Laboratorio de Sistemas Inteligentes 25/26

Sitio: Campus Virtual
Curso: SISTEMAS INTELIGENTES (ESI)
Libro: Laboratorio de Sistemas Inteligentes 25/26

Imprimido por: MANUEL VILLANUEVA ALISES
Día: martes, 2 de diciembre de 2025, 14:49

Descripción

En este libro se fijan los conceptos y objetivos del **Laboratorio de Sistemas Inteligentes** para el curso 2025/26

Tabla de contenidos

1. Tarea 0. Control de versiones del software

El control del software se realiza mediante git con un repositorio **compartido en GitHub** entre los profesores y el estudiante.

Si no conoces **git** pásate por learngitbranching que te ayudará en los primeros pasos.

1.1. Gestión del repositorio en github

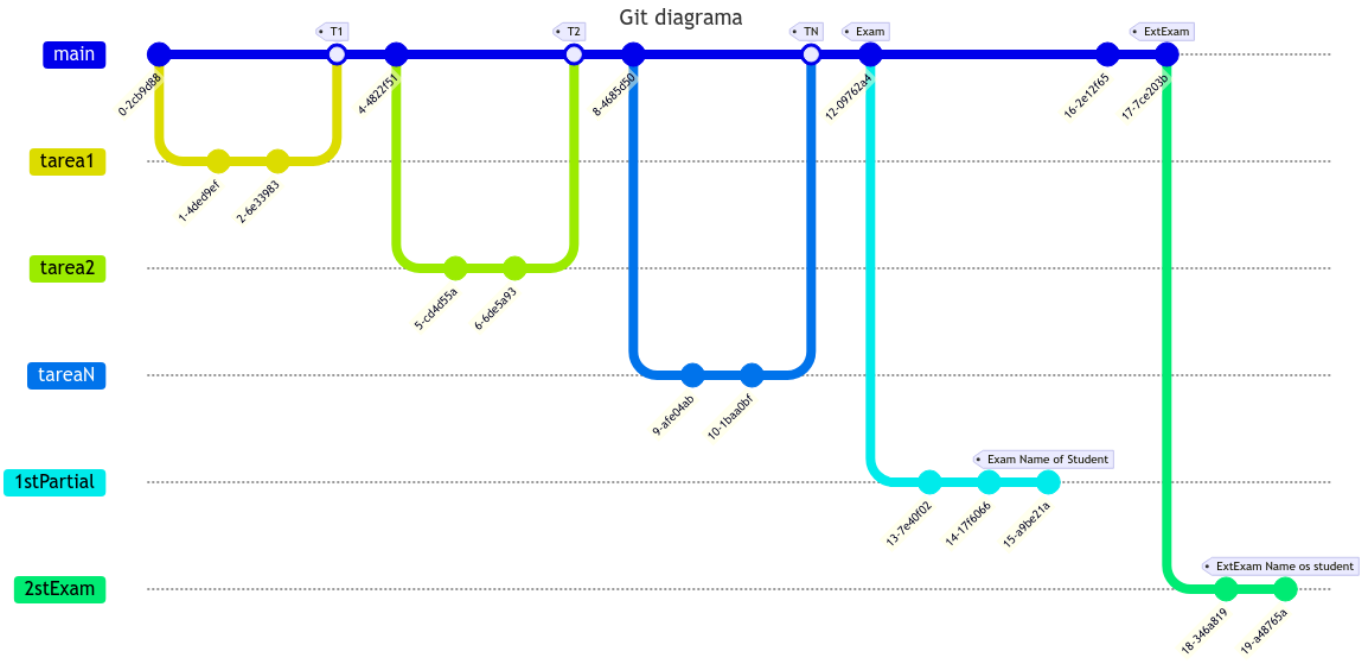
Todos los alumnos han de tener una cuenta en GitHub Education creada como **estudiantes de la UCLM**.

Con la utilización de repositorios se consigue que el **código fuente** de todos las tareas desarrolladas en el laboratorio esté actualizado y **disponible**.

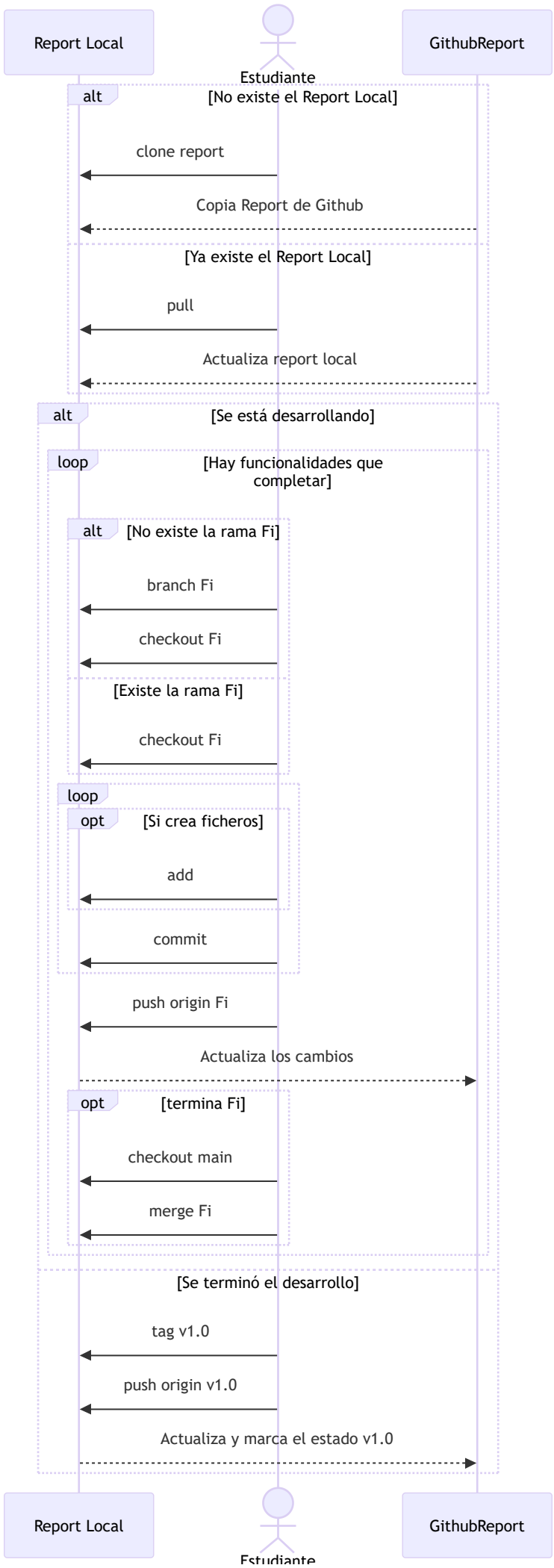
Los profesores habilitarán un enlace en Moodle para que cada estudiante cree su repositorio.

Al acceder al enlace aparece un listado de estudiantes. El estudiante seleccionará su nombre y aceptará la asignación. Entonces, se creará el repositorio de manera automática.

1.2. Esquema temporal de utilización del Repositorio



1.3. Interacciones básicas


Report Local
Estudiante
GithubReport
alt
[No existe el Report Local]
clone report
Copia Report de Github
[Ya existe el Report Local]
pull
Actualiza report local
alt
[Se está desarrollando]
loop
[Hay funcionalidades que completar]
alt
[No existe la rama Fi]
branch Fi
checkout Fi
[Existe la rama Fi]
checkout Fi
loop
opt
[Si crea ficheros]
add
commit
push origin Fi
Actualiza los cambios
opt
[termina Fi]
checkout main
merge Fi
[Se terminó el desarrollo]
tag v1.0
push origin v1.0
Actualiza y marca el estado v1.0
Report Local
Estudiante
GithubReport

Estudiante

1.4. Ejecución del Software y Comentarios

El software que se ha de realizar consistirá en un programa que se ejecutará desde la línea de comandos llamado **rushhour** que ha de admitir una serie de acciones junto con sus parámetros.

A modo de ejemplo, según sea el lenguaje de desarrollo utilizado (se pueden utilizar otros lenguajes diferentes a los de los ejemplos), tendremos:

JAVA

> **java -jar rushhour.jar <acción> <parámetros>**
>
> java -jar rushhour.jar successors -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM
>
> Donde rushhour.jar ha de contener todas las clases necesarias para su ejecución en un entorno estándar de Java

PYTHON

> **python rushhour.py <acción> <parámetros>**
>
> python rushhour.py successors -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM
>
> En el mismo path se ha de disponer de un fichero requirements.txt donde se especifiquen todos los paquetes necesarios que no están en la distribución estándar de Python.

COMENTARIOS

De manera **obligatoria**, todos las procedimientos, funciones o métodos que se implementen **han de tener un comentario** justo antes de que comience el método con dos partes:

1. **QUE**. Donde se describe lo que hace el método de forma concisa.
2. **POR QUE**. Cuál es motivo por el que se ha implementado este método, procedimiento o función.

2. Tarea 1: Dominio del problema

Puzzle Rush Hour (Hora Punta)

¿Qué es Rush Hour?

Rush Hour es un juego de mesa tipo rompecabezas creado por *ThinkFun*. Está diseñado para un solo jugador y simula un atasco de tráfico en una cuadrícula. El *objetivo es liberar un coche rojo* que está atrapado entre otros vehículos, moviéndolos dentro de un espacio limitado.

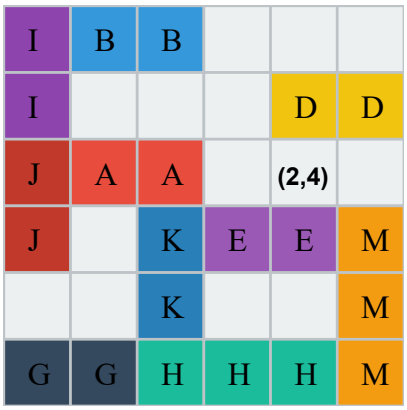
¿Cómo se juega?

