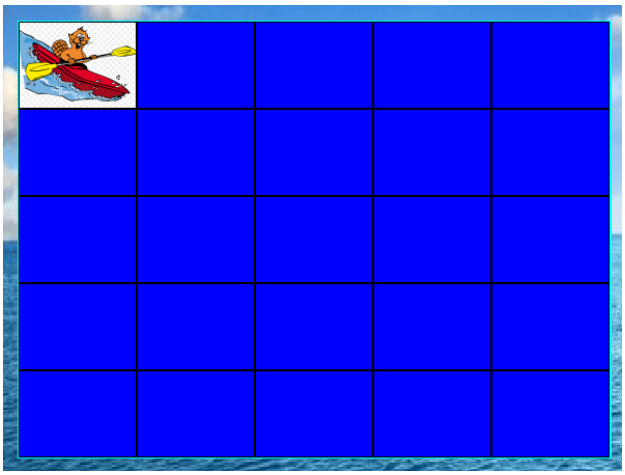


Tableros muy grandes (mayores de 20 x 20) se empiezan a ver con casillas demasiado pequeñas y no es recomendable pero el algoritmo funciona igualmente y la representación gráfica es correcta.

Panel con las Casillas disponibles:

*Las etiquetas Peligroso y Eliminar tienen distinto color
por mera claridad visual, no hay ningún otro motivo

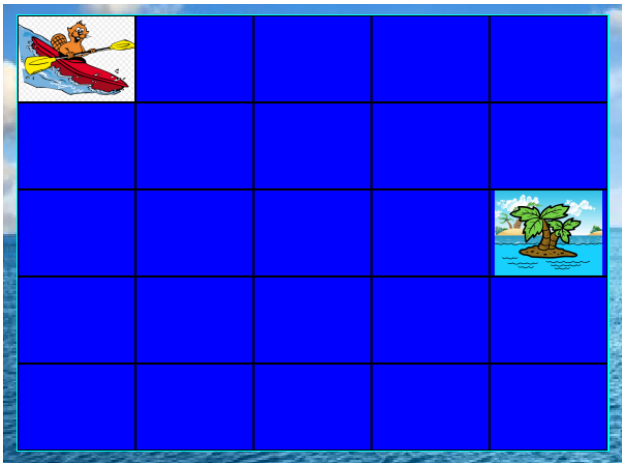
Para **colocar una casilla**, por ejemplo Inicio, hacemos click sobre la casilla Inicio del panel anterior y pulsamos sobre la casilla del tablero donde queremos colocar el Inicio:

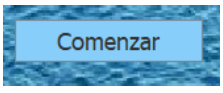


Cada vez que queramos colocar una casilla en el tablero, debemos volver a pulsar sobre el tipo de casilla en el panel de tipos de casilla.

Para **eliminar una casilla** podemos pulsar directamente sobre la casilla del tablero que queremos eliminar. Es lo más intuitivo, pero si no se nos ocurre también podemos pulsar la casilla de tipo Eliminar y luego en la casilla del tablero que queramos.

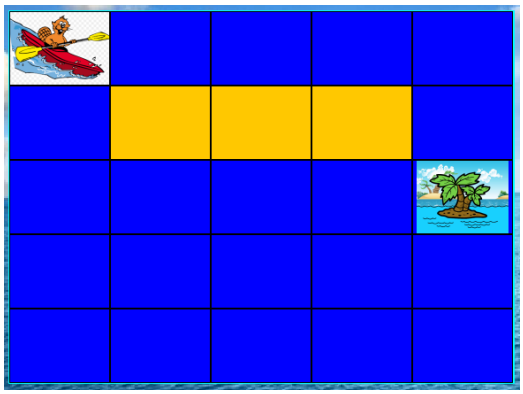
Para comenzar a ejecutar el algoritmo debe haber como mínimo un Inicio y una Meta colocados. En cualquier caso si no los colocamos nos saltará una notificación para que los coloquemos.



