Área académica de Ingeniería en Computadores

Lenguajes, compiladores e intérpretes – CE3104

TAREA CORTA 2

Estudiantes:

Daniel Brenes Gómez

Saúl Gómez Ramírez

Esteban Madrigal Sandoval

Profesor:

Marco Rivera Meneses

IIS - 2020

1. Descripción de los hechos y reglas implementadas.

Se utilizaron hechos de Prolog para guardar los restaurantes, los tipos de menú, de bebida, etc., como si fuesen una base de datos predefinida. Esto para facilitar la búsqueda por medio de hechos, según lo que el usuario ingrese. Lo anterior, se implementó con el siguiente formato:

- restaurante([Nombre, TipoMenu, [Provincia, Lugar], Capacidad, Obligaciones]).
- menu(TipoComida, [RestaurantesDisponibles], [SaboresDisponibles]).
- bebida(TipoBebida, [RestaurantesDisponibles]).

También, se implementaron hechos y reglas para validar el sintagma verbal y nominal, los cuales funcionan bajo la notación de Backus-Naur Form (BNF), usado para expresar gramáticas libres de contexto, es decir, una manera para describir un lenguaje formal. Cabe resaltar que, para validar el sintagma de la oración ingresada, es necesario el uso de algunos hechos básicos como lo son determinante_n(A,B), determinante_m(A,B), determinante_f(A,B), sustantivo_g(A,B), verbo(A,B), los cuales ayudan a verificar si se ingresó un determinante masculino, femenino o neutro, así como también un verbo o un sustantivo.

Se implementaron algunas reglas básicas para el correcto funcionamiento del programa y estas son:

- input_to_list(Oracion): Esta regla retorna una lista con las palabras de la oración ingresada.
- input_to_string(Oracion): Esta regla retorna un string con las palabras de la oración ingresada.
- list_to_string(Elemento, Lista): Esta regla retorna un string con las palabras de la lista ingresada.
- string_to_list_of_atoms(Lista, String): Esta regla retorna una lista con las palabras del string ingresado.
- lista_vacia(Lista, Resultado): Esta regla retorna un true si la lista ingresada está vacía.

- miembro(Elemento, Lista): Esta regla retorna un true si el elemento dado se encuentra en una lista ingresada.

Para validar si los datos ingresados por el usuario se encuentran en la base de datos predefinida, se implementaron las siguientes reglas:

- compareTipoDeMenu(Lista, Elemento): Esta regla retorna un tipo de menú específico válido si el elemento dado se encuentra en la base de datos, de lo contrario le pide al usuario que ingrese nuevamente la oración.
- compareComida(Lista, Elemento): Esta regla retorna una comida válida si el elemento dado se encuentra en la base de datos, de lo contrario le pide al usuario que ingrese nuevamente la oración.
- compareSaborComida(Lista, Elemento): Esta regla retorna un sabor de comida válido si el elemento dado se encuentra en la base de datos, de lo contrario le pide al usuario que ingrese nuevamente la oración.
- compareBebida(Lista, Elemento): Esta regla retorna una bebida válida si el elemento dado se encuentra en la base de datos, de lo contrario le pide al usuario que ingrese nuevamente la oración.
- compareLugar(Lista, Elemento): Esta regla retorna un lugar válido si el elemento dado se encuentra en la base de datos, de lo contrario le pide al usuario que ingrese nuevamente la oración.
- compareCantidad(Lista, Elemento): Esta regla retorna una cantidad de personas válida si el elemento dado se encuentra en la base de datos, de lo contrario le pide al usuario que ingrese nuevamente la oración.

Entre las reglas principales para que el programa funcione se encuentran:

a. inicio():

   Este es la regla principal que inicia el programa llamando a la regla encabezado, que imprime un mensaje de bienvenida, y a la regla pedirDatos.

b. pedirDatos:

   Tiene la siguiente sintaxis: pedirDatos(NombreRestaurante, TipoMenu, TipoComida, SaborComida, TipoBebida, LugarDeseado, CantidadDeseada).

