

TAREA PROGRAMADA 2



Jurgen Arley Elizondo, Luis Eladio Castro Montenegro, Juan Ignacio Aguilera Mora

TEC Administración de Tecnología de Información

Tabla de contenido

Resumen ejecutivo	1
¿Qué es lo que se va a hacer?	1
¿Quiénes lo van a hacer?.....	1
¿Cómo lo van a hacer?	2
¿Por qué se va a hacer?	2
Propósitos	3
Descripción del programa:	3
Modo definición:	3
Modo consulta:	3
Funcionalidades.....	5
1. Definición.....	5
2. Consulta	6
Funciones:	8
Bibliografía.....	11

Resumen ejecutivo

El proyecto consiste en el desarrollo de una aplicación que permite al usuario utilizar un intérprete para un lenguaje de programación lógico similar a Prolog, el cual es *“lenguaje de programación ideado a principios de los años 70 en la Universidad de Aix-Marseille I (Marsella, Francia) por los estudiantes Alain Colmerauer y Philippe Roussel. Nació de un proyecto que no tenía como objetivo la traducción de un lenguaje de programación, sino la clasificación algorítmica de lenguajes naturales. Alain Colmerauer y Robert Pasero trabajaban en la parte del procesamiento del lenguaje natural y Jean Trudel y Philippe Roussel en la parte de deducción e inferencia del sistema. Interesado por el método de resolución SL, Trudel persuadió a Robert Kowalski para que se uniera al proyecto, dando lugar a una versión preliminar del lenguaje Prolog a finales de 1971 y apareciendo la versión definitiva en 1972.3 Esta primera versión de Prolog fue programada en ALGOL W.”* (Wikipedia, s.f.)

El usuario puede abrir el programa hecho en Python, esta aplicación podrá ser accedida desde Python 3.0 en adelante de los sistemas operativos Windows y Ubuntu, y este mostrará una consola similar a la del intérprete de Prolog, mostrándosele al iniciar la siguiente sentencia.

```
Welcome to TEC-Prolog (Multi-threaded, 64 bits, Version 7.1.23)
Copyright (c) 2014 Tecnológico de Costa Rica, Cartago Costa Rica
TEC-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Seleccione el modo en el que desea trabajar: |
```

El usuario tienen la opción de ingresar instrucciones, pero estas son procesadas por el compilador, y algoritmos de búsquedas implementados por los estudiantes del proyecto. Una vez realizado este paso, se mostrará los resultados en pantalla. Por lo tanto el usuario inserta la base de conocimientos para obtener luego los resultados.

¿Qué es lo que se va a hacer?

Se desarrollará un programa que permite consultas y almacenamiento temas que el usuario desee por lo tanto se puede utilizar este programa como una base de datos, ya que permite realizar consultas y almacenamiento.

¿Quiénes lo van a hacer?

- Jurgen Arley Elizondo
- Ignacio Aguilera Mora
- Luis Eladio Castro Montenegro

Estudiantes del curso:

- Lenguajes de Programación, de la carrera de Administración de Tecnología de la Información

Universidad:

- Instituto Tecnológico de Costa Rica

¿Cómo lo van a hacer?

Con el lenguaje de programación Python, se desarrollará el sistema de consultas y almacenamiento. Los algoritmos serán realizados en un 100% por los estudiantes del curso.

¿Por qué se va a hacer?

El programa se realizará para facilitar la búsqueda en una base de conocimientos realizado por el usuario.

Propósitos

En esta sección se presentará la descripción del programa, y los requerimientos de diseño que presenta la misma.

Descripción del programa:

La aplicación está diseñada para que los usuarios realicen diversas consultas sobre los temas que quieran debido a que el usuario tiene libre uso este programa, así que en el modo definición se puede insertar la base de conocimientos que desee y en el modo consultar, consultar lo que desee, y este le devolverá la respuesta de YES o NO.

El programa no almacenará la base de conocimientos debido a que esto fue pedido por el profesor del curso, para poder almacenarlo este no debe ejecutado una vez ya iniciado el programa.

Cuando el usuario arranca el programa, este le pedirá si el usuario quiere entrar a modo consulta y a modo definición.

Modo definición:

En este modo el usuario insertará la base de conocimientos que desee el usuario por ejemplo, si el usuario quiere insertar una base de conocimientos de animales. El usuario deberá insertar por ejemplo:

- ? Perro(ruffo).
- ? Perro(sammy).
- ? Perro(bruno).
- ? Perro(terry).
- ? Gato(vegetta).
- ? Gato(tomy).
- ? Vivenjuntos(terry, tommy).

Entonces para el intérprete conocerá que los perros serán Ruffo, Sammy, Bruno y Terry. Y los gatos serán Vegetta y Tommy.

Modo consulta:

En este modo el usuario insertará las consultas que desee realizar, ya que el ejemplo anterior fue de una base de conocimientos de animales las consultas también serán de animales.

- ? Vivenjuntos(X, tommy).
X=terry.
- ? Vivenjuntos (terry, tommy).
Yes.
- ? Perro (juan).
No.
- ? Perro (terry).
Yes.
- ? Perro (X).
X=terry.

De esta forma el usuario podrá saber las consultas que le dará el intérprete siendo Sí o No, cuando en el modo consulta se inserta(X), devuelve el primer resultado que encuentra en la base de conocimientos.

Requerimientos del sistema:

Para poder ejecutar el programa se necesita tener:

Windows:

- Un sistema operativo Windows .
- 512 MB de Memoria Ram.
- Instalar la versión 3.2 de Python.

Ubuntu:

- Ubuntu 12 en adelante
- 512 MB de Memoria Ram.
- Instalar la versión 3.2 de Python.

Funcionalidades

La aplicación presenta dos funcionalidades:

1. Definición

La aplicación permitirá a la base de conocimientos los elementos que desee, tiene una sintaxis que a continuación se mostrará:

- a. perro: la palabra que identificará debe estar en minúscula y no puede contener caracteres especiales.
- b. (). : Después debe tener los paréntesis que adentro contendrá información que describe.
- c. ruffo: Lo que va adentro del paréntesis eso va en minúscula.

Ejemplo:

perro(ruffo).

perro(ruffo,ana).