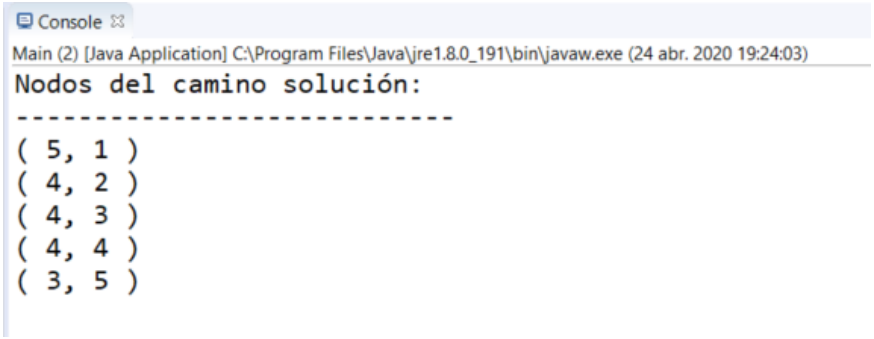
Pulsamos 

par ejecutar el algoritmo y que se pinte el camino más corto. El camino se representa en naranja.

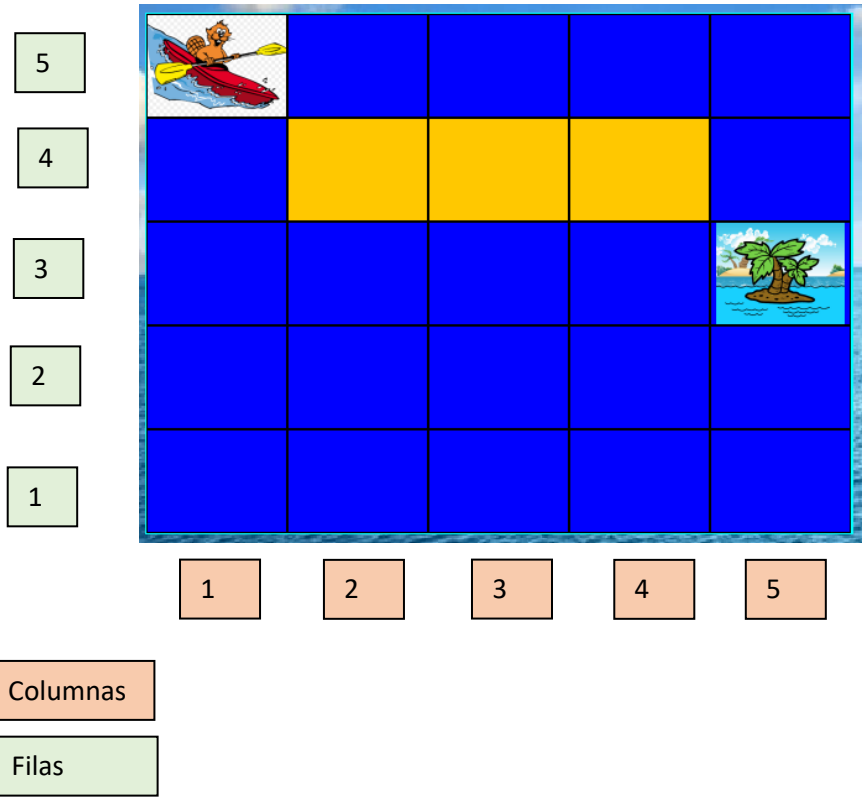
Las casillas way Points, en caso de haber colocado una o varias, no cambian de color porque son una imagen, pero se sabe que forman parte del camino.



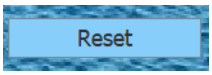
La solución de Nodos que representa el camino solución también se pintan por consola, para mayor seguridad de los nodos seguidos (Empieza en las coordenadas del nodo Inicio y termina en las coordenadas del Nodo Meta):



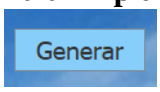
Esta numeración de nodos es de acuerdo a la siguiente numeración de filas y columnas del tablero:



(5,1) es el nodo Inicio (fila 5, columna 1).

