

2018

Proyecto de Representación del Conocimiento

SWI - Prolog

Sistema que se fundamenta en una Base de Conocimiento de forma estructurada para determinar las Propiedades y Relaciones en Clases y Objetos.

Equipo:

Jessica Sarahi Méndez Rincón
Juan Daniel Lawrence Pedroza
Marco Tulio Sánchez Rodríguez
Nahet Cortez Fuerte
Rodrigo Terpán Arenas

MAESTRÍA EN CIENCIA E INGENIERÍA DE LA
COMPUTACIÓN
INTELIGENCIA ARTIFICIAL
Semestre 1 2019-1

Profesores:

- Dr. Luis A. Pineda Cortes, IIMAS, UNAM
- Dr. Arturo Rodríguez García, Facultad de Ingeniería, UNAM
- Mtro. Iván Torres Rodríguez, PCIC, UNAM

Fecha de entrega Jueves 15 de Noviembre de 2018



Contenido

• Objetivo	3
• Base de Conocimientos.....	3
• Ilustración 1 Diagrama de Taxonomía.....	4
• Archivo de la Base de Conocimiento	5
• Diseño y Desarrollo de Módulos	6
• Ejecución.....	7
• Módulos del Sistema	11
• Módulo de Consulta	11
• Consulta de Clases.....	11
• Consulta de Propiedades de Clases.....	11
• Consulta de Preferencias de Propiedades de Clases.....	12
• Consulta de Relaciones de Clases.....	12
• Consulta de Preferencias de Relaciones de Clases.....	13
• Consulta de Objetos.....	14
• Consulta de Propiedades de Objetos	14
• Consulta de Preferencias de Propiedades de Objetos.....	15
• Consulta de Relaciones de Objetos.....	16
• Consulta de Preferencias de Relaciones de Objetos	16
• Módulo de Añadir	17
• Añadir de Clases	17
• Añadir de Propiedades de Clases.....	18
• Añadir de Preferencias de Propiedades de Clases.....	19
• Añadir de Relaciones de Clases	19
• Añadir de Preferencias de Relaciones de Clases	20
• Añadir de Objetos	21
• Añadir de Propiedades de Objetos.....	21
• Añadir de Preferencias de Propiedades de Objetos.....	22
• Añadir de Relaciones de Objetos	23
• Añadir de Preferencias de Relaciones de Objetos	23
• Módulo de Eliminar.....	25

• Eliminar de Clases.....	25
• Eliminar de Propiedades de Clases	25
• Eliminar de Preferencias de Propiedades de Clases	26
• Eliminar de Relaciones de Clases	27
• Eliminar de Preferencias de Relaciones de Clases.....	27
• Eliminar de Objetos	29
• Eliminar de Propiedades de Objetos	29
• Eliminar de Preferencias de Propiedades de Objetos	29
• Eliminar de Relaciones de Objetos	30
• Eliminar de Preferencias de Relaciones de Objetos	31
• Módulo de Modificar.....	32
• Modificar de Clases	33
• Modificar de Propiedades de Clases	33
• Modificar de Preferencias de Propiedades de Clases	34
• Modificar de Relaciones de Clases	34
• Modificar de Preferencias de Relaciones de Clases	35
• Modificar de Objetos	36
• Modificar de Propiedades de Objetos.....	36
• Modificar de Preferencias de Propiedades de Objetos	37
• Modificar de Relaciones de Objetos	37
• Modificar de Preferencias de Relaciones de Objetos	38

Objetivo

Llevar a cabo la comprensión de los temas de Inteligencia Artificial de Representación del Conocimiento con el Lenguaje de Programación Prolog mediante un Proyecto que ayude a construir una Base de Conocimiento para consultar la taxonomía de individuos en un dominio en particular con relaciones y propiedades.

De igual forma:

- ✂ Establecer el conocimiento de forma económica
- ✂ Usar la inferencia de manera eficiente

Base de Conocimientos

Debido a que se requiere una taxonomía como ejemplo para el arranque del Sistema con pruebas y validación de los diversos módulos, el equipo analizó, diseño y estructuro una base con taxonomía del reino animal y con individuos que servirán de ejemplos para establecer propiedades y relaciones.

Definiéndose la siguiente estructura:

En base a Clases e Individuos u objetos se estructuraron sus propiedades y relaciones de la siguiente forma:

Para las Clases:

Class(nombre_clase, nombre_clase_padre, nombre_propiedad,nombre_Relacion,objetos)

Class(nombre_clase,padre_clase,[],[],[]) =>Clase sin propiedades y sin relaciones

Para los objetos:

Id=>nombre_objeto, nombre_propiedad, nombre_Relacion

Id =>nombre_objeto,[],[] =>Objeto sin propiedades y sin relaciones

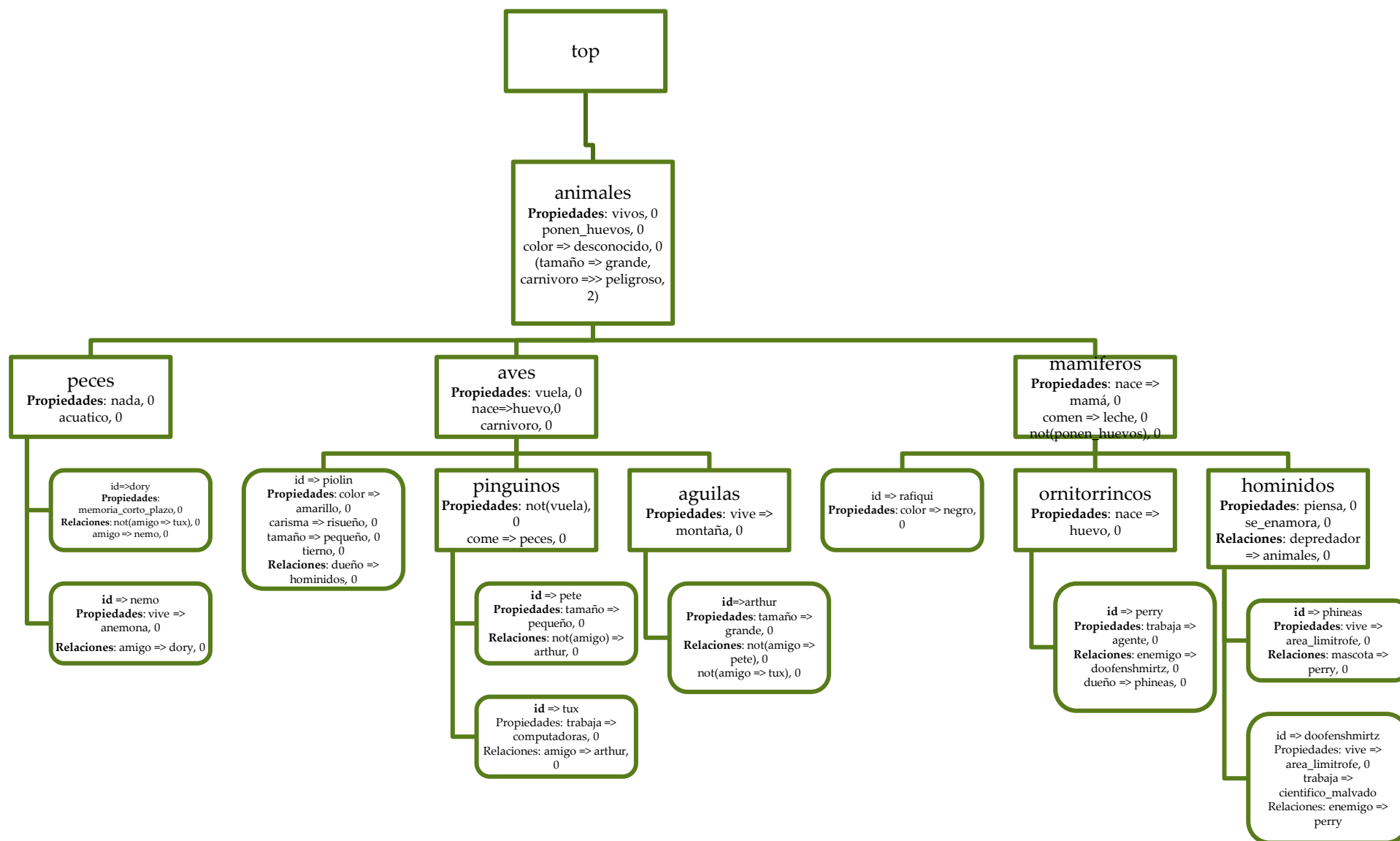


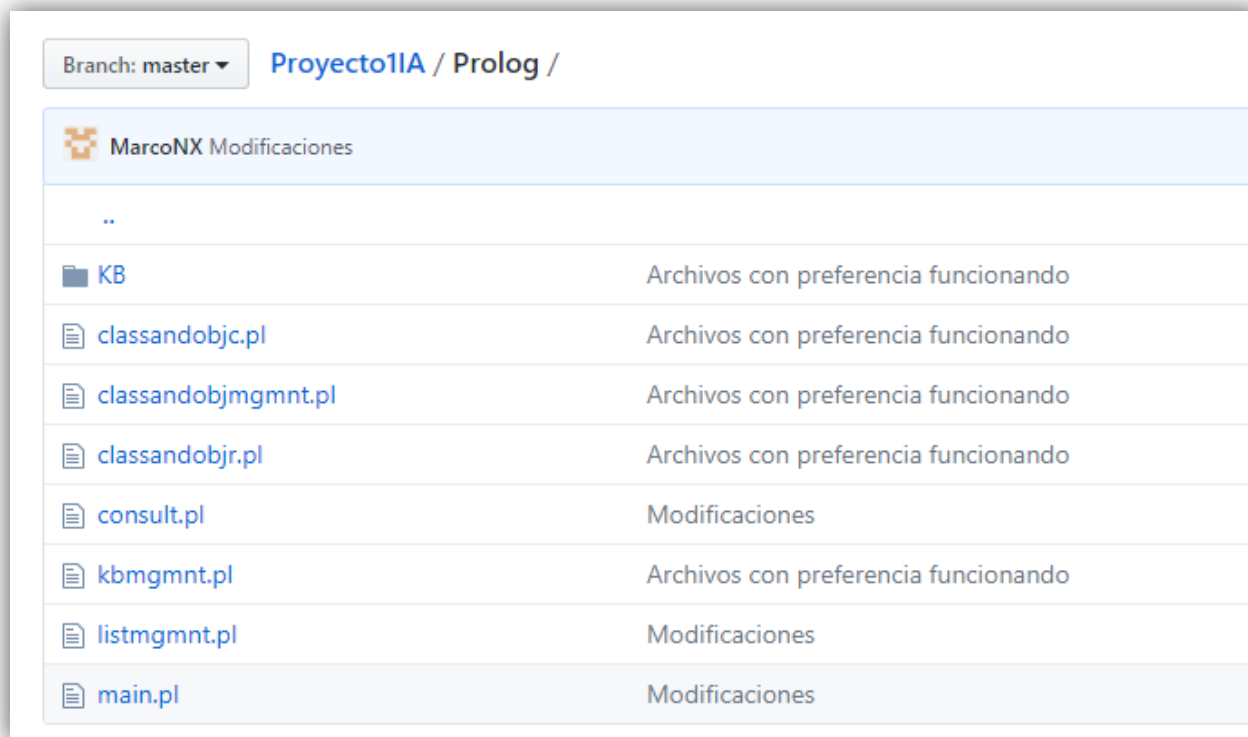
Ilustración 1 Diagrama de Taxonomía

Archivo de la Base de Conocimiento

```
[
class(top,none,[],[]),
class(animales,top,[[vivos,0],[ponen_huevos,0],[color=>desconocido,0],[[tamano=>grande,carnivoro]==>[peligroso,2]],[],[]),
class(peces,animales,[[nadan,0],[acuatico,0]],[],[
    [ id=>dory,      [[memoria_corto-plazo,0] ],      [[not(amigo=>tux),0],[amigo=>nemo,0] ]],
    [ id=>nemo,      [[vive=>anemona,0] ],      [[amigo=>dory,0] ]]]),
class(aves,animales,[[vuela,0],[nace=>huevo,0],[carnivoro,0]],[],[
    [ id=>piolin,    [[color=>amarillo,0],[carisma=>risueno,0],[tamano=>pequeno,0],[tierno,0] ]],
    [[mascota=>hominidos,0] ]])],
class(pinguinos,aves,[[not(vuelan),0]],[[come=>peces,0]],[
    [ id=>pete,      [[tamano=>pequeno,0] ],      [[not(amigo=>arthur),0] ]],
    [ id=>tux,       [[trabaja=>computadoras,0] ],      [[amigo=>arthur,0] ]]]),
class(aguilas,aves,[[vive=>montana,0]],[],[
    [ id=>arthur,    [[tamano=>grande,0] ],      [[not(amigo=>pete),0],[not(amigo=>tux),0] ]]]),
class(mamiferos,animales,[[nace=>mama,0],[comen=>leche,0],[not(ponen_huevos),0]],[],[
    [ id=>rafiqui,    [[color=>negro,0] ],      []]),
class(ornitorrincos,mamiferos,[[nace=>huevo,0],[ponen_huevos,0]],[],[
    [ id=>perry,      [[trabaja=>agente,0] ],      [[enemigo=>doofenshmirtz,0],[dueno=>phineas,0] ]]]),
class(hominidos,mamiferos,[
    [piensa=>yes,0], [seenamora=>yes,0]],[[depredadorDe=>animales,0]],[
    [ id=>phineas,[[vive=>arealimitrofe,0]],[[mascota=>perry,0] ],
    [ id=>doofenshmirtz,[[vive=>arealimitrofe,0],[trabaja=>maldad,0]],[[enemigo=>perry,0] ] ]])
]
```

Diseño y Desarrollo de Módulos

Se diseñaron los archivos que contenían las diversas funcionalidades del Sistema en Prolog, de acuerdo a la estructura que se muestra en la imagen:



Branch: master ▾ Proyecto1IA / Prolog /	
MarcoNX Modificaciones	
..	
KB	Archivos con preferencia funcionando
classandobjc.pl	Archivos con preferencia funcionando
classandobjmgmnt.pl	Archivos con preferencia funcionando
classandobjr.pl	Archivos con preferencia funcionando
consult.pl	Modificaciones
kbmgmnt.pl	Archivos con preferencia funcionando
listmgmnt.pl	Modificaciones
main.pl	Modificaciones

Teniendo en cuenta los módulos de

main.pl

Contiene el código para invocar las funcionalidades del Archivo así como para invocar el conjunto de funcionalidades que lleven a cabo una tarea para la transacción de información y guardado en el archivo de entrega final.

listmgmnt.pl

Contiene el código Core del resto de las funciones, son operaciones directas dentro de las listas que lee y codifica directamente del archivo en la base de conocimientos.

kbmgmnt.pl

Contiene los servicios de consulta de las Clases y de las propiedades, así como de relaciones de objetos.

consult.pl

Contiene los servicios de consulta de las Clases, las propiedades, las preferencias y relaciones .

classandobjr.pl

Contiene los servicios para el módulo de Eliminar, todas las funciones que se usan tanto para clases como para objetos, y la validación de las listas con las características necesarias.

classandobjmgmnt.pl

Contiene los servicios para el módulo de Modificar, todas las funciones para cambiar por ejemplo el nombre de una clase.

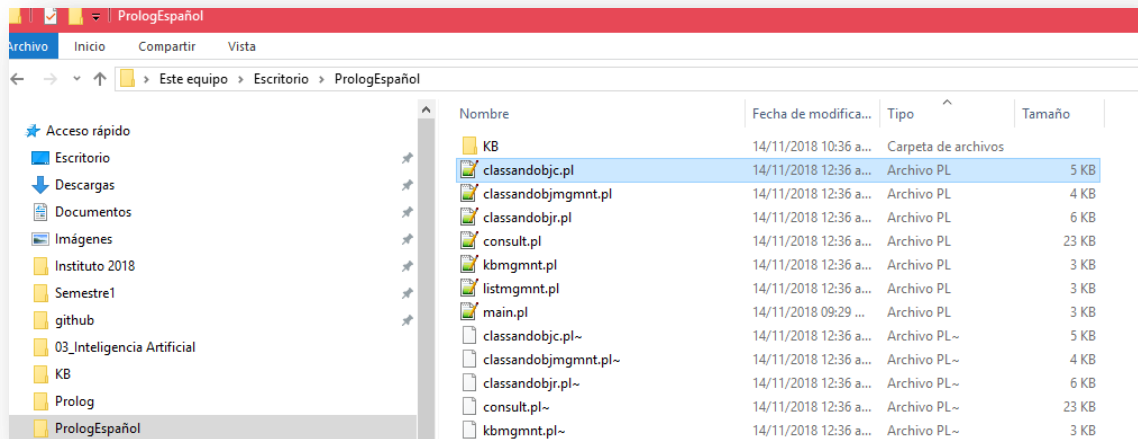
Classandobjc.pl

Contiene los servicios para el módulo de Agregar, todas las funciones para agregar clases y objetos, así como para agregar propiedades y relaciones tanto de clases como de objetos.