La constante en este paréntesis es ruffo y para esta funcionalidad las constantes deben ser solo letras y números, comienzan por minúscula, pueden contener “_” (subrayado), si van entre comillas simples (,).

Constantes válidas:

- a
- vacio
- “jorge-perez”
- -->
- jorge_perez
- algo1234

Constantes no válidas:

- 304algo
- jorge-perez
- Vacio
- _alfa

2. Consulta

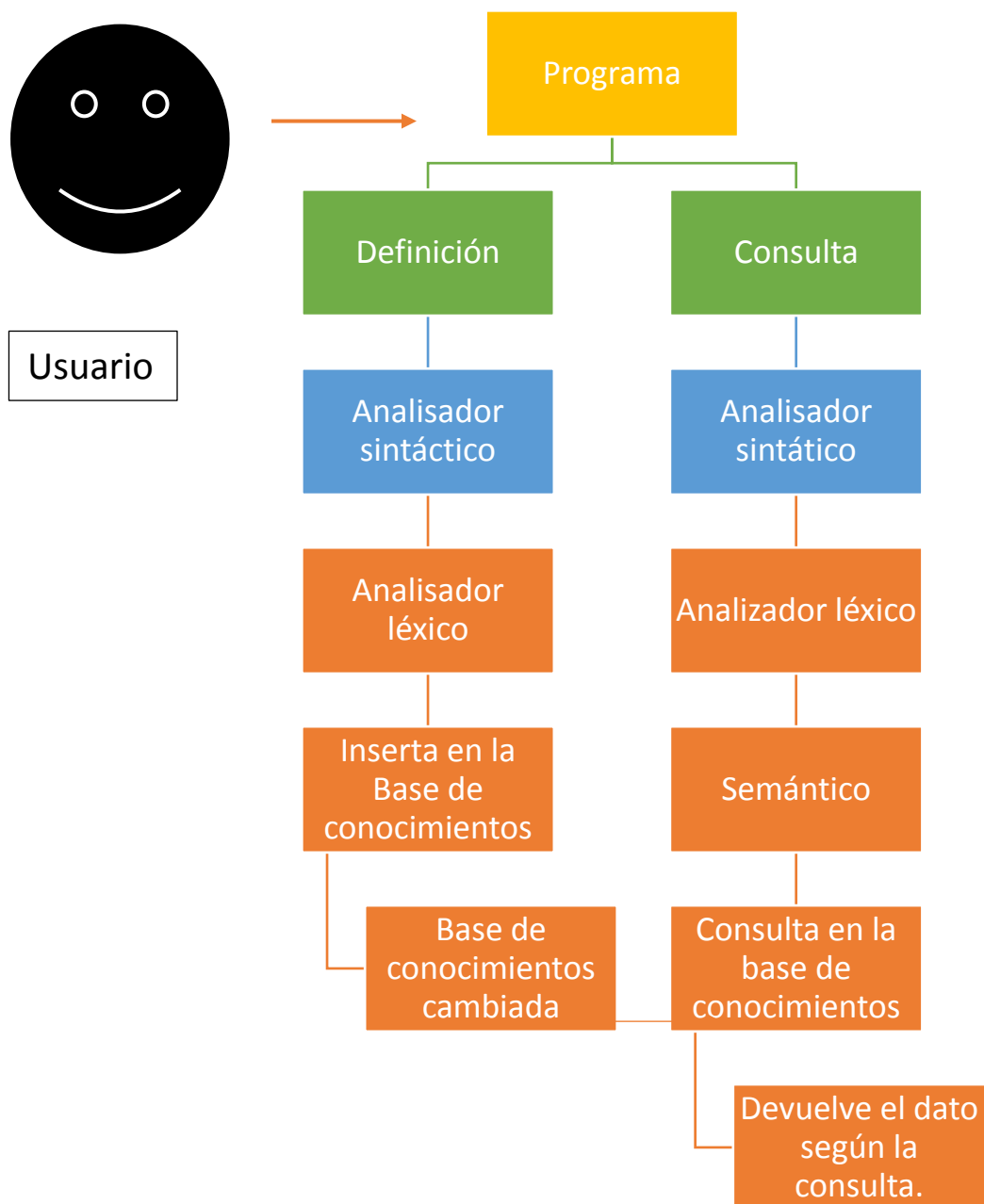
El programa permitirá hacer consultas por el usuario, estas consultas son verificadas en la base de conocimientos, cuando se inserta la consulta, esta es verificada por un parser hecho por los estudiantes de este proyecto:

- a. perro: la palabra que identificará debe estar en minúscula y no puede contener caracteres especiales.
- b. (). : Después debe tener los paréntesis que adentró contendrá información que describe.
- c. ruffo: la variable puede ser en mayúscula o en minúscula siendo lo que va a comparar ejemplo una constante.

Variables

- Representan objetos no específicos (a los que aún no sabemos cómo nombrar).
- Comienzan por Mayúscula ó “_” (subrayado)
 - X
 - Y
 - Z
 - Padre
 - Madre
 - Algo
 - _alfa
- “_” es una variable (anónima)
 - No queda ligada (no necesitaremos su valor)
 - ?- le_gusta_a (_, juan), le_gusta_a(_, luis)
 - Comparar con
 - ?- le_gusta_a (X, juan), le_gusta_a(X, luis)

Diagrama de representación del programa



Descripción detallada

El código del programa realiza, un analizador, sintáctico y léxico, por lo tanto hay varias funciones que permiten el análisis de este. Identificará debe estar en minúscula y no puede contener caracteres especiales.

