

UNIVERSIDAD SANTIAGO DE CHILE



Laboratorio de Paradigmas de Programación
Laboratorio N° 2: "Lenguaje Prolog: Sistema de Archivos"

Integrantes:

Claudio Hernández Hernández

Profesor:

Edmund Leiva Lobos

1. Introducción

El presente documento de carácter analítico tiene como objetivo que el estudiante logre realizar una explicación y resolución al problema presentado en el segundo laboratorio de Paradigmas de Programación. Para ello se debe realizar una breve descripción tanto del problema como del paradigma y conceptos que se usen en este proyecto, realizar un análisis de esto, donde se debe especificar los requisitos a cumplir, diseñar la solución explicando el enfoque que se decide tomar para resolver el problema, aspectos implementados para la resolución del mismo, ejemplos de instrucciones funcionales del programa, un resumen de los resultados que se obtienen, donde se explica cuáles fueron exitosas y cuáles no, y argumentar por qué fue así, y por último, un cierre del trabajo hablando de las limitaciones, dificultades del paradigma, y una comparativa respecto al laboratorio trabajado anteriormente.

2. Descripción del Problema

En esta ocasión se debe realizar un Sistema de Archivos usando el Paradigma Lógico en el lenguaje de programación Prolog. Este sistema debe permitir una gran cantidad de funcionalidades, tales como registrar e iniciar sesión con algún usuario, la modificación de archivos y carpetas, restaurar