**UNIVERSIDAD SIMÓN BOLÍVAR**

**DEPARTAMENTO DE COMPUTACIÓN Y TECNOLOGÍA DE LA INFORMACIÓN**

**CI2693 – LABORATORIO DE LENGUAJES DE PROGRAMACIÓN I**

**Trimestre:** Abril - Julio 2022

**Profesor:** Fernando Torre

**Informe del Proyecto #2**

**Estudiantes:**

José González 15-10627

Ana Santos. 17-10602

Sartenejas, Julio del 2022

**Introducción**

El Curso CI-3661 Laboratorio de Lenguajes de Programación propone estudiar la programación lógica. Prolog es un lenguaje lógico creado en la década de los 70`s, su nombre es una referencia al paradigma lógico que determina lo que debe hacerse como consecuencia de características específicas, es lugar del cómo hacerlo.

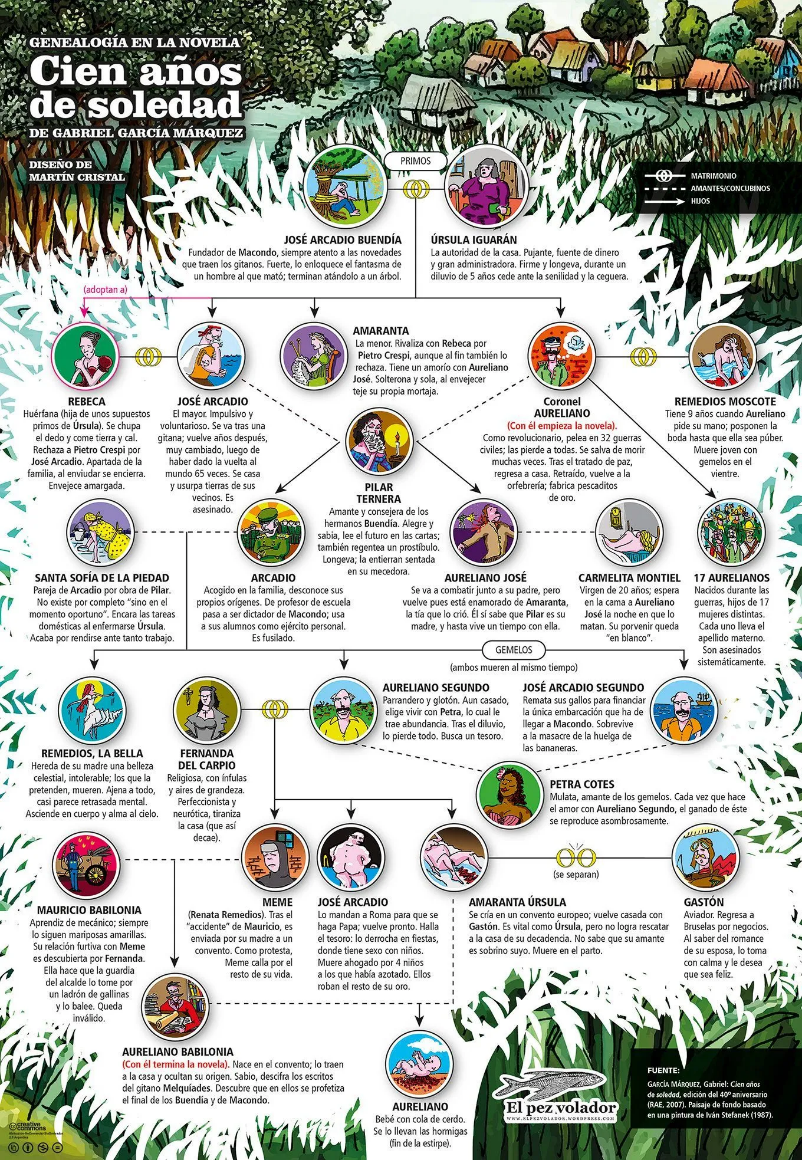
Por ello, se les propone a los estudiantes de Ingeniería de la Computación la representación o esquematización de un ***Árbol genealógico*** de alguna familia descrita en la literatura, cine, juegos, entre otros. El objetivo específico de este proyecto es lograr definir hechos relativos a los parentescos familiares a partir de hechos básico como el género y parentescos básicos, tomando en cuenta la necesidad de demostrar que las consultas relacionadas sobre relaciones filiales de los fans de personajes ficticios se pueden resolver sencillamente por la comunidad de software libre.

