Instituto Politécnico Nacional (IPN).

Escuela Superior de Computo.

Materia.	Fundamentos de Inteligencia Artificial.	**Grupo.**	4BM2.
Periodo.	Segundo departamental.	**Carrera.**	Ingeniería en Inteligencia Artificial.
Tema:	Practica 06.		
Profesor	Hernández Cruz Macario	**Alumnos:**	Carrillo Barreiro José Emiliano. Escobar Montoya Patricio-
Fecha de entrega	Noviembre 13 del 2023.		

Tema*: *Representación mediante marcos.

Resumen.

La Práctica 06 se centra en la creación de un sistema experto combinando Python y Prolog. Esta fusión estratégica aprovecha la versatilidad de Python y la lógica declarativa de Prolog para desarrollar un sistema inteligente. Basándonos en la programación lógica de Prolog, incorporamos hechos y heurísticas para potenciar el razonamiento. La interfaz de usuario en Python facilita la interacción, permitiendo a los usuarios ingresar datos sobre una especie y obtener información relevante del programa Prolog. Este enfoque híbrido destaca la colaboración efectiva entre dos lenguajes, proporcionando una solución eficiente para la creación de sistemas expertos.

Introducción.

En la práctica 06, exploramos las posibilidades de desarrollo de un sistema experto mediante la fusión estratégica de Python y Prolog. Este proyecto, fundamentado en la sinergia entre la versatilidad intrínseca de Python y la lógica declarativa característica de Prolog, se erige como un testimonio de la capacidad innovadora de combinar tecnologías distintas para alcanzar un objetivo común.

Prolog, un lenguaje arraigado en la programación lógica, despliega su potencial al modelar el conocimiento humano y establecer procedimientos inferenciales a través de hechos y reglas. Inspirado por la definición de Edward Feigenbaum, el sistema experto resultante se conforma de hechos y heurísticas, utilizando reglas de razonamiento plausible para abordar problemas específicos.

En esta implementación, hemos incrustado tanto hechos como heurísticas en el entorno Prolog, facilitando el razonamiento a través de reglas de inferencia que potencian nuestras consultas. La interfaz de usuario desarrollada en Python añade una capa intuitiva, permitiendo a los usuarios introducir información sobre una especie y acceder a detalles relevantes derivados del programa Prolog.

Este maridaje singular entre Python y Prolog no solo proporciona una solución eficaz para la creación de sistemas expertos, sino que también destaca la colaboración armoniosa de dos lenguajes de programación distintos, cada uno contribuyendo con sus características distintivas para forjar un sistema completo y eficiente.

Desarrollo de la práctica.

El código proporcionado en PROLOG representa una base de conocimiento que incluye información nutricional sobre diferentes alimentos, como frutas, proteínas, cereales,

verduras, carbohidratos y grasas saludables. Además, el código implementa un programa interactivo que permite a los usuarios calcular su Índice de Masa Corporal (IMC) y recibir recomendaciones de dieta saludable.

A continuación, se describe la funcionalidad principal del código:

1. Base de Conocimiento (PROLOG):

Este código en Prolog establece una serie de hechos que definen relaciones entre diferentes clases de animales, sus propiedades y ejemplos específicos de esas clases. Aquí hay un resumen de lo que hace el código:

1.1. Definición de Clases y Relaciones:

- Se definen clases de animales como `elephantidae`, `equidae`, `ursidae`, etc.

- Se establecen relaciones de subclase (`subclase_de`) entre las clases, como, por ejemplo, `elephantidae` es subclase de `proboscidea`.

1.2. Propiedades de las Clases:

- Se definen propiedades específicas para cada clase de animal, como el hecho de que los `elefantes` son de la clase `elephantidae` y tienen la propiedad de ser elefantes (`son(elefantes)`).

1.3. Instancias Específicas:

- Se definen instancias específicas de animales y se les asignan propiedades, como `elefante_asiatico` que es una instancia de `elephantidae` y tiene un nombre científico y una imagen asociada.