   Esta regla es llamada por inicio(). y realiza la función de establecer una conversación con el usuario, así como de llamar a las reglas necesarias para verificar la validez de los datos ingresados las cuales son verificación_gramatical(Oracion). que ejecuta el sintagma nominal y verbal, y alguna de todas las reglas de tipo compareXXX(Lista, Elemento), que verifica los datos ingresados con los de la base de datos.

c. buscarRestauranteConDatosIngresados:

   Esta regla recibe como parámetro las descripciones necesarias para buscar el restaurante deseado por el usuario y estos son el resultado de la regla pedirDatos(NombreRestaurante, TipoMenu, TipoComida, SaborComida, TipoBebida, LugarDeseado, CantidadDeseada).

d. crearReferencia:

   Tiene la siguiente sintaxis:
   crearReferencia(NombreRest, DireccionTemp, Obligaciones).

   Esta regla escribe en consola la referencia encontrada según los datos ingresados y el resultado de ejecutar la regla anterior, es decir, buscarRestauranteConDatosIngresados(NombreRestaurante, TipoMenu, TipoComida, SaborComida, TipoBebida, LugarDeseado, CantidadDeseada).

e. buscarNuevamente:

   Esta regla le indica al usuario si desea buscar nuevamente un restaurante con los datos ingresados y en caso de que esto ocurra, se llama nuevamente a la regla pedirDatos, de lo contrario, se llama a la regla despedida que lo que hace es mostrar un mensaje de despedida.

2. Descripción de las estructuras de datos desarrolladas.

Las estructuras de datos desarrolladas fueron listas que se manejaron como hechos de Prolog que contienen a dichas listas. Lo anterior se realizó con la finalidad de facilitar las búsquedas y permitir un mejor acceso a los datos preestablecidos. Es necesario mencionar que, todas las oraciones ingresadas por el usuario son manejadas como listas de átomos a lo interno del programa.

3. Descripción detallada del algoritmo general de solución.

El mecanismo implementado para brindar la referencia de un restaurante específico con base en los datos ingresados por el usuario convierte la oración leída en consola en una lista para luego verificar con la regla de validación_gramatical todo lo respecto al sintagma, ya sea nominal o verbal. De esta manera, si una oración contiene un sintagma incorrecto para el habla de los humanos, será rechazado y esto se verifica cada vez que el usuario brinda una respuesta a la pregunta del sistema experto.

Es importante mencionar que la sintaxis que se debe utilizar para que el sistema experto reconozca una oración válida es la siguiente:

- [Sujeto] [Verbo] [Predicado]
- [Verbo] [Predicado]

Además, los sujetos que el sistema experto reconoce como válidos son:

- “Yo”.
- “yo”.
- “Nosotros”.
- “nosotros”.

También, los verbos y combinaciones de estos que el sistema experto reconoce como válidos son:

1. “comer”, “tomar”, “beber”, “quiero”, “queremos”.

2. “Quiero”, “Queremos”.

3. “quiero comer”, “quiero tomar”, “quiero beber”, “queremos comer”, “queremos tomar”, “queremos beber”.

4. “Quiero comer”, “Quiero tomar”, “Quiero beber”, “Queremos comer”, “Queremos tomar”, “Queremos beber”.