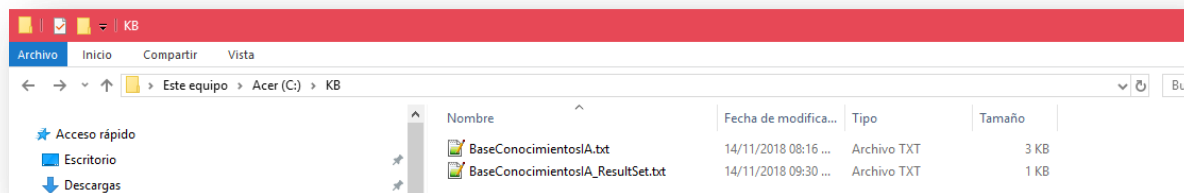
Ejecución del Sistema

La forma más correcta en el diseño del sistema para el desarrollo de los diversos módulos, era separar en varios archivos el código y así de forma más ordenada llevar a cabo los trabajos de mantenimiento en la programación, ya que si se efectuaban cambios, estos sólo afectarían a los del módulo.

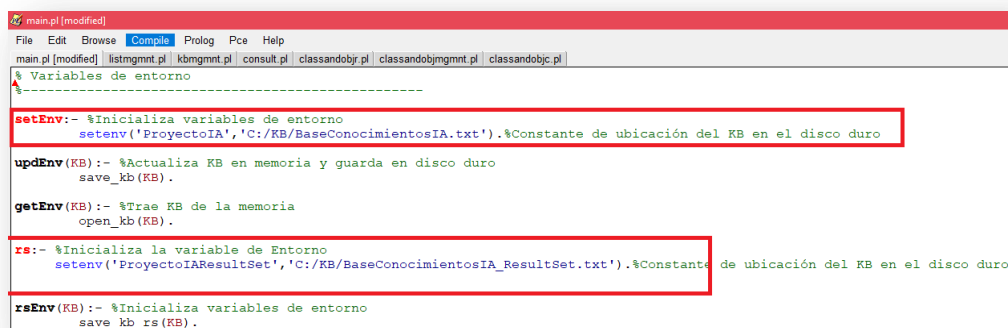
Después para que se ejecuten las consultas y las transacciones a la base de conocimiento, es necesario y de forma obligatoria estar dentro de una misma carpeta cuidando que el nombre de las funciones fuera único en toda la carpeta del proyecto, se logró crear los 7 archivos que se deben de compilar y abrir en Prolog.



La carpeta de la base de conocimiento deberá estar en la raíz de del Disco C, preferentemente para que no se cambie la variable global. En caso de que no se pueda realizar esto se deberá de modificar dentro del archivo main.pl la ruta de ubicación de la base de conocimiento y de igual forma el nombre si se cambia, así como el archivo de Result Set o de resultados de consulta.

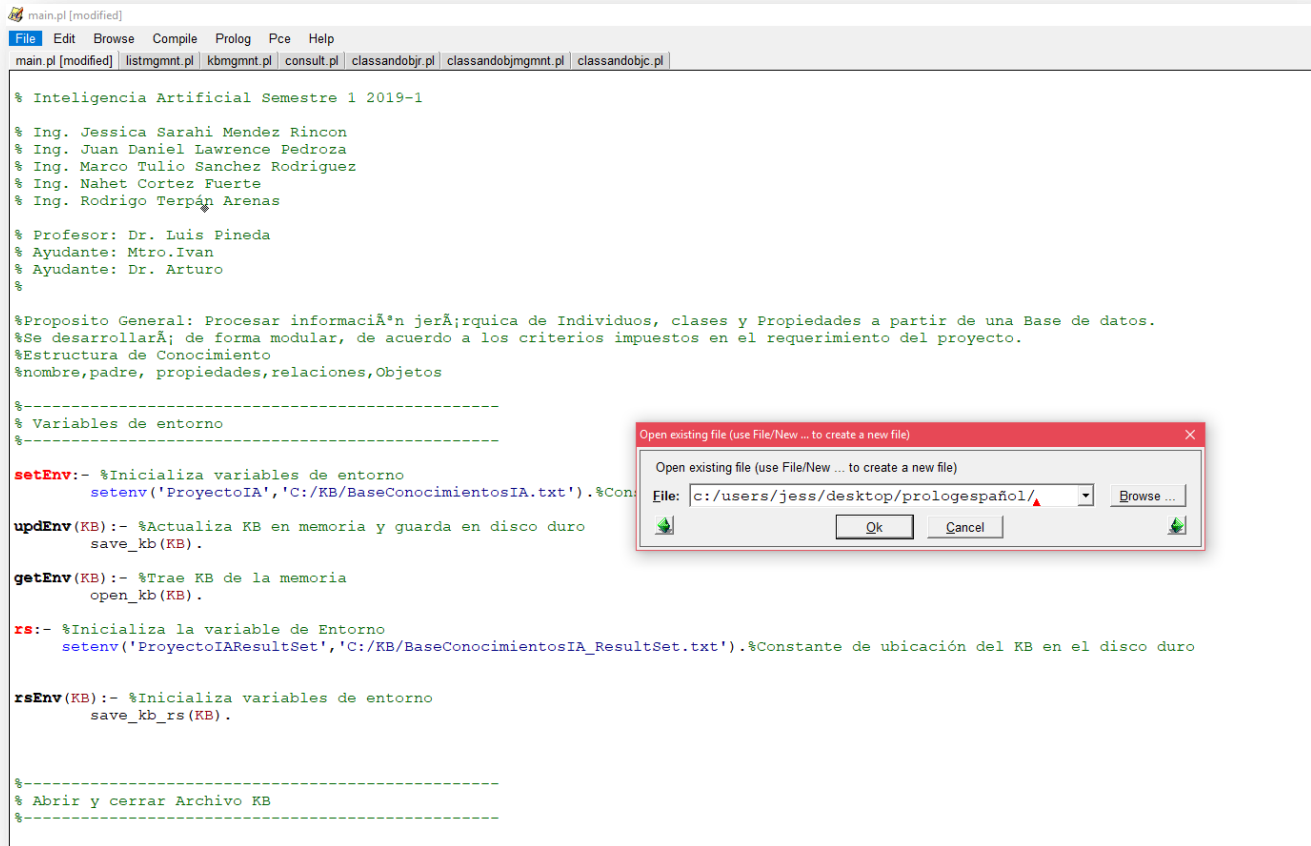


En caso de que se desee cambiar la ruta de la base de conocimientos, se necesita ir al archivo de main.pl, y modificar la ruta del archivo, en las líneas que se indican en la siguiente imagen.



Se deberá de abrir cada uno de los archivos de la carpeta en una misma sesión de Prolog.

De tal forma que se puedan contemplar todos los archivos.