Para un mayor entendimiento de la implementación de un ***Árbol genealógico*** a través de la programación lógica o declarativa, se mostrarán a continuación la familia escogida y sus características, las decisiones en el planteamiento en cuanto a los nombres de los personajes y la adición de predicados especiales, ejemplificación de llamada y especificaciones de la implementación.

1. **Detalles de la Familia Ficticia:**
   1. Familia escogida y sus características:

La familia escogida para la realización de este proyecto es la *Familia Buendía* sacada de la obra literaria *Cien años de Soledad* de Gabriel García Márquez. La escogencia de esta viene dada porque es una familia muy conocida, ofrece parentescos filiales sencillos y es lo suficientemente enrevesada por su relaciones extramaritales e incestuosas para comprobar el buen funcionamiento de las declaraciones del programa con sus 25 miembros genealógicos.

* 1. Árbol genealógico:



* 1. Predicados especiales y aclaraciones de los nombres:

Como se puede ver en el árbol genealógico anterior, la relación de amantes juega un papel fundamental en la procreación de descendientes en esta familia. Por esta razón, los autores de este proyecto han decidido agregar el hecho especial *elAmante* y el predicado especial *laAmante*. La funcionalidad de estos últimos viene dada por la misma interacción que *esposo* y *esposa.*

Por otro lado, se quiere precisar la relación entre los nombres en el árbol genealógico y su representación como constantes en el código:

* José Arcadio Buendía (primer patriarca) como *joseArcadioBuendia.*
* Úrsula Iguarán como *ursula.*
* Rebeca como *rebeca.*
* José Arcadio (hijo de José Arcadio Buendía y Úrsula Iguarán) como *joseArcadio.*
* Amaranta como *amaranta.*
* Coronel Aureliano como *coronelAureliano.*
* Remedios Moscote como *remediosMoscote.*
* Pilar Ternera como *pilar.*
* Santa Sofía de la Piedad como *santaSofia.*
* Arcadio como *arcadio.*
* Aureliano José como *aurelianoJose.*
* Carmelita Montiel como *carmelita.*
* Los 17 Aurelianos (procreados en las guerras) como *aurelianos17.*
* Remedios La Bella como *remediosLaBella.*
* Fernanda del Carpio como *fernanda.*
* Aureliano Segundo como *aurelianoII.*
* José Arcadio Segundo como *joseArcadioII.*
* Petra Cotes como *petra.*
* Mauricio Babilonia como *mauricio.*
* Meme como *meme.*
* José Arcadio (hijo de Fernanda del Carpio y Aureliano Segundo) como *joseArcadioIII.*
* Amaranta Úrsula como *amarantaUrsula.*
* Gastón como *gaston.*
* Aureliano Babilonia como *aurelianoBabilonia.*
* Aureliano como *aureliano.*

1. **Decisiones de diseño:**
   1. Diseño general de la solución:

Primeramente, para la implementación de este proyecto se definieron los hechos o relaciones básicas, *persona*, *padre, madre, esposo* y *elAmante*, tomando los nombres como constantes y la definición del género de cada uno con masc para masculino y fem para femenino. El primer hecho define el género, mientras que los siguientes tres establecen las relaciones respectivas relaciones familiares; por ejemplo, sin perder generalidad *padre(X,Y)* indica que X es el padre de Y.

A partir de estos hechos, se definen las relaciones *esposa, laAmante, hermano, hermana, abuelo, abuela, hijo, hija, nieto, nieta, tio, tia, sobrina, sobrino, cuñado, cuñada, suegro* y *suegra.* Tomando X y Y como las variables para establecer parentesco, se define:

* X es esposa de Y, si Y es el esposo de X.
* X es la amante de Y, si Y es el amante de X.
* X es el hermano de Y, si X es masculino y X tiene el mismo padre o madre que Y. Respectivamente, funciona para hermana siendo X femenina.
* X es el abuelo de Y, si X es masculino y X es el padre del padre o la madre de Y. Respectivamente, funciona para abuela siendo X femenina.
* X es el hijo de Y, si X es masculino y Y es el padre o la madre de X. Respectivamente, funciona para hija siendo X femenina.
* X es el tío de Y, si X es el hermano de padre o madre de Y. Respectivamente, funciona para tía siendo X la hermana.
* X es el sobrino de Y, si X es el hijo del hermano o hermana de Y. Respectivamente, función para sobrina siendo X la hija.
* X es el cuñado de Y, si X es el esposo de hermano o hermana de Y, o si X es el hermano de la esposa o esposo de Y. Respectivamente, funciona para cuñada siendo X la esposa o la hermana.
* X es el suegro de Y, si X es el padre del esposo o esposa de Y. Respectivamente, funciona para suegra siendo X la madre.
  1. Organización del código y representación escogida:

El código se organizó en un único archivo llamado *familias.pro o familias.pl* (El desarrollo de este proyecto fue realizado en Windows y el SWI-Prolog no soporta de manera adecuado los archivos con extensión .pro por lo tanto decidimos agregar ambos archivos, lo cuales tienen el mismo contenido de manera que en caso de no poder ejecutar uno, se pueda ejecutar el otro). La separación interna del código se realizó con sus debidos comentarios, primero se plantearon los hechos de la siguiente manera:

* Todos los personajes de la novela con su respectivo género.
* Todos los padres de la novela.
* Todas las madres de la novela.
* Todos los esposos de la novela.
* Todos los amantes de género masculino de la novela.

Finalmente, se plantearon los predicados en el siguiente orden, considerando que algunos serían necesarios para el codeo de los próximos predicados:

* El predicado de la relación filial esposa.
* El predicado de la relación filial amante de género femenino.
* Los predicados de las relaciones filiales hermano y hermana.
* Los predicados de las relaciones filiales abuelo y abuela.
* Los predicados de las relaciones filiales hijo y hija.
* Los predicados de las relaciones filiales nieto y nieta.
* Los predicados de las relaciones filiales hermano y hermana.
* Los predicados de las relaciones filiales tío y tía.
* Los predicados de las relaciones filiales sobrino y sobrina.
* Los predicados de las relaciones filiales cuñado y cuñada.
* Los predicados de las relaciones filiales suegro y suegra.

Otra preciso importante, es sobre la representación escogida para los datos, la cual se implementó como átomos. Esta decisión estuvo basada en que los átomos es la manera más sencilla de representar una persona y al considerar que la consultas se harán de manera regular en Prolog, no se consideró necesario el uso de strings o listas para la representación de cada miembro familiar.

* 1. Llamadas ejemplificadas:

1. **Dificultades en la implementación:**

Durante la implementación del proyecto en Prolog, se encontró como dificultad que al no tener un IDE convencional con extensiones que ayuden al momento de escribir texto; sin embargo, es agradable el hecho de que la consola de ejecución tenga incluidos el modo *debug* y *traced.*

Uno de los problemas enfrentado durante la implementación del proyecto, fue al consultar por el predicado *hermano(X,Y)*, se obtenía como resultado que X y Y podían ser iguales, es decir, que un personaje era hermano de sí mismo. Para solucionar este problema, se agregó a la condición del predicado que X debía ser distinto de Y, eliminando de esta manera que considerara casos como que X podía ser hermano o hermana de Y.

**Conclusiones y Recomendaciones**

Representar un árbol genealógico a partir de hechos lógicos fue la manera en la que los autores de este proyecto, encontraron para estudiar la programación lógica con Prolog, escogiendo de esta manerauna familia muy conocida en el mundo de la Literatura del Realismo Mágico. La *familia Buendía* de la obra literaria *Cien años de soledad* de Gabriel García Márquez posee alrededor de 25 miembros y sus parentescos son lo suficientemente enrevesados entre ellos para poner a prueba la comprobación de los hechos y predicados de relaciones filiales en este lenguaje lógico.

Es importante, destacar que el proyecto para la definición de los predicados, tomo en cuenta sólo los parentescos sanguíneos para los parentescos de tíos, tías, sobrinos y sobrinas. Sería interesante para próximos proyectos, ampliar los hechos relativos que se quieren estudiar para representar parentescos políticos, además, agregar relaciones como primos, yernos, entre otros y diferenciar hermanos directos de hermanastros o madres de madrastras, por ejemplo.

Finalmente, se puede concluir que la representación genealógica es clave para el entendimiento y compresión de la programación lógica con lenguajes como Prolog, ampliando, gracias a esto, el conocimiento de los estudiantes de Ingeniería de la Computación y mostrando claramente que este tipo de consultas se puede dejar a manos de la comunidad de software libre.

**Referencias**

*Infobae*. (31 de Mayo de 2017). Recuperado el 29 de Julio de 2022, de Cultura: https://s3.amazonaws.com/arc-wordpress-client-uploads/infobae-wp/wp-content/uploads/2017/05/31165850/garcia-marquez-arbol-Cien-anos-de-soledad.jpg

*SeabrookeWindows*. (s.f.). Recuperado el 27 de Julio de 2022, de Tecnología: https://www.seabrookewindows.com/BQqNE3nMK/