



## Instituto Tecnológico de Costa Rica

Lenguajes de Programación

TI-3404

Tarea Programada 2

Andrei Fuentes Leiva

## Integrantes

Alonso Jiménez

Javier Rojas

Fernando Mata

Edwin Fernández

Jose Andrés Hernández

## 1. Descripción del Problema.

Como parte de la evaluación del curso TI-3404, Lenguajes de Programación, se solicita a los estudiantes realizar una simulación de un lenguaje de programación perteneciente al paradigma lógico, en este caso, se solicita simular PROLOG. La simulación debe crearse basado en 4 lenguajes a escoger, entre ellos C, C++, Java y Python.

Independientemente del lenguaje a utilizar, se deben recrear funcionalidades básicas de este lenguaje de programación, tales como un modo de administración, el cual permitirá al usuario introducir hechos y reglas a una base de conocimientos, y un modo de consulta, el cual deberá cargar en memoria todos los hechos y reglas digitados, mediante el uso de estructuras de datos, que le permitan al usuario poder obtener un resultado booleano al final de la consulta.

Dentro de los aspectos técnicos de la simulación de PROLOG, se debe tomar en cuenta características como **Built-In Predicates** tales como **fail**, **write** y **nl**, la implementación de **backtracking**, y el análisis léxico y sintáctico, siguiendo la estructura original de como se representa un hecho y una regla en PROLOG.

## 2. Diseño del Programa.

Debido a las características que posee el lenguaje de programación Java, y el fácil manejo de objetos que permite, se escogió este lenguaje para realizar la tarea programada. Además, que por haber aprobado anteriormente el curso de Estructuras de Datos, y Java fue utilizado para desarrollar habilidades de programación en este curso, se planeó la reutilización de código creado de estructuras tales como las listas simples, listas dobles y arreglos, para poder cargar toda la información en memoria temporal.

Al utilizar Java, se decidió manejar las secciones más amplias del proyecto de forma separada, en diferentes clases; por ejemplo el scanner, el parser, la interfaz gráfica, y la solución son clases totalmente diferentes. La base de conocimiento se almacena en un arreglo de Strings, es por esto que utilizamos la interfaz gráfica, para simplificar el ingreso de datos por parte del usuario y así, conocer el tamaño del arreglo que se crea para la base de conocimientos, además para poseer un programa mas intuitivo.

Se definieron algunas características para establecer estándares para la aceptación de hechos, reglas, y la creación de predicados. Como parte de la especificación de la tarea, se debían seguir los mismos estándares de PROLOG como la utilización de caracteres como "-" y "." dentro de los strings a utilizar para la creación de la base de conocimientos.

En cuanto a los algoritmos utilizados para el ingreso de la base de conocimientos, se utiliza la interfaz gráfica para indicar el tamaño del arreglo donde se guardará la base de conocimientos, luego se envía a la clase del scanner donde se evalúa cada carácter que sea válido en los caracteres aceptados por el programa y posteriormente se envía el arreglo al parser para

que se valide la estructura. Si logra ser satisfactorio se envía el arreglo a la clase solución donde el motor de inferencia evalúa cada predicado con respecto a la base de datos y las variables unificadas en la tabla de valores que se utiliza mediante una lista. También se implementó el backtracking en caso de que algún predicado falle, elimina las unificaciones que se hicieron en ese predicado y se revisa en los otros que unifiquen.

Dentro de algunas decisiones tomadas mientras se desarrollaba el código de la tarea programada se puede mencionar un cambio de la estructura para representar la condición “,”, reemplazada por un “&”, esto debido a que se presentaba un error al momento de **tokenizar** e intentar separar los predicados mediante ese carácter, creando una pérdida de información.

### 3. Librerías.

No se utilizaron librerías externas para la realización de esta tarea programada. Para la interfaz gráfica de usuario, se utilizó la librería interna de Java, **Swing**, y para su diseño, un complemento perteneciente al IDE utilizado, en este caso, **Eclipse**.

## 4. Análisis de Resultados

En conclusión, se puede decir que se obtuvieron un 90% de los resultados esperados, logrando la mayoría de las funcionalidades solicitadas, sin embargo, se mencionan algunas complicaciones que se dieron:

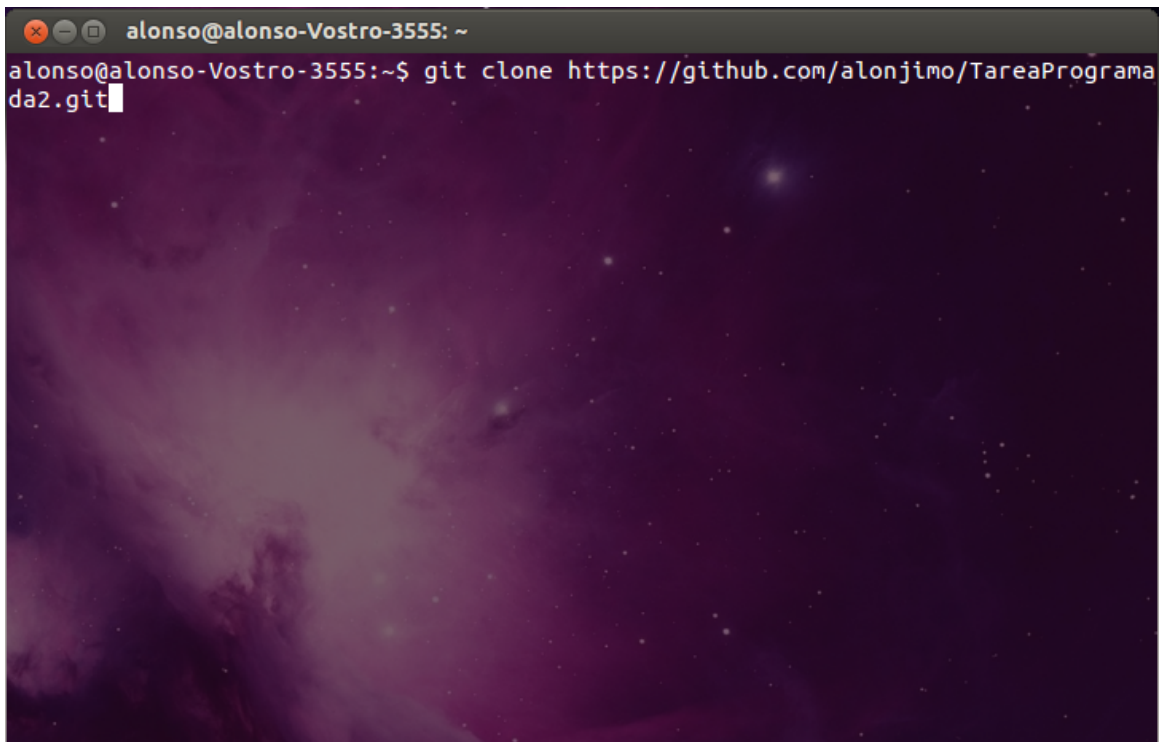
- ▲ **Backtracking del predicado Fail:** no se logró implementar un correcto funcionamiento de este predicado, sin embargo, mediante el uso de una bandera y la evaluación del predicado, si se reconoce que se está tratando con este predicado, se retorna **False**, en este caso, de forma bruta, sin el funcionamiento correcto mediante la evaluación por medio de backtracking.
- ▲ **Evaluación del Fail por el parser:** Se logró la evaluación de este predicado por el analizador léxico y sintáctico, sin embargo, si se evalúa un predicado que incluya fail por parte del parser, valida que todo el predicado es correcto, aunque el predicado no se encuentre sintácticamente correcto.

## 5. Manual de Usuario.

Para iniciar el programa, primero debe obtener desde el repositorio de GitHub.

Para poder descargar el programa se deben seguir los siguientes comandos desde la terminal. Luego de todo comando se debe digitar la tecla **ENTER**.

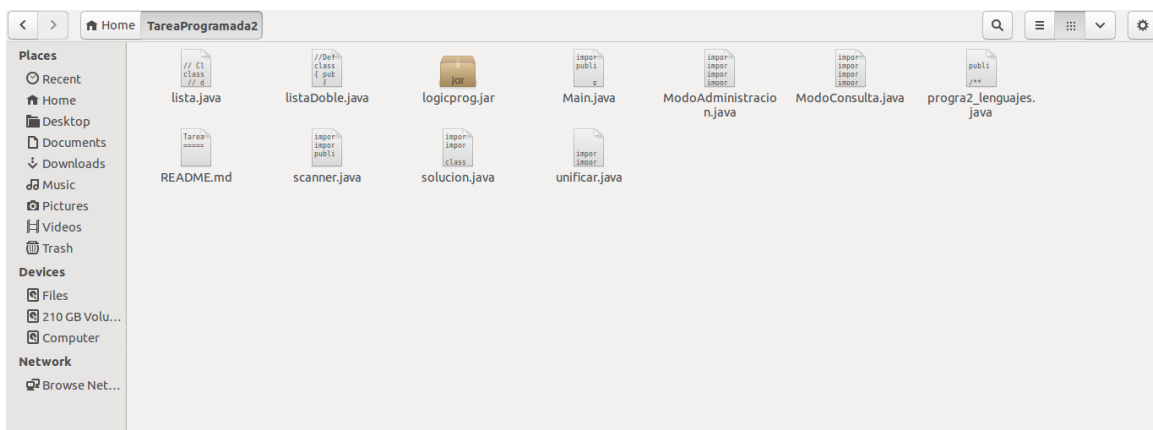
```
git clone https://github.com/alonjimo/TareaProgramada2.git
```