Para resetear el tablero y **borrar solo el camino** pulsar 

Si tras pulsar comenzar y obtener un camino, colocamos nuevas casillas en el tablero y pulsamos comenzar de nuevo sin pulsar Reset, no pasa nada. El algoritmo se reinicia y calcula el nuevo camino aunque hasta que no termine de ejecutarse no se borra el camino de la interfaz y no aparece la nueva solución.

Para **limpiar por completo el tablero que tenemos y dejarlo vacío** pulsar 

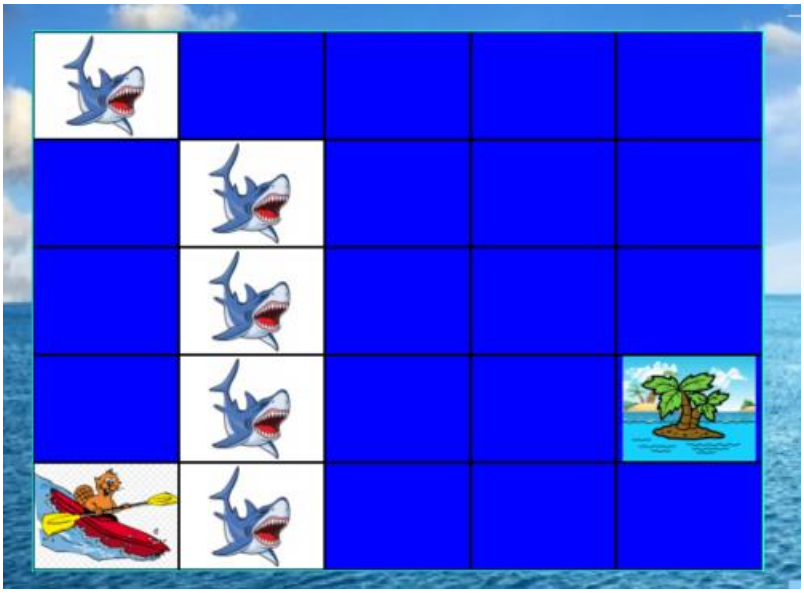
sin cambiar ni las filas ni las columnas.

En caso de duda ver el video adjunto.

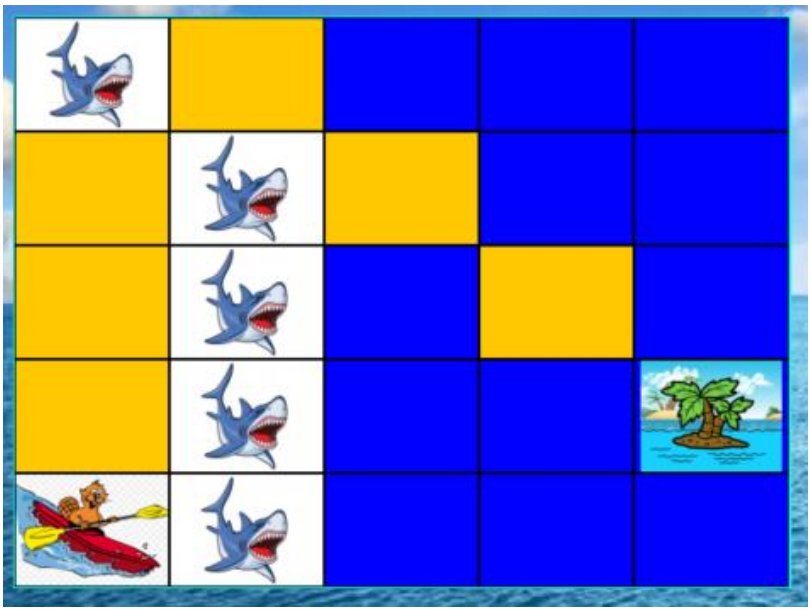
5. Ejemplos de ejecución

Ejemplo 1 (el ejemplo del enunciado)