1. **Tablero**: Es una cuadrícula de 6x6 casillas.
2. **Vehículos**: Hay coches (2 casillas de largo) y camiones (3 casillas de largo), colocados horizontal o verticalmente.
3. **Coche rojo**: Es el vehículo que debes sacar del atasco. Siempre empieza en una fila central y debe salir por el borde derecho del tablero.
4. **Movimientos**: Sólo puedes mover los vehículos hacia adelante o hacia atrás en la dirección en la que están orientados (no se pueden girar).
5. **Objetivo**: Mover los vehículos de forma estratégica para despejar el camino del coche rojo hacia la salida.

2.1. Nivel Rush Hour

El nivel de Rush Hour se va a representar con una cadena de caracteres que muestra la serialización del tablero con un tamaño 6x6 casillas. Las casillas que ocupan los vehículos se representa por una letra mayúscula, siendo la letra 'A' la que representa al coche rojo, y las casillas vacías se representa por la letra 'o' (o minúscula).

Por ejemplo la cadena IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM representa el tablero mostrado en la siguiente figura, donde también se indica la posición de la casilla (2,4) , fila 2 y columna 4, que está vacía, para especificar cómo se van a identificar las posiciones.



La **casilla de salida** es (2,5). El puzzle se resuelve cuando alguna de las casillas del coche rojo 'A' es la (2,5).

2.2. Resultados para la tarea T1

Desarrollar dos acciones: **verify** y **question** que servirán para verificar que una cadena de entrada corresponde con un nivel válido y para realizar distintas preguntas sobre un determinado nivel, respectivamente.

Verificar un nivel

La acción se invocará como **rushhour verify -s <cadena del nivel>** y la salida será un entero dependiendo del error, tal y como se muestra a continuación:

1. Si el tamaño del string no se corresponde con un nivel de 6x6.
2. Si el string contiene caracteres no válidos (no ('A'-'Z' u 'o')).
3. No está el vehículo rojo ('A') en el string.
4. Si el vehículo rojo no está en la fila de salida.
5. Si el vehículo rojo no está situado horizontalmente.
6. Vehículo con longitud mayor que 3 o menor que 2.
7. Si hay vehículos con la misma letra duplicados (deben estar separados por una casilla vacía).

Si no hay error el resultado a devolver es 0.

Cuestiones sobre el nivel

La acción se invocara como **rushhour question -s <cadena de nivel> <diferentes parámetros>**

Dependiendo de la pregunta concreta los diferentes parámetros son:

1. rushhour **question** -s <cadena de nivel> **--whereis** <*vehículo*>. Devuelve la lista de casillas (fila,columna) del vehículo. Al presentar la lista de posiciones no aparecerá ningún espacio en blanco.
2. rushhour **question** -s <cadena de nivel> **--what** <*celda*>. Que hay en la celda f,c
3. rushhour **question** -s <cadena de nivel> **--size** <*vehículo*>. Que tamaño tiene el vehículo <vehículo>.
4. rushhour **question** -s <cadena de nivel> **--howmany**. Cantidad de vehículos.

Ejemplos

```
rushhour  verify -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLoo
0
rushhour  verify -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLo
1
rushhour  verify -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIoGLoo
6
rushhour  verify -s BBJoooHoJDDMHAAooMHoKEEMIoKoFFIGGKKo
7
rushhour  question -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLoo --whereis I
(4,0)(5,0)

rushhour  question -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLoo --howmany
12

rushhour  question -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLoo --size F
2

rushhour question -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLoo --what 5,3
L

rushhour question -s BBJoooHoJDDMHAAooMHoKEEMIoKLFFIGGLoo --what 0,3
o
```

2.3. Entrega y Plazos

El plazo aproximado para la realización de la primera tarea es de dos semanas.

La **rama T1** correspondiente a esta tarea **no puede ser modificada** en las 24 horas anteriores de la sesión de cada grupo.

Por tanto, las fechas límite de modificación de la rama son:

Grupo B1: Miércoles 8 de Octubre. 11:30h.

Grupo B2: Domingo 5 de Octubre. 11:30h.

Grupo C1: Martes 7 de Octubre. 13:00h.

Grupo C2: Miércoles 8 de Octubre: 13:00h.

2.4. Tests

verify -s IBBoooIoooDDJAAoooJoKEEMooKooM	1
verify -s IBBoooIoooDDJAAoooJo	1
verify -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHMX	1
verify -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHH	1
verify -s *BBoooIoooDDJAAoooJoKEEMooKooMGGHHHM	2
verify -s IBBoooIooo*DJAAoooJoKEEMooKooMGGHHHM	2
verify -s IBBoooIoooDDJAAoooJo*EEMooKooMGGHHHM	2
verify -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHH*	2
verify -s IBBoooIoooDDJooooJoKEEMooKooMGGHHHM	3
verify -s IBBoooIoooDDJooooJoKEEMooKooMGGHHHM	3
verify -s IBBoooIoooDDJooooJoKEEMooKooMGGHHHM	3
verify -s IBBoooIoooDDJooooJoKEEMooKooMGGHHHM	3
verify -s IBBoooIAAoDDJooooJoKEEMooKooMGGHHHM	4
verify -s IBBoooIoooDDJooooJoKEEMAAKooMGGHHHM	4
verify -s IBBAAoIoooDDJooooJoKEEMooKooMGGHHHM	4
verify -s IBBoooIoAADDJooooJoKEEMooKooMGGHHHM	4
verify -s ooooooooooooAooooAooooooooooooooooo	5
verify -s ooooooooooooooAooooAooooooooooooooo	5
verify -s ooooooooooooooAooooAooooooooooooooo	5
verify -s oooooooooooooooAooooAooooooooooooo	5
verify -s IBBZooIoooDDJAAoooJoKEEMooKooMGGHHHM	6
verify -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHH	6
verify -s IBBoooIoooDDJAAoooJoKEEMBBKooMGGHHHM	7
verify -s IBBoooIoooDDJAAoooJoKEEMoKKooMGGHHHM	7

question -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM --whereis A	(2,1)(2,2)
question -s BBDDCoEoooCKEoAACKGooooGooooGoHHHo --whereis B	(0,0)(0,1)
question -s BBDDCoEoooCKEoAACKGooooGooooGoHHHo --whereis C	(0,4)(1,4)(2,4)
question -s IBBBoDIooJoDoAAJooooJoMKKoooMooHHHo --whereis H	(5,2)(5,3)(5,4)
question -s IoBBCCIDooooIDoAAoooJooMKKJooMoooHHH --whereis J	(3,2)(4,2)
question -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM --what 2,2	A
question -s BBDDCoEoooCKEoAACKGooooGooooGoHHHo --what 0,0	B
question -s IoBBCCIDooooIDoAAoooJooMKKJooMoooHHH --what 5,5	H
question -s IBBBoDIooJoDoAAJooooJoMKKoooMooHHHo --what 3,5	M
question -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM --what 1,4	D

question -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM --size A	2
question -s BBDDCoEoooCKEoAACKGooooGooooGoHHHo --size B	2
question -s BBDDCoEoooCKEoAACKGooooGooooGoHHHo --size C	3
question -s IBBBoDIooJoDoAAJooooJoMKKoooMooHHHo --size H	3
question -s IoBBCCIDooooIDoAAoooJooMKKJooMoooHHH --size M	2
question -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM --howmany	10
question -s BBDDCoEoooCKEoAACKGooooGooooGoHHHo --howmany	8
question -s IBBBoDIooJoDoAAJooooJoMKKoooMooHHHo --howmany	8
question -s IoBBCCIDooooIDoAAoooJooMKKJooMoooHHH --howmany	9
question -s IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM --howmany	10

3. Tarea 2: Problema (Definición)

Como se indicó en la tarea anterior, el objetivo de este laboratorio es obtener una secuencia de acciones que permitan salir del atasco al vehículo rojo.

Antes de encontrar una solución al problema, éste ha de formularse de manera correcta. Para esto se tienen que especificar:

1. El **espacio de estados**: Representación de Estado y Función Sucesores
2. El **estado Inicial**.
3. La **función objetivo.**

3.1. Espacio de Estados

ESTADO

Un estado ***es la descripción de la situación en un instante de tiempo determinado***. Lo más habitual para representar un estado es determinar qué elementos del problema/entorno/agente son relevantes, los identificaremos como ***variables*** y determinaremos sus ***dominios*** como los posibles valores que puedan tomar.

Establecidas las variables y sus dominios, un ***estado es una asignación concreta de valores a dichas variables***.

Variables y Dominios

En nuestro caso se va a **seleccionar como representación de estado la cadena de texto del nivel Rush Hour tal y como se estableció en la Tarea 1, por ejemplo,** IBBoooIoooDDJAAoooJoKEEMooKooMGGHHHM

FUNCIÓN SUCESOR

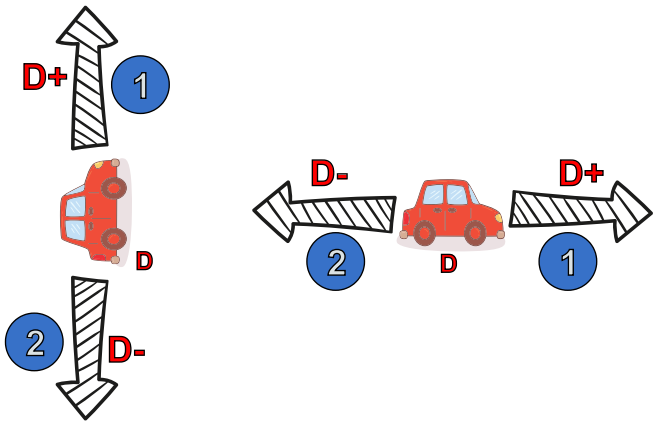
En el problema propuesto, la única forma de modificar un estado es mediante las siguientes acciones: **Mover un vehículo un número de casillas**.

Movimiento de un vehículo.