Funciones:

- **Menu():** Esta función es donde empieza el programa, imprime un mensaje de la historia de Prolog y donde fue hecho, después le pide al usuario en qué modo desea trabajar si en el modo definición o el modo consulta, en caso de que el usuario no lo escriba correctamente, esta función lo validará. Cuando el usuario entre a modo definición, este llama la función `define()`, cuando entra al modo consulta esta función llama a la función `consult()`:
- **Consult():** Esta función hace un ciclo infinito por lo que siempre imprime el `-?`, después llama a la función `predicado(predicado)`, una vez cumpliéndose las funciones anteriores llama a la función `evalua(input)`: que fue insertado por el usuario.
- **Predicado():** Esta función verifica y determina el tipo de predicado(Hecho o regla), genera un ciclo y determina si el predicado es un hecho, si el `if` se cumple, el predicado es una regla o la sintaxis del hecho es invalida, por lo tanto verifica si es regla o un hecho y hace las variables globales que se llamarán hecho o regla. En caso de que sea regla, se llama a una función llamada `sintaxisRegla(predicado)` y si es un hecho llama a la función `sintaxis(predicado)`.
- **sintaxisHecho(predicado):** Esta función realiza el análisis léxico y sintáctico del hecho, después valida la sintaxis del hecho, después realiza la validación de argumentos, después se guarda en una variable y obtiene la aridad del argumento. Después si todo se cumple se llama a la función `ingresarHecho(predicado)` finalizando haciendo una variable global donde ingresa el predicado a una variable global.
- **Ingresahchos(predicado):** Esta función realiza una lista que genera a partir de lo que recibe la función, después abre el `txt` y escribe el predicado y luego cierra el `txt`.
- **Sintaxisregla(predicado):** Esta función hace un análisis léxico y sintáctico de la regla, después quita la `,` `o` `;` que queda y en caso de que esta no cumpla con el analizador sintáctico o léxico imprime "Error en la escritura de la regla".

- **Evalua(predicado):** Función que evalúa que la aridad y el predicado sean iguales para proceder a realizar la consulta y al final devuelve la consulta. Para verificar la base llama a una función llamada **consultaH(predicado)** la cual devuelve variables y después si eso se cumple llama a la función **consultar(predicado)**.
- **consultaH(predicado):** Esta función se encarga de resolver consultas de hechos, por lo tanto recorre toda la base de conocimientos, de esta forma si encuentra la consulta que necesita le devuelve el resultado.
- **consultaR(predicado):** Esta función se encarga de resolver consultas de reglas, para realizar este proceso llama a la función **cambia()**, por lo tanto recorre toda la base de conocimientos, de esta forma si encuentra la consulta que necesita le devuelve el resultado.
- **Cambia():** Función que acomoda la lista de reglas para facilitar las consultas, hace una variable temporal para no modificar la lista de reglas, for para pasar la información de **baseR** a **TempL** con un **Split**. Luego hace un **while** para darle formato a las reglas.
- **cambia_predicado(var):** Función que evalúa que la aridad y el predicado sean iguales para proceder a realizar la consulta
- **write(sentencia):** Esta función es donde si el usuario consulta para que se pueda imprimir y verifica si el analizador sintáctico y léxico está bien escrito para luego imprimir con o sin salto de línea dependiendo de la necesidad del usuario.

Manual de usuario



A continuación se mostrará el manual de usuario para el programa TEC-PROLOG realizado en el lenguaje de Python con un enfoque en el sistema operativo.

Requisito para poder iniciar este manual de usuario:

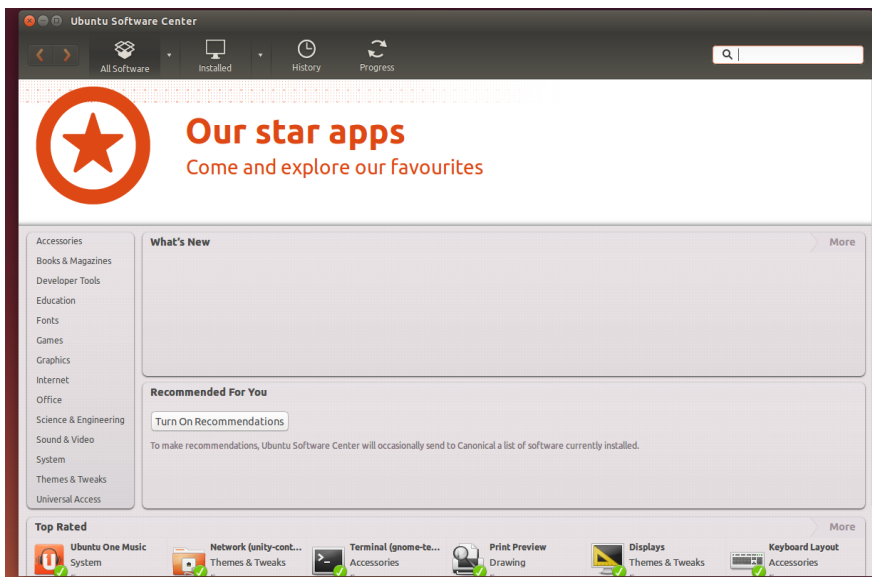
- Tener conexión a internet
- Tener instalado Ubuntu.
- Saber la contraseña del administrador del sistema operativo

Como primer paso, se debe abrir la tienda de Ubuntu 14.04 la cual puede ser encontrada a continuación, se la da clic y debe mostrar una ventana que será observado en el paso 2.

Nota: Ubuntu actualiza sus sistemas operativos constante mente pero con este manual se puede seguir los pasos e igual probablemente los cambios sean mínimos o no sean tan bruscos.