Una vez que se abren se compilan, para ello puede ser desde el Editor seleccionando la opción de:

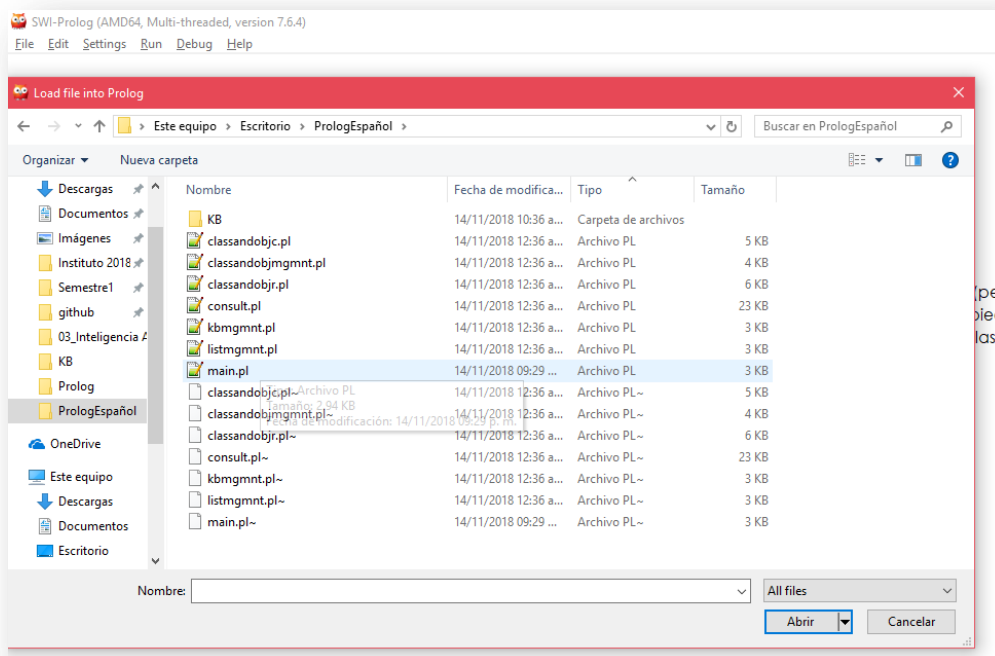
Compile>Compile Buffer

O también en la pantalla de Ejecución de Prolog seleccionando la opción de:

File>Consult...

Y buscar el archivo, o tambien una vez que se abra el archivo se puede aplicar la función

File>Reload modified files



Módulos del Sistema

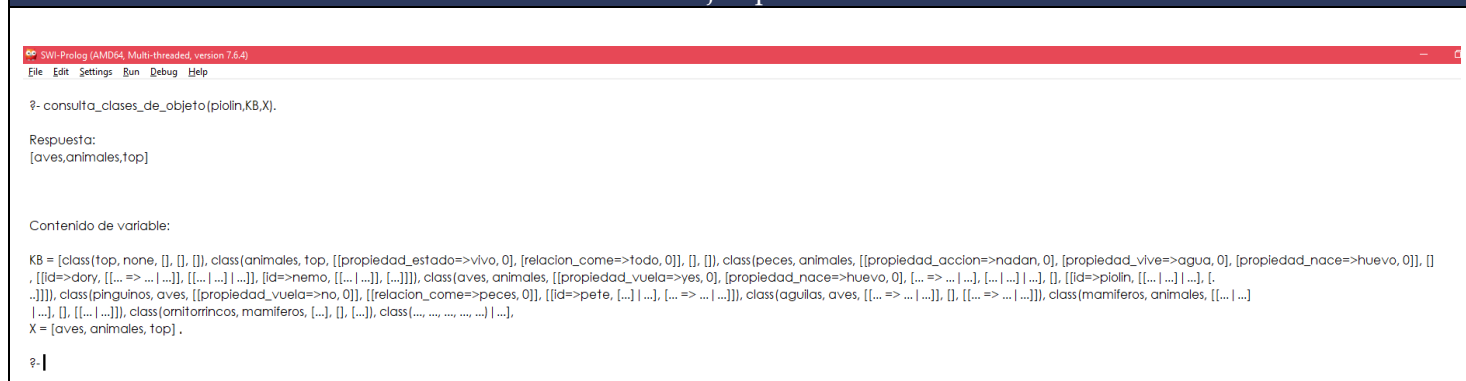
A continuación se dará una descripción de las funciones de los módulos así como un ejemplo de ejecución.

Módulo de Consulta

Consulta de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

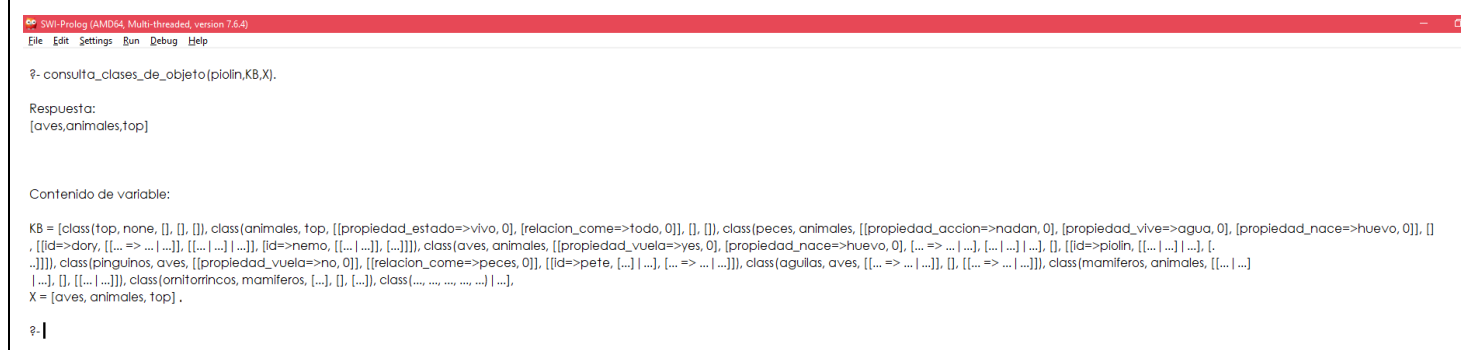


Consulta de Propiedades de Clases

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	---

Ejemplo

 <pre>?- consulta_clases_de_objeto(piolin,KB,X). Respuesta: [aves,animales,top] Contenido de variable: KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>... ...], [...], [id=>nemo, [... ...], [... ...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... ...], [...], [id=>piolin, [...], [... ...], [... ...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [[id=>pete, [... ...], [...=>... ...]], class(agallas, aves, [...=>... ...]), [...=>... ...]), class(mamiferos, animales, [... ...]), class(ornitornicos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...)], X = [aves, animales, top]. ?- </pre>
--

Consulta de Preferencias de Propiedades de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...|...]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]), [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].

?-|
```

Consulta de Relaciones de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]


Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...|...]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]), [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].

?-|
```

Consulta de Preferencias de Relaciones de Clases

Función	Descripción
---------	-------------

<pre> %Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_, 'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	
	

Consulta de Objetos

Función	Descripción
<pre> %Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_, 'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```

SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [[id=>pete, [...|...], [...=>...|...]], class(agullas, aves, [...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [...|...|...|...]), class(omitorincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...)], X = [aves, animales, top].

```

Consulta de Propiedades de Objetos

Función	Descripción
<pre> %Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```

SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [[id=>pete, [...|...], [...=>...|...]], class(agullas, aves, [...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [...|...|...|...]), class(omitorincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...)], X = [aves, animales, top].

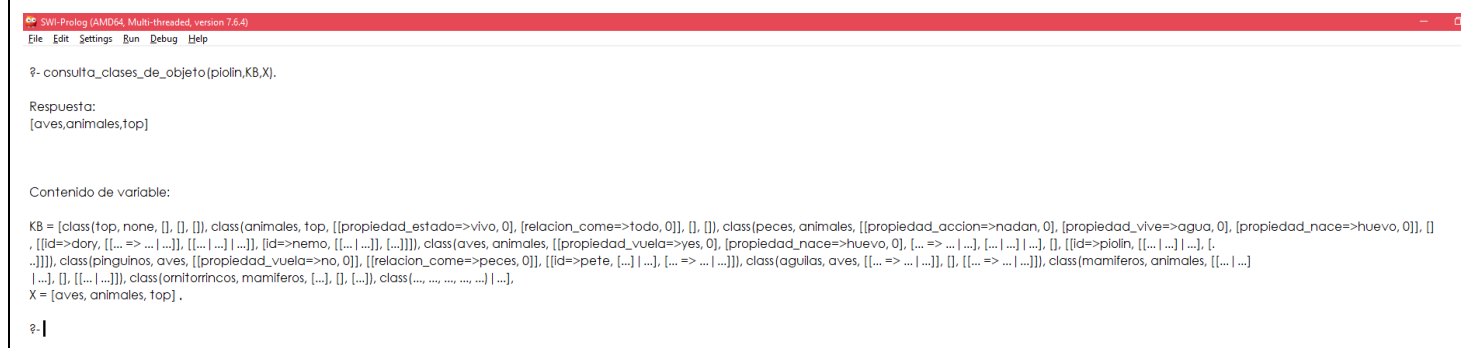
```

Consulta de Preferencias de Propiedades de Objetos

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
---	--

Ejemplo

 <pre>?- consulta_clases_de_objeto(piolin,KB,X). Respuesta: [aves,animales,top] Contenido de variable: KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... ...], [[...]], [id=>nemo, [[... ...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... ...], [...], [], [id=>piolin, [[...], [... ...], [... ...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [... ...], [...=>... ...]], class(agallas, aves, [[...=>... ...]], [], [...=>... ...]), class(mamiferos, animales, [[... ...], [... ...], [... ...]), class(ornitorrinos, mamiferos, [...], [], [...]), class(..., ..., ..., ...,)], X = [aves, animales, top]. ?- </pre>
--

Consulta de Relaciones de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...|...]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...|...|...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]), [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...)],
X = [aves, animales, top].
```

Consulta de Preferencias de Relaciones de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...|...]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...|...|...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]), [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...)],
X = [aves, animales, top].
```

Módulo de Añadir

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Ejemplo

--

Añadir de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...], [[...|...|...]], [id=>nemo, [[...|...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...], [], [id=>piolin, [[...|...|...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [[...|...|...], [], [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].