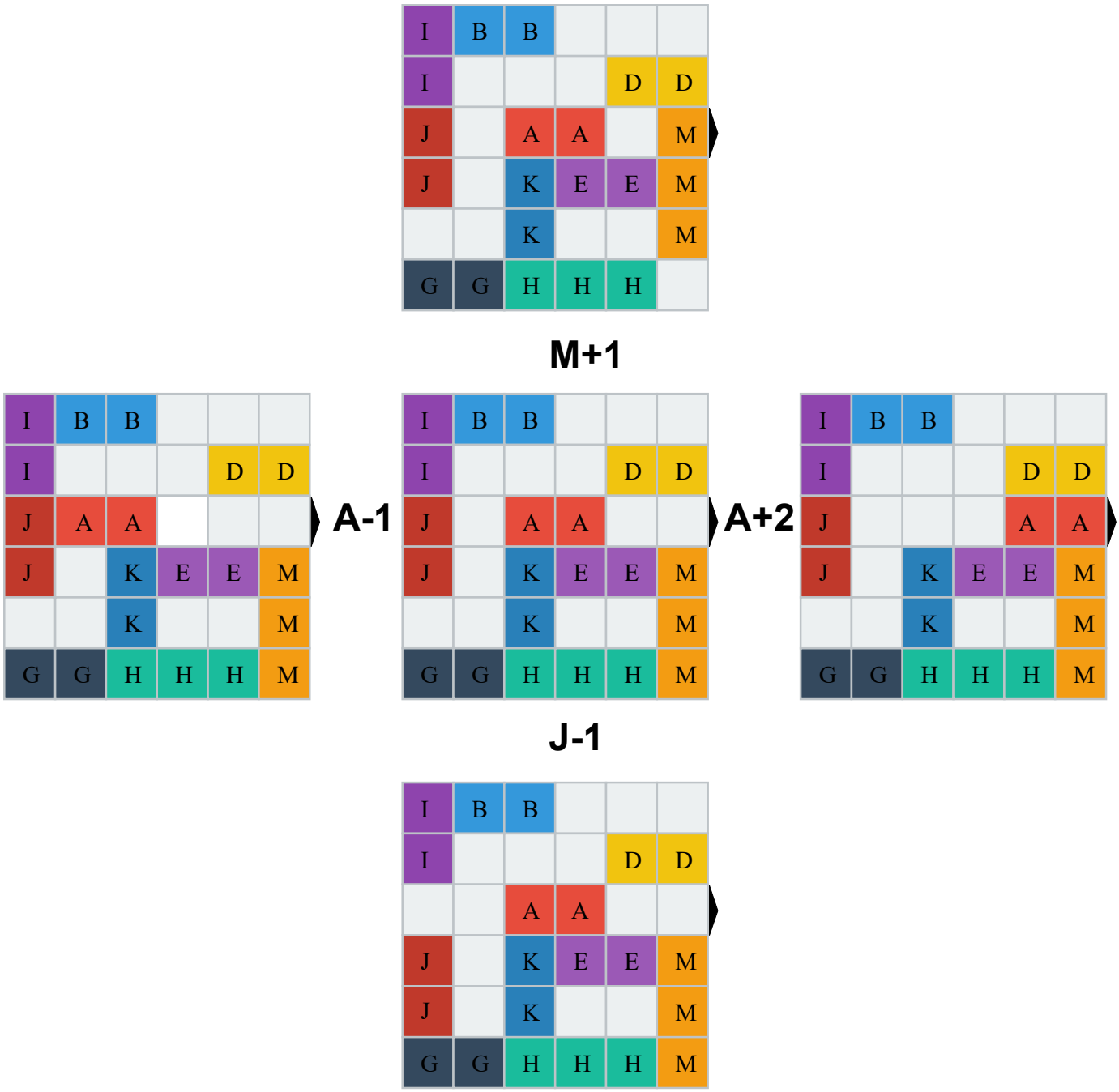
Un vehículo puede encontrarse en una posición vertical u horizontal luego podemos moverlo hacia arriba o hacia abajo, hacia la derecha o izquierda un número de casillas. Si un vehículo se puede desplazar un máximo de n casillas, también se puede desplazar n-1, n-2, .., 1 casilla. De esta forma se puede nombrar de forma unívoca cualquier movimiento con la siguiente representación:

<vehículo><dirección><número de casillas>

En las siguientes figuras se muestras como nombrar las posibles direcciones y movimiento y cual de ellos se realizará antes (etiquetas 1 y 2)



En la siguiente figura se muestra un ejemplo de diferentes movimientos **a partir del nivel rush hour del centro de la imagen.**



Costo:

Para terminar de definir los elementos necesarios para la función sucesores, se debe establecer el **costo** de cada acción. Este se establecerá para cualquier acción con un valor igual a **seis menos el número de casillas que se ha desplazado el vehículo.**

Sucesor.

Por tanto, un sucesor **SUC** para un estado vendrá dado por la tupla:

[<acción>, <nuevo_estado>, <costo>]

Por tanto, la función SUCESORES del estado **e** será una lista de tripletas: <acción_válida, nuevo_estado, costo>:

SUCESORES(e)=[<SUC1>,<SUC2>,..,]

Para establecer una uniformidad en la lista de sucesores se consideran las siguientes reglas:

1. Los vehículos que se puede mover se mueven en **orden alfabético**, primero el vehículo A, luego el B si lo hubiera y así sucesivamente.
2. Si un vehículo puede desplazarse en ambos sentidos primero el **sentido indicado con +,** y luego el **indicado con -**
3. **Orden creciente del número casillas movidas** cuando hay movimientos de varias casillas.

En la figura anterior esta es la secuencia ordenada de movimientos posibles : A+1,A+2,A-1,B+1,B+2,B+3,D-1,D-2,D-3,J-1,M+1

3.2. Problema

Definido el espacio de estados con la representación de estado y la función sucesores, se debe definir un estado inicial y una función objetivo.

ESTADO INICIAL

Cualquier nivel de Rush Hour válido.

FUNCIÓN OBJETIVO

La forma más directa de reconocer que en un estado **e** se ha resuelto un nivel es cuando alguna de las casillas del vehículo A coincide con la casilla de salida (2,5).

FUNCION_OBJETIVO(e)=Alguna casilla de 'A' es la (2,5)

3.3. Resultados

Desarrollar una nueva acción **successors** para rushhour que admita un parámetro -s <nivel> que devuelva la lista ordenada de los sucesores del nivel. Además, añadir dos nuevas cuestiones a la acción **question**: **--goal** que devuelva TRUE o FALSE si el nivel es objetivo, y **--move <lista de acciones separadas por comas>** que devuelva el estado resultado tras aplicar la lista de acciones de forma consecutiva.

Ejemplo

rushhour successors -s IBBooolooooDDJoAAooJoKEEMooKooMGGHHHM
[A+1,IBBooolooooDDJooAAoJoKEEMooKooMGGHHHM,5]
[A+2,IBBooolooooDDJoooAAJoKEEMooKooMGGHHHM,4]
[A-1,IBBooolooooDDJAAoooJoKEEMooKooMGGHHHM,5]
[B+1,IoBBooolooooDDJoAAooJoKEEMooKooMGGHHHM,5]
[B+2,IooBBoIooooDDJoAAooJoKEEMooKooMGGHHHM,4]
[B+3,IoooBBIooooDDJoAAooJoKEEMooKooMGGHHHM,3]
[D-1,IBBooolooDDoJoAAooJoKEEMooKooMGGHHHM,5]
[D-2,IBBoooIoDDooJoAAooJoKEEMooKooMGGHHHM,4]
[D-3,IBBooolDDoooJoAAooJoKEEMooKooMGGHHHM,3]
[J-1,IBBooolooooDDooAAooJoKEEMJoKooMGGHHHM,5]
[M+1,IBBooolooooDDJoAAoMJoKEEMooKooMGGHHHo,5]

rushhour question -s IBBooolooooDDJoAAooJoKEEMooKooMGGHHHM --goal
FALSE

rushhour question -s IBBooolooooDDJoAAooJoKEEMooKooMGGHHHM --move A+1,A+1
IBBooolooooDDJoooAAJoKEEMooKooMGGHHHM

3.4. Entrega y Plazos

El plazo aproximado para la realización de la segunda tarea es de **tres semanas.**

La **rama T2** correspondiente a esta tarea **no puede ser modificada** en las 24 horas anteriores a la fecha de entrega de cada grupo.

Por tanto, las fechas límite de modificación de la rama son:

Grupo B1: Miércoles 29 de Octubre. 11:30h.

Grupo B2: Domingo 26 de Octubre. 11:30h.

Grupo C1: Martes 28 de Octubre. 13:00h.

Grupo C2: Miércoles 29 de Octubre: 13:00h.

4. Tarea 3: Algoritmo de búsqueda

En esta tarea desarrollaremos el **algoritmo de búsqueda** general para lo que necesitaremos desarrollar previamente los artefactos **frontera**, **nodo del árbol** y conjunto de **estados visitados** .

4.1. La frontera

La **frontera** ha de ser una estructura que almacene nodos del árbol de búsqueda.

Las características que le pedimos son:

1. Debe ser una **estructura ordenada** por el *valor* de los nodos del árbol. Si existen varios nodos con *igual valor se ordenaran estos por los ID's* de los nodos.
2. Por razones de eficiencia la frontera tendrá una **inserción ordenada**, evitando su ordenación tras cada inserción.

4.2. Nodos del árbol de búsqueda

Un nodo del árbol de búsqueda deberán tener los siguiente elementos:

- **ID**: Un número entero. Es único para cada nodo y se van asignando de forma creciente.
- **Padre**: Algún tipo de referencia o valor del nodo padre que lo generó o el valor **none** si no existe.
- **Estado**: Un estado. Es el estado del nodo
- **Valor**: Un número entero. Este valor se calculará mediante las diferentes estrategias como se ha visto en clase.
- **Profundidad**: Un número entero.
- **Costo**: El costo del camino hasta el nodo considerando el número de acciones ejecutadas.
- **Heurística**: Un número real. El valor de la función heurística para el estado del nodo. Su valor concreto se verá en la siguiente tarea,
- **Acción**: Sería el nombre de la acción, siguiendo la sintaxis explicada en la Tarea 2, que ha generado dicho nodo**.** La acción de no hacer nada, la acción que genera el nodo raíz, se representa como '___' .