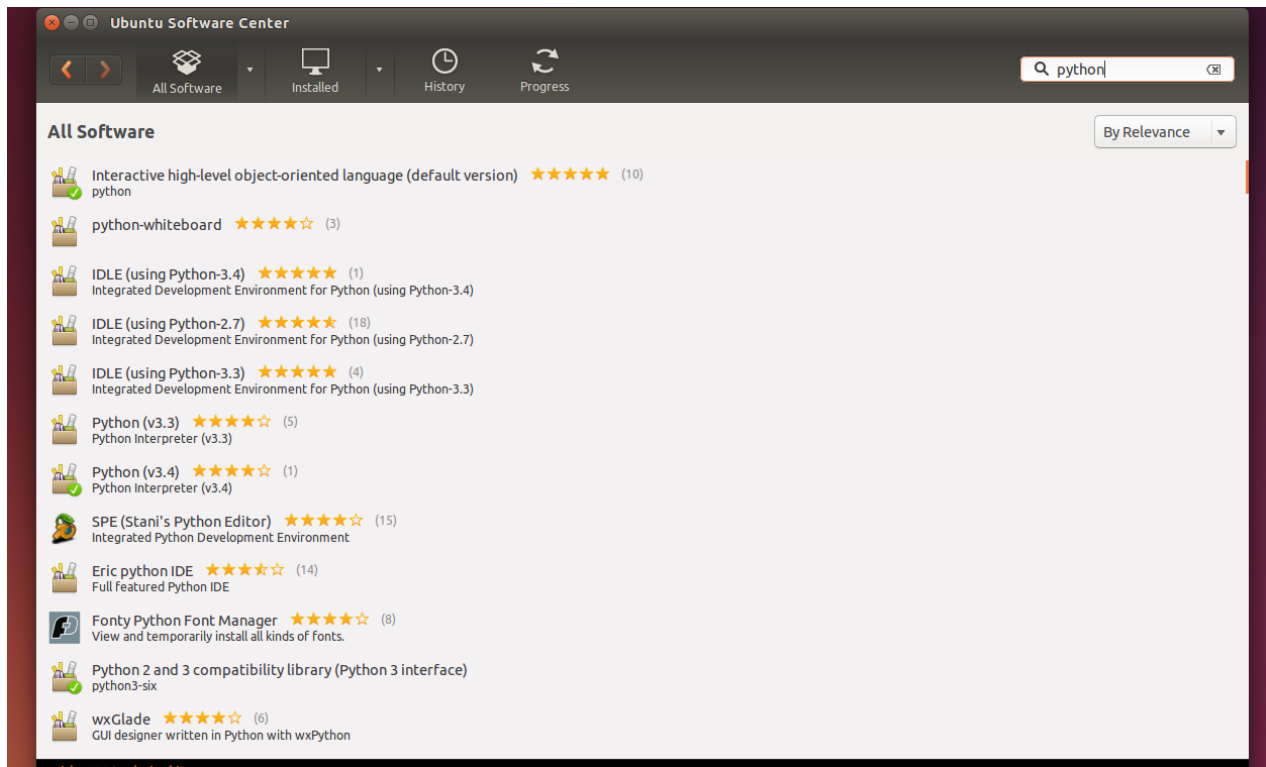
1. . Se debe verificar que a la hora de la instalación debe ingresar en el usuario administrativo, en caso de que esto no pueda suceder la instalación no podrá ser efectuada debido a que ubuntu debe dar permisos para la instalación.

(Imagen 1)



2. Como segundo paso se abrirá la siguiente ventana, se debe verificar el lugar donde se encuentra el textbox, para proceder con el tercer paso de la instalación de Python.

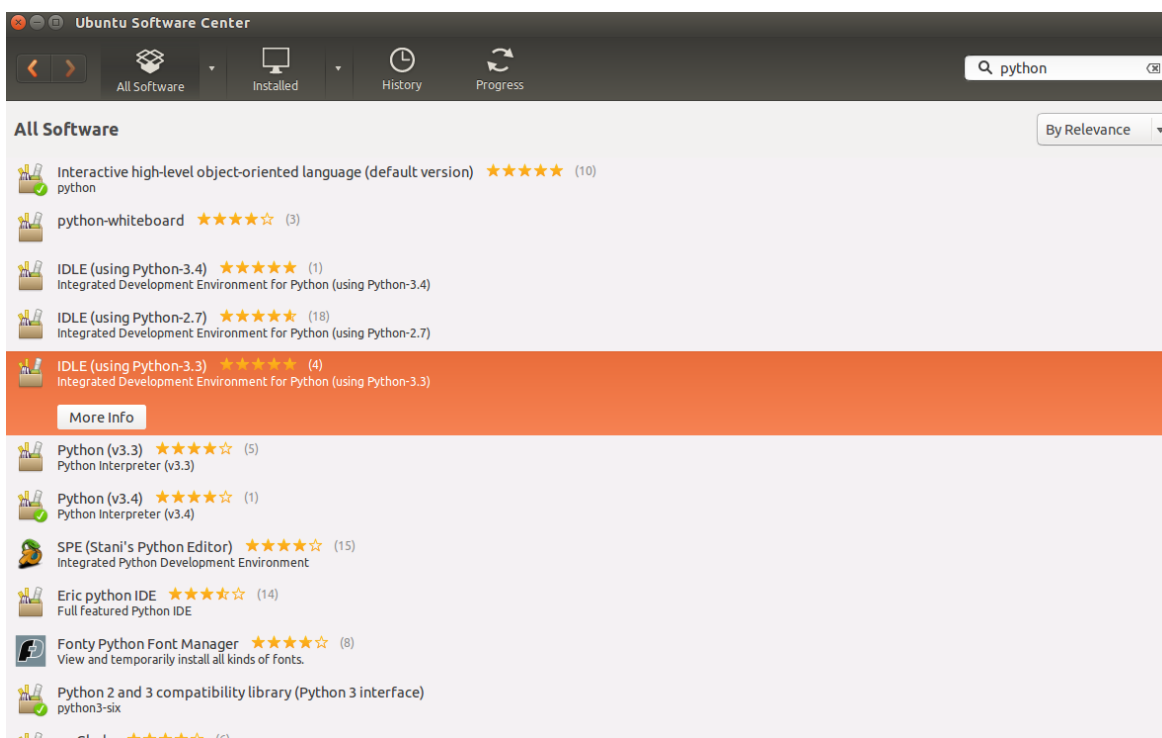
(Imagen2)



(Imagen 3)

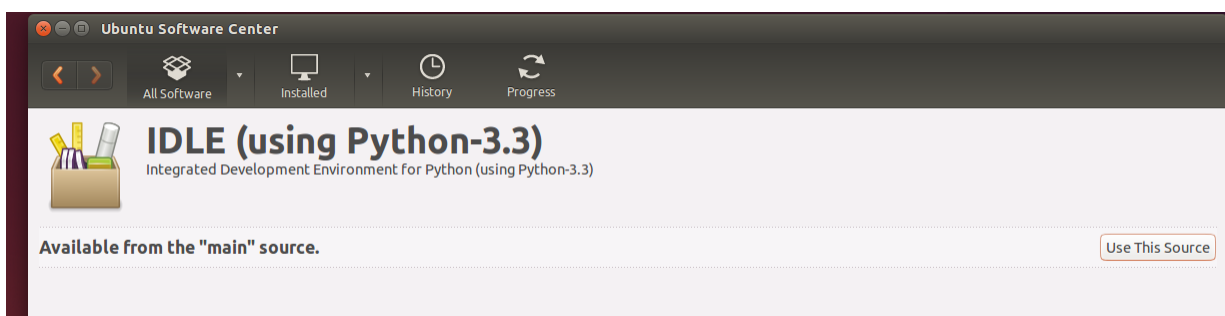
3. En el tercer paso procederemos a escribir la palabra “Python” en el textbox ubicado en la esquina superior izquierda y sin teclear enter, la aplicación sola hará una búsqueda y encontrará esos repertorios que contienen el IDLE de Python.

