ENTRADAS y SALIDAS

UNIVERSIDAD DE VALLADOLID

Índice:

Escritura de términos (I)

- Predicado predefinido "write(X)", que no se resatisface.
  - Si X está instanciada, se muestra en pantalla.
  - Si no, se saca una variable numerada('_225').

```
?- write(Beatriz).
_G244
Beatriz = _G244 ;
No
?- write("Beatriz").
[66, 101, 97, 116, 114, 105, 122]
Yes
?- write('Beatriz').
Beatriz
Yes
```

Escritura de términos (II)

- "nl" provoca un salto de línea. (?- nl.)
- "tab(X)" escribe X espacios en blanco. No coincide con la función de una máquina de escribir convencional.
  - Ambas sólo se satisfacen una sola vez.

```
?- nl, write('Beatriz'), tab(5), write('Gonzalez'), nl,
write('Francisco'), tab(5), write('Perez').

Beatriz     Gonzalez
Francisco     Perez

Yes
```

Escritura de términos (III)

- Ejemplo: escritura estructurada de listas.
  - La complejidad de la escritura de listas se debe al posible anidamiento con otras listas o estructuras.
  - pp(X, I): "*pretty print*"
    - X debe instanciar a la lista que se quiere mostrar
    - Objetivo: pp([1, 2, 3]

      1
      2
      3
    - Objetivo: pp[1,2,[3,4], 5, 6]

      1
      2
          3
          4
      5
      6

Escritura de términos (IV)

```
pp([H|T], I) :- !, J is I+3, pp(H, J), ppx(T, J), nl.
pp(X, I) :- tab(I), write(X), nl.

ppx([], _).
ppx([H|T], I) :- pp(H, I), ppx(T, I).

?- pp([1,2,[3,4], 5, 6], 0).
1
2
   3
   4
5
6

Yes
```

- I es un contador de columna

```
?- pp([1,2,[3,4], 5, 6], 7).
       1
       2
          3
          4
       5
       6

Yes
```

Escritura de términos (V)

```
?- write(a+b*c/d), nl, display(a+b*c/d), nl.
a+b*c/d
+(a, /(*(b, c), d))
Yes
```

- "write" tiene no en cuenta la declaración de los operadores; "display", sí.
- Por ejemplo, el operador "+" es declarado infijo con dos argumentos.
- "display" puede ser útil para recordar el orden de precedencia:

```
?- write((a+b)*c/d), nl, display((a+b)*c/d), nl.
(a+b)*c/d
/(*(+(a, b), c), d)
Yes
```

Lectura de términos (I)

- "read(X)", lee por teclado un término, que se instanciará a la variable X.
- El término debe ir seguido de "." y un carácter no imprimible como el espacio o el retorno de carro.

```
hola :- write('Nombre: '), read(Nombre),
   write('1º Apellido: '), read(Apellido1),
   write('2º Apellido: '), read(Apellido2), nl,
   write('Hola '), write(Nombre), tab(1),
   write(Apellido1), tab(1), write(Apellido2).

?- hola.
Nombre: Francisco.
1º Apellido: Perez.
2º Apellido: Garcia.

Hola _I131 _I132 _I133      (Funciona con minúsculas sólo)
```

Lectura de términos (II)

```
?- hola.
Nombre: francisco.
1º Apellido: perez.
2º Apellido: garcia.

Hola francisco perez  garcia
```

- Las mayúsculas se utilizan para designar a variables, por eso se mostraba el valor del puntero a estas supuestas variables: Francisco, Perez y Garcia.

Escritura y lectura de caracteres (I)

- El carácter es la unidad más pequeña objeto de lectura o escritura.
- Si una variable está instanciada a un código ASCII (entero positivo):

```
?- put(104), put(111), put(108), put(97).
hola
```

- "put(X)" escribe el correspondiente carácter. No admite resatisfacción.

Escritura y lectura de caracteres (II)

- Ejercicio 1:
  - Crear un predicado denominado espacios(X), cuya función sea similar a tab(X) mediante el empleo de put(32) de manera recursiva.
- Ejercicio 2:
  - Empleando una estrategia similar al ejercicio anterior, simular igualmente al predicado "write" mediante otro denominado "escribecadena" utilizando el desglose de las listas –cadenas de caracteres- (cabeza y cola) y "put".