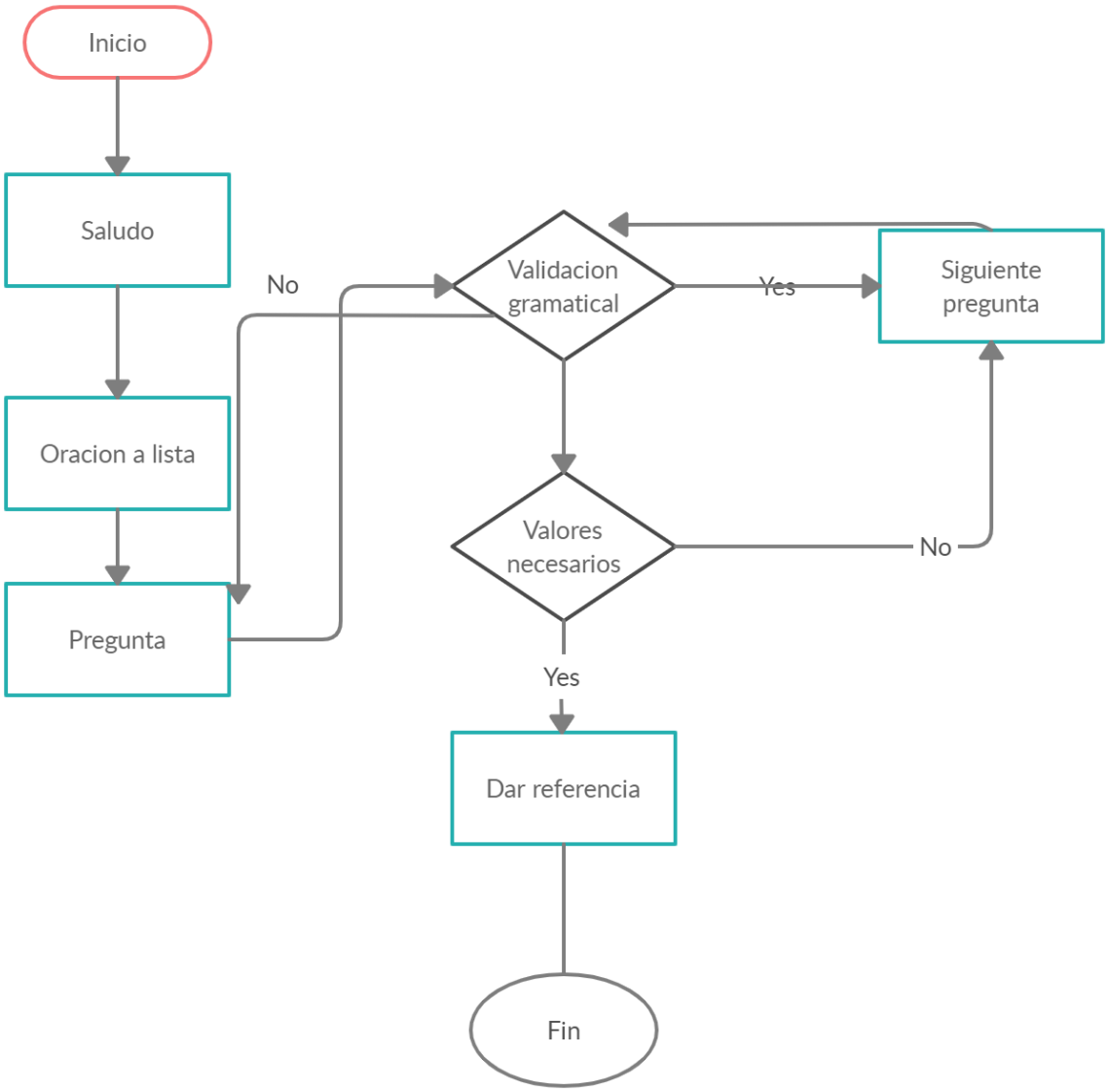
5. “quiero estar cerca de”, “quiero algo cerca de”, “quiero estar alrededor de”, “quiero algo alrededor de”, “queremos estar cerca de”, “queremos algo cerca de”, “queremos estar alrededor de”, “queremos algo alrededor de”.

6. “Quiero estar cerca de”, “Quiero algo cerca de”, “Quiero estar alrededor de”, “Quiero algo alrededor de”, “Queremos estar cerca de”, “Queremos algo cerca de”, “Queremos estar alrededor de”, “Queremos algo alrededor de”.

