



PRÁCTICAS DE MATEMÁTICAS 1

2018-2019



- >> ¿Plman lleva objeto? ¿Qué objeto lleva?
- >> Predicado: **havingObject**
- >> Modulariza el código con subreglas
- >> Resuelve mapas Fase 2





¿Algún problema con este mapa?

plman/maps/fase2/mapa0.pl

```
#####  
# . | @          a#  
#####
```

¿Plman se mueve tal como habías previsto?





¿ PLMAN LLEVA OBJETO ?

PREDICADO: **havingObject/0**

tiene **éxito** si Plman lleva un objeto encima, sea el objeto que sea.

Fracasa si no lo lleva.

EJEMPLO

...

```
do(move(down)) :- havingObject.
```

```
do(move(down)) :- not(havingObject).
```

...

```
do(move(down)) :- havingObject, see(normal, down, ' ').
```

```
do(move(down)) :- see(normal, down, ' '), not(havingObject).
```

El predicado se puede escribir en cualquier parte del cuerpo de la regla



¿ QUÉ OBJETO LLEVA ?

Plman cambia su comportamiento en función del objeto que lleva encima

havingObject/1

havingObject(appearance(OBJ))

OBJ: objeto que lleva Plman.

>> el predicado tiene **éxito** si Plman lleva el objeto referido en el argumento

EJEMPLO

```
do(use(left)) :- havingObject(appearance('a')), see(normal, left, '|').
```



ELECCIÓN de havingObject

1º : Si las acciones de Plman **NO** dependen del objeto que lleva pero es necesario tener en cuenta si Plman lleva o no el objeto usar:

havingObject/0

2º : Si las acciones de Plman **SÍ** dependen del objeto que lleva usar :

havingObject/1





Ejemplos del uso de **havingObject** en solución **mapa0.pl**

Resuélvelo

>> Que se mueva a la derecha cuando no lleve ningún objeto:

do(move(right)) :- see(normal,right,' '), **not(havingObject).**

>> Que se mueva a la derecha cuando no lleve el objeto 'a':


do(move(right)) :- see(normal,right,' '), **not(havingObject(appearance(a))).**


>> Que use el objeto 'a' a la izquierda si lo lleva encima:


do(use(left)) :- see(normal,left,'|'), **havingObject(appearance(a)).**



Si quieres, puedes usar “alias”

 `s(DIR, OBJ) :- see(normal, DIR, OBJ).`
`do(move(right)) :- s(right, ' '), not(havingObject).`

 `hO(OBJ) :- havingObject(appearance(OBJ)).`
`do(use(left)) :- hO(a), s(left, '|').`

 `hO :- havingObject.`
`do(move(left)) :- hO, s(left, '.').`



Hay que resolver....

plman/maps/fase2/mapa00.pl

Pero espera...

Para resolver estos mapas es conveniente que “organices” tu código teniendo en cuenta si llevas o no objeto y según qué objeto lleves hacer “según qué” acciones

Reglas que se “activan” en el cuerpo de otras reglas:

subreglas

```
#####
#  || ..... #
# #####
# # #
# # a #
# # #
# # ###
# # @ ###
# #####r#
# || #
#####
```




Modulariza el código >> Subreglas

%% Regla principal que lanza subreglas

do(ACT) :- hO(OBJ), do1(OBJ, ACT).

do(ACT) :- do1(n, ACT).

%% Reglas para cuando Plman lleva objeto 'a'

do1(a, use(right)) :- s(right, '|').

do1(a, drop(left)) :- s(up, 'r').

.....

%% Reglas para cuando Plman lleva objeto 'r'

do1(r, move(right)) :- s(right, '.').

do1(r, move(left)) :- s(left, ' ').

....

%% Reglas para cuando Plman no lleva objeto

do1(n, move(right)) :- s(right, ' ').

do1(n, move(up)).



Puedes necesitar hacer evaluaciones aritméticas

Operador **is**: $Z \text{ is } X + Y$

$X - Y$

$X * Y$

X / Y

$X \text{ mod } Y$

y comparar números

$X = Y$

$X \neq Y$

$X < Y$

$X > Y$

$X \geq Y$

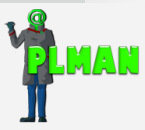
$X \leq Y$

EJEMPLO

`do(get(down)) :- s(down, '+'), s(down-left, X), s(down-right, Y), $X + Y > 5$.`

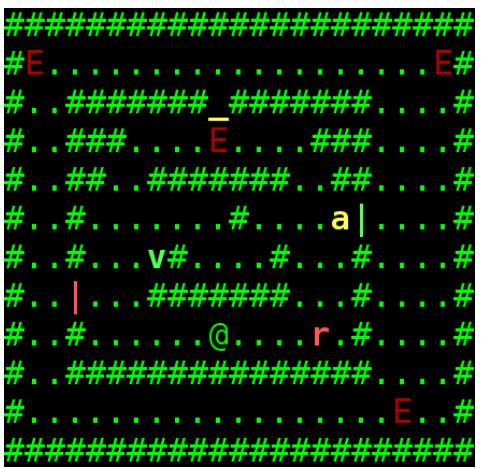
`do(get(up)) :- s(right, '+'), s(right-up, X), s(down-right, Y),`

`$Z \text{ is } X + Y$, write(Z), s(up, Z).`



Resuelve mapas de fase 2

1º los de maps/fase2



maps/fase2/mapa5.pll

- Comienzo FASE 2: **29 octubre hasta 11 noviembre**
- Nº mapas de fase 2: **3** de diferente dificultad.
- Mapa resuelto $\geq 75\%$ >> permite descargar siguiente mapa.
- Hasta 100% 3 intentos más >> la nota nunca baja.

FASE	DIFICULTAD				
	D1	D2	D3	D4	D5
0	0,100				
1	0,350	0,450	0,500	0,550	0,650
2	0,525	0,675	0,750	0,825	0,975
3	1,225	1,575	1,750	1,925	2,228
4		2,025	2,250	2,475	