Estado inicial



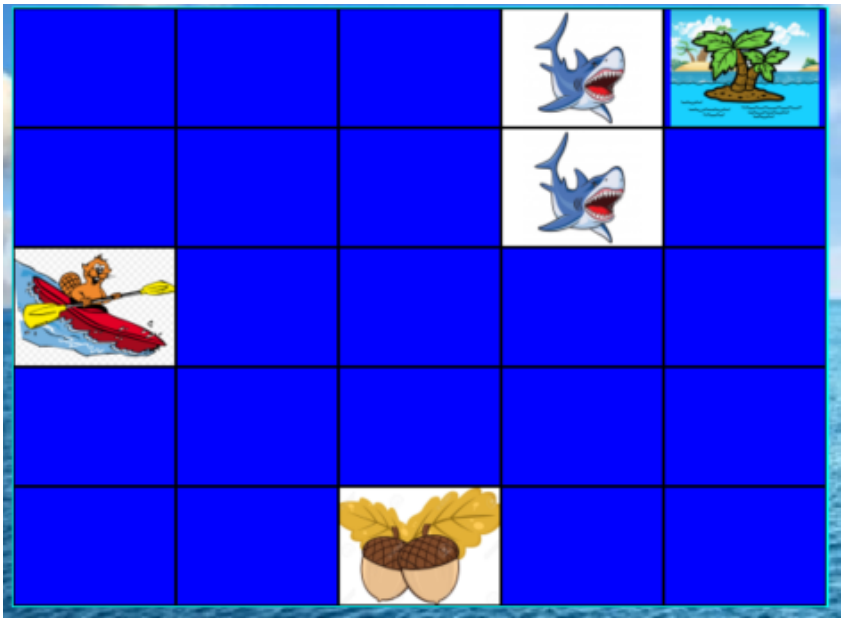
Solución



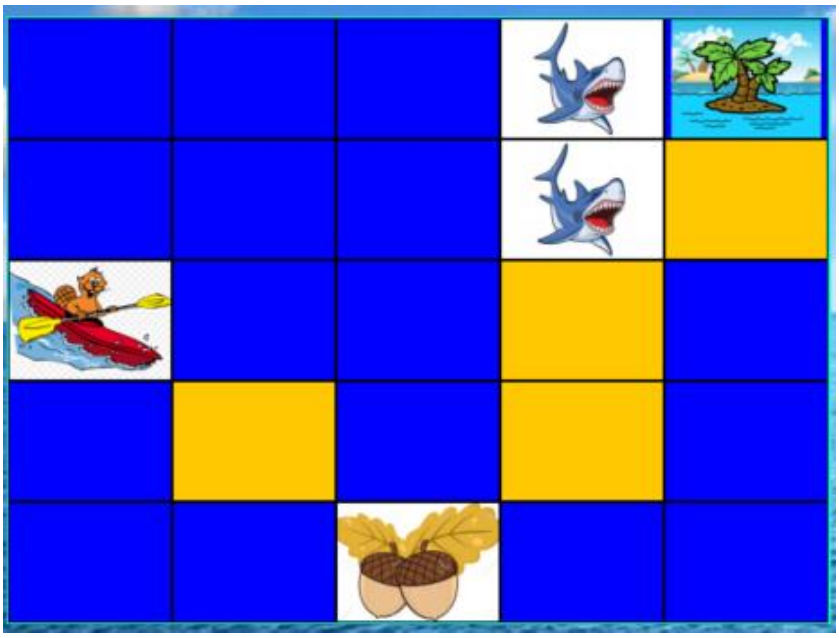
```
Nodos del camino solución:
---------------------------
( 1, 1 )
( 2, 1 )
( 3, 1 )
( 4, 1 )
( 5, 2 )
( 4, 3 )
( 3, 4 )
( 2, 5 )
```

