

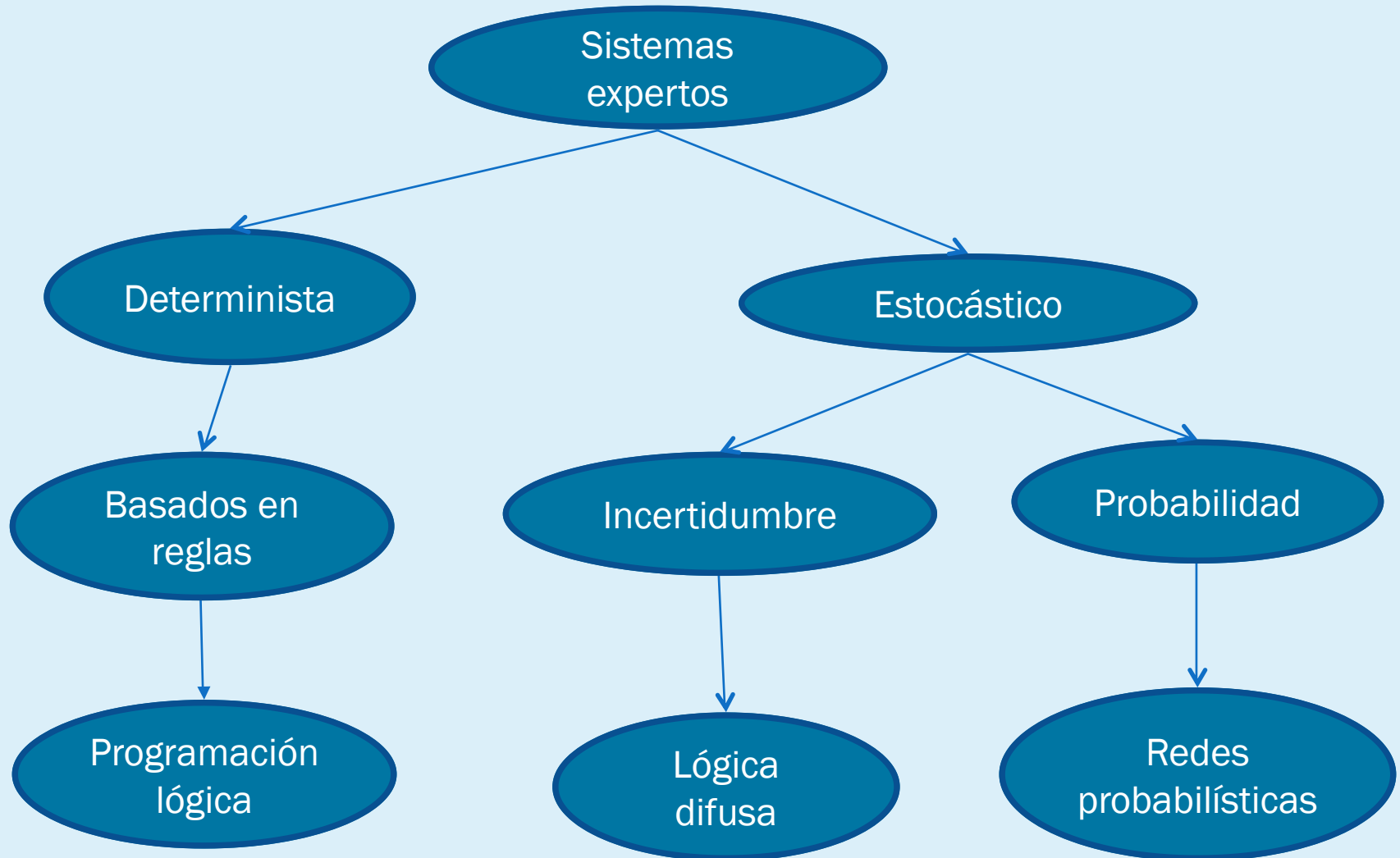


# CONCLUSIÓN SISTEMAS EXPERTOS

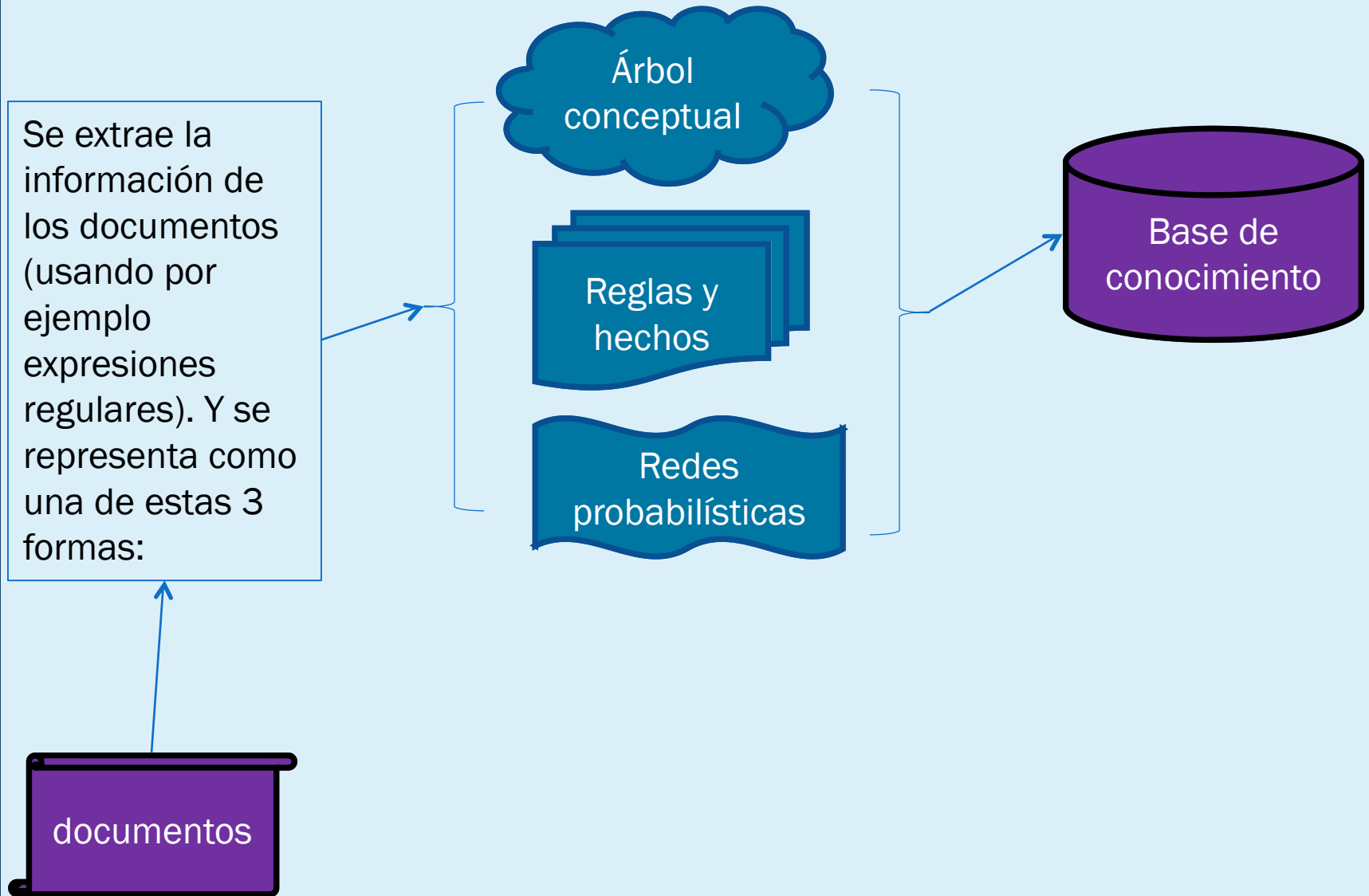
JAFH