?- |
```

Añadir de Propiedades de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...], [[...|...|...]], [id=>nemo, [[...|...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...], [], [id=>piolin, [[...|...|...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [[...|...|...], [], [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].

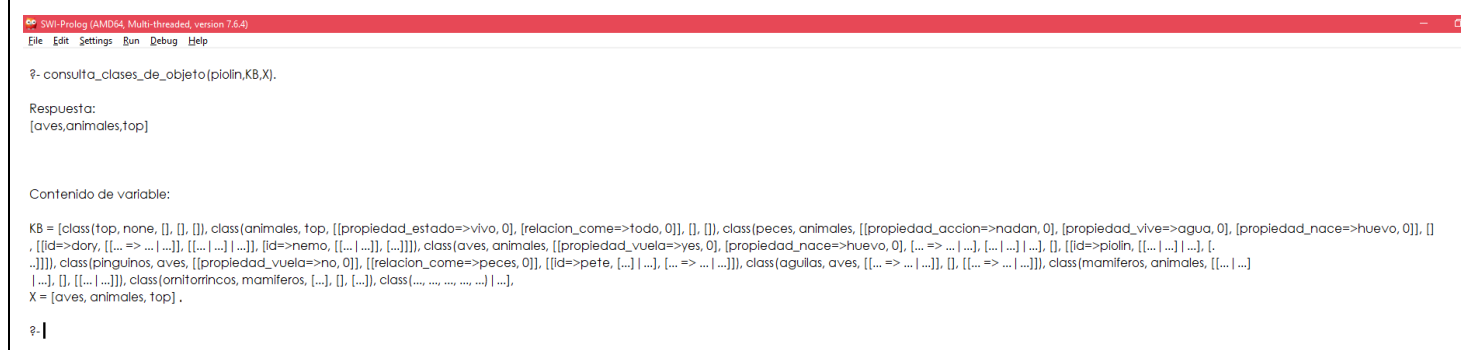
?- |
```

Añadir de Preferencias de Propiedades de Clases

Función	Descripción
---------	-------------

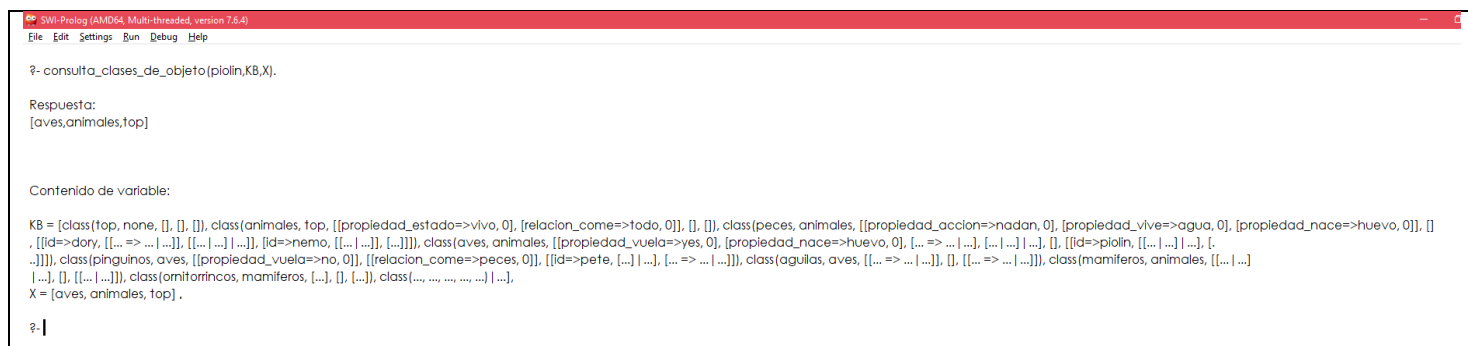
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Ejemplo


--

Añadir de Relaciones de Clases

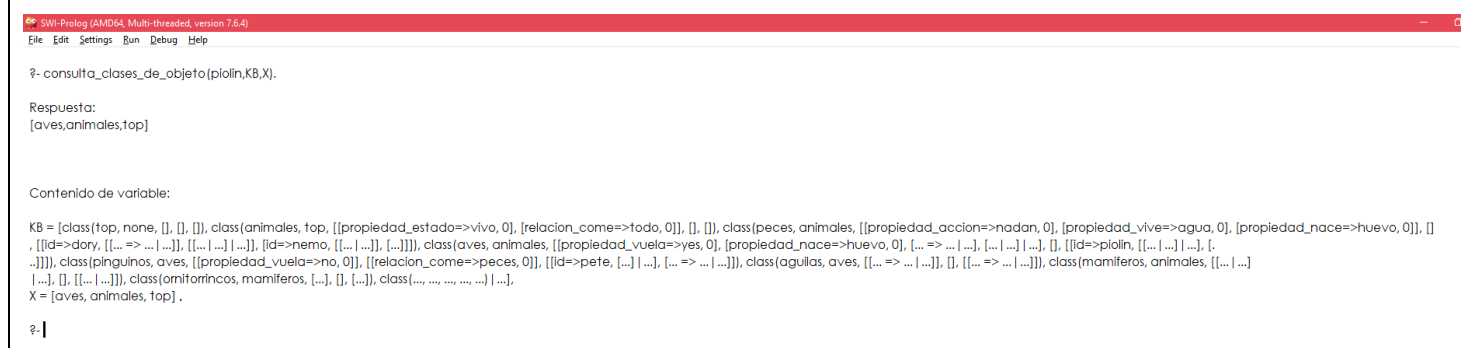
Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	



Añadir de Preferencias de Relaciones de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). .</pre> <p>clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases).</p> <p>clasesDeObjeto(, 'no lo se').</p>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

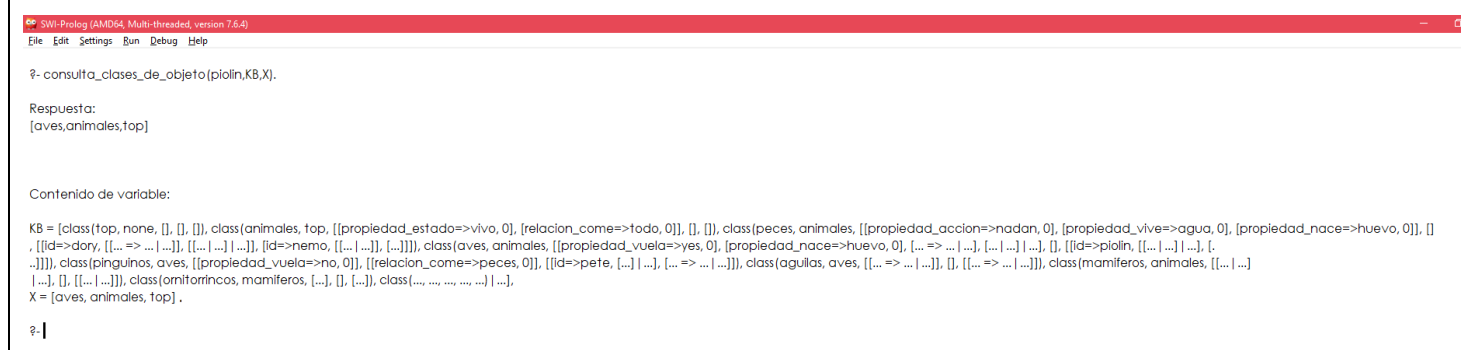


Añadir de Objetos

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Ejemplo


--

Añadir de Propiedades de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]], class(mamiferos, animales, [[...|...|...], [...|...|...]], class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...),
X = [aves, animales, top].