Ejemplo 2

Situación inicial



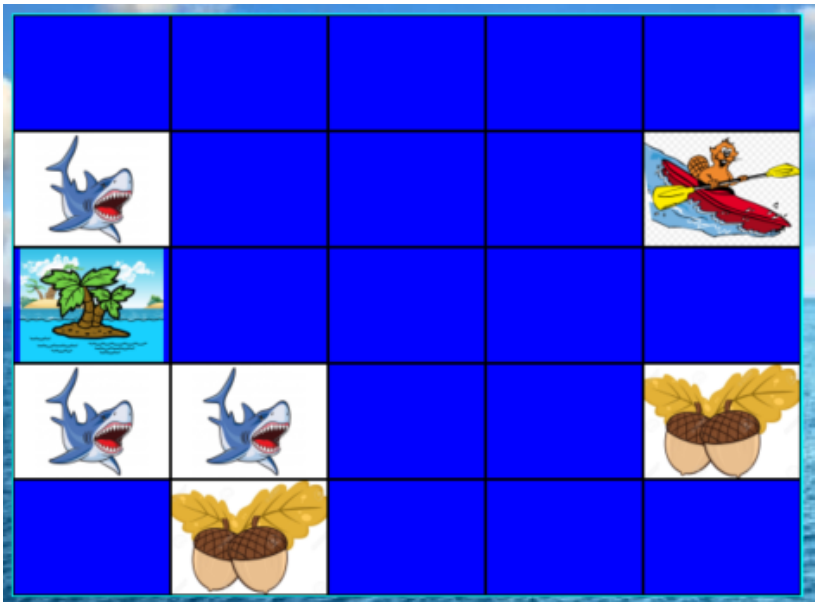
Solución



```
Nodos del camino solución:
---------------------------
( 3, 1 )
( 2, 2 )
( 1, 3 )
( 2, 4 )
( 3, 4 )
( 4, 5 )
( 5, 5 )
```