# Tipos de sistemas expertos



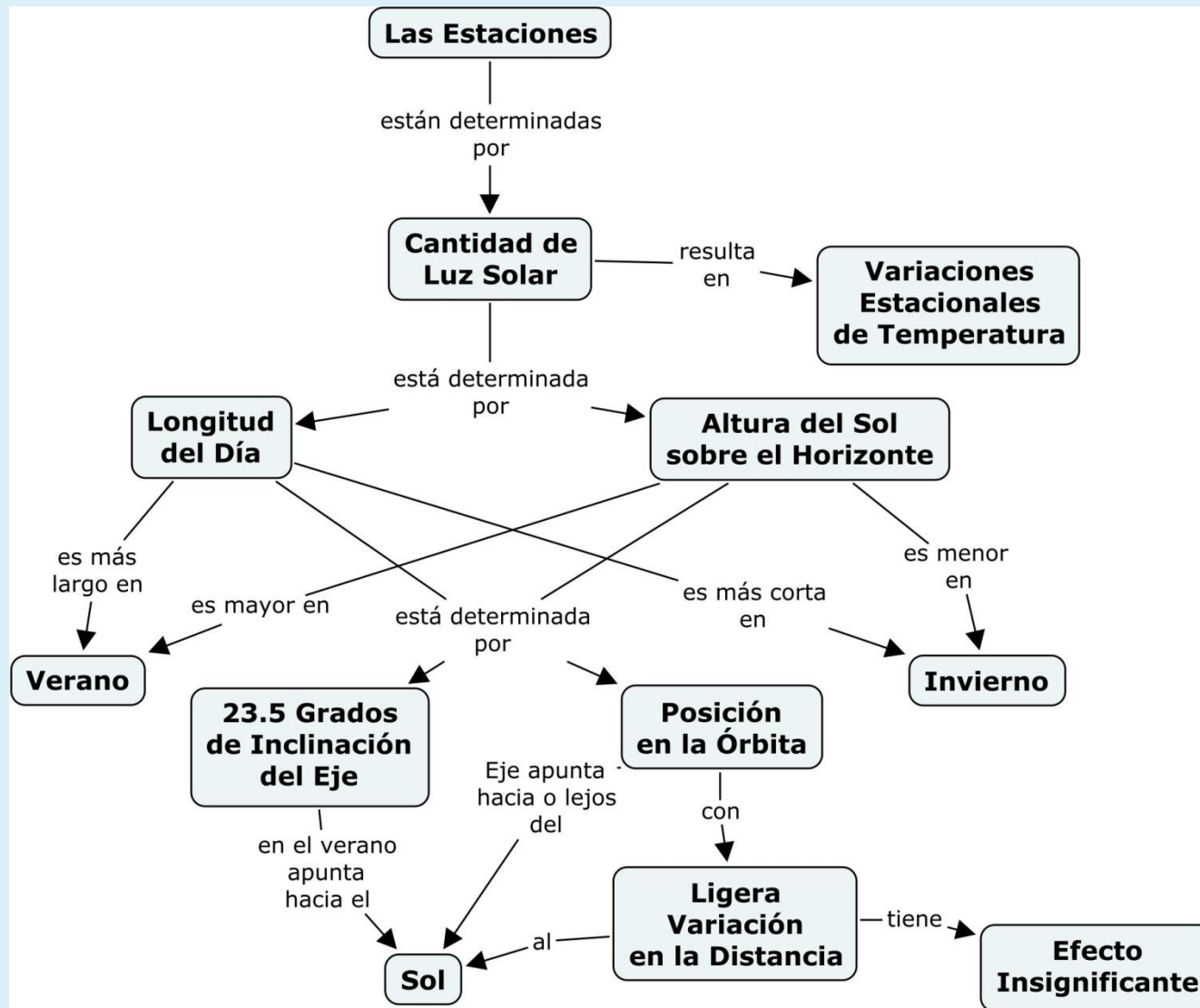
## Base de Conocimiento, se representa con:



# Expresiones regulares ver:

- [Jflorespampano.blogspot.mx](http://Jflorespampano.blogspot.mx)
  - *Entrada 9 octubre 2007 (polaca inversa)*

# Ejemplo de conocimiento representado como árbol conceptual



Para el conocimiento representado  
en:

Redes probabilísticas, tenemos:

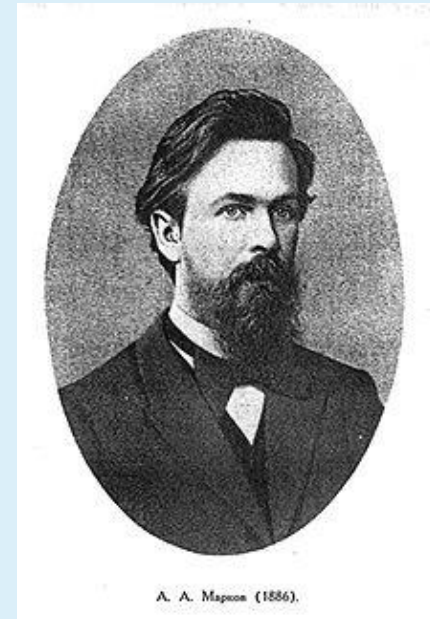
## ■ Redes Bayesianas

- <https://sites.google.com/site/cuerpoacademicounacarca35/Resources>

## ■ Cadenas de Markov.



Thomas Bayes  
1702-1761



Andréi Márkov  
1856-1922

# Teorema de Bayes

Thomas Bayes matemático Inglés del siglo XVI trabajo en las leyes probabilísticas y particularmente en las probabilidades de sucesos condicionados, una de sus principales aportaciones es la del teorema que lleva su nombre

$$P(A_i | B) = \frac{P(A_i) P(B | A_i)}{\sum_{i=1}^n P(A_i) P(B | A_i)}$$

# Cadenas de Markov

Es una sucesión de ensayos similares u observaciones en la cual cada ensayo tiene el mismo número finito de resultados posibles y en donde la probabilidad de cada resultado para un ensayo dado depende sólo del resultado del ensayo inmediatamente precedente y no de cualquier resultado previo.



Por otra parte tenemos el conocimiento basado en sistemas expertos Deterministas

Basados en

- Reglas

- Hechos

Determinista por que se basa en inferencias lógicas

A implica B

A	B	$A \Rightarrow B$
V	V	V
V	F	F
F	V	V
F	F	V

A si y solo si B

A	B	$A \Leftrightarrow B$
V	V	V
V	F	F
F	V	F
F	F	V

# Conceptos

Tautología. Resulta verdadera para cualquier combinación.

Contradicción. Es falsa para cualquier combinación.

Falacia. Inferir una falsedad a partir de premisas verdaderas.

# Representar la BC- Prolog

Para expresar reglas y hechos de nuestra base de conocimiento (BC), usaremos el lenguaje ***Prolog*** (Programación lógica) descargue su implementación libre en:

<http://www.swi-prolog.org/download/stable>

# Hechos

Un hecho es una relación entre objetos

*relación(objeto,objeto)*

Ejemplo:

`aprobo(juan,programacion1).`

Donde aprobó es el predicado, juan y programacion1 son los argumentos.

Los nombres de predicados y argumentos deben iniciar con minúscula.

Esto se lee como: juan aprobó programación 1

# Reglas

Una regla consta de 2 partes, cabeza y cuerpo:

consecuente:-antecedente.

El antecedente consta de varios hechos.

consecuente:-hecho1,hecho2,...,hechon.

Se lee: El consecuente es verdad si el antecedente es verdad

# Ejemplo

`programa(P):-aprobo(P,programacion1).`

Donde P es una variable (las variables inician con mayúscula),.

Se lee el elemento P programa (sabe programar) si P aprobó programación 1.

# Archivo 'uno.pl'

Ejemplo, suponga que tiene el siguiente conjunto de hechos, esta será nuestra base de conocimiento

```
aprobo(juan,programacion1).  
aprobo(maria,programacionoo).  
reprobo(juan,programacion2).  
reprobo(maria,estructuradedatos).  
reprobo(juan,programacionoo).  
programa(P):-aprobo(P,programacion1).  
programaoo(P):-aprobo(P,programacionoo).
```



# Consultas

?-reprobo(maria,X).

?-programaoo(juan).

?-call(aprobo,juan,programacion1).