?-|
```

Añadir de Preferencias de Propiedades de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]], class(mamiferos, animales, [[...|...|...], [...|...|...]], class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...),
X = [aves, animales, top].

?-|
```

Añadir de Relaciones de Objetos

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
---	--

Ejemplo

 <pre>?- consulta_clases_de_objeto(piolin,KB,X). Respuesta: [aves,animales,top] Contenido de variable: KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... ...], [[...]], [id=>nemo, [[... ...], [...]]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... ...], [...], [], [id=>piolin, [[...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [id=>pete, [... ...], [...=>... ...]], class(agallas, aves, [[...=>... ...]], [], [...=>... ...]), class(mamiferos, animales, [[...], [], [... ...]], class(ornitorrinos, mamiferos, [...], [], [...]), class(..., ..., ..., ...,)], X = [aves, animales, top]. ?- </pre>
--

Añadir de Preferencias de Relaciones de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_comer=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [[relacion_comer=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]], class(mamiferos, animales, [[...|...|...]], [], [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].

?- |
```

Módulo de Eliminar

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta: '),nl, write(X),nl,nl,nl,nl, write('Contenido de variable: '),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesor(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesor => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_comer=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [[relacion_comer=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]], class(mamiferos, animales, [[...|...|...]], [], [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].

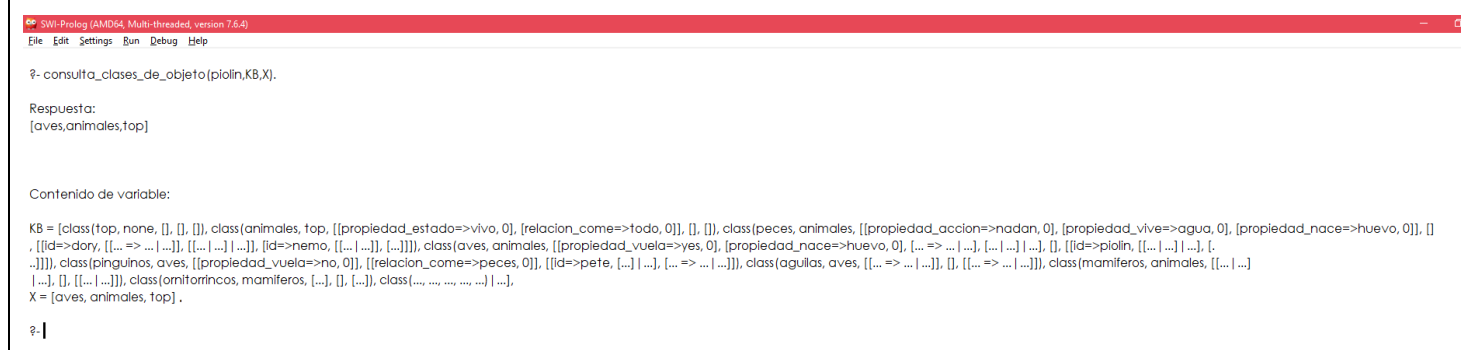
?- |
```

Eliminar de Clases

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Ejemplo

 <pre>?- consulta_clases_de_objeto(piolin,KB,X). Respuesta: [aves,animales,top] Contenido de variable: KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... ...], [[...]], [id=>nemo, [[... ...], [...]]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... ...], [...], [], [id=>piolin, [[...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [[relacion_come=>peces, 0]], [[id=>pete, [... ...], [...=>... ...]], class(agallas, aves, [[...=>... ...]], [], [...=>... ...]], class(mamiferos, animales, [[...], [], [... ...]], class(ornitorrinos, mamiferos, [...], [], [...]), class(..., ..., ..., ...,)], X = [aves, animales, top]. ?- </pre>

Eliminar de Propiedades de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]], class(mamiferos, animales, [[...|...|...], [...|...|...]], class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...),
X = [aves, animales, top].

?-|
```

Eliminar de Preferencias de Propiedades de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [...=>...|...]], class(mamiferos, animales, [[...|...|...], [...|...|...]], class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...),
X = [aves, animales, top].

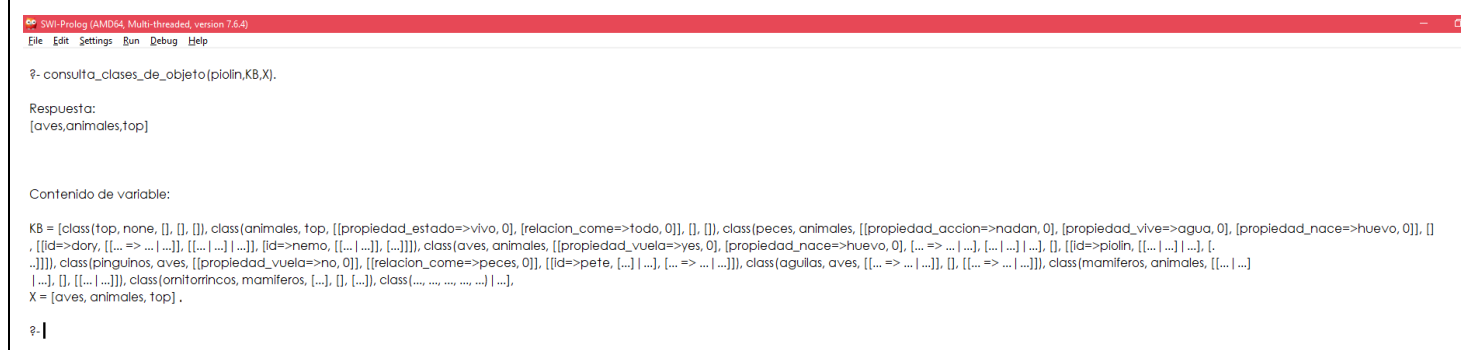
?-|
```

Eliminar de Relaciones de Clases

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
---	--

Ejemplo

 <pre>?- consulta_clases_de_objeto(piolin,KB,X). Respuesta: [aves,animales,top] Contenido de variable: KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... ...], [[...]], [id=>nemo, [[... ...], [...]]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... ...], [...], [], [id=>piolin, [[...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0], [relacion_come=>peces, 0]], [[id=>pete, [... ...], [...=>... ...]], class(agallas, aves, [[...=>... ...]], [], [...=>... ...]], class(mamiferos, animales, [[...], [], [... ...]], class(ornitorrinos, mamiferos, [...], [], [...]), class(..., ..., ..., ...,)], X = [aves, animales, top]. ?- </pre>

Eliminar de Preferencias de Relaciones de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...|...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [[relacion_come=>peces, 0]], [[id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [[...=>...|...]], class(mamiferos, animales, [[...|...|...]], [], [[...|...|...]], class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...),
X = [aves, animales, top].