Nota: Debemos saber que los comandos de Python 2.7 y (3.4 , 3.3) son muy diferentes así que para el uso de este programa es necesario la instalación del 3.4 o el 3.3. En este caso utilizaremos el 3.3



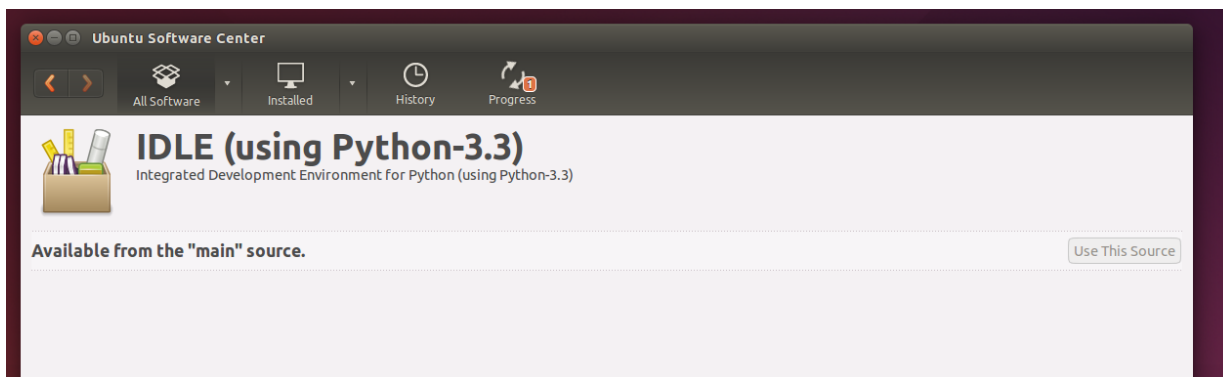
(Imagen 4)

En la imagen 4 se puede observar donde hay que presionar la opción que dice IDLE (using Python3.3) , opcional mente también funciona IDLE (using Python3.4) y seleccionar clic donde dice more info, como se muestra en la imagen 4.



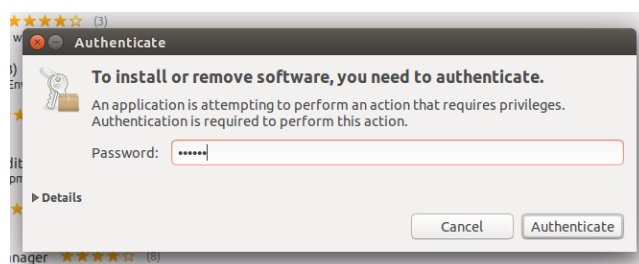
(Imagen 5)

En esta imagen s se puede observar lo cómo se abrirá la ventana teniendo el IDLE 3.3, en este caso se procederá a seleccionar el botón que dice USE THIS SOURCE, esto en español significa utilizar esta fuente.



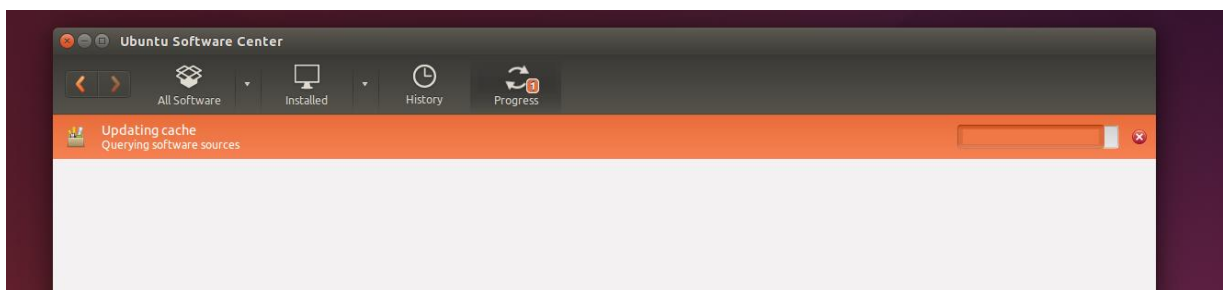
(Imagen 6)

Una vez seleccionado el botón se debe observar bloqueado el botón ya que la información seleccionada ha sido guardada, y en los siguientes pasos se mostrara la instalación



(Imagen 7)

Se deberá esperar a que abra la ventana de auto identificación que realiza Ubuntu cada vez que se elimine, o instale un software, se pondrá la contraseña del administrador ejemplo , cuando se va a iniciar sesión con el usuario de administrador Ubuntu pide la contraseña, en este caso sería esta.



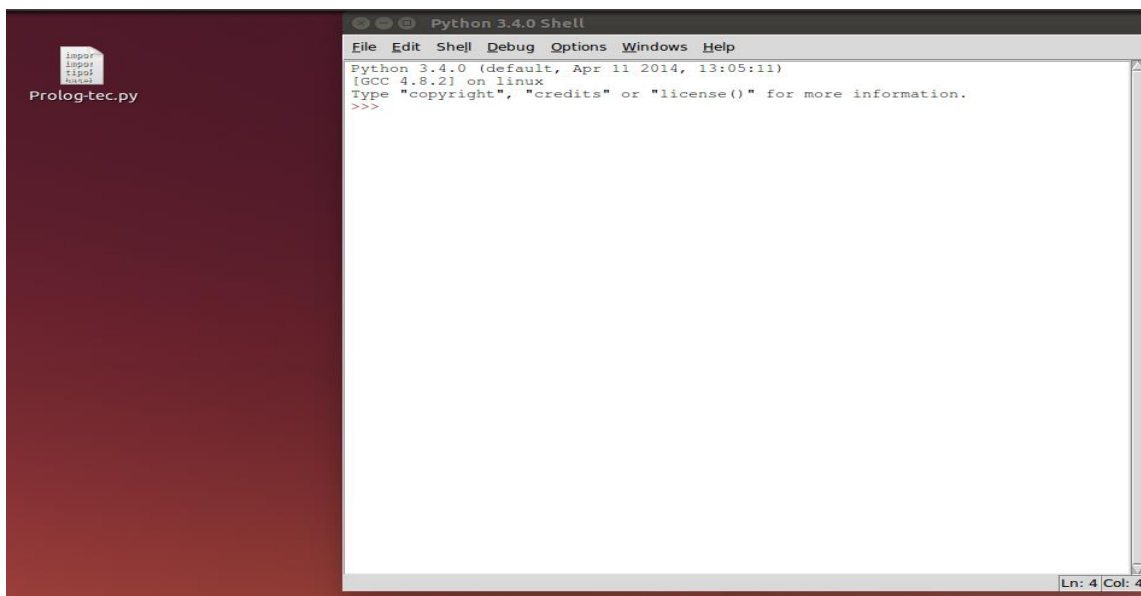
(Imagen 8)