1.4. Consultas:

- Se proporciona un conjunto de consultas que permiten obtener información sobre las clases y propiedades de los animales.

1.5. Reglas de Inferencia:

- Se definen reglas que permiten determinar a qué clase pertenece un animal y qué propiedades tiene.

```
frame(elephantidae, subclase_de(proboscidea),
propiedades([son(elefantes)])).
frame(equidae, subclase_de(perissodactyla),
propiedades([son(compatibles)])).
frame(rhinocerotidae, subclase_de(perissodactyla),
propiedades([tien(cuerno)])).
```

```
frame(ursidae, subclase_de(carnivora), propiedades([son(osos)])).
frame(noctilionoidea, subclase_de(chiroptera),
propiedades([es(pescador)])).
frame(animal, subclase_de(objeto), propiedades([puede(sentir),
puede(respirar)])).
frame(mamifero, subclase_de(animal), propiedades([puede(mamar),
respira(aire)])).
frame(artiodactilo, subclase_de(mamifero),
propiedades([tiene(pesugnas_pares), comen(plantas)])).
frame(carnivora, subclase_de(mamifero), propiedades([comen(carne)])).
frame(primates, subclase_de(mamifero),
propiedades([tiene(cerebro_desarrollado)])).
frame(proboscidea, subclase_de(mamifero), propiedades([es(grande)])).
frame(perissodactyla, subclase_de(mamifero),
propiedades([tiene(pesugnas_impares)])).
frame(chiroptera, subclase_de(mamifero), propiedades([tien(alas),
es(roedor)])).
frame(camelidos, subclase_de(artiodactilo),
propiedades([familia_de(camellos)])).
frame(canidae, subclase_de(carnivora),
propiedades([puede(comer_vegetales)])).
frame(suidae, subclase_de(artiodactilo),
propiedades([son(inteligentes)])).
frame(hominidae, subclase_de(primates),
propiedades([son(grandes_simios)])).
frame(felidae, subclase_de(carnivora), propiedades([son(felinos)])).

frame(elefante_asiatico, subclase_de(elephantidae),
propiedades([nombre_cientifico(elephas_maximus),
imagen('elefante.png')])).
frame(caballo, subclase_de(equidae),
propiedades([nombre_cientifico(equus_caballus), ruido(relincha),
imagen('caballo.png')])).
frame(rinoceronte_blanco, subclase_de(rhinocerotidae),
propiedades([nombre_cientifico(ceratotherium_simum),
imagen('rino.png')])).
frame(oso_pardo, subclase_de(ursidae),
propiedades([nombre_cientifico(ursus_arctos), pelaje(marron),
imagen('pardo.png')])).
frame(oso_polar, subclase_de(ursidae),
propiedades([nombre_cientifico(ursus_maritimus), pelaje(blanco),
imagen('polar.png')])).
frame(panda, subclase_de(ursidae),
propiedades([nombre_cientifico(ailuropoda_melanoleuca),
pelaje(blanco_y_negro), imagen('panda.png')])).
```

```
frame(murcielago, subclase_de(noctilionoidea),