?- |
```

Eliminar de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta: '),nl, write(X),nl,nl,nl,nl, write('Contenido de variable: '),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesor(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesor => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...], [...|...|...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [[relacion_come=>peces, 0]], [[id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [[...=>...|...]], [], [[...=>...|...]], class(mamiferos, animales, [[...|...|...]], [], [[...|...|...]], class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...),
X = [aves, animales, top].

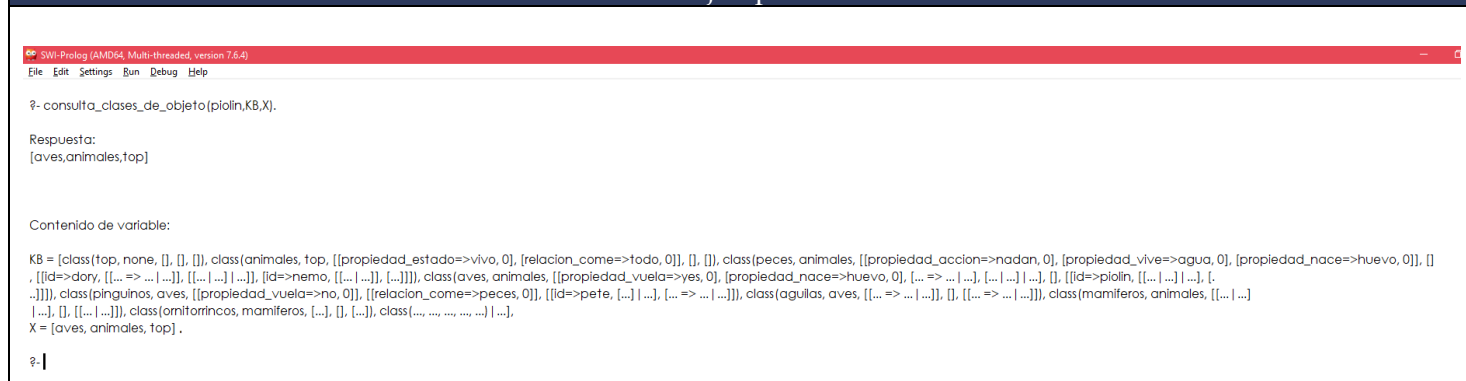
?- |
```

Eliminar de Propiedades de Objetos

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Ejemplo


--

Eliminar de Preferencias de Propiedades de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...|...]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]), [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].
```

Eliminar de Relaciones de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).


Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...|...]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...|...], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]], [id=>pete, [...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]), [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...|...), X = [aves, animales, top].
```

Eliminar de Preferencias de Relaciones de Objetos

Función	Descripción
---------	-------------

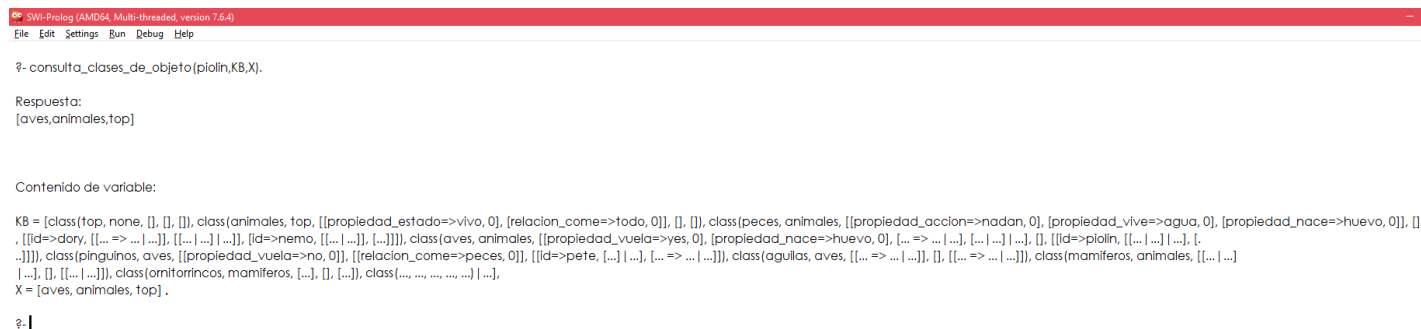
<pre> %Para realizar Consultas %%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_, 'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	
	

Módulo de Modificar

Función	Descripción
<pre> %Para realizar Consultas %%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB, yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

clasesDeObjeto(_,'no lo se').

Ejemplo



```

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

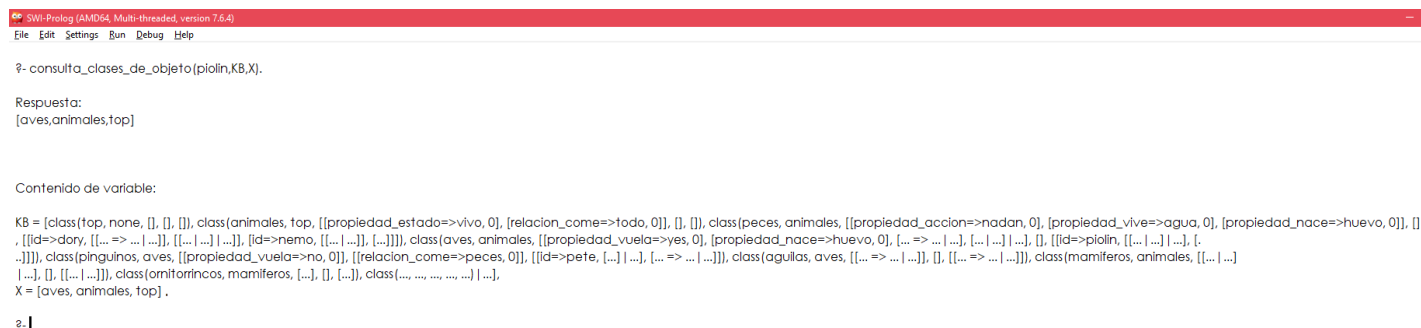
KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... | ...], [[... | ...] | ...]], [id=>nemo, [[... | ...], [...]]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... | ...], [... | ...] | ...], [], [id=>piolin, [[... | ...] | ...], [...]]], class(pinguinos, aves, [[propiedad_vuela=>no, 0], [relacion_come=>peces, 0]], [[id=>pete, [... | ...], [...=>... | ...]], class(aguilas, aves, [[...=>... | ...]], [], [...=>... | ...]], class(mamiferos, animales, [[... | ...] | ...], [], [... | ...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ... | ...),
X = [aves, animales, top].

```

Modificar de Clases

Función	Descripción
<pre> %Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesor(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesor => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo



```

?- consulta_clases_de_objeto(piolin,KB,X).

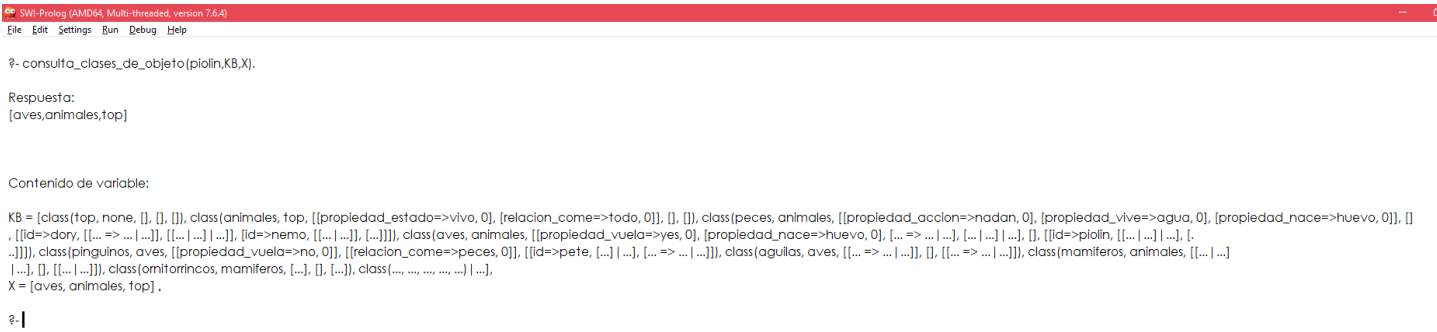
Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... | ...], [[... | ...] | ...]], [id=>nemo, [[... | ...], [...]]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... | ...], [... | ...] | ...], [], [id=>piolin, [[... | ...] | ...], [...]]], class(pinguinos, aves, [[propiedad_vuela=>no, 0], [relacion_come=>peces, 0]], [[id=>pete, [... | ...], [...=>... | ...]], class(aguilas, aves, [[...=>... | ...]], [], [...=>... | ...]], class(mamiferos, animales, [[... | ...] | ...], [], [... | ...]), class(ornitorrincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ... | ...),
X = [aves, animales, top].

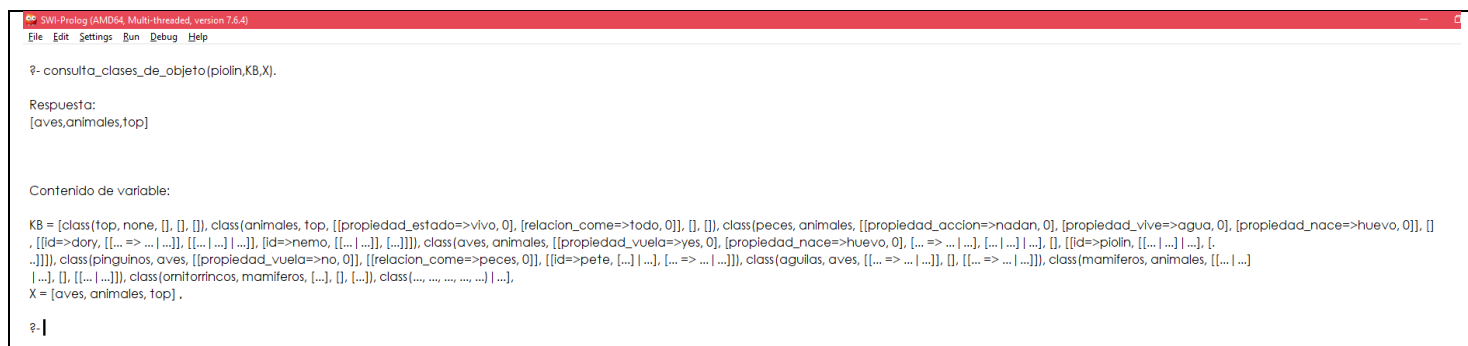
```