Escritura y lectura de caracteres (III)

- "get0(X)" y "get(X)" son objetivos que se satisfacen si X no está instanciada y no se resatisfacen.
- Diferencia: "get0(X)" espera a instanciar en X el primer carácter tecleado, mientras que "get(X)" captura el primero imprimible.

```
?- get(X).          ?- get0(X).
|:<CR>              |: <CR>
|: .
X = 46              X = 10 ;
```

Escritura y lectura de caracteres (IV)

- Ejercicio 3:

?- leer(S).

El ladron, que es muy habil con las manos, le birlo la cartera a Juan.

S=[el, ladron, `,', que, es, muy, habil, con, las, manos, `,', le, birlo, la, cartera, a, juan]

- Consideraciones:
  - Los espacios son separadores de palabras.
  - , ; : ? ! . Son palabras por sí mismas
  - . ! ? Son la última palabra de la frase

Escritura en fichero (I)

- Canal de salida activo: por defecto, la pantalla.
- "tell(X)", establece la salida según lo instanciado por X.
- Es un predicado que requiere que X esté instanciada; en caso contrario, se produce un error.
- Como objetivo, no permite la resatisfacción.

Escritura en fichero (II)

- Ejercicio:
  - Desviar la salida del predicado "pp" hacia el fichero c:\tmp\pp.txt

```
?- tell('c:/tmp/pp.txt')
?- pp([1, 2, [3, 4], 5, 6], -3).
?- told.
```

- Este último predicado cierra el fichero y restablece la pantalla como canal de salida.

Escritura en fichero (III)

- Para averiguar el canal activo, se emplea el predicado "telling(X)". Aquí X está por instanciar.

```
?- telling(X).
X = '$stream'(65086996) ;
No

?- tell('d:/tmp/salida.prolog').
Yes

?- telling(X).
X = '$stream'(112102) ;
No

?- told.
Yes
```

Escritura en fichero (IV)

- Para volver a la pantalla como canal activo, sin cerrar el fichero abierto previamente, se emplea: "tell(user)".

```
?- tell('c:/tmp/pp1.txt'),
pp([1, 2, [3, 4], 5, 6], -3),
tell(user), pp([1, 2, [3, 4], 5, 6]),
tell('c:/tmp/pp2.txt'),
pp([7, [8, [9, [0]]]], 0),
told, pp([7, [8, [9, [0]]]], 0), told.
```

- Ejercicio: insertar "telling" en esta secuencia para averiguar el canal activo.

Lectura de ficheros (I)

- Las mismas consideraciones, sólo que:
  - El canal de entrada por defecto será el teclado (user).
  - El predicado "tell" se sustituye por "see":
    - ?- see(X).
    - ?- seeing(X).
    - ?- seen.
- Si la lectura se efectúa para ampliar la base de conocimiento, se empleará "consult".

Lectura de ficheros (II)

- Ejemplo: construir una base de conocimiento fruto de la unión otras tres.

```
?- consult('PrettyPrint').
% PrettyPrint compiled 0.00 sec, 40 bytes

?- consult('ReyesCasaAustria').
% ReyesCasaAustria compiled 0.00 sec, 0 bytes

?- consult('TresEnRaya').
% TresEnRaya compiled 0.02 sec, 0 bytes


?- ['PrettyPrint', 'ReyesCasaAustria', 'TresEnRaya'].
% PrettyPrint compiled 0.00 sec, 0 bytes
% ReyesCasaAustria compiled 0.00 sec, 0 bytes
% TresEnRaya compiled 0.00 sec, 0 bytes
```

Declaración de operadores (I)

- Utilización de un predicado predefinido:
  - ?- op(Precedencia, Especificaciones, Nombre).
  - Precedencia $\in$ [1, 255] y depende de implementaciones.
  - Especificaciones:
    - Notación:
      - Infijo: xfy, yfx, xfx, yfy
        (f representa operador; x e y operandos)
      - Prefijo: fx, fy.
      - Sufijo: xf, yf

Declaración de operadores (II)

- Asociatividad:
  - "y" indica que el argumento puede contener operadores de igual o menor precedencia.
  - "x" obliga a que sea estrictamente menor.
  - Ejemplo: "+" declarado como yfx.
    "a+b+c": a+(b+c) ó (a+b)+c
    La segunda opción queda eliminada porque el segundo argumento no puede contener un operador de precedencia igual.
  - Esto se conoce como asociatividad por la izquierda (yfx), por la derecha (xfy) o indistintamente (xfx); según esto, no tendría sentido (yfy).