propiedades([nombre_cientifico(noctilio_albiventris),
imagen('dracula.png')])).
frame(vicugna, subclase_de(camelidos),
propiedades([nombre_cientifico(vicugna_vicugna),
imagen('vicugna_vicugna.png')])).
frame(guanaco, subclase_de(camelidos),
propiedades([nombre_cientifico(lama_guanicoe),
imagen('guanaco.png')])).
frame(llama, subclase_de(camelidos),
propiedades([nombre_cientifico(lama_pacos), imagen('llama.png')])).
frame(perro, subclase_de(canidae),
propiedades([nombre_cientifico(canis_lupus_familiaris),
imagen('guagua.png')])).
frame(lobo_rojo, subclase_de(canidae),
propiedades([nombre_cientifico(canis_rufus),
imagen('lobo_rojo.png')])).
frame(coyote, subclase_de(canidae),
propiedades([nombre_cientifico(canis_latrans),
imagen('coyote.png')])).
frame(babirusa, subclase_de(suidae),
propiedades([nombre_cientifico(babyrousa_babyrussa),
imagen('babirusa.png')])).
frame(jabali, subclase_de(suidae),
propiedades([nombre_cientifico(pumba_o_sus_scrofa),
imagen('jabali.png')])).
frame(chimpance, subclase_de(hominidae),
propiedades([nombre_cientifico(pan_troglodytes), vive_en(selvas),
imagen('chimpance.png')])).
frame(gorila, subclase_de(hominidae),
propiedades([nombre_cientifico(gorilla_gorilla),
vive_en(bosques_costeros), imagen('gorila.png')])).
frame(orangutan, subclase_de(hominidae),
propiedades([nombre_cientifico(pongo_pygmaeus), vive_en(selvas),
imagen('orangutan.png')])).
frame(puma, subclase_de(felidae),
propiedades([nombre_cientifico(puma_concolor), es(pequegna),
emite(maullidos), imagen('puma.png')])).
frame(leopardo, subclase_de(felidae),
propiedades([nombre_cientifico(panthera_pardus), es(rapido),
imagen('leopardo.png')])).
frame(gato_montes, subclase_de(felidae),
propiedades([nombre_cientifico(leopardus_geoffroyi), es(chiquito),
imagen('montes.png')])).
```

```
que_es(X):-((instancia(X),es(Clase,X));
(subclase(X),subc(X,Clase))),Clase\=objeto,write('Es
'),writeln(Clase),fail.
es(Clase,Obj):- frame(Obj,instancia_de(Clase),_).
es(Clase,Obj):- frame(Obj,instancia_de(Clasep),_),subc(Clasep,Clase).
subc(C1,C2):- frame(C1,subclase_de(C2),_).
subc(C1,C2):- frame(C1,subclase_de(C3),_),subc(C3,C2).
subclase(X):-frame(X,subclase_de(_),_).
instancia(X):-frame(X,instancia_de(_),_).
propiedadesc(objeto):-!.
propiedadesc(X):-
frame(X,subclase_de(Y),propiedades(Z)),imprime(Z),propiedadesc(Y).
propiedadesi(X):-
frame(X,instancia_de(Y),propiedades(Z)),imprime(Z),propiedadesc(Y).
props(X):-
(instancia(X),propiedadesi(X));(subclase(X),propiedadesc(X)).
imprime([]):-!.
imprime([H|T]):-writeln(H),imprime(T).
about(X):-que_es(X);props(X).
```

