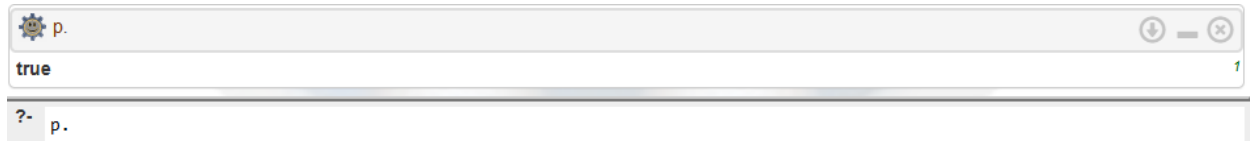


TALLER 7 PDF CONSULTAS.

PUNTO 1:

Prog1:



A screenshot of a Prolog window. The title bar shows a gear icon, a 'p.' label, and standard window controls. The main text area contains the text 'true' followed by a small '1' in the bottom right corner. Below the text area, there is a prompt '?-' followed by 'p.'.

Prog2:



A screenshot of a Prolog window. The title bar shows a gear icon, a 'p.' label, and standard window controls. The main text area contains the text 'procedure `t` does not exist' and 'Reachable from:' followed by a list of variables 's' and 'p'. Below the text area, there is a prompt '?-' followed by 'p.'.

Prog3:



A screenshot of a Prolog window. The title bar shows a gear icon, a 'p.' label, and standard window controls. The main text area contains the text 'procedure `s` does not exist' and 'Reachable from:' followed by a list of variables 'r', 'q', and 'p'. Below the text area, there is a prompt '?-' followed by 'p.'.





PUNTO 2:



A screenshot of a Prolog window. The title bar shows a gear icon, a 'p' label, and standard window controls. The main text area contains a small icon. Below the text area, there is a prompt '?-' followed by 'p.'.

En este caso, el programa es un ciclo infinito, ya que intenta demostrar que p es verdadero solo si p es verdadero, es decir, p depende de si mismo y no existe un caso base o un punto de salida que permita romper la dependencia recursiva. Esta clausula de horn es insatisfacible por eso que el programa no puede encontrar una respuesta.

PUNTO 3:

 `tiene(Persona, Mascota).`   

Mascota = gato,
Persona = maria
Mascota = gato,
Persona = pedro
Mascota = perro,
Persona = laura
Mascota = gato,
Persona = juan
Mascota = perro,
Persona = juan
Mascota = perro,
Persona = pedro

?- `tiene(Persona, Mascota).`

PUNTO 4:

 `verdura(papa).`   

false

 `verdura(Verdura).`   

Verdura = zanahoria
Verdura = cebolla

 `vegetariano(Persona).`   

Persona = jose
Persona = james

 `gusta(jose, X).`   

X = zanahoria
X = cebolla

 `gusta(Persona, cebolla).`   

Persona = jose

 `ama(Persona, cebolla).`   

Persona = jose
Persona = james

?- Your query goes here ...

PUNTO 5:

 `viajar(Persona).`

Persona = john

Next 10 100 1,000 Stop

 `viajero(Persona).`

Persona = john

Next 10 100 1,000 Stop


 `sano(Persona), adinerado(Persona).`


Persona = john

Next 10 100 1,000 Stop

?- `sano(Persona), adinerado(Persona).`

PUNTO 6:

 `riddle(Houses, FishOwner).`

 Singleton variables: [Nationality1,Color1,Cigar1,Beverage1,Pet1,Nationality2,Color2,Cigar2,Beverage2,Pet2,Nationality3,Color3,Cigar3,Beverage3,Pet3,Nationality4,Color4,Cigar4,Beverage4,Pet4,Nationality5,Color5,Cigar5,Beverage5,Pet5]

FishOwner = aleman,

Houses =

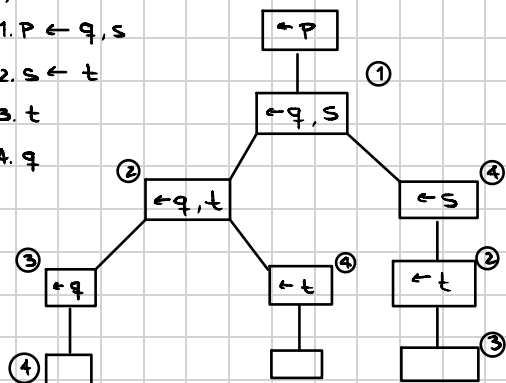
[house(1,noruego,amarillo,dunhill,agua,gato),
house(2,danes,azul,brends,te,caballo),
house(3,britanico,rojo,pallMall,leche,pajaro),
house(4,aleman,verde,prince,cafe,fish),
house(5,sueco,blanco,bluemasters,cerveza,perro)]

Next 10 100 1,000 Stop

?- `riddle(Houses, FishOwner).`

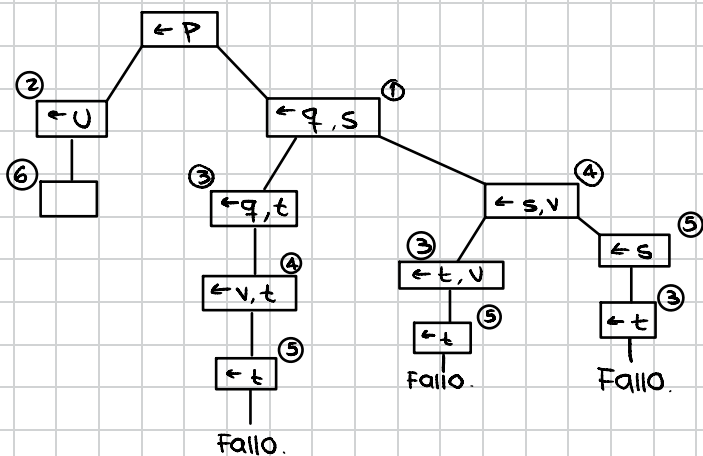
1)

1. $P \leftarrow q, s$
2. $s \leftarrow t$
3. t
4. q



2)

1. $P \leftarrow q, s$
2. $P \leftarrow u$
3. $s \leftarrow t$
4. $q \leftarrow v$
5. v
6. u



3)

1. $P \leftarrow q, s$
2. $q \leftarrow r$
3. $r \leftarrow s$
4. r