Dada esta pequeña base de conocimiento en prolog podemos hacer consultas como estas, a lo que el prolog responderá con cierto o falso

# Comandos básicos PROLOG

?-pwd. //muestra carpeta corriente

?-cd('c:/trabajo'). //modifica carpeta de trabajo

?-consult('archivo'). //Carga clausulas

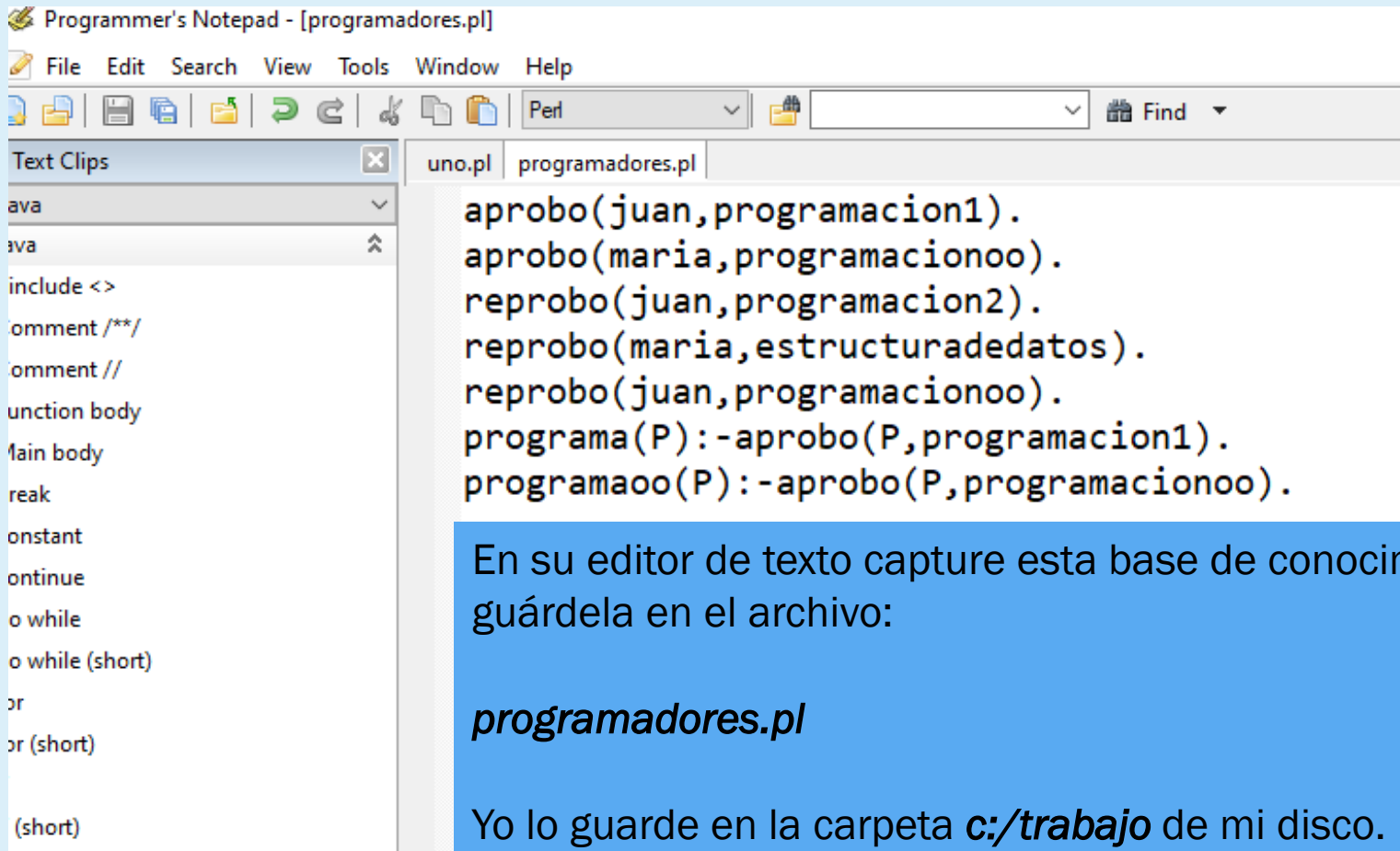
?.[archivo]. //carga clausulas

?-edit('archivo')

# call

call comprueba la veracidad de un predicado, pasado como primer parámetro, el resto de los argumentos son los parámetros del predicado.

# Probar el ejemplo



Programmer's Notepad - [programadores.pl]

File Edit Search View Tools Window Help

Text Clips

- ava
- ava
- include <>
- omment /\*\*/
- omment //
- unction body
- lain body
- reak
- onstant
- ontinue
- o while
- o while (short)
- or
- or (short)
- (short)

```
aprobo(juan,programacion1).
aprobo(maria,programacionoo).
reaprobo(juan,programacion2).
reaprobo(maria,estructuradedatos).
reaprobo(juan,programacionoo).
programa(P):-aprobo(P,programacion1).
programaoo(P):-aprobo(P,programacionoo).
```