Ya con la contraseña puesta, se empezará a descargar el programa e instalar automáticamente, aproximadamente dura 5 min con un internet de un Megabyte.



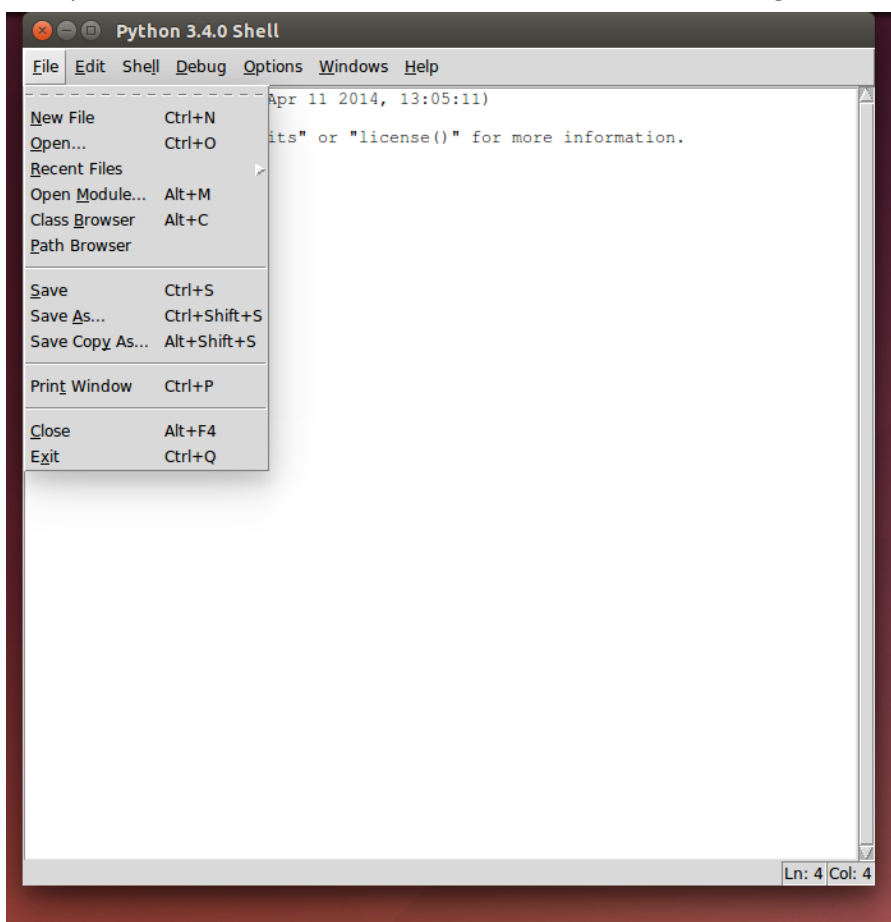
Al final de la instalación, la ventana anterior se cerrará automáticamente y ya se podrá utilizar el IDLE del lenguaje de programación Python. Procederemos a abrir el programa azul con amarillo que se encuentra en la esquina inferior de la barra.

(Imagen 10)



(Imagen 11)

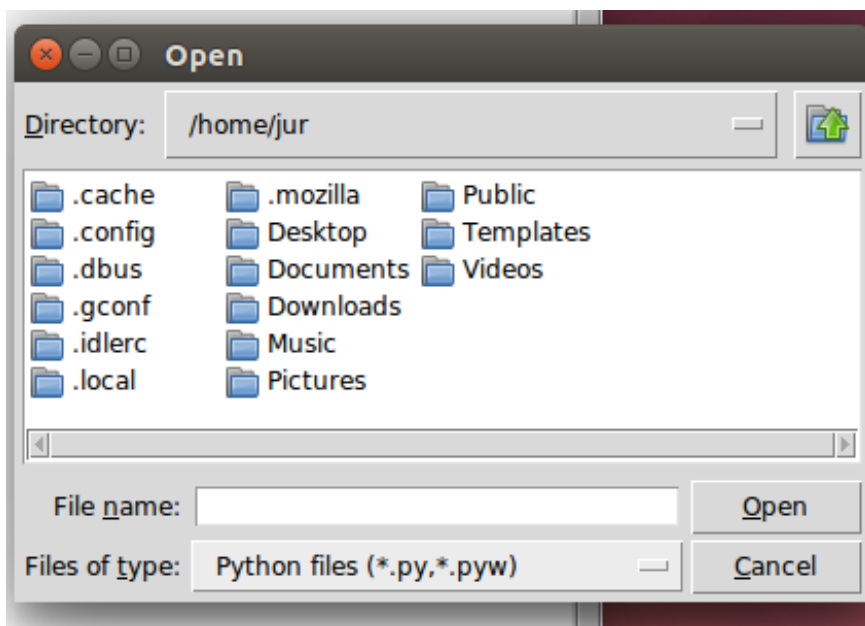
Se verá una ventana parecida a esta, y procederemos a mover el archivo del programa al escritorio para la comodidad del usuario. Se debe ver como en la imagen.



