




Un ejemplo de sistema experto (PROLOG)

Ingeniería de Conocimiento
3º Grado de Ingeniería Informática





Planteamiento

- Un programa en Prolog para averiguar el **nombre** de un polígono en función de lados, paralelismo y ángulos.
 - Novedades:
 - Va a seguir un sistema de preguntas, pero guiado por las contestaciones dadas anteriormente.
 - Por tanto, habrá preguntas fijas, pero otras no.
 - Se pretende dotar al prototipo de capacidad de aprender, es decir, de incorporar nuevas reglas en función de las entradas.
 - Esto supone la modificación de la base de conocimiento en la tiempo de ejecución
 - Factible, ya que PROLOG es un intérprete
- 



Triángulo - Conocimiento

- Todo polígono de **tres** lados.
- Lados iguales:
 - Isósceles
 - Equilátero
- Ángulo recto:
 - Triángulo rectángulo
- Mezcla de los anteriores:
 - Triángulo rectángulo isósceles





Triángulos - Reglas

nombre(triangulo):- **orden**(3).

nombre(trianguloIsosceles):- **nombre**(triangulo),
ladosIguales(2).

nombre(trianguloRectangulo):- **nombre**(triangulo),
anguloRecto(si).

nombre(trianguloRectanguloIsosceles):-
nombre(trianguloIsosceles),
nombre(trianguloRectangulo).

nombre(trianguloEquilatero):- **nombre**(triangulo),
ladosIguales(3).





Cuadrilátero - Conocimiento

- Todo polígono de **cuatro** lados.
- Lados paralelos:
 - Trapecio (2)
 - Paralelogramo (4)
 - Ángulo recto: rectángulo
 - Lados iguales:
 - Rombo
 - Angulo recto:
 - Cuadrado





Cuadriláteros - Reglas

nombre(cuadrilatero):- **orden**(4).

nombre(trapezio):- nombre(cuadrilatero), **ladosParalelos**(2).

nombre(paralelogramo):- nombre(cuadrilatero), **ladosParalelos**(4).

nombre(rectangulo):- nombre(paralelogramo), **anguloRecto**(si).

nombre(rombo):- nombre(paralelogramo), **ladosIguales**(4).

nombre(cuadrado):- nombre(rombo), nombre(rectangulo).





Predicados

- :-dynamic memory/2
 - Crea un predicado (memory)
 - /2 indica que tiene dos argumentos
 - Puede cambiar su definición en tiempo de ejecución
 - Se va a usar para ver si existe un atributo y su valor
- Para ello, se crea un predicado ask con tres parámetros:
 - El hecho buscado
 - La pregunta al usuario
 - La respuesta



Predicado “ask”

- Caso de que se conozca el hecho y su valor:
 - `ask(Pred, _, X):- memory(Pred, X).`
- Se conoce el hecho, pero no su valor o no importa.
 - `ask(Pred, _, _):- memory(Pred, _), !, fail.`
- No se ha encontrado el hecho, por lo que se plantea preguntarle al usuario:
 - `ask(Pred, Question, X):- write(Question), read(Y),
asserta(memory(Pred, Y)), X == Y.`






Preguntas ante hechos desconocidos

`ladosIguales(X):- ask(ladosIguales, '¿Cuántos lados iguales tiene la figura? ', X).`

`anguloRecto(X):- ask(anguloRecto, '¿La figura posee ángulos rectos (sí, no)? ', X).`

`ladosParalelos(X):- ask(ladosParalelos, '¿Cuántos lados paralelos tiene la figura (0, 2 o 4)? ', X).`

`orden(X):- ask(orden, '¿Cuántos lados? ', X).`





Metaintérprete

solve:-

```
retractall(memory(_,_)),
```

- Unifica “memory”, si está lo borra
 - Si no, lo crea dinámicamente con la ayuda de “dynamic”

```
findall(X, nombre(X), R),
```

- Crea una lista de instanciaciones (X), mediante la satisfacción del término (nombre), con resultado R.

```
write(R).
```

Crear el programa PROLOG y ejecutarlo con:

?solve.