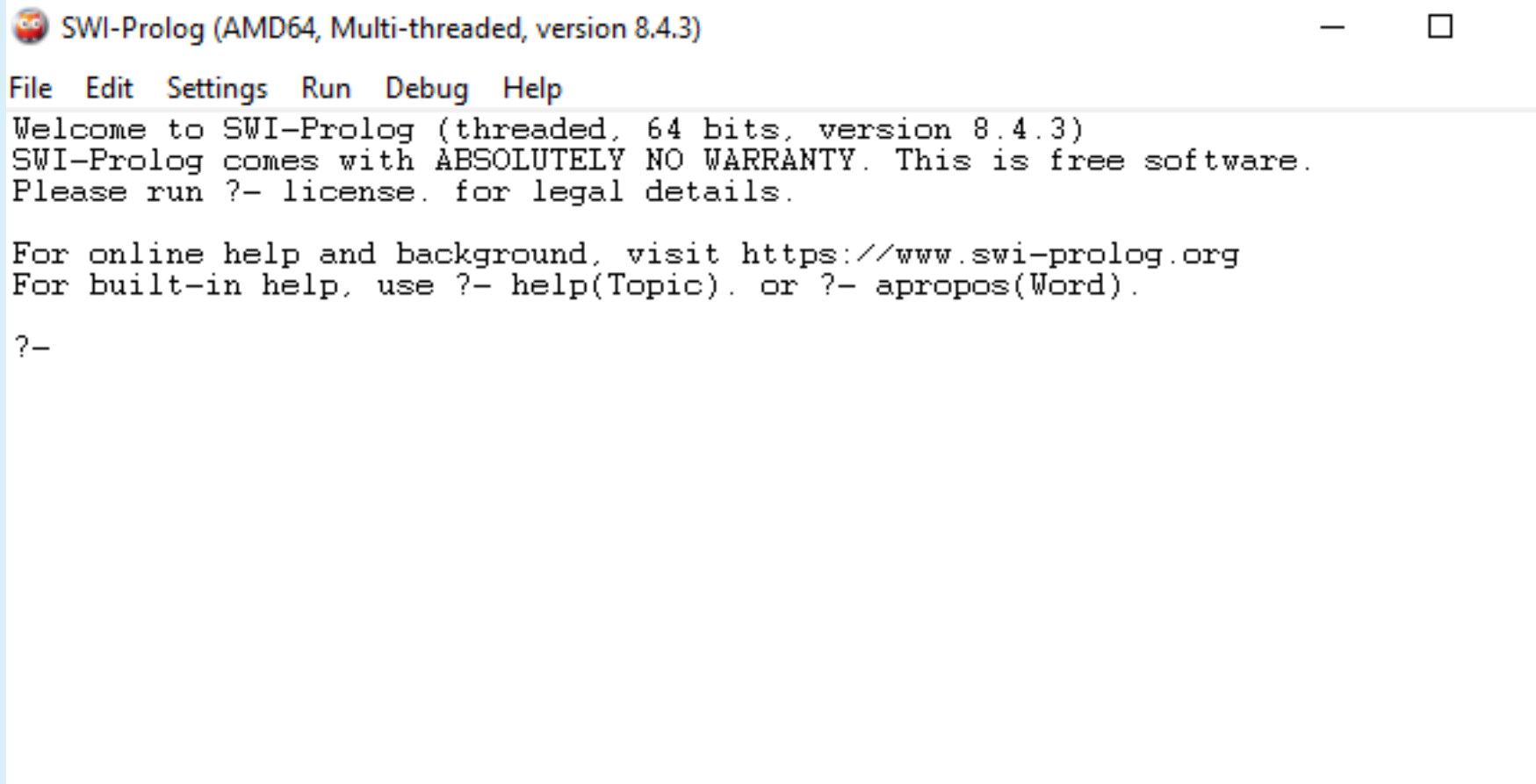
En su editor de texto capture esta base de conocimiento y guárdela en el archivo:

***programadores.pl***

Yo lo guarde en la carpeta ***c:/trabajo*** de mi disco.

Si no dispone de un editor de texto vea la diapositiva 26-30

# Ejecute swi-prolog

A screenshot of the SWI-Prolog (AMD64, Multi-threaded, version 8.4.3) application window. The window has a title bar with the application name and version, and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with 'File', 'Edit', 'Settings', 'Run', 'Debug', and 'Help'. The main text area displays a welcome message and instructions for using the software.

```
SWI-Prolog (AMD64, Multi-threaded, version 8.4.3)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
```

## Vamos a la carpeta donde esta nuestra base de conocimiento

 SWI-Prolog (AMD64, Multi-threaded, version 8.4.3)

File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
```

```
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
```

```
?- cd('c:/trabajo').
true.
```

```
?-
```

Para cambiar a su carpeta de trabajo (o donde haya guardado su BC), escriba el comando:

***cd('carpeta').***

Como se muestra en la imagen

No olvide que las sentencias en prolog deben terminar con punto, si devuelve **true**, todo va bien.

# Cargue la base de conocimiento

```
SWI-Prolog (AMD64, Multi-threaded, version 8.4.3)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- cd('c:/trabajo').
true.

?- consult('programadores.pl').
true.

?-
```

# Haga sus consultas

Por ejemplo para preguntar si maría reprobó alguna materia, usamos la consulta:

***reprobo(maria,X).***

Donde X es una variable (recuerde que las variables van con mayúsculas) y ***prolog*** nos devuelve X con el valor de la materia reprobada por maría si esta existe.

 SWI-Prolog (AMD64, Multi-threaded)

File Edit Settings Run Debug

Welcome to SWI-Prolog (threaded)  
SWI-Prolog comes with ABSOLUTELY NO WARRANTY.  
Please run ?- license. for details.

For online help and background information  
For built-in help, use ?- help.

?- cd('c:/trabajo').

**true.**

?- consult('programadores.pl').

**true.**

?- reprobo(maria,X).

X = estructuradedatos.

?-



# O podemos consultar lo siguiente:



SWI-Prolog (AMD64, Multi-threaded, version 8.6.3)

File Edit Settings Run Debug Help

Welcome to SWI-Prolog (threaded)  
SWI-Prolog comes with ABSOLUTELY NO WARRANTY.  
Please run `?- license.` for legal details.

For online help and background information, see  
For built-in help, use `?- help.`