7. “seria”, “seriamos”, “somos”.

8. “Seria”, “Seriamos”, “Somos”.

El diagrama de flujo que representa el funcionamiento del programa es el siguiente:



4. Problemas encontrados

DESCRIPCIÓN	INTENTOS DE SOLUCIÓN SIN ÉXITO	SOLUCIONES ENCONTRADAS	RECOMENDACION	CONCLUSION	PROBLEMA CORREGIDO	BIBLIOGRAFÍA
El sistema no identificaba las palabras clave como hamburguesa	1	Corregir el uso de listas	Revisar paréntesis	Tener cuidado al momento de ubicar los paréntesis de las listas y donde ponerlos	✓	-
El sistema no daba una respuesta en la mayoría de los casos	2	Expandir bastante la base de datos	Poner más tipos de comida, menús, lugares…	Tomar en cuenta una gran base de datos para encontrar respuesta	✓	-
El sistema solo reconoce restaurantes y comidas en minúscula, y si tienen dos palabras o más deben ir pegadas	1	En este caso no hay solución ya que el sistema se desarrolló en la manera que las validaciones fueron realizadas	Cambiar como se puede identificar las palabras clave	Tomar en cuenta estos detalles a la hora de registrar elementos en un programa	**X**	-

5. Plan de Actividades realizadas por estudiante

TIPO DE TAREA	DESCRIPCIÓN DE TAREA	TIEMPO ESTIMADO	RESPONSABLE	FECHA
Funcionamiento lógico del programa	• Programar el saludo y la despedida del sistema experto.	1h	**Esteban Madrigal**	**4/10/2020**
	• Programar y definir los verbos posibles.	1h	**Saúl Gómez**	**7/10/2020**
	• Programar y definir los sustantivos posibles.	1h	**Saúl Gómez**	**7/10/2020**
	• Programar y definir el sintagma nominal.	3h	**Esteban Madrigal**	**9/10/2020**
	• Programar y definir el sintagma verbal.	3h	**Esteban Madrigal**	**9/10/2020**
	• Definir la base de datos de restaurantes, comida, lugar, capacidad, etc.	3h	**Saúl Gómez**	**7/10/2020**
	• Programar las respuestas necesarias según los datos ingresados.	1h	**Daniel Brenes**	**10/10/2020**
	• Programar la identificación de palabras clave de entrada.	3h	**Daniel Brenes**	**10/10/2020**
	• Programar la sugerencia de restaurantes al usuario según los datos ingresados.	4h	**Daniel Brenes**	**10/10/2020**
Documentación	• Realizar la descripción de los hechos y reglas implementadas.	1h	**Todos**	**10/10/2020**
	• Realizar la descripción de las estructuras de datos desarrolladas.	2h	**Todos**	**10/10/2020**
	• Realizar la descripción detallada del algoritmo general de solución.	2,5h	**Todos**	**11/10/2020**
	• Realizar la descripción de los problemas sin solución.	1h	**Todos**	**11/10/2020**
	• Realizar la descripción de los problemas encontrados.	2h	**Todos**	**11/10/2020**
	• Realizar la plantilla del documento para la documentación externa.	1h	**Esteban Madrigal**	**4/10/2020**
	• Terminar la sección de conclusiones del proyecto.	1h	**Todos**	**11/10/2020**
	• Terminar la sección de recomendaciones.	1h	**Todos**	**11/10/2020**
	• Terminar la sección de bibliografía consultada en todo el proyecto.	1h	**Todos**	**11/10/2020**

6. Conclusiones y recomendaciones.

Gracias a este proyecto sobre el trabajo de un sistema experto se analizó como este se comporta y compone, mediante la utilización de bases de datos, listas, sintagmas, entre otros, los cuales son una pieza fundamental en la construcción de programas que ayudan a encontrar la mejor opción según las especificaciones y deseos del usuario. De esta forma, es posible crear nuevas versiones que aporten al desarrollo de aplicaciones que tengan el fin de proveer soluciones a aquellos problemas que puedan ser enfrentados mediante un sistema experto. Además de comprender más a fondo el concepto anterior, se aprendió a como diseñarlo mediante el lenguaje Prolog, como también la administración de listas con reglas que brindan la solución deseada y por ende, la finalización del proyecto.