2. Interfaz Gráfica y de Usuario (Python):

Este código de Python utiliza la biblioteca Tkinter para crear una interfaz gráfica que interactúa con un programa en Prolog. A continuación, se explican las funciones principales del código:

2.1. Configuración de la Interfaz Gráfica:

- `Tk()`: Crea una instancia de la clase `Tk` para la interfaz gráfica.

- `title()`: Establece el título de la ventana.

- `iconbitmap()`: Asigna un ícono a la ventana.

- `maxsize()`: Establece el tamaño máximo de la ventana.

2.2. Creación de Elementos de la Interfaz:

- Se utilizan las clases `Label`, `Entry`, `Text`, y `Button` para crear etiquetas, campos de entrada, áreas de texto y botones, respectivamente.

- Se establece la apariencia y posición de estos elementos en la ventana mediante métodos como `grid()` y configuraciones adicionales como `config()`.

2.3. Listado de Especies:

- Se define una cadena (`especieslista`) que contiene un listado de especies de animales disponibles.

2.4. Función `buscar () `:

- Se recoge el nombre de la especie ingresada por el usuario en la interfaz.

- Se escribe una consulta en un archivo llamado "input.txt" en formato Prolog (`about(nombre_de_especie). `).

- Se ejecuta un programa en Prolog (`swipl especies.pl < input.txt > output.txt`) para procesar la consulta y obtener información sobre la especie.

- Se lee el resultado desde un archivo llamado "output.txt".

- Se actualiza el área de texto (`textResult`) con la información obtenida y se muestra la imagen correspondiente de la especie.

2.5. Interacción con Prolog:

- Se utiliza un archivo "especies.pl" en Prolog que contiene reglas y hechos relacionados con las especies animales.

2.6. Presentación de Resultados:

- Se muestra la información obtenida en un área de texto (`textResult`).

- Se muestra la imagen correspondiente a la especie consultada en una etiqueta (`image_label`).

2.7. Botón de Consulta:

- Se crea un botón (`btnConsultar`) que, al hacer clic, activa la función `buscar()`.

2.8. Configuración y Ejecución de la Ventana:

- Se configura el fondo de la ventana (`bg`) y se ejecuta el bucle principal de la interfaz (`mainloop()`).

```
from tkinter import *
import os

root = Tk()
root.title("Práctica 06 - FIA")
root.iconbitmap("images/pk.ico")
root.maxsize(700,770)

especieslista="\npuma\nleopardo\ngato_montes\nelefante_asiatico\ncaballo\nrinoceronte_blanco\noso_pardo\noso_polar\npanda\nmurcielago\nvicugna\nguanaco\nllama\nperro\nlobo_rojo\ncoyote\nbabirusa\njabalí\nchimpancé\ngorila\norangutan"
```

```
Label(root,  text="PRACTICA 06 - FIA 4BM1 \nEQUIPO:", bg="white",
font=("Helvetica Bold", 25)).grid(pady=5, row=0, column=0,
columnspan=7)
Label(root, text="CORONA REYES MAURICIO DASSEL\nMARTINEZ MENDEZ
DIEGO\nPACHECO SANCHEZ RODRIGO\n", bg="white", font=("Helvetica Bold",
15)).grid( pady=5, row=1, column=0, columnspan=7)
Label(root,text="Ingresa una especie: ", bg="white", font=("Helvatical
bold", 15)).grid( pady=5, row=2, column=0)


Label(root, text="Especies disponibles: ", bg="white", font=("Arial
bold", 15)).grid(pady=5, row=2, column=3)
listaesp =Label(root, justify=LEFT,text=especieslista)
listaesp.config(bg="white")
listaesp.grid(pady=5, row=1, rowspan = 7, column= 3)

especieInput = Entry(root, width=30)
especieInput.config(highlightbackground="gray",
highlightcolor="black", highlightthickness=2)
especieInput.grid(pady=20, row=3, column=0)

textResult = Text(
    width=50,
    height=5,
    font=("Arial", 12)
)
textResult.grid(pady=5, row=5, column=0)

image_label = Label(root)
image_label.grid(pady=40, row=6, column=0)
image_path = "images/pk.png"
image = PhotoImage(file=image_path)
image_label.image = image
image_label.configure(image=image, height=200, width=200, bg="white")

textoaux = "Escriba un animal del listado!"

def buscar():
    specie = especieInput.get()
    with open("input.txt", "w") as file:
        file.write(f"about({specie}).")
    os.system(f"swipl especies.pl < input.txt > output.txt")
    result = ""
    with open("output.txt", "r") as file:
```