```
?- cd('c:/trabajo').  
true.
```

```
?- consult('programadores.pl').  
true.
```

```
?- reprobo(maria,X).  
X = estructuradedatos.
```

```
?- programaoo(juan).  
false.
```

```
?- call(aprobo,juan,programacion1).  
true.
```

```
?-
```

Preguntamos si juan programa orientado a objetos

***programaoo(juan).***

Preguntamos si juan aprobó programación 1

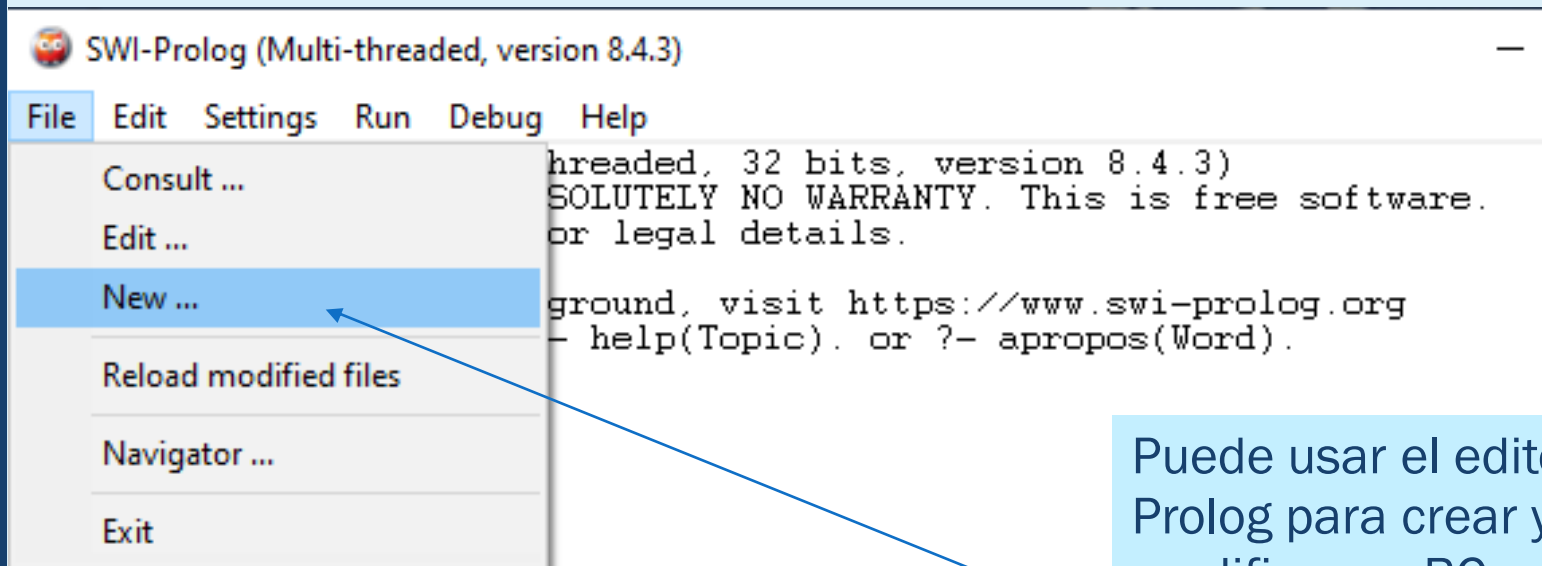
***call(aprobó,juan,programacion1).***

Que a la luz de nuestra base de conocimiento nos devuelve false y true

Recuerde que nuestra base de conocimiento contiene:

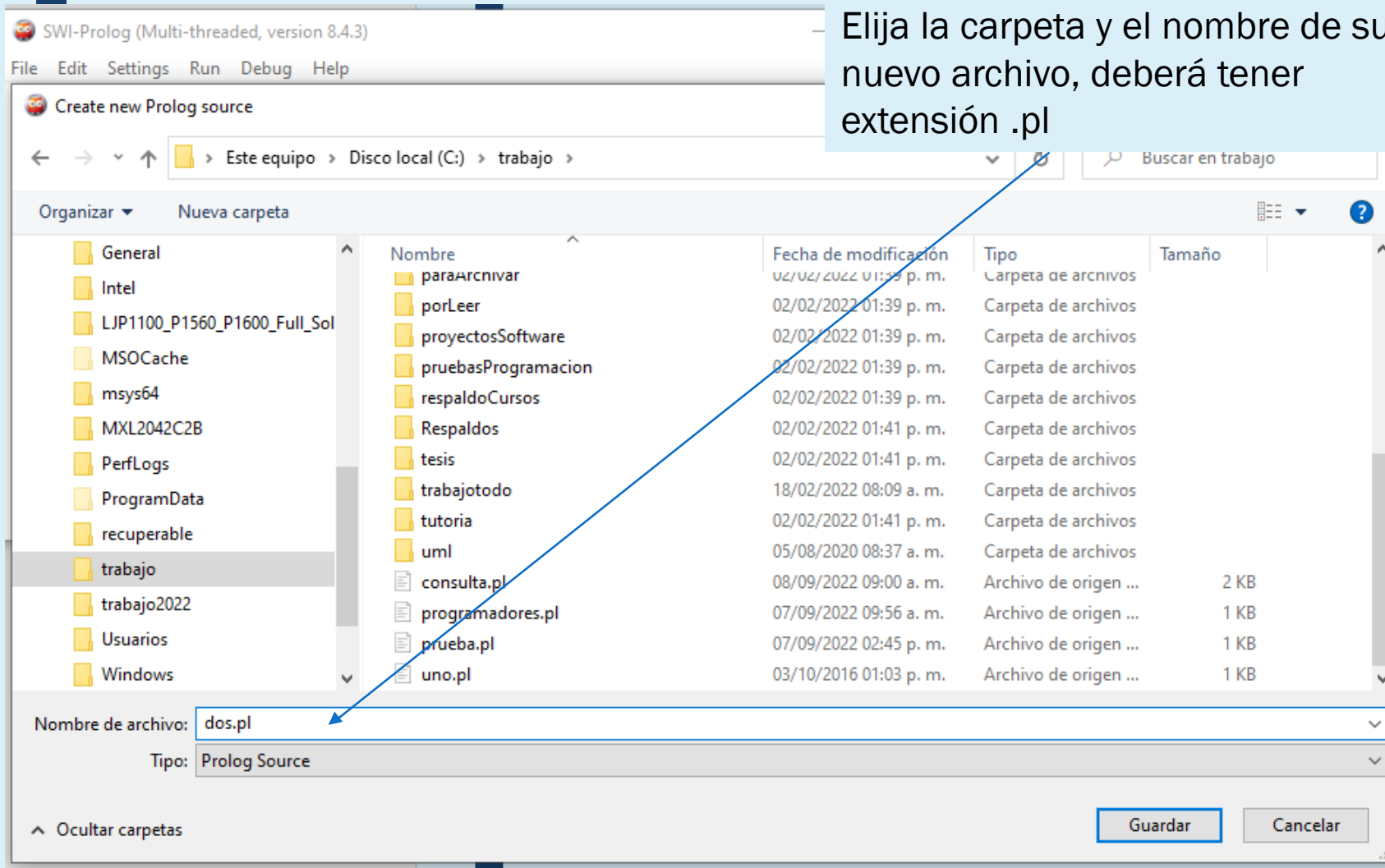
```
aprobo(juan,programacion1).  
aprobo(maria,programacionoo).  