Modificar de Propiedades de Clases

Función	Descripción
<pre> %Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	
	

Modificar de Preferencias de Propiedades de Clases

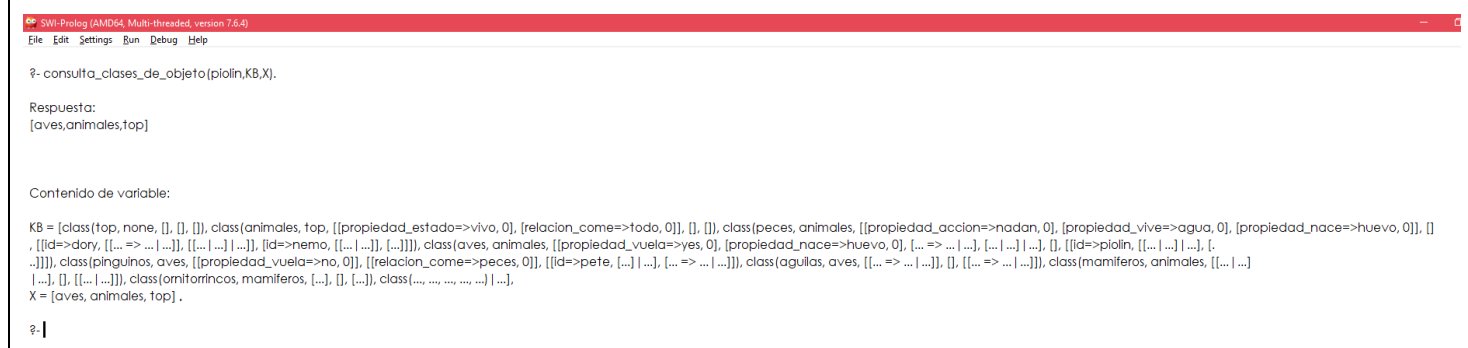
Función	Descripción
<pre> %Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se'). </pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	



Modificar de Relaciones de Clases

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). .</pre> <p>clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases).</p> <p>clasesDeObjeto(, 'no lo se').</p>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo



Modificar de Preferencias de Relaciones de Clases

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X).</pre> <p>clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesoros(X,KB,Y), append([X],Y,ListaClases).</p> <p>clasesDeObjeto(____,'no lo se').</p>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia => verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesoros => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Modificar de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [[id=>pete, [...|...], [...=>...|...]], class(agullas, aves, [...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(omitorincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...)],
X = [aves, animales, top].

?-|
```

Modificar de Propiedades de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesor(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesor => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>

Ejemplo

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [...=>...|...], [...|...|...]], [id=>nemo, [...|...], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [...|...|...], [...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [relacion_come=>peces, 0]), [[id=>pete, [...|...], [...=>...|...]], class(agullas, aves, [...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(omitorincos, mamiferos, [...], [], [...]), class(..., ..., ..., ..., ...)],
X = [aves, animales, top].

?-|
```

Modificar de Preferencias de Propiedades de Objetos

Función	Descripción
---------	-------------

<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
--	--

Ejemplo

 <pre>?- consulta_clases_de_objeto(piolin,KB,X). Respuesta: [aves,animales,top] Contenido de variable: KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_come=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>... ...], [[...]], [id=>nemo, [[... ...], [...]]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>... ...], [...], [], [id=>piolin, [[...], [...]]]], class(pinguinos, aves, [[propiedad_vuela=>no, 0], [relacion_come=>peces, 0]], [[id=>pete, [... ...], [...=>... ...]], class(agallas, aves, [[...=>... ...]], [], [...=>... ...]], class(mamiferos, animales, [[...], [...], [], [... ...]]), class(ornitorrinos, mamiferos, [...], [], [...]), class(..., ..., ..., ...,)], X = [aves, animales, top]. ?- </pre>
--

Modificar de Relaciones de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta:'),nl, write(X),nl,nl,nl,nl, write('Contenido de variable:'),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecedentes(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecedentes => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	

```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:

KB = [class(top, none, [], [], []), class(animales, top, [[propiedad_estado=>vivo, 0], [relacion_comer=>todo, 0]], [], []), class(peces, animales, [[propiedad_accion=>nadan, 0], [propiedad_vive=>agua, 0], [propiedad_nace=>huevo, 0]], [], [[id=>dory, [[...=>...|...]], [[...|...|...]], [id=>nemo, [[...|...]], [...]]], class(aves, animales, [[propiedad_vuela=>yes, 0], [propiedad_nace=>huevo, 0], [...=>...|...], [...|...|...]], [], [[id=>piolin, [[...|...|...]], [...|...|...]], class(pinguinos, aves, [[propiedad_vuela=>no, 0]], [[relacion_comer=>peces, 0]], [id=>pete, [...|...|...], [...=>...|...]], class(aguilas, aves, [...=>...|...]], [], [...=>...|...]), class(mamiferos, animales, [...|...|...]), class(ornitorrincos, mamiferos, [...|...]), class(..., ..., ..., ..., ...|...)],
X = [aves, animales, top].
```

Modificar de Preferencias de Relaciones de Objetos

Función	Descripción
<pre>%Para realizar Consultas %%%CLASES consulta_clases_de_objeto(Objeto,KB,X):- getEnv(KB), clasesDeObjeto(Objeto,KB,X),nl, write('Respuesta: '),nl, write(X),nl,nl,nl,nl, write('Contenido de variable: '),nl,nl, rsEnv(X). clasesDeObjeto(Objeto,KB,ListaClases):- verificarObjeto(Objeto,KB,yes), obtenerHerencia(Objeto,KB,X), obtenerAntecesor(X,KB,Y), append([X],Y,ListaClases). clasesDeObjeto(_,'no lo se').</pre>	<p>Función que muestra todas las clases a las que pertenece un objeto.</p> <p>VerificarObjeto => verifica si existe el objeto en la base de conocimiento</p> <p>obtenerHerencia=> verifica el nombre de la clase del objeto, el primer relacionado al objeto es decir, la clase padre</p> <p>obtenerAntecesor => extrae las clases padre del objeto</p> <p>Append => agrega los nombres a una lista para el result set.</p>
Ejemplo	


```
SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

?- consulta_clases_de_objeto(piolin,KB,X).

Respuesta:
[aves,animales,top]

Contenido de variable:
```

```
KB = [class(top,none,[],[]),class(animales,top,[{propiiedad_estado=>vivo,0},{relacion_come=>toda,0}],[],[]),class(peces,animales,[{propiedad_accion=>nadan,0},{propiedad_vive=>agua,0},{propiedad_nace=>huevo,0}],[]),[{id=>dory,[...=>...|...]],[[...|...]|...]],[{id=>nemo,[...|...]|...}]]],class(aves,animales,[{propiedad_vuela=>yes,0},{propriedad_nace=>huevo,0},[...=>...|...]),[[],[{id=>piolin,[...|...]|...}|,...|...]],class(pinguinos,aves,[{propiedad_vuela=>no,0},{relacion_come=>peces,0}],[{id=>pete,[...|...]|...},[...=>...|...]}]),class(aguilas,aves,[...=>...|...]),[[],[{...=>...|...}]],class(mamiferos,animales,[[...|...]|...|...|...]),class(ornitorrincos,mamiferos,[...|...]|...]),class(...,...,<...>|...)|...],X=[aves,animales,top].
```