(Imagen 12)

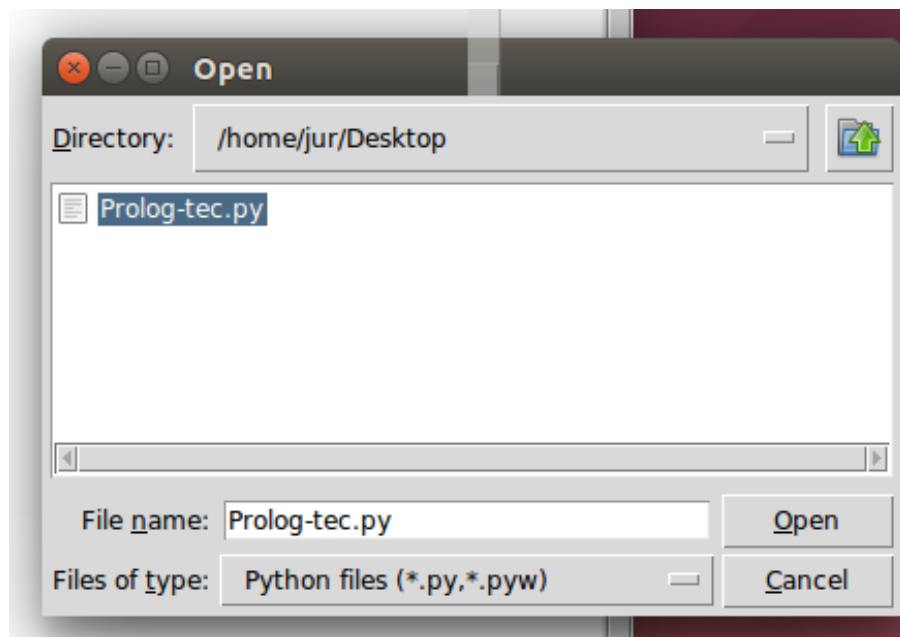
Se seleccionará el botón de File, y después el botón de Open, o podemos abrirla desde los comandos del teclado escribiendo CTRL+O.

(Imagen 12.1)



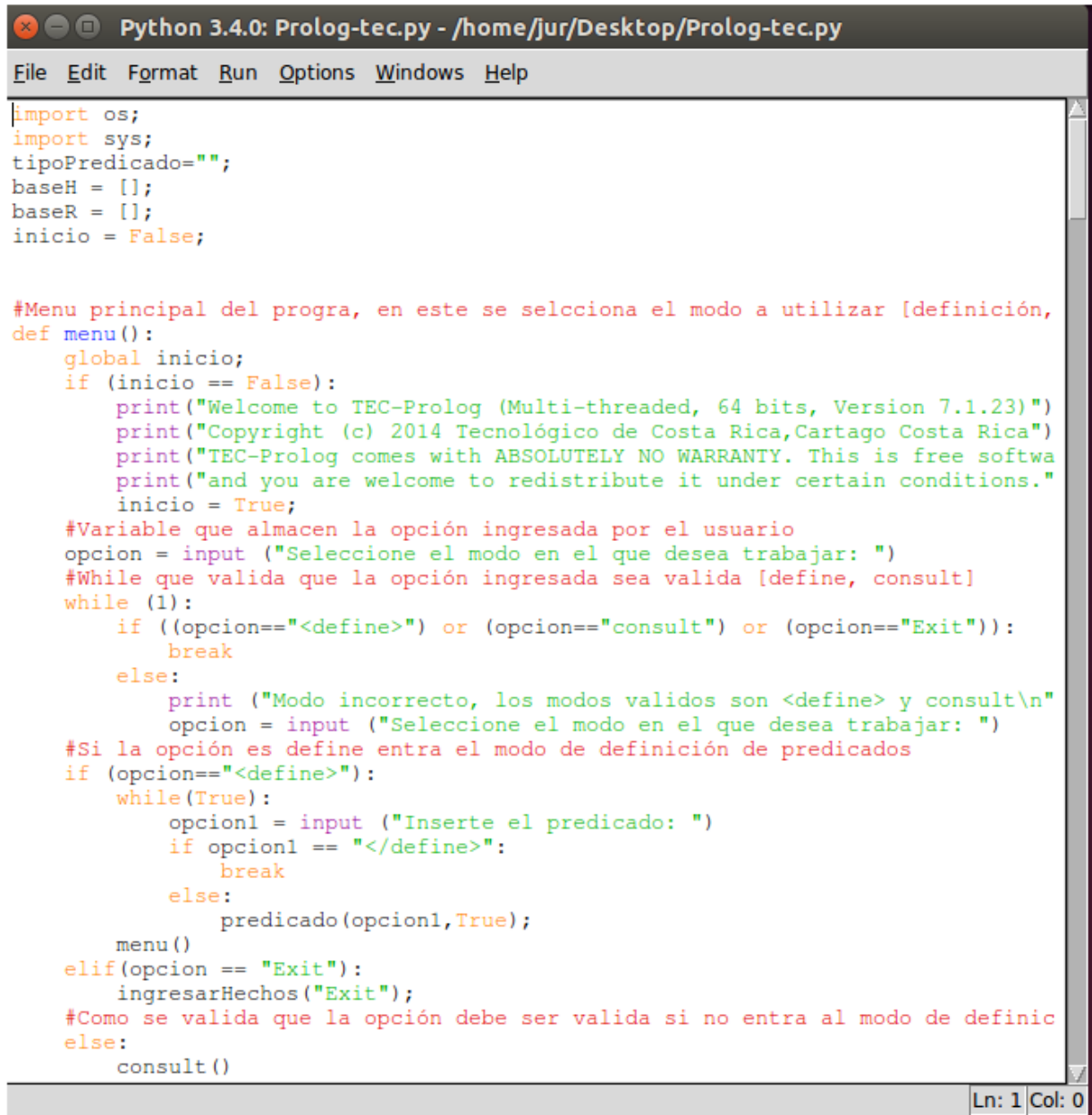
Se abrirá la siguiente ventana, como anteriormente se mencionó, el programa se almacenará en el escritorio, buscaremos la carpeta Desktop y luego Prolog-TEC.py

(Imagen 13)



Ya encontrado el archivo Prolog-TEC.py se deberá seleccionar la casilla donde dice Open que significa abrir en español.