Ejemplo 3

Situación inicial

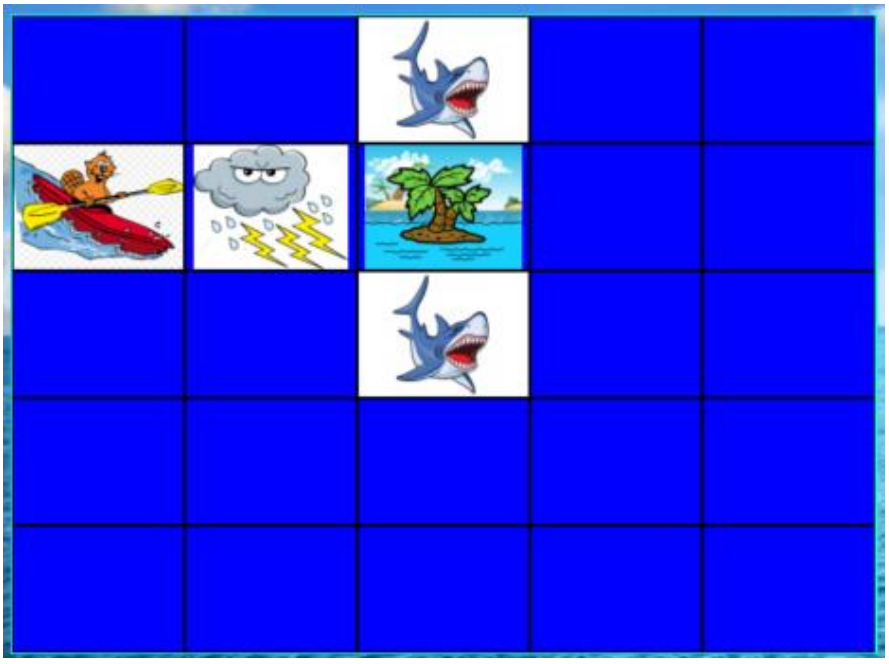


Solución

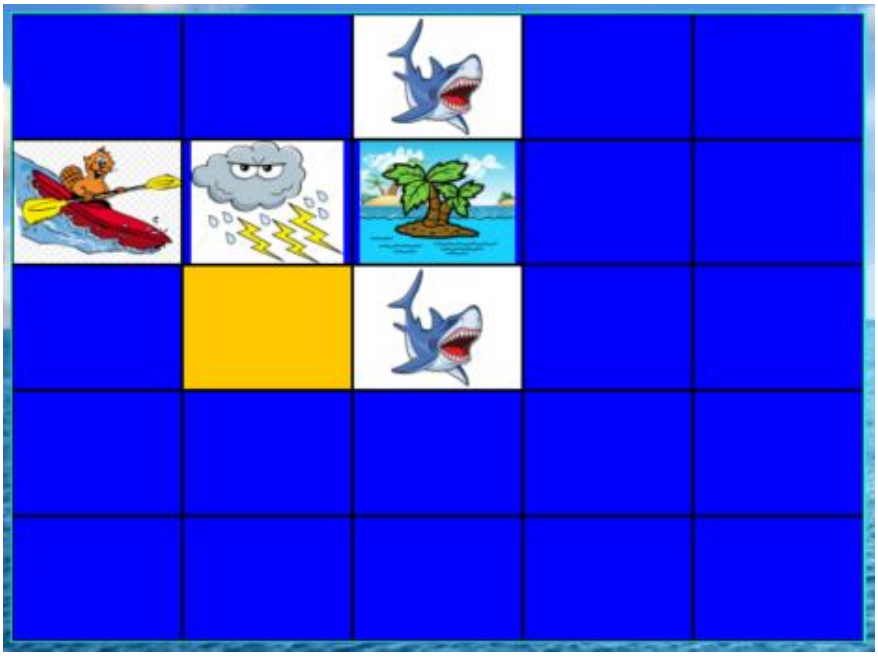
```
Nodos del camino solución:
---------------------------
( 4, 5 )
( 3, 5 )
( 2, 5 )
( 2, 4 )
( 1, 3 )
( 1, 2 )
( 2, 3 )
( 3, 2 )
( 3, 1 )
```