Se recomienda tener una vasta base de datos, para apoyar los deseos del usuario, y que este pueda encontrar su solución, en este caso, un restaurante con las especificaciones dadas mediante las preguntas hechas por el sistema. Además de eso, contar con un diseño de sintagmas nominales y verbales para poder ser más flexible con la escritura de las respuestas del usuario, de forma que sean lo más humanas posibles.

7. Bibliografía consultada en todo el proyecto

[1] Peinado V. (2014) Prolog: Listas. Disponible en: http://vitojph.github.io/ling/mrc/Prolog-Listas.pdf

[2] Guillen A. (2015) Sistemas Expertos: Definición, Aplicaciones y Ejemplos. Disponible en: https://www.tecnologias-informacion.com/sistemas-expertos.html

[3] Lezcano M. (2000) Prolog y los sistemas Expertos. Disponible en: https://www.researchgate.net/profile/Mateo_Lezcano2/publication/310314071_Prolog_y_los_Sistemas_Expertos/links/582b3fa208ae004f74afab90/Prolog-y-los-Sistemas-Expertos.pdf

[4] Soler F. Programación de gramáticas clausales en Prolog. Disponible en: https://personal.us.es/fsoler/papers/05capgram.pdf}

8. Bitácora

N° ACTIVIDAD	ACTIVIDAD	FECHA	DURACIÓN	ENCARGADO	OBSERVACIONES
1	Definición de una regla para que el sistema experto realice un saludo al usuario	**3/10/2020**	**0.5h**	**Esteban Madrigal**	Queda pendiente la despedida
2	Llenando la base de datos con verbos	**3/10/2020**	**1h**	**Saúl Gómez**	-
3	Terminado la regla para despedir al usuario	**4/10/2020**	**0.5h**	**Esteban Madrigal**	Todo funciona correctamente en el saludo y despedida
4	Definición de regla para buscar restaurante del usuario	**5/10/2020**	**4h**	**Daniel Brenes**	Queda pendiente búsqueda por menú
5	Llenando base de datos con hechos y reglas restantes	**5/10/2020**	**4h**	**Saúl Gómez**	-
6	Programado sintagma nominal y verbal	**6/10/2020**	**3.5h**	**Esteban Madrigal**	Sin problema alguno.
7	Terminando búsqueda por menú	**6/10/2020**	**2h**	**Daniel Brenes**	-
8	Testeando y terminado la búsqueda por hechos de restaurantes	**7/10/2020**	**4h**	**Daniel Brenes**	Sin errores aparentes
9	Creada validación de sintagma nominal y verbal	**7/10/2020**	**4.5h**	**Esteban Madrigal**	Falta testeo
10	Creada validación de datos ingresados	**9/10/2020**	**2h 3h**	**Esteban Madrigal**	-
11	Testeo y validación de casos de error en el ingreso de datos por consola	**10/10/2020**	**2h**	**Daniel Brenes**	Funciona dentro de los parámetros establecidos
12	Llenado de hechos verbales y nominales para la verificación del sintagma	**10/10/2020**	**3.5h**	**Daniel Brenes**	-
13	Definición de una regla para preguntarle al usuario si desea volver a buscar	**10/10/2020**	**3h**	**Esteban Madrigal**	Funciona correctamente Sin errores
14	Testeada la validación del sintagma	**11/10/2020**	**1h**	**Esteban Madrigal**	Funciona correctamente
15	Testeo general	**12/10/2020**	**2h**	**Todos**	-