reprobo(juan,programacion2).  
reprobo(maria,estructuradedatos).  
reprobo(juan,programacionoo).  
programa(P):-aprobo(P,programacion1).  
programaoo(P):-aprobo(P,programacionoo).
```

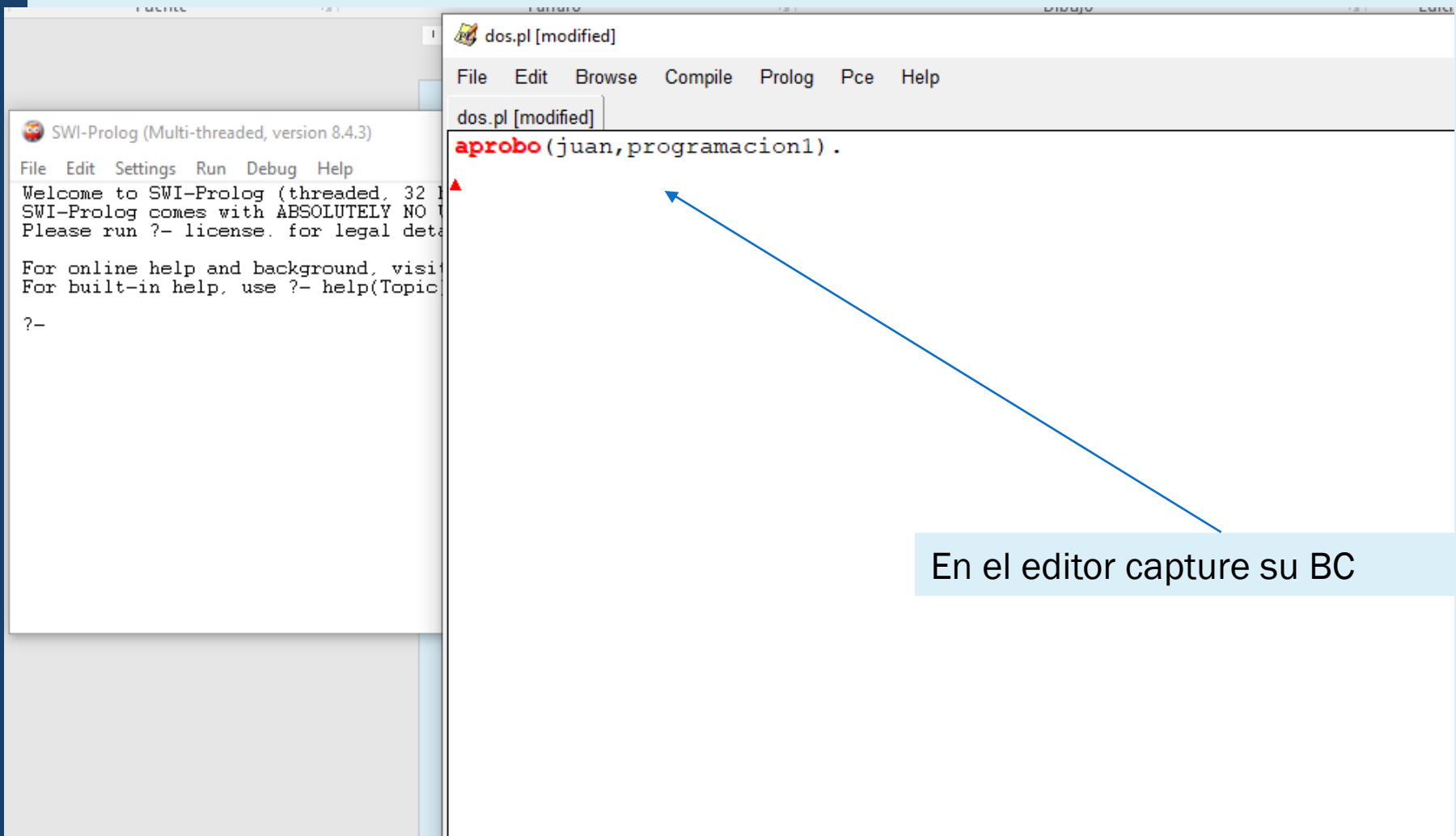
# Otra forma de crear la base de conocimiento

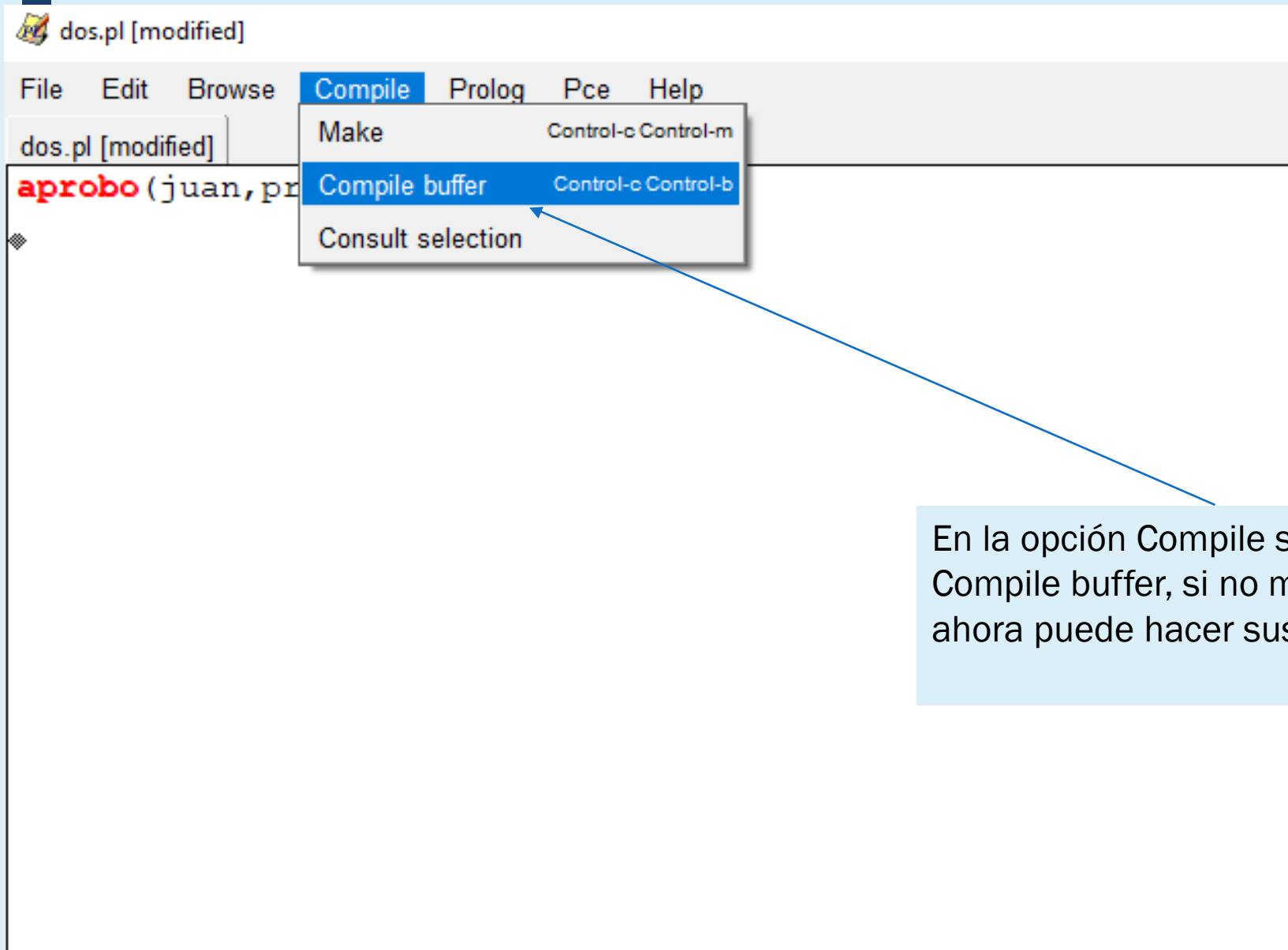


Puede usar el editor de Swi Prolog para crear y modificar su BC.

En el menú file elija la opción new







En la opción Compile seleccione  
Compile buffer, si no marca error,  
ahora puede hacer sus consultas

dos.pl

File Edit Browse Compile Prolog Pce Help

dos.pl

```
aprobo(juan,programacion1).
```

SWI-Prolog (Multi-threaded, version 8.4.3)

File Edit Settings Run Debug Help

Welcome to SWI-Prolog (threaded, 32 bits, version 8.4.3)  
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.  
Please run ?- license. for legal details.

For online help and background, visit <https://www.swi-prolog.org>  
For built-in help, use ?- help(Topic). or ?- apropos(Word).

```
?- aprobo(juan,programacion1).  
true.
```

```
?- aprobo(X,programacion1).  
X = juan.
```

```
?- █
```

Ahora puede hacer sus con

# Fin

Este código se probó con swiprolog 8.4.3 para 64 bits, usted debe cargar el de 32 o 64 bits según su equipo de cómputo.

Ligas:

- <http://www.swi-prolog.org/download/stable>
- [Jflorespampano.blogspot.com](http://Jflorespampano.blogspot.com)