(Imagen 14)



```
Python 3.4.0: Prolog-tec.py - /home/jur/Desktop/Prolog-tec.py
File Edit Format Run Options Windows Help

import os;
import sys;
tipoPredicado="";
baseH = [];
baseR = [];
inicio = False;

#Menu principal del progra, en este se selcciona el modo a utilizar [definición,
def menu():
    global inicio;
    if (inicio == False):
        print("Welcome to TEC-Prolog (Multi-threaded, 64 bits, Version 7.1.23)")
        print("Copyright (c) 2014 Tecnológico de Costa Rica, Cartago Costa Rica")
        print("TEC-Prolog comes with ABSOLUTELY NO WARRANTY. This is free softwa")
        print("and you are welcome to redistribute it under certain conditions.")
        inicio = True;
    #Variable que almacene la opción ingresada por el usuario
    opcion = input ("Seleccione el modo en el que desea trabajar: ")
    #While que valida que la opción ingresada sea valida [define, consult]
    while (1):
        if ((opcion=="<define>") or (opcion=="consult") or (opcion=="Exit")):
            break
        else:
            print ("Modo incorrecto, los modos validos son <define> y consult\n")
            opcion = input ("Seleccione el modo en el que desea trabajar: ")
    #Si la opción es define entra el modo de definición de predicados
    if (opcion=="<define>"):
        while(True):
            opcion1 = input ("Inserte el predicado: ")
            if opcion1 == "</define>":
                break
            else:
                predicado(opcion1,True);
        menu()
    elif(opcion == "Exit"):
        ingresarHechos("Exit");
    #Como se valida que la opción debe ser valida si no entra al modo de definic
    else:
        consult()
```

Ln: 1 Col: 0

(Imagen 15)

Una vez abierto el programa se mostrará el código del programa, el cual no debe ser modificado por ningún motivo, Este tiene una documentación interna en español y está escrita de la siguiente manera #.... en color rojo. En esta documentación se pueden observar más detalladas las funciones de cada función ya que por este medio se explica en prosa.

```

Python 3.4.0: Prolog-tec.py - /home/jur/Desktop/Prolog-tec.py
File Edit Format Run Options Windows Help
Python Shell
Check Module Alt+X
Run Module F5

import os;
import sys;
tipoPredicado="
baseH = [];
baseR = [];
inicio = False;

#Menu principal del progra, en este se selcciona el modo a utilizar [definición,
def menu():
    global inicio;
    if (inicio == False):
        print("Welcome to TEC-Prolog (Multi-threaded, 64 bits, Version 7.1.23)")
        print("Copyright (c) 2014 Tecnológico de Costa Rica, Cartago Costa Rica")
        print("TEC-Prolog comes with ABSOLUTELY NO WARRANTY. This is free softwa")
        print("and you are welcome to redistribute it under certain conditions.")
        inicio = True;
    #Variable que almacene la opción ingresada por el usuario
    opcion = input ("Seleccione el modo en el que desea trabajar: ")
    #While que valida que la opción ingresada sea valida [define, consult]
    while (1):
        if ((opcion=="<define>") or (opcion=="consult") or (opcion=="Exit")):
            break
        else:
            print ("Modo incorrecto, los modos validos son <define> y consult\n")
            opcion = input ("Seleccione el modo en el que desea trabajar: ")
    #Si la opción es define entra el modo de definición de predicados
    if (opcion=="<define>"):
        while(True):
            opcion1 = input ("Inserte el predicado: ")
            if opcion1 == "</define>":
                break
            else:
                predicado(opcion1,True);
        menu()
    elif(opcion == "Exit"):
        ingresarHechos("Exit");
    #Como se valida que la opción debe ser valida si no entra al modo de definic
    else:
        consult()

```

Ln: 1 Col: 0

(Imagen 16)

Después seleccionaremos la pestaña RUN que se muestra arriba después de eso le daremos RUN MODULE, También está la opción de hacerlo en el teclado, presionando F5, las teclas especiales que se encuentran en la parte de arriba del teclado, en caso de que no funcione puede realizarse con la tecla fn que está en la parte inferior izquierda del teclado y f5 a la misma vez.

```
>>>
Welcome to TEC-Prolog (Multi-threaded, 64 bits, Version 7.1.23)
Copyright (c) 2014 Tecnológico de Costa Rica, Cartago Costa Rica
TEC-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Seleccione el modo en el que desea trabajar: <define>
Inserte el predicado: perro(ruffo).
Inserte el predicado: gato(vegetta).
Inserte el predicado: perro(Thor).
Error
Inserte el predicado: perro(thor).
Inserte el predicado: perro(bruno)
Predicado incorrecto
Inserte el predicado: perro(bruno).
Inserte el predicado: </define>
Seleccione el modo en el que desea trabajar: |
```

Ln: 35 Col: 45

Modo: <define> (Imagen 17)

El programa ejecutará la siguiente sentencia y pedirá al usuario al modo que desee entrar. Existen 2 modos el de consulta y el de definición. Además se tiene la opción de salir mediante la sentencia "Exit".

El modo definición se entra escribiendo <define>, Luego este le pedirá al usuario que inserte el predicado, si este tiene errores en la escritura de devolverá algún tipo de error, por ejemplo con el uso de perro(Thor). , esto debido a que adentro de los paréntesis solo puede ir en minúscula o perro(bruno) debido a que debe tener un punto al final pero como no lo tiene el programa no permite la inserción a la base de conocimientos, Para salir de este modo lo único que es necesario es escribir la siguiente sentencia: </define>.

Se ingresa al modo de consulta con la sentencia <consult>, en el cual se mostrara una interfaz similar a la de prolog. Al indicar "?-" se solicita al usuario ingresar la consulta respectiva, luego presionar la tecla "enter" y devuelve el resultado según lo solicitado. Para salir del modo de consulta se escribe la sentencia </consult>. La imagen 18 muestra un ejemplo de ello.

```
Seleccione el modo en el que desea trabajar: <consult>
-? perro(motita) .

Yes
-? perro(pinky) .

No
-? animal(veggeta) .

No
-? animal(vegeta) .

No
-? animal(vegetta) .

Yes
-? perro(X) .

No
-?
-? </consult>
Seleccione el modo en el que desea trabajar: Exit
>>> |
```

Modo consulta (Imagen 18).

Bibliografía

Wikipedia. (s.f.). <http://es.wikipedia.org/wiki/Prolog#Historia>. Obtenido de Wikipedia.