Ejemplo 4

Situación inicial



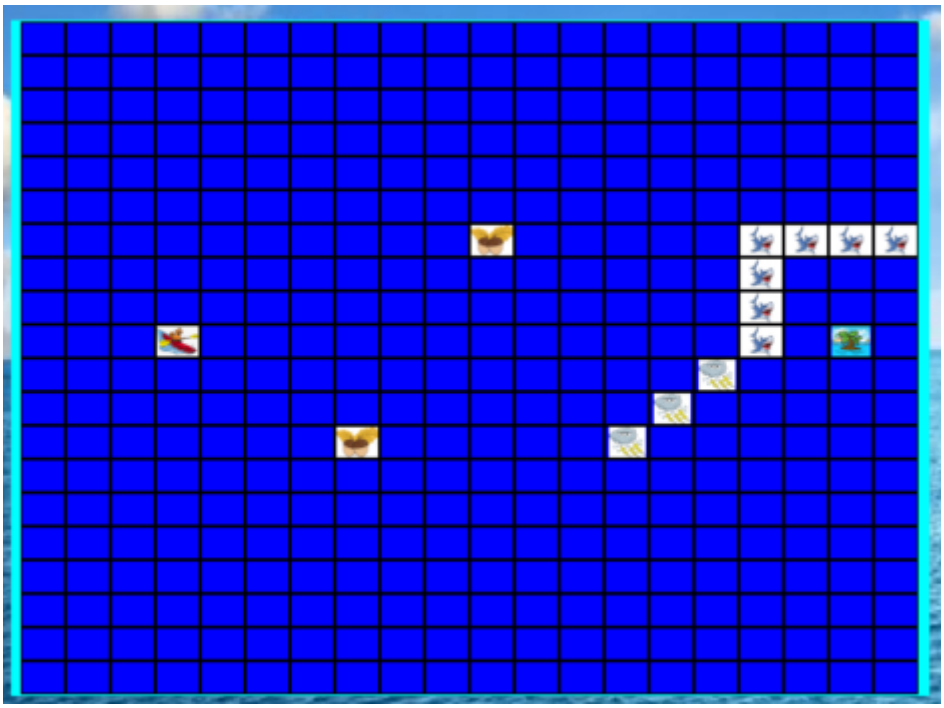
Solución



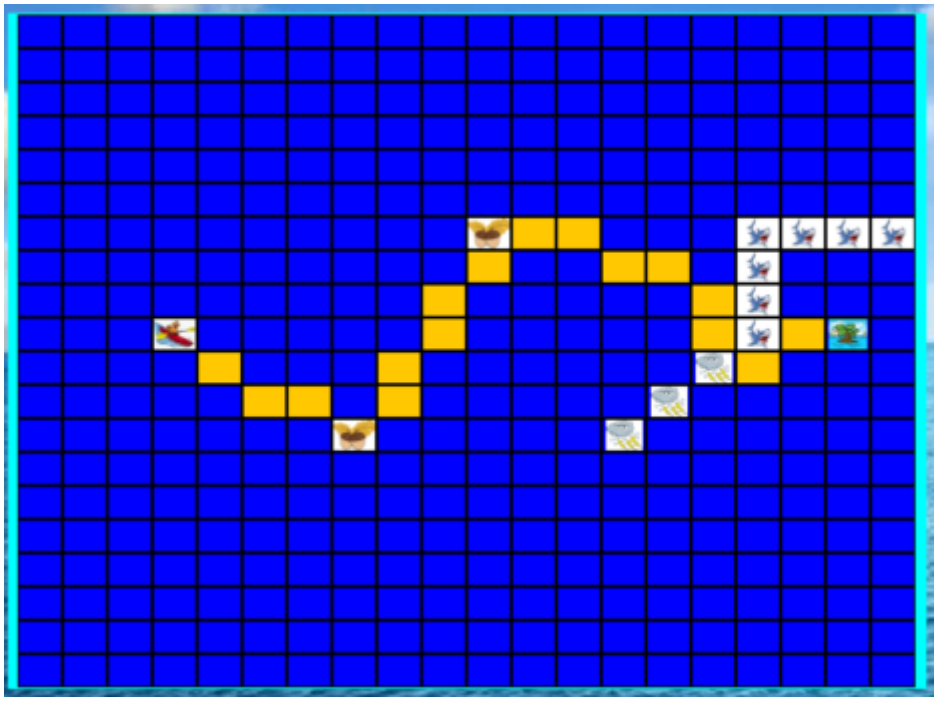
```
Nodos del camino solución:
---------------------------
( 4, 1 )
( 3, 2 )
( 4, 3 )
```

Ejemplo 5 (Tablero 20 x 20)