Y una **función camino** que devuelva una lista de nodos desde el nodo raíz a dicho nodo.

Un nodo tendrá una representación textual como una cadena con la forma:

[<Node ID>,<Parent ID>,<ACCIÓN>,<str ESTADO>,<COSTO>,<PROFUNDIDAD>,<HEURISTICA>,<VALOR>]

Los elementos **<PROFUNDIDAD>,<COSTO>, <HEURISTICA> y <VALOR>** son números enteros.

En las estrategias sin heurística su valor sera **0**

Por ejemplo:

[10059,9849,L+1,BBIKLMooIKLMAAoooMGHJCCCGHJoDDGoEEFF,183,42,0,42]

4.3. Expansión de un nodo del árbol

Expansión

La expansión de un nodo se realiza mediante la *lista de sucesores de su estado*, manteniendo el orden establecido para dicha lista.

El **nodo.valor** se asignará según la estrategia propuesta:

- **Anchura:** nodo.valor <- nodo.profundidad
- **Profundidad**: nodo.valor <- - nodo.profundidad
- **Costo Uniforme**: nodo.valor <- costo.distancia recorrida

Los demás valores de un nodo se asignarán de la siguiente forma:

- **nodo.ID**<-Total de nodos + 1
- **nodo.padre** <-padre
- **nodo.profundidad** <- padre.profundidad + 1
- **nodo.costo** <- padre.costo + step(cost)
- **nodo.acción** <- nombre de la acción
- **nodo.estado** <- estado siguiente del sucesor considerado

4.4. Conjunto de Estados Visitados

La implementación de la poda de nodos con estados visitados se realizará tal y como se ha explicado en clase de teoría.

4.5. Estadísticas

El algoritmo de búsqueda, dependiendo de las estrategias y las decisiones tomadas de implementación, realiza su objetivo con distintos desempeños y eficiencia.

Con el objetivo de tener una estimaciones de la complejidad y la eficiencia en la implementación definiremos un conjunto de estadísticas.

- **ET**: Tiempo de ejecución del algoritmo en microsegundos. Se tomará el momento t0 justo antes de comenzar el bucle principal y el momento cuando se abandona el bucle t1. El ET se calcula como t1-t0 en microsegundos.
- **TN**: Número total de nodos generados al terminar el algoritmo.
- **CN:** Número total de nodos podados. Incluye también los podados por superar la profundidad máxima (CUT_DEPTH).
- **EN**: Número de nodos expandidos.
- **DF**: Profundidad máxima alcanzada.

4.6. Resultados

Desarrollar una nueva acción **solver** para rushhour que admita los parámetros: **-s** cuyo valor sea una cadena que representa el nivel que queremos resolver , **--strategy** cuyo valor será la cadena que representa la estrategia a utilizar entre BFS (anchura), DFS (Profundidad) y UCS (Costo uniforme) y el parámetro **--depth** cuyo valor será un valor entero para la máxima profundidad si la estrategia es DFS.

El resultado será: **La secuencia de nodos del camino solución.**

Ejemplo:

python rushhour.py solver -s oooGoooFoGBBEFAAHJEoCCHJEDDDIJooooIo --strategy DFS --depth 20

java -jar rushhour.jar solver -s oooGoooFoGBBEFAAHJEoCCHJEDDDIJooooIo --strategy DFS --depth 20

Que genera la salida:

[0,none,___,oooGoooFoGBBEFAAHJEoCCHJEDDDIJooooIo,0,0,0,0]
[1,0,C-1,oooGoooFoGBBEFAAHJECCoHJEDDDIJooooIo,5,1,0,-1]
[9,1,E+1,oooGooEFoGBBEFAAHJECCoHJoDDDIJooooIo,10,2,0,-2]
[14,9,C+1,oooGooEFoGBBEFAAHJEoCCHJoDDDIJooooIo,15,3,0,-3]
[22,14,D-1,oooGooEFoGBBEFAAHJEoCCHJDDDoIJooooIo,20,4,0,-4]
[29,22,C-1,oooGooEFoGBBEFAAHJECCoHJDDDoIJooooIo,25,5,0,-5]
[37,29,E+1,EooGooEFoGBBEFAAHJoCCoHJDDDoIJooooIo,30,6,0,-6]
[40,37,C+1,EooGooEFoGBBEFAAHJooCCHJDDDoIJooooIo,35,7,0,-7]
[47,40,C-2,EooGooEFoGBBEFAAHJCCooHJDDDoIJooooIo,39,8,0,-8]
[55,47,D+1,EooGooEFoGBBEFAAHJCCooHJoDDDIJooooIo,44,9,0,-9]
[58,55,C+1,EooGooEFoGBBEFAAHJoCCoHJoDDDIJooooIo,49,10,0,-10]
[63,58,C+1,EooGooEFoGBBEFAAHJooCCHJoDDDIJooooIo,54,11,0,-11]
[77,63,F+1,EFoGooEFoGBBEoAAHJooCCHJoDDDIJooooIo,59,12,0,-12]
[4826,77,A-1,EFoGooEFoGBBEAAoHJooCCHJoDDDIJooooIo,64,13,0,-13]
[4837,4826,C-1,EFoGooEFoGBBEAAoHJoCCoHJoDDDIJooooIo,69,14,0,-14]
[4848,4837,D-1,EFoGooEFoGBBEAAoHJoCCoHJDDDoIJooooIo,74,15,0,-15]
[5576,4848,G-3,EFooooEFooBBEAAoHJoCCGHJDDDGIJooooIo,77,16,0,-16]
[6837,5576,B-2,EFooooEFBBooEAAoHJoCCGHJDDDGIJooooIo,81,17,0,-17]
[7049,6837,H+2,EFooHoEFBBHoEAAooJoCCGoJDDDGIJooooIo,85,18,0,-18]
[7119,7049,J-1,EFooHoEFBBHoEAAooooCCGoJDDDGIJoooolJ,90,19,0,-19]
[7180,7119,A+3,EFooHoEFBBHoEoooAAoCCGoJDDDGIJoooolJ,93,20,0,-20]

Mostrando las estadísticas

python rushhour.py solver -s oooGoooFoGBBEFAAHJEoCCHJEDDDIJooooIo --strategy DFS --depth 20 --stats

java -jar rushhour.jar solver -s oooGoooFoGBBEFAAHJEoCCHJEDDDIJooooIo --strategy DFS --depth 20 --stats

Que genera la salida:

[0,none,___,oooGoooFoGBBEFAAHJEoCCHJEDDDIJooooIo,0,0,0,0]
[1,0,C-1,oooGoooFoGBBEFAAHJECCoHJEDDDIJooooIo,5,1,0,-1]
[9,1,E+1,oooGooEFoGBBEFAAHJECCoHJoDDDIJooooIo,10,2,0,-2]
[14,9,C+1,oooGooEFoGBBEFAAHJEoCCHJoDDDIJooooIo,15,3,0,-3]
[22,14,D-1,oooGooEFoGBBEFAAHJEoCCHJDDDoIJooooIo,20,4,0,-4]
[29,22,C-1,oooGooEFoGBBEFAAHJECCoHJDDDoIJooooIo,25,5,0,-5]
[37,29,E+1,EooGooEFoGBBEFAAHJoCCoHJDDDoIJooooIo,30,6,0,-6]
[40,37,C+1,EooGooEFoGBBEFAAHJooCCHJDDDoIJooooIo,35,7,0,-7]
[47,40,C-2,EooGooEFoGBBEFAAHJCCooHJDDDoIJooooIo,39,8,0,-8]
[55,47,D+1,EooGooEFoGBBEFAAHJCCooHJoDDDIJooooIo,44,9,0,-9]
[58,55,C+1,EooGooEFoGBBEFAAHJoCCoHJoDDDIJooooIo,49,10,0,-10]
[63,58,C+1,EooGooEFoGBBEFAAHJooCCHJoDDDIJooooIo,54,11,0,-11]
[77,63,F+1,EFoGooEFoGBBEoAAHJooCCHJoDDDIJooooIo,59,12,0,-12]
[4826,77,A-1,EFoGooEFoGBBEAAoHJooCCHJoDDDIJooooIo,64,13,0,-13]
[4837,4826,C-1,EFoGooEFoGBBEAAoHJoCCoHJoDDDIJooooIo,69,14,0,-14]
[4848,4837,D-1,EFoGooEFoGBBEAAoHJoCCoHJDDDoIJooooIo,74,15,0,-15]
[5576,4848,G-3,EFooooEFooBBEAAoHJoCCGHJDDDGIJooooIo,77,16,0,-16]
[6837,5576,B-2,EFooooEFBBooEAAoHJoCCGHJDDDGIJooooIo,81,17,0,-17]
[7049,6837,H+2,EFooHoEFBBHoEAAooJoCCGoJDDDGIJooooIo,85,18,0,-18]
[7119,7049,J-1,EFooHoEFBBHoEAAooooCCGoJDDDGIJoooolJ,90,19,0,-19]
[7180,7119,A+3,EFooHoEFBBHoEoooAAoCCGoJDDDGIJoooolJ,93,20,0,-20]

ET: 804814
TN: 7192
EN: 866

CN: 6226
DF: 20

4.7. Entrega

El plazo para la realización de esta tarea es de **tres semanas.**

La rama **T3 no puede ser modificada** en las 24 horas anteriores de la sesión de cada grupo.

Por tanto, las fechas límite de modificación de la rama son:

Grupo B1: Miércoles 19 de Noviembre. 11:30h.

Grupo B2: Domingo 16 de Noviembre 11:30h.

Grupo C1: Martes 18 de Noviembre. 13:00h.

Grupo C2: Miércoles 19 de Noviembre 13:00h.

4.8. Configuración Tests Automaticos. Tareas T1 y T2.

Antes de realizar cualquier operación, el código de las ramas T1 y T2 debe estar fusionado en la rama main ya que los tests sólo se ejecutan en dicha rama.

Si no se pasan los tests, las modificaciones para corregir los errores se deben realizar en la rama actual de trabajo, es decir **en la rama T3.**

Los tests sólo los puede ejecutar el profesor.

Para automatizar la tarea de la ejecución de la batería de tests es necesario que cada estudiante cree un fichero de texto en **la rama main** denominado **test.txt** con dos líneas:

1. Lenguaje de programación utilizado en mayúsculas: JAVA o PYTHON (Otro lenguaje es necesario hablarlo con el profesor de prácticas).
2. El path relativo al repositorio donde se encuentra el fichero rushhour.jar o rushour.py.

Un ejemplo sería:

```
PYTHON
src
```

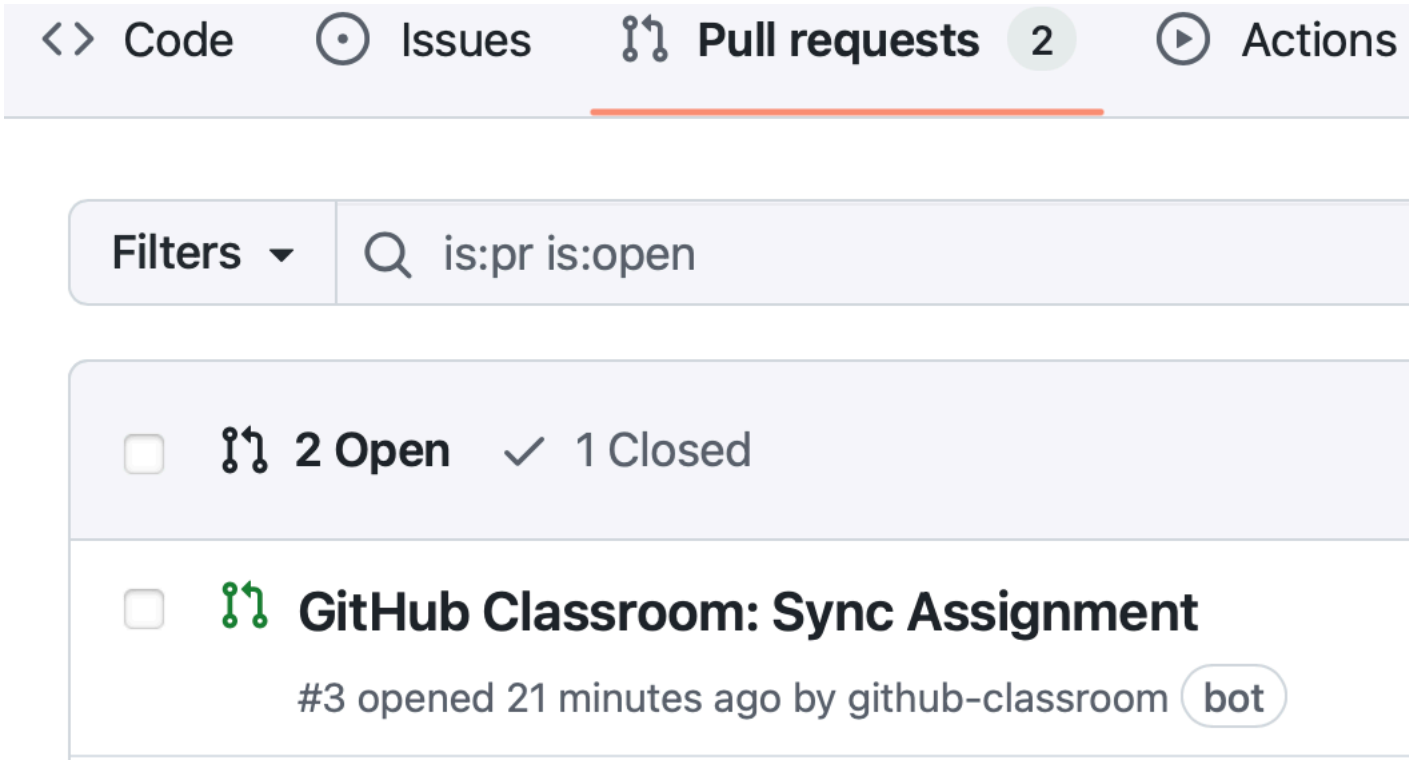
Que indicaría que se ha desarrollado en Python y que el fichero rushhour.py se encuentra en ./src/rushhour.py.

Otro ejemplo sería:

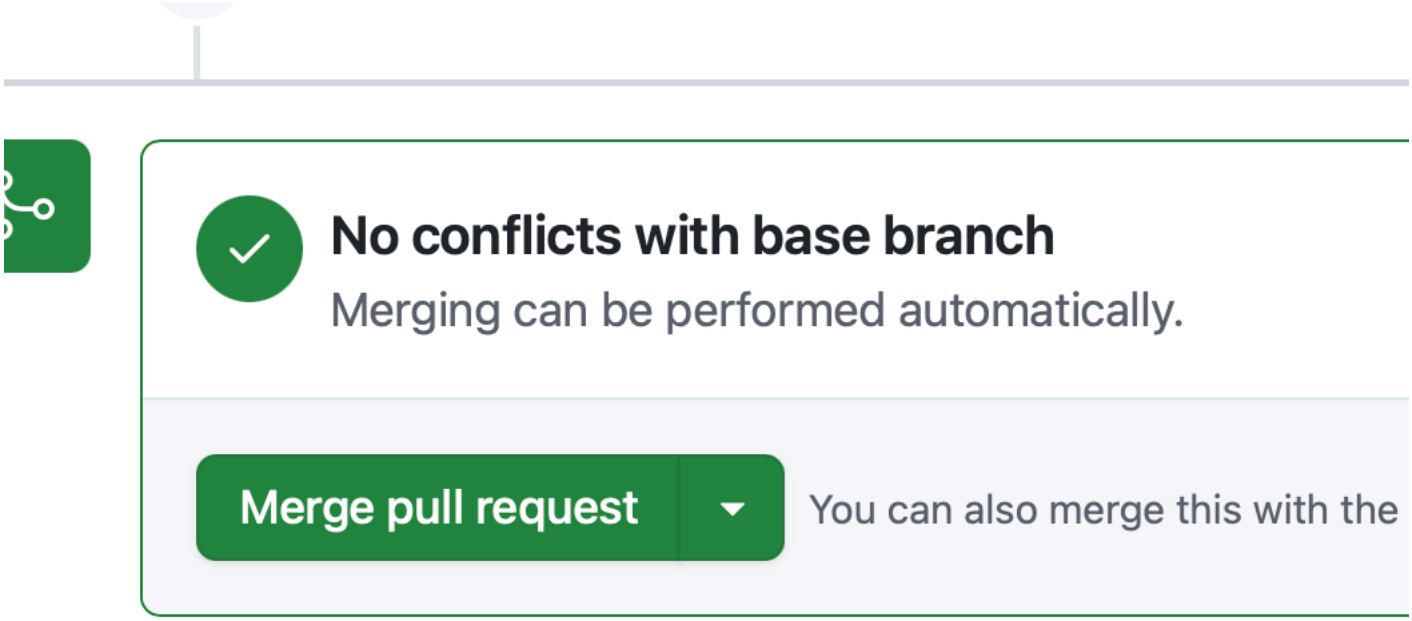
```
JAVA
.
```

Que indicaría que se ha desarrollado en Java y que el fichero rushhour.jar se encuentra en ./rushhour.jar

Posteriormente, en el laboratorio, el estudiante debe esperar la indicación del profesor y aceptar el siguiente pull request:



pulsando sobre "Merge Pull Request".


No conflicts with base branch
Merging can be performed automatically.
Merge pull request
You can also merge this with the

4.9. Tests Adicionales a los automáticos

`solver -s IBBoooIooLDDJAALooJoKEEMFFKooMGGHHHM --depth 10 --strategy DFS`	`[0,none,___,IBBoooIooLDDJAALooJoKEEMFFKooMGGHHHM,0,0,0,0]`
	`[1,0,B+1,IoBBooIooLDDJAALooJoKEEMFFKooMGGHHHM,5,1,0,-1]`
	`[6,1,B+1,IooBBoIooLDDJAALooJoKEEMFFKooMGGHHHM,10,2,0,-2]`
	`[10,6,B+1,IoooBBIooLDDJAALooJoKEEMFFKooMGGHHHM,15,3,0,-3]`
	`[17,10,L+1,IooLBBIooLDDJAAoooJoKEEMFFKooMGGHHHM,20,4,0,-4]`
	`[19,17,A+1,IooLBBIooLDDJoAAooJoKEEMFFKooMGGHHHM,25,5,0,-5]`
	`[24,19,A+1,IooLBBIooLDDJooAAoJoKEEMFFKooMGGHHHM,30,6,0,-6]`
	`[28,24,A+1,IooLBBIooLDDJoooAAJoKEEMFFKooMGGHHHM,35,7,0,-7]`

`solver -s IBBoooIooLDDJAALooJoKEEMFFKooMGGHHHM --depth 10 --strategy DFS --stats`	`[0,none,___,IBBoooIooLDDJAALooJoKEEMFFKooMGGHHHM,0,0,0,0]`
	`[1,0,B+1,IoBBooIooLDDJAALooJoKEEMFFKooMGGHHHM,5,1,0,-1]`
	`[6,1,B+1,IooBBoIooLDDJAALooJoKEEMFFKooMGGHHHM,10,2,0,-2]`
	`[10,6,B+1,IoooBBIooLDDJAALooJoKEEMFFKooMGGHHHM,15,3,0,-3]`
	`[17,10,L+1,IooLBBIooLDDJAAoooJoKEEMFFKooMGGHHHM,20,4,0,-4]`
	`[19,17,A+1,IooLBBIooLDDJoAAooJoKEEMFFKooMGGHHHM,25,5,0,-5]`
	`[24,19,A+1,IooLBBIooLDDJooAAoJoKEEMFFKooMGGHHHM,30,6,0,-6]`
	`[28,24,A+1,IooLBBIooLDDJoooAAJoKEEMFFKooMGGHHHM,35,7,0,-7]`
	
	`ET: 1381`
	`TN: 35`
	`EN: 7`
	`CN: 3`
	`DF: 7`

`solver -s ooJBBMCCJKoMIAAKooIDDLEEFFFLooooHHHo --strategy BFS --stats`	`[0,none,___,ooJBBMCCJKoMIAAKooIDDLEEFFFLooooHHHo,0,0,0,0]`
	`[3,0,H-2,ooJBBMCCJKoMIAAKooIDDLEEFFFLooHHHooo,4,1,0,1]`
	`[16,3,L-1,ooJBBMCCJKoMIAAKooIDDoEEFFFLooHHHLoo,9,2,0,2]`
	`[34,16,E-1,ooJBBMCCJKoMIAAKooIDDEEoFFFLooHHHLoo,14,3,0,3]`
	`[64,34,M-3,ooJBBoCCJKooIAAKooIDDEEMFFFLoMHHHLoo,17,4,0,4]`
	`[115,64,B+1,ooJoBBCCJKooIAAKooIDDEEMFFFLoMHHHLoo,22,5,0,5]`
	`[212,115,K+1,ooJKBBCCJKooIAAoooIDDEEMFFFLoMHHHLoo,27,6,0,6]`
	`[362,212,A+3,ooJKBBCCJKooIoooAAIDDEEMFFFLoMHHHLoo,30,7,0,7]`
	
	`ET: 18729`
	`TN: 614`
	`EN: 101`
	`CN: 261`
	`DF: 8`

`solver -s ooJBBMCCJKoMIAAKooIDDLEEFFFLooooHHHo --strategy UCS --stats`	`[0,none,___,ooJBBMCCJKoMIAAKooIDDLEEFFFLooooHHHo,0,0,0,0]`
	`[3,0,H-2,ooJBBMCCJKoMIAAKooIDDLEEFFFLooHHHooo,4,1,0,4]`
	`[8,3,L-1,ooJBBMCCJKoMIAAKooIDDoEEFFFLooHHHLoo,9,2,0,9]`
	`[24,8,E-1,ooJBBMCCJKoMIAAKooIDDEEoFFFLooHHHLoo,14,3,0,14]`
	`[55,24,M-4,ooJBBoCCJKooIAAKooIDDEEoFFFLoMHHHLoM,16,4,0,16]`
	`[87,55,B+1,ooJoBBCCJKooIAAKooIDDEEoFFFLoMHHHLoM,21,5,0,21]`
	`[163,87,K+1,ooJKBBCCJKooIAAoooIDDEEoFFFLoMHHHLoM,26,6,0,26]`
	`[295,163,A+3,ooJKBBCCJKooIoooAAIDDEEoFFFLoMHHHLoM,29,7,0,29]`
	
	`ET: 8703`
	`TN: 450`
	`EN: 78`
	`CN: 190`
	`DF: 7`

solver -s BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo --depth 30 --strategy DFS --stats	[0,none,___,BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo,0,0,0,0] [1,0,B+1,oBBJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo,5,1,0,-1] [8,1,G+1,oBBJooGHoJKLGHAAKLoDDDKMooIEEMFFIooo,10,2,0,-2] [14,8,B-1,BBoJooGHoJKLGHAAKLoDDDKMooIEEMFFIooo,15,3,0,-3] [23,14,D-1,BBoJooGHoJKLGHAAKLDDDoKMooIEEMFFIooo,20,4,0,-4] [29,23,B+1,oBBJooGHoJKLGHAAKLDDDoKMooIEEMFFIooo,25,5,0,-5] [36,29,G+1,GBBJooGHoJKLoHAAKLDDDoKMooIEEMFFIooo,30,6,0,-6] [40,36,D+1,GBBJooGHoJKLoHAAKLoDDDKMooIEEMFFIooo,35,7,0,-7] [48,40,G-3,oBBJoooHoJKLoHAAKLGDDDKMGoIEEMFFIooo,38,8,0,-8] [52,48,B-1,BBoJoooHoJKLoHAAKLGDDDKMGoIEEMFFIooo,43,9,0,-9] [62,52,K+1,BBoJKooHoJKLoHAAKLGDDDoMGoIEEMFFIooo,48,10,0,-10] [65,62,B+1,oBBJKooHoJKLoHAAKLGDDDoMGoIEEMFFIooo,53,11,0,-11] [73,65,D+1,oBBJKooHoJKLoHAAKLGoDDDMGoIEEMFFIooo,58,12,0,-12] [80,73,B-1,BBoJKooHoJKLoHAAKLGoDDDMGoIEEMFFIooo,63,13,0,-13] [91,80,G+1,BBoJKooHoJKLGHAAKLGoDDDMooIEEMFFIooo,68,14,0,-14] [97,91,B+1,oBBJKooHoJKLGHAAKLGoDDDMooIEEMFFIooo,73,15,0,-15] [106,97,D-1,oBBJKooHoJKLGHAAKLGDDDoMooIEEMFFIooo,78,16,0,-16] [114,106,B-1,BBoJKooHoJKLGHAAKLGDDDoMooIEEMFFIooo,83,17,0,-17] [127,114,L+1,BBoJKLoHoJKLGHAAKoGDDDoMooIEEMFFIooo,88,18,0,-18] [152837,127,G-1,BBoJKLoHoJKLoHAAKoGDDDoMGoIEEMFFIooo,93,19,0,-19] [168248,152837,M-1,BBoJKLoHoJKLoHAAKoGDDDooGoIEEMFFIooM,98,20,0,-20] [168292,168248,D+2,BBoJKLoHoJKLoHAAKoGooDDDGoIEEMFFIooM,102,21,0,-21] [168306,168292,H-2,BBoJKLoooJKLooAAKoGHoDDDGHIEEMFFIooM,106,22,0,-22] [168443,168306,A-2,BBoJKLoooJKLAAooKoGHoDDDGHIEEMFFIooM,110,23,0,-23] [168779,168443,I+4,BBIJKLooIJKLAAooKoGHoDDDGHoEEMFFoooM,112,24,0,-24] [172738,168779,A+2,BBIJKLooIJKLooAAKoGHoDDDGHoEEMFFoooM,116,25,0,-25] [175849,172738,E-1,BBIJKLooIJKLooAAKoGHoDDDGHEEoMFFoooM,121,26,0,-26] [178843,175849,H+2,BBIJKLoHIJKLoHAAKoGooDDDGoEEoMFFoooM,125,27,0,-27] [179621,178843,D-2,BBIJKLoHIJKLoHAAKoGDDDooGoEEoMFFoooM,129,28,0,-28] [179643,179621,K-3,BBIJoLoHIJoLoHAAooGDDDKoGoEEKMFFooKM,132,29,0,-29] [179676,179643,A+2,BBIJoLoHIJoLoHooAAGDDDKoGoEEKMFFooKM,136,30,0,-30] ET: 2416938 TN: 179689 EN: 16541 CN: 162987 DF: 30

solver -s BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo --depth 40 --strategy DFS --stats	[0,none,___,BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo,0,0,0,0] [1,0,B+1,oBBJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo,5,1,0,-1] [8,1,G+1,oBBJooGHoJKLGHAAKLoDDDKMooIEEMFFIooo,10,2,0,-2] [14,8,B-1,BBoJooGHoJKLGHAAKLoDDDKMooIEEMFFIooo,15,3,0,-3] [23,14,D-1,BBoJooGHoJKLGHAAKLDDDoKMooIEEMFFIooo,20,4,0,-4] [29,23,B+1,oBBJooGHoJKLGHAAKLDDDoKMooIEEMFFIooo,25,5,0,-5] [36,29,G+1,GBBJooGHoJKLoHAAKLDDDoKMooIEEMFFIooo,30,6,0,-6] [40,36,D+1,GBBJooGHoJKLoHAAKLoDDDKMooIEEMFFIooo,35,7,0,-7] [48,40,G-3,oBBJoooHoJKLoHAAKLGDDDKMGoIEEMFFIooo,38,8,0,-8] [52,48,B-1,BBoJoooHoJKLoHAAKLGDDDKMGoIEEMFFIooo,43,9,0,-9] [62,52,K+1,BBoJKooHoJKLoHAAKLGDDDoMGoIEEMFFIooo,48,10,0,-10] [65,62,B+1,oBBJKooHoJKLoHAAKLGDDDoMGoIEEMFFIooo,53,11,0,-11] [73,65,D+1,oBBJKooHoJKLoHAAKLGoDDDMGoIEEMFFIooo,58,12,0,-12] [80,73,B-1,BBoJKooHoJKLoHAAKLGoDDDMGoIEEMFFIooo,63,13,0,-13] [91,80,G+1,BBoJKooHoJKLGHAAKLGoDDDMooIEEMFFIooo,68,14,0,-14] [97,91,B+1,oBBJKooHoJKLGHAAKLGoDDDMooIEEMFFIooo,73,15,0,-15] [106,97,D-1,oBBJKooHoJKLGHAAKLGDDDoMooIEEMFFIooo,78,16,0,-16] [114,106,B-1,BBoJKooHoJKLGHAAKLGDDDoMooIEEMFFIooo,83,17,0,-17] [124,114,G+1,BBoJKoGHoJKLGHAAKLoDDDoMooIEEMFFIooo,88,18,0,-18] [129,124,B+1,oBBJKoGHoJKLGHAAKLoDDDoMooIEEMFFIooo,93,19,0,-19] [138,129,D+1,oBBJKoGHoJKLGHAAKLooDDDMooIEEMFFIooo,98,20,0,-20] [146,138,B-1,BBoJKoGHoJKLGHAAKLooDDDMooIEEMFFIooo,103,21,0,-21] [158,146,D-2,BBoJKoGHoJKLGHAAKLDDDooMooIEEMFFIooo,107,22,0,-22] [165,158,B+1,oBBJKoGHoJKLGHAAKLDDDooMooIEEMFFIooo,112,23,0,-23] [174,165,G+1,GBBJKoGHoJKLoHAAKLDDDooMooIEEMFFIooo,117,24,0,-24] [178,174,D+1,GBBJKoGHoJKLoHAAKLoDDDoMooIEEMFFIooo,122,25,0,-25] [184,178,D+1,GBBJKoGHoJKLoHAAKLooDDDMooIEEMFFIooo,127,26,0,-26] [198,184,H-2,GBBJKoGooJKLooAAKLoHDDDMoHIEEMFFIooo,131,27,0,-27] [228338,198,G-3,oBBJKooooJKLooAAKLGHDDDMGHIEEMFFIooo,134,28,0,-28] [249672,228338,A-2,oBBJKooooJKLAAooKLGHDDDMGHIEEMFFIooo,138,29,0,-29] [249683,249672,B-1,BBoJKooooJKLAAooKLGHDDDMGHIEEMFFIooo,143,30,0,-30] [249691,249683,L+1,BBoJKLoooJKLAAooKoGHDDDMGHIEEMFFIooo,148,31,0,-31] [257578,249691,M-1,BBoJKLoooJKLAAooKoGHDDDoGHIEEMFFIooM,153,32,0,-32] [257594,257578,D+1,BBoJKLoooJKLAAooKoGHoDDDGHIEEMFFIooM,158,33,0,-33] [257607,257594,I+4,BBIJKLooIJKLAAooKoGHoDDDGHoEEMFFoooM,160,34,0,-34] [259402,257607,A+2,BBIJKLooIJKLooAAKoGHoDDDGHoEEMFFoooM,164,35,0,-35] [261558,259402,E-1,BBIJKLooIJKLooAAKoGHoDDDGHEEoMFFoooM,169,36,0,-36] [264552,261558,H+2,BBIJKLoHIJKLoHAAKoGooDDDGoEEoMFFoooM,173,37,0,-37] [265330,264552,D-2,BBIJKLoHIJKLoHAAKoGDDDooGoEEoMFFoooM,177,38,0,-38] [265352,265330,K-3,BBIJoLoHIJoLoHAAooGDDDKoGoEEKMFFooKM,180,39,0,-39] [265385,265352,A+2,BBIJoLoHIJoLoHooAAGDDDKoGoEEKMFFooKM,184,40,0,-40] ET: 3780329 TN: 265398 EN: 24742 CN: 240441 DF: 40

solver -s BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo --strategy UC --stats	[0,none,___,BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo,0,0,0,0] [4,0,K+1,BBoJKooHoJKLGHAAKLGDDDoMooIEEMFFIooo,5,1,0,5] [33,4,M-1,BBoJKooHoJKLGHAAKLGDDDooooIEEMFFIooM,10,2,0,10] [162,33,D+2,BBoJKooHoJKLGHAAKLGooDDDooIEEMFFIooM,14,3,0,14] [234,162,I+1,BBoJKooHoJKLGHAAKLGoIDDDooIEEMFFoooM,19,4,0,19] [687,234,F+2,BBoJKooHoJKLGHAAKLGoIDDDooIEEMooFFoM,23,5,0,23] [1203,687,H-3,BBoJKooooJKLGoAAKLGoIDDDoHIEEMoHFFoM,26,6,0,26] [2030,1203,G-2,BBoJKooooJKLooAAKLooIDDDGHIEEMGHFFoM,30,7,0,30] [3404,2030,A-2,BBoJKooooJKLAAooKLooIDDDGHIEEMGHFFoM,34,8,0,34] [5424,3404,I+3,BBIJKoooIJKLAAooKLoooDDDGHoEEMGHFFoM,37,9,0,37] [7582,5424,D-3,BBIJKoooIJKLAAooKLDDDoooGHoEEMGHFFoM,40,10,0,40] [11089,7582,E-1,BBIJKoooIJKLAAooKLDDDoooGHEEoMGHFFoM,45,11,0,45] [20923,11089,K-3,BBIJooooIJoLAAoooLDDDoKoGHEEKMGHFFKM,48,12,0,48] [31100,20923,L+1,BBIJoLooIJoLAAooooDDDoKoGHEEKMGHFFKM,53,13,0,53] [57849,31100,A+4,BBIJoLooIJoLooooAADDDoKoGHEEKMGHFFKM,55,14,0,55] ET: 980945 TN: 74119 EN: 6480 CN: 36183 DF: 15
solver -s BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo --strategy BFS --stats	[0,none,___,BBoJoooHoJKLGHAAKLGDDDKMooIEEMFFIooo,0,0,0,0] [3,0,G-1,BBoJoooHoJKLoHAAKLGDDDKMGoIEEMFFIooo,5,1,0,1] [24,3,K+1,BBoJKooHoJKLoHAAKLGDDDoMGoIEEMFFIooo,10,2,0,2] [128,24,L+1,BBoJKLoHoJKLoHAAKoGDDDoMGoIEEMFFIooo,15,3,0,3] [413,128,M-1,BBoJKLoHoJKLoHAAKoGDDDooGoIEEMFFIooM,20,4,0,4] [958,413,D+2,BBoJKLoHoJKLoHAAKoGooDDDGoIEEMFFIooM,24,5,0,5] [1786,958,H-2,BBoJKLoooJKLooAAKoGHoDDDGHIEEMFFIooM,28,6,0,6] [2931,1786,A-2,BBoJKLoooJKLAAooKoGHoDDDGHIEEMFFIooM,32,7,0,7] [4615,2931,I+4,BBIJKLooIJKLAAooKoGHoDDDGHoEEMFFoooM,34,8,0,8] [7156,4615,A+2,BBIJKLooIJKLooAAKoGHoDDDGHoEEMFFoooM,38,9,0,9] [11064,7156,E-1,BBIJKLooIJKLooAAKoGHoDDDGHEEoMFFoooM,43,10,0,10] [17681,11064,H+2,BBIJKLoHIJKLoHAAKoGooDDDGoEEoMFFoooM,47,11,0,11] [29043,17681,D-2,BBIJKLoHIJKLoHAAKoGDDDooGoEEoMFFoooM,51,12,0,12] [47800,29043,K-3,BBIJoLoHIJoLoHAAooGDDDKoGoEEKMFFooKM,54,13,0,13] [76420,47800,A+2,BBIJoLoHIJoLoHooAAGDDDKoGoEEKMFFooKM,58,14,0,14] ET: 1495878 TN: 114087 EN: 9883 CN: 66537 DF: 15

5. Tarea 4: Heurística y estrategias Voraz y A*

Una vez finalizada la implementación completa del algoritmo de búsqueda, la tarea final del laboratorio consiste en incorporar las estrategias Voraz y A*.

Para ello, se propone la implementación de una serie de funciones heurísticas que subestimen la cantidad de movimientos que tendrá que realizar el coche rojo A para alcanzar el estado que verifica la función objetivo.

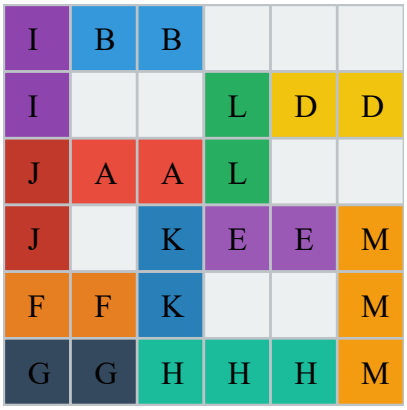
5.1. Funciones Heurísticas

Mediante una función heurística, calculada sobre un estado, se consigue una estimación del costo necesario para alcanzar algún estado objetivo.

Consideraremos las siguientes heurísticas:

- Una heurística simple, que llamaremos **heurística 0,** que calcula el número de casillas a la derecha del coche rojo A.
- Una segunda heurística que muestra el número de coches que hay que mover para despejar el camino. Su valor será el número de coches que ocupan casillas de la fila de salida y están a la derecha del coche rojo A. Esta heurística sera la **heurística 1**.
- Por último tendremos otra **heurística 2** que será la suma de las dos heurísticas anteriores (heurística 0 + heurística 1).

Ejemplo:



El valor de las diferentes heurística para este nivel son: **3** (heurística 0), **1** (heurística 1) y **4** (heurística 2).

5.2. Resultado

Modificar la acción **solver** para que el parámetro --strategy pueda tomar los valores para las estrategias GBF (voraz) y AStar(A estrella). Además, se añadirá un nuevo parámetro --heuristic con un valor entero que representa la heurística seleccionada para la ejecución. Para los ejemplo de más abajo, serían la heurística 0 y 2 respectivamente.

rushhour solver -s BBBIKLCCoIKLAAoJKMGDDJoMGoHEEMFFHooo --strategy GBF --heuristic 0 --stats

La salida es:

[0,none,___,BBBIKLCCoIKLAAoJKMGDDJoMGoHEEMFFHooo,0,0,4,4]
[1,0,A+1,BBBIKLCCoIKLoAAJKMGDDJoMGoHEEMFFHooo,5,1,3,3]
[6,1,C+1,BBBIKLoCCIKLoAAJKMGDDJoMGoHEEMFFHooo,10,2,3,3]
[13,6,G+2,BBBIKLGCCIKLGAAJKMoDDJoMooHEEMFFHooo,14,3,3,3]
[36,13,D-1,BBBIKLGCCIKLGAAJKMDDoJoMooHEEMFFHooo,19,4,3,3]
[86,36,H+1,BBBIKLGCCIKLGAAJKMDDHJoMooHEEMFFoooo,24,5,3,3]
[140,86,F+3,BBBIKLGCCIKLGAAJKMDDHJoMooHEEMoooFFo,27,6,3,3]
[196,140,H-1,BBBIKLGCCIKLGAAJKMDDoJoMooHEEMooHFFo,32,7,3,3]
[257,196,D+1,BBBIKLGCCIKLGAAJKMoDDJoMooHEEMooHFFo,37,8,3,3]
[329,257,G-3,BBBIKLoCCIKLoAAJKMoDDJoMGoHEEMGoHFFo,40,9,3,3]
[398,329,D-1,BBBIKLoCCIKLoAAJKMDDoJoMGoHEEMGoHFFo,45,10,3,3]
[516,398,H+1,BBBIKLoCCIKLoAAJKMDDHJoMGoHEEMGooFFo,50,11,3,3]
[680,516,F-2,BBBIKLoCCIKLoAAJKMDDHJoMGoHEEMGFFooo,54,12,3,3]
[845,680,M-1,BBBIKLoCCIKLoAAJKoDDHJoMGoHEEMGFFooM,59,13,3,3]
[990,845,L-1,BBBIKooCCIKLoAAJKLDDHJoMGoHEEMGFFooM,64,14,3,3]
[1089,990,A-1,BBBIKooCCIKLAAoJKLDDHJoMGoHEEMGFFooM,69,15,4,4]
[1783,1089,H+1,BBBIKooCCIKLAAHJKLDDHJoMGooEEMGFFooM,74,16,4,4]
[2313,1783,E-2,BBBIKooCCIKLAAHJKLDDHJoMGEEooMGFFooM,78,17,4,4]
[3527,2313,J-2,BBBIKooCCIKLAAHoKLDDHooMGEEJoMGFFJoM,82,18,4,4]
[5657,3527,I-2,BBBoKooCCoKLAAHIKLDDHIoMGEEJoMGFFJoM,86,19,4,4]
[8446,5657,K-2,BBBooooCCooLAAHIKLDDHIKMGEEJKMGFFJoM,90,20,4,4]
[12339,8446,B+3,oooBBBoCCooLAAHIKLDDHIKMGEEJKMGFFJoM,93,21,4,4]
[17693,12339,C+2,oooBBBoooCCLAAHIKLDDHIKMGEEJKMGFFJoM,97,22,4,4]
[21406,17693,H+2,ooHBBBooHCCLAAoIKLDDoIKMGEEJKMGFFJoM,101,23,4,4]
[23696,21406,A+1,ooHBBBooHCCLoAAIKLDDoIKMGEEJKMGFFJoM,106,24,3,3]
[23702,23696,D+1,ooHBBBooHCCLoAAIKLoDDIKMGEEJKMGFFJoM,111,25,3,3]
[23709,23702,G+4,GoHBBBGoHCCLoAAIKLoDDIKMoEEJKMoFFJoM,113,26,3,3]
[23737,23709,D-1,GoHBBBGoHCCLoAAIKLDDoIKMoEEJKMoFFJoM,118,27,3,3]
[23811,23737,E-1,GoHBBBGoHCCLoAAIKLDDoIKMEEoJKMoFFJoM,123,28,3,3]
[23895,23811,A-1,GoHBBBGoHCCLAAoIKLDDoIKMEEoJKMoFFJoM,128,29,4,4]
[24984,23895,H-3,GooBBBGooCCLAAoIKLDDHIKMEEHJKMoFFJoM,131,30,4,4]
[25300,24984,A+1,GooBBBGooCCLoAAIKLDDHIKMEEHJKMoFFJoM,136,31,3,3]
[25314,25300,C-2,GooBBBGCCooLoAAIKLDDHIKMEEHJKMoFFJoM,140,32,3,3]
[25351,25314,G-1,oooBBBGCCooLGAAIKLDDHIKMEEHJKMoFFJoM,145,33,3,3]
[25547,25351,B-3,BBBoooGCCooLGAAIKLDDHIKMEEHJKMoFFJoM,148,34,3,3]
[26160,25547,I+2,BBBIooGCCIoLGAAoKLDDHoKMEEHJKMoFFJoM,152,35,3,3]
[27417,26160,A+1,BBBIooGCCIoLGoAAKLDDHoKMEEHJKMoFFJoM,157,36,2,2]
[27432,27417,K-1,BBBIooGCCIoLGoAAoLDDHoKMEEHJKMoFFJKM,162,37,2,2]
[27465,27432,A+1,BBBIooGCCIoLGooAALDDHoKMEEHJKMoFFJKM,167,38,1,1]
[27478,27465,L+1,BBBIoLGCCIoLGooAAoDDHoKMEEHJKMoFFJKM,172,39,1,1]
[27498,27478,A+1,BBBIoLGCCIoLGoooAADDHoKMEEHJKMoFFJKM,177,40,0,0]

ET: XXXXXX
TN: 27506
EN: 3014
CN: 22781
DF: 40

rushhour solver -s HBBCCCHDDKoMAAJKoMEEJFFMoIooLooIGGLo --strategy AStar --heuristic 2 --stats

La salida es:

[0,none,___,HBBCCCHDDKoMAAJKoMEEJFFMoIooLooIGGLo,0,0,7,7]
[1,0,J-1,HBBCCCHDDKoMAAoKoMEEJFFMoIJoLooIGGLo,5,1,6,11]
[7,1,A+1,HBBCCCHDDKoMoAAKoMEEJFFMoIJoLooIGGLo,10,2,5,15]
[19,7,H-1,oBBCCCHDDKoMHAAKoMEEJFFMoIJoLooIGGLo,15,3,5,20]
[30,19,B-1,BBoCCCHDDKoMHAAKoMEEJFFMoIJoLooIGGLo,20,4,5,25]
[43,30,C-1,BBCCCoHDDKoMHAAKoMEEJFFMoIJoLooIGGLo,25,5,5,30]
[55,43,M+1,BBCCCMHDDKoMHAAKoMEEJFFooIJoLooIGGLo,30,6,5,35]
[66,55,F+1,BBCCCMHDDKoMHAAKoMEEJoFFoIJoLooIGGLo,35,7,5,40]

[76,66,K-2,BBCCCMHDDooMHAAooMEEJKFFoIJKLooIGGLo,39,8,4,43]
[78,76,A+2,BBCCCMHDDooMHooAAMEEJKFFoIJKLooIGGLo,43,9,2,45]
[90,78,D+2,BBCCCMHooDDMHooAAMEEJKFFoIJKLooIGGLo,47,10,2,49]
[114,90,J+2,BBCCCMHoJDDMHoJAAMEEoKFFoIoKLooIGGLo,51,11,2,53]
[134,114,E+1,BBCCCMHoJDDMHoJAAMoEEKFFoIoKLooIGGLo,56,12,2,58]
[147,134,H-3,BBCCCMooJDDMooJAAMoEEKFFHIoKLoHIGGLo,59,13,2,61]
[148,147,E-1,BBCCCMooJDDMooJAAMEEoKFFHIoKLoHIGGLo,64,14,2,66]
[160,148,J-2,BBCCCMoooDDMoooAAMEEJKFFHIJKLoHIGGLo,68,15,2,70]
[166,160,D-3,BBCCCMDDoooMoooAAMEEJKFFHIJKLoHIGGLo,71,16,2,73]
[176,166,A-3,BBCCCMDDoooMAAoooMEEJKFFHIJKLoHIGGLo,74,17,5,79]
[255,176,J+2,BBCCCMDDJooMAAJooMEEoKFFHIoKLoHIGGLo,78,18,6,84]
[336,255,K+2,BBCCCMDDJKoMAAJKoMEEooFFHIooLoHIGGLo,82,19,7,89]
[441,336,F-2,BBCCCMDDJKoMAAJKoMEEFFooHIooLoHIGGLo,86,20,7,93]
[547,441,L+3,BBCCCMDDJKLMAAJKLMEEFFooHIooooHIGGoo,89,21,8,97]
[687,547,F+2,BBCCCMDDJKLMAAJKLMEEooFFHIooooHIGGoo,93,22,8,101]
[926,687,G+2,BBCCCMDDJKLMAAJKLMEEooFFHIooooHIooGG,97,23,8,105]
[1323,926,J-3,BBCCCMDDoKLMAAoKLMEEooFFHIJoooHIJoGG,100,24,7,107]
[1658,1323,K-3,BBCCCMDDooLMAAooLMEEooFFHIJKooHIJKGG,103,25,6,109]
[2011,1658,A+2,BBCCCMDDooLMooAALMEEooFFHIJKooHIJKGG,107,26,4,111]
[2525,2011,D+2,BBCCCMooDDLMooAALMEEooFFHIJKooHIJKGG,111,27,4,115]
[3958,2525,E+2,BBCCCMooDDLMooAALMooEEFFHIJKooHIJKGG,115,28,4,119]
[6186,3958,H+3,BBCCCMHoDDLMHoAALMooEEFFoIJKoooIJKGG,118,29,4,122]
[8268,6186,I+3,BBCCCMHIDDLMHIAALMooEEFFooJKooooJKGG,121,30,4,125]
[10475,8268,E-2,BBCCCMHIDDLMHIAALMEEooFFooJKooooJKGG,125,31,4,129]
[13679,10475,J+1,BBCCCMHIDDLMHIAALMEEJoFFooJKooooooKGG,130,32,4,134]
[16824,13679,K+1,BBCCCMHIDDLMHIAALMEEJKFFooJKooooooGG,135,33,4,139]
[19078,16824,G-4,BBCCCMHIDDLMHIAALMEEJKFFooJKooGGoooo,137,34,4,141]
[19959,19078,J-1,BBCCCMHIDDLMHIAALMEEoKFFooJKooGGJooo,142,35,4,146]
[21383,19959,K-1,BBCCCMHIDDLMHIAALMEEooFFooJKooGGJKoo,147,36,4,151]
[22465,21383,F-2,BBCCCMHIDDLMHIAALMEEFFooooJKooGGJKoo,151,37,4,155]
[23199,22465,L-3,BBCCCMHIDDoMHIAAoMEEFFooooJKLoGGJKLo,154,38,3,157]
[23466,23199,M-3,BBCCCoHIDDooHIAAooEEFFoMooJKLMGGJKLM,157,39,2,159]
[23735,23466,A+2,BBCCCoHIDDooHIooAAEEFFoMooJKLMGGJKLM,161,40,0,161]

ET: XXXXX
TN: 23977
EN: 3062
CN: 20329
DF: 40

6. Entrega final

El estudiante debe:

1. Haber superado todos los **tests de autograde** de github actions en la sesión de laboratorio de su grupo de la semana del 24 de noviembre o en la sesión de la semana del 1 de diciembre.
2. **Entregar en Moodle el código fuente de la rama main** tal y como se explica en: https://campusvirtual.uclm.es/mod/assign/view.php?id=787221. **La fecha límite de subida es el 5 de diciembre.**
3. Durante las semanas del 8 de diciembre, 15 de diciembre y 22 de diciembre, se convocará mediante correo electrónico a los estudiantes que hayan **completado los puntos 1 y 2 a la/s prueba/s de autoría**.