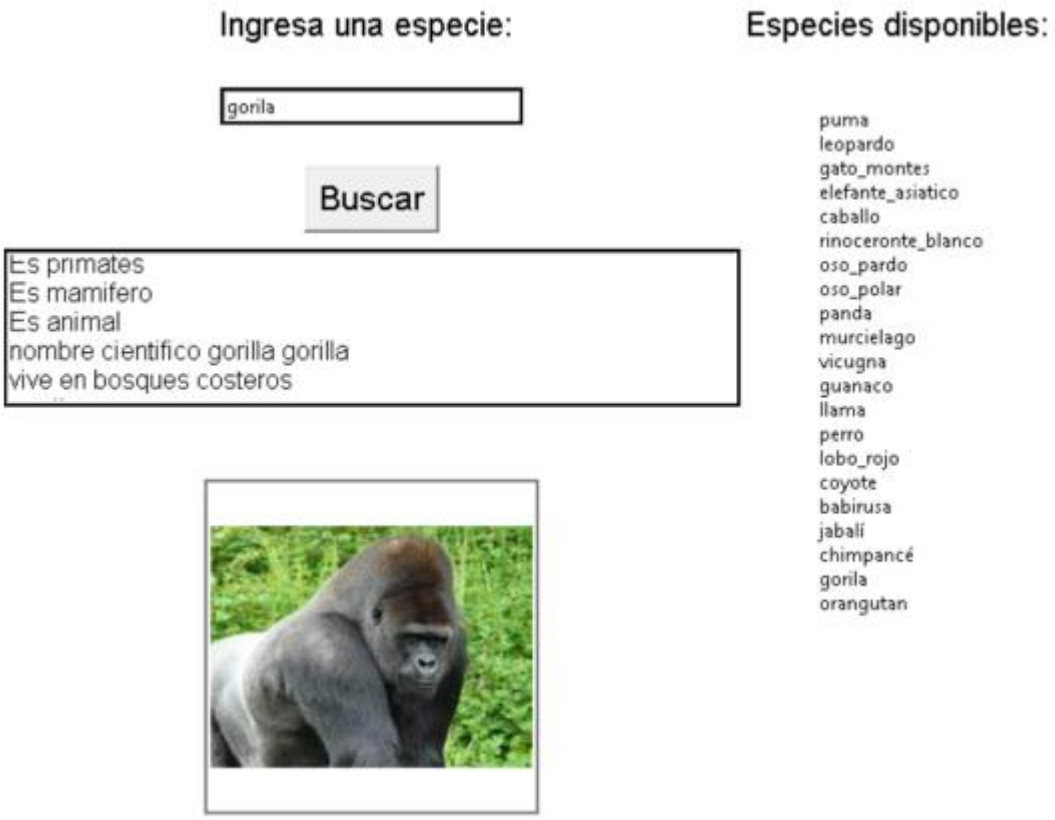
```
        result = file.readlines()
        cosa = ""
        for x in range(0, len(result)):
            cosa = cosa + "\n" +str(result[x]).replace("\n", "
").replace("(", " ").replace(")", " ").replace("_", "
").replace("imagen","").replace(".png", "")
    textResult.delete("1.0", "end")
    textResult.insert(INSERT, str(cosa)[:-11])
    textResult.config(highlightbackground="black",
highlightthickness=2)

    for r in result:
        if "imagen" in r:
            image_path = r.split("(")[1].split(")")[0]
            break
    else:
        textResult.insert(INSERT, str(textoaux))
        image_path = "pk.png"

    image = PhotoImage(file="images/"+image_path)
    image_label.image = image
    image_label.configure(image=image, height=200, width=200,
highlightbackground="gray", highlightthickness=2)

btnConsultar = Button(
    text="Buscar",
    font=("Arial", 16),
    command=buscar
).grid(pady=5, row=4, column=0)

root.config(bg="white")
root.mainloop()
```

En resumen, este código crea una interfaz gráfica simple que permite al usuario ingresar el nombre de una especie de animal y obtener información sobre ella mediante consultas a un programa Prolog. La interfaz muestra la información en un área de texto y la imagen correspondiente a la especie consultada.

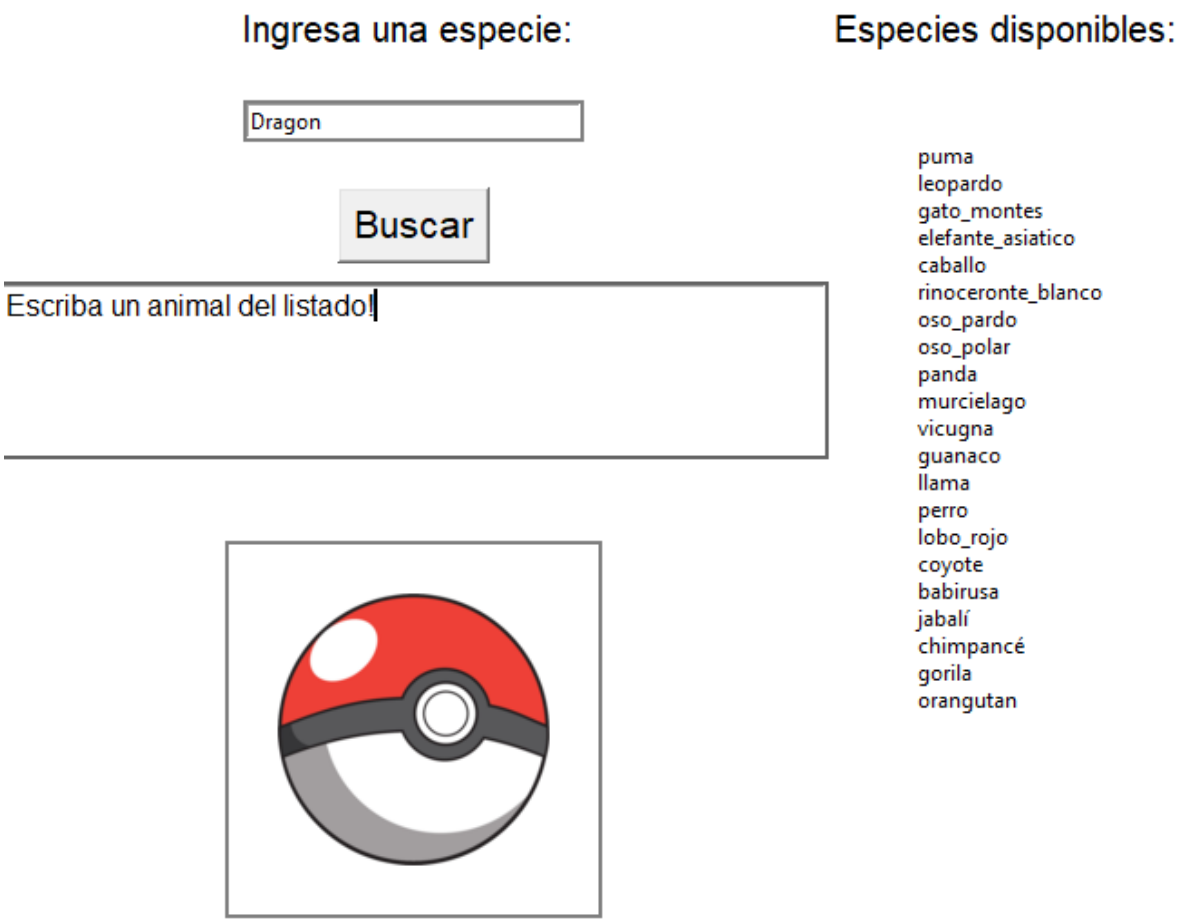
Demostración del funcionamiento:

- Pantalla de inicio (Menu).

- Opción 1. (La opción se encuentra en la base de conocimiento).



- Opción 2. (La opción NO se encuentra en la base de conocimiento).



Conclusión.

La práctica destaca la colaboración entre Python y Prolog para desarrollar un sistema experto. Prolog modela jerarquías y reglas lógicas, mientras que Python proporciona una interfaz gráfica amigable. La combinación de ambos lenguajes permite clasificar especies animales, ofreciendo consultas intuitivas y una experiencia de usuario eficiente. La práctica resalta la importancia de la integración de lenguajes en el desarrollo de sistemas expertos especializados y user-friendly.