Situación inicial



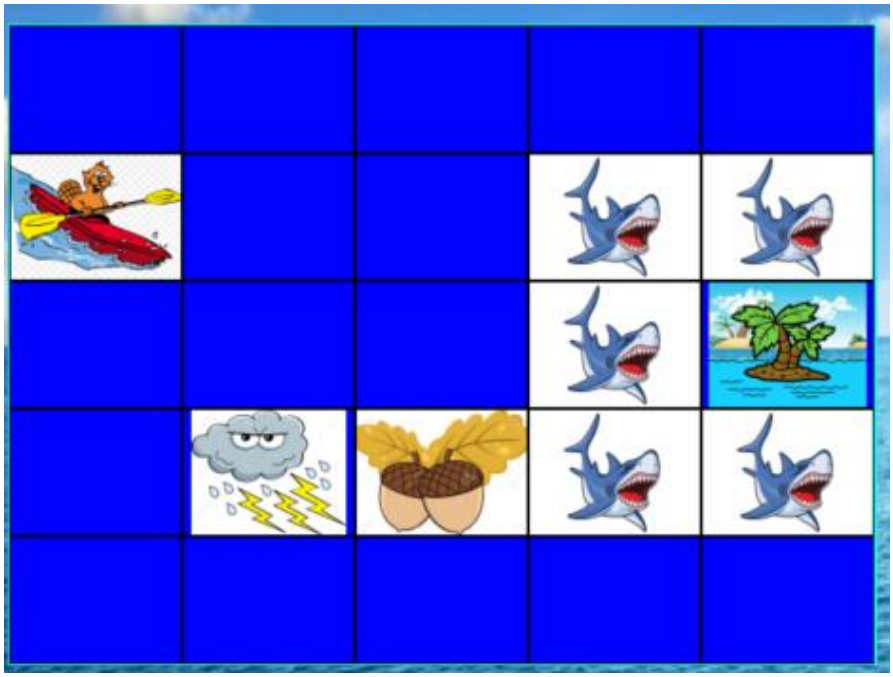
Solución



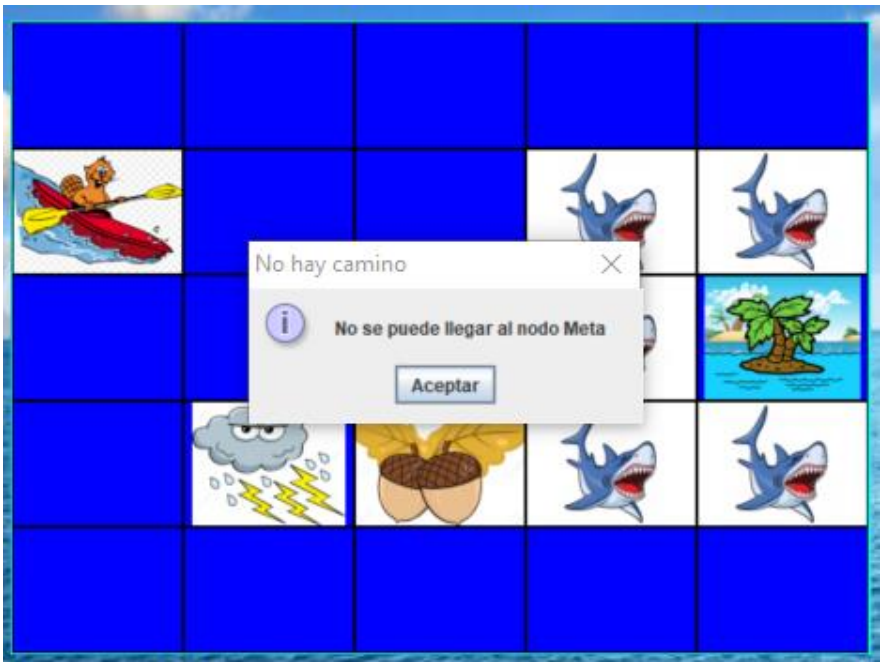
```
Nodos del camino solución:
---------------------------
( 11, 4 )
( 10, 5 )
( 9, 6 )
( 9, 7 )
( 8, 8 )
( 9, 9 )
( 10, 9 )
( 11, 10 )
( 12, 10 )
( 13, 11 )
( 14, 11 )
( 14, 12 )
( 14, 13 )
( 13, 14 )
( 13, 15 )
( 12, 16 )
( 11, 16 )
( 10, 17 )
( 11, 18 )
( 11, 19 )
```

Ejemplo 6

Situación inicial



Solución


No hay camino
No se puede llegar al nodo Meta
Aceptar