Al final debe retornar este resultado si el resultado es correcto.

```
alonso@alonso-Vostro-3555: ~  
alonso@alonso-Vostro-3555:~$ git clone https://github.com/alonjimo/TareaProgramada2.git  
Cloning into 'TareaProgramada2'...  
remote: Counting objects: 3, done.  
remote: Total 3 (delta 0), reused 0 (delta 0)  
Unpacking objects: 100% (3/3), done.  
alonso@alonso-Vostro-3555:~$
```

Luego se deben dirigir a la carpeta en donde se encuentra el proyecto descargado. Ingresan a la carpeta **TareaProgramada2** y ejecutan el archivo.jar para iniciar el programa.



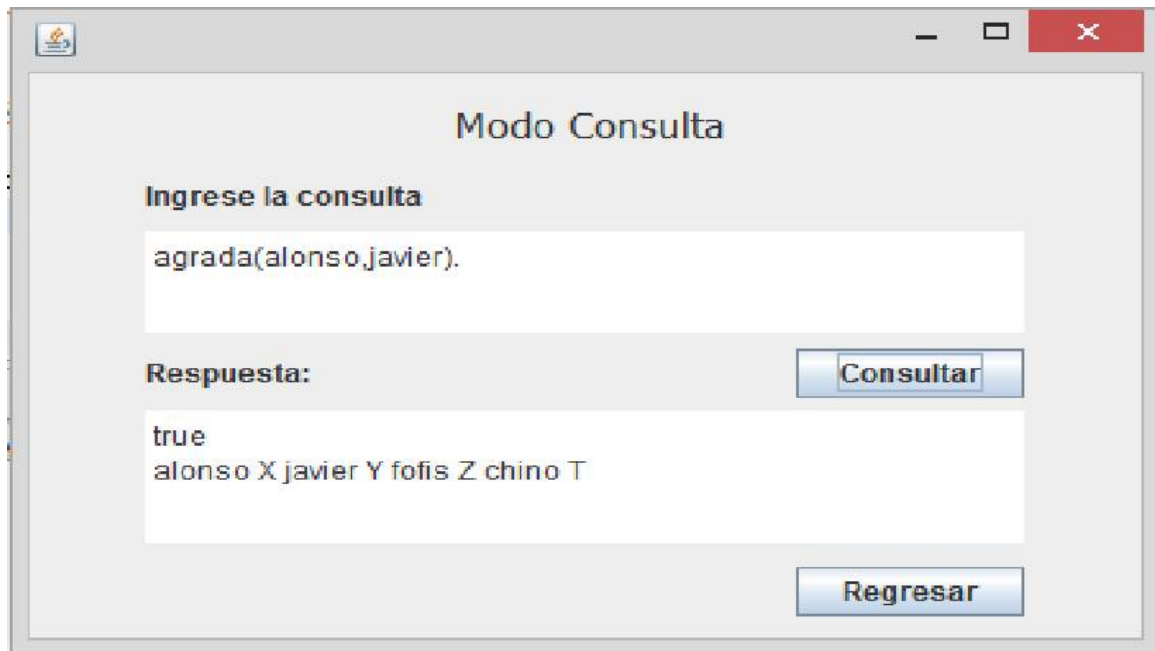
Al ejecutar el archivo logicprog.jar se iniciará el programa en modo administración.



En esta ventana podrá incluir todos los hechos y reglas en la base de conocimientos, una vez incluidos, se hace click a consulta, para poder iniciar el proceso de inferencia.



Por último, se muestra el resultado **True** o **False**



The image shows a Java Swing window titled "Modo Consulta". Inside the window, there is a label "Ingrese la consulta" above a text input field containing the query "agrada(alonso,javier).". Below the input field, there is a label "Respuesta:" followed by a text area displaying the result "true" and "alonso X javier Y fofis Z chino T". To the right of the "Respuesta:" label is a button labeled "Consultar". At the bottom right of the window is a button labeled "Regresar".

Consideramos que la tarea programada no solamente ha puesto a prueba nuestras cualidades como programadores y estudiantes de una carrera que vive y vivirá de la tecnología, si no que tambien ha puesto a prueba nuestras cualidad interpesonales, al vernos realmente en la necesidad de trabajar como un verdadero equipo y poder sacar adelante el proyecto, y llegar a la meta satisfactoria.