PRÁCTICAS DE MATEMÁTICAS 1

2018-2019

- Más sobre las reglas en Prolog vs Plman.

- Link a Youtube de las clases del Prof. Francisco Gallego (2017-18).
  https://www.youtube.com/watch?v=7I6X5NZd2kw&list=PLmxqg54iaXrgiF20LM2JfetrnZF87UMBD

>> **logica.i3a.ua.es**

Iniciar sesión

Usuario: --------@ (EPS)

Contraseña: NIF letra mayúscula → cambiar

Descarga mapa / Entrega de solución / Obtienes nota.

Fichero_solución.pl

- Si solución >= 75% >> Mapa superado
- Sólo 3 intentos más.
- Puedes descargar siguiente mapa.
- Eoc entrega soluciones cada 10 min

Rutina para resolver mapas

Editar fichero solución (extensión .pl):

```
$ gedit   solucion.pl   &
```

Ejecutar solución de un mapa

```
$ ./plman   mapa.pl   solucion.pl
```

Escribir en la 1ª línea del fichero solución.pl

```
:- use_module('pl-man-game/main').
```

Resto de líneas

```
Código Prolog con acciones para Plman
```

ACCIONES de Plman

Acciones: **do(ACCION).**

ACCION = { move(D1), get(D1), drop(D1), use(D1) }

D1 = {up, down, left, right}

Plman No se mueve: do(move(none)).

Sensor de visión: **see(normal, D2, OBJ)**

D2 = { right, left, down, up, here, down-right, down-left, up-right, up-left }

OBJ: Cualquier caracter en el mapa

Ojo con letras mayúsculas y símbolos expeciales

Regla sencilla

do(ACCION(D1)) :- see(normal, D2, OBJ).

PARA HACER REGLAS MÁS COMPLEJAS PUEDES AÑADIR VARIAS CONDICIONES EN EL CUERPO DE LA REGLA

>> **CONJUNCIÓN (,)**

Puedes **añadir** varias condiciones en el cuerpo de la regla, separadas por la conjunción (,)

>> varios predicados see/3

Ej. do(move(down)) :- see(normal, down, '.'), see(normal, up, ‘ ').

>> predicado predeficnido para escribir mensajes: **writeln/1**

Ej. do(move(down)) :- see(normal, down, '.'), writeln(‘veo coco abajo’).

>> otros predicados que ya veremos en fases sucesivas.

>> **NEGACIÓN (not)** Cualquier predicado se puede negar, p.ej., el *predicado see/3*:

Ej. do(move(down)) :- not(see(normal,down,‘E‘)).

not(see(normal,down,‘E’)) → tendrá éxito si no hay enemigo abajo.

MÁS SOBRE HECHOS y REGLAS: CLÁUSULAS PROLOG

>> En cada **golpe de espaciador** se ejecuta una acción para ello Prolog se sitúa en la primera cláusula que encuentra y de **arriba/abajo** ejecuta la primera que sea cierta**.**

>> En el **cuerpo de la regla** se escriben todas las condiciones que Plman debe cumplir para que se realice la acción correspondiente indicada en la cabeza.

>> Si la cláusula es una regla se comprueba de **izda a dcha** las condiciones de su cuerpo mientras no fracase ninguna.

- Si todas las condiciones son ciertas se ejecuta la acción indicada en la cabeza.
- Si alguna condición falla el sistema comprueba si la siguiente cláusula se puede ejecutar.

Esto significa que el **orden** en que escribas las reglas es **MUY IMPORTANTE**

||| Compruébalo con cualquier código que hayas escrito con varias reglas

Aquí tienes un ejemplo:

solucion.pl del mapa : maps/ejemplos/**mapaej8.pl**

Cambia el orden de algunas reglas y comprueba.

```
:- use_module('pl-man-game/main').
do(get(down)) :- see(normal,down,'l'),
do(move(up)) :- see(normal,up,'.'),
do(move(right)) :- see(normal,right,'.'),
do(move(up)) :- see(normal,up,' '),
do(use(right)) :- - see(normal,right,'E'),
do(get(down)) :- see(normal,down,'l'),
do(move(up)) :- see(normal,up,'.').
```

MÁS SOBRE HECHOS y REGLAS: CLÁUSULAS PROLOG

>> Escribe **tantas reglas** como condiciones puedan pasarle a Plman.

>> **Añade** reglas de una en una y comprueba antes de seguir.

REGLA DE ORO:

No te dejes cocos

al pasar por un sitio,

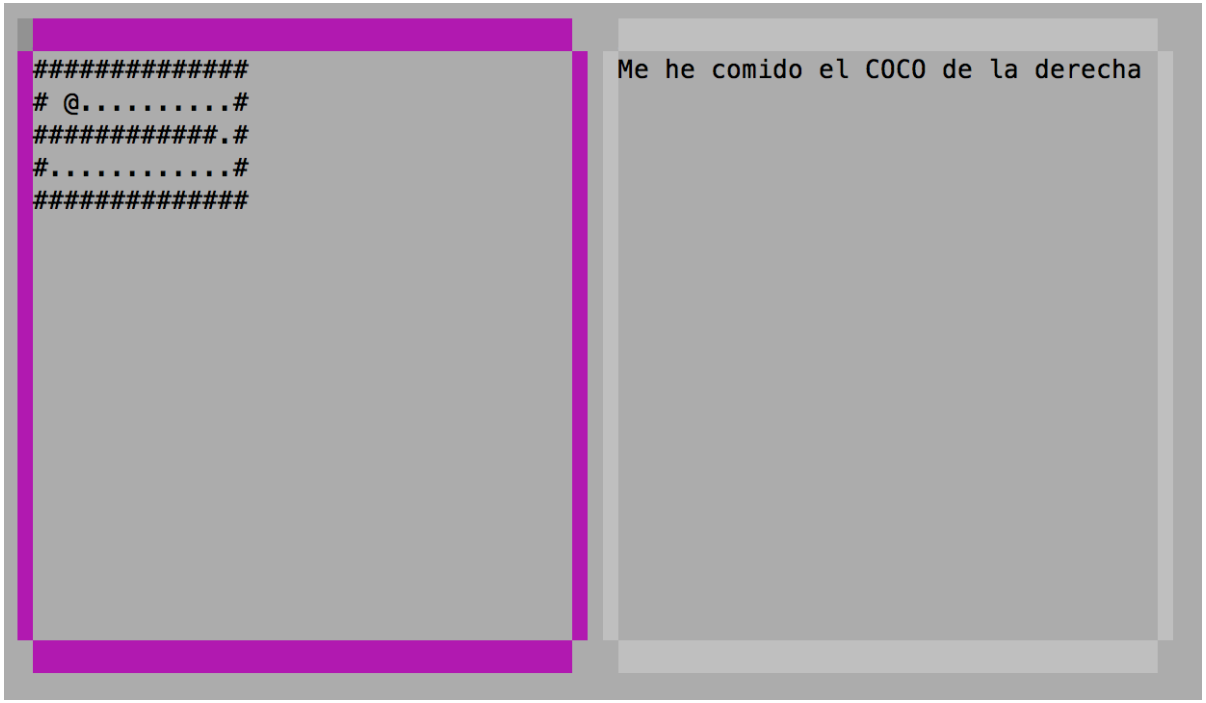
puede ser imposible regresar.

Escribe ”mensajes” en el cuerpo de la regla para saber lo que está pasando.

writeln(‘mensaje’): imprime por la pantalla : mensaje

```
:- use_module('pl-man-game/main.pl').

do(move(X)) :- see(normal, X, '.'), writeln('Me he comido el COCO de la derecha').
```



Errores comunes en línea de comandos

```
German-5:plman sierratech$ ./plman maps/fase1/mapa51.pl  sols/fase1/mapa5.pl
Compilando...
Archivos compilados con éxito :)
Ejecutando...
1 ERROR:  El archivo de mapa no existe o no tiene el formato correcto. [maps/fase1/mapa51.pl]
German-5:plman sierratech$
```


Errores comunes en línea de comandos

```
German-5:plman sierratech$ ./plman maps/fase1/mapa5.pl  sols/fase1/mapa51.pl
Compilando...
************************************************
********** ERROR DE COMPILACION ***************
************************************************

ERROR: source_sink `'sols/fase1/mapa51.pl'' does not exist
% autoloading prolog_codewalk:must_be/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/error
% autoloading prolog_codewalk:portray_clause/1 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/listing
% autoloading prolog_codewalk:clause_info/4 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/prolog_clause
% autoloading prolog_codewalk:initialization_layout/4 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/prolog_clause
% autoloading prolog_debug:backtrace/1 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/prolog_stack
% autoloading oset:reverse/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading record:member/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading qsave:current_foreign_library/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/shlib
% autoloading error:assertion/1 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/debug
% autoloading prolog_codewalk:clause_name/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/prolog_clause
% Autoloader: iteration 1 resolved 10 predicates and loaded 10 files in 0.077 seconds.  Restarting ...
% Autoloader: loaded 10 files in 2 iterations in 0.103 seconds
```

Errores comunes en línea de comandos

```
German-5:plman sierratech$ ./plman maps/fase1/mapa5.pl  sols/fase1/mapa5.pl
Compilando...
************************************************
********** ERROR DE COMPILACION ***************
************************************************

ERROR: /Users/sierratech/Dropbox/UA/Matematicas 1/plman/sols/fase1/mapa5.pl:4:63: Syntax error: Operator expected
% autoloading prolog_codewalk:must_be/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/error
% autoloading oset:reverse/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading 'pl-man':nth0/3 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading 'pl-man':member/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading 'pl-man':maplist/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/apply
% autoloading 'pl-man':swritef/3 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/writef
% autoloading 'pl-man':subtract/3 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading 'pl-man':string_to_atom/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/backcomp
% autoloading backward_compatibility:maplist/3 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/
% autoloading mapManager:nth0/3 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.
% autoloading dynamicProperties:member/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/s
% autoloading dynamicProperties:maplist/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/
% autoloading dynamicProperties:append/3 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/s
% autoloading cheeseEngine:string_to_list/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/li
% autoloading
% autoloading
% autoloading
% autoloading
% autoloading
% autoloading
% autoloading
% autoloading
% autoloading prolog_codewalk:clause_info/4 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/prolog_clause
% autoloading prolog_codewalk:initialization_layout/4 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/prolog_clause
% autoloading record:member/2 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/lists
% autoloading quintus:shell/0 from /usr/local/Cellar/swi-prolog/7.6.4/libexec/lib/swipl-7.6.4/library/shell
```

```
1  :- use_module('pl-man-game/main.pl').
2
3
4  do(move(none))  :- see(normal, down-right, 'E'). % Avoid 1st E
5  do(move(none))  :- see(normal, down,       'E'). % Avoid 1st E
6  do(get(up))     :- see(normal, up,          p ). % Get p
```

NORMAS para evaluación de mapas

- **Mapas <u>limitados</u>** en cada grupo.
- En cada fase (menos Fase0) los mapas serán de diferente dificultad.
- Mapa seleccionado **no se puede cambiar.**
- Entrega **<u>ilimitada</u>** (cada **10 min**) de soluciones hasta obtener **75%**
- Si solución **>= 75% → mapa superado → Sólo 3** entregas más.
- Sucesivas entregas **no bajan** nota.
- Se puede **descargar nuevo** mapa si anterior está superado.
- Al entregar **revisa** el nombre del fichero solución que resuelve el mapa.
- Si entregas fuera de **fecha límite →** !25% penalización!
- **Plagio** supone nota = 0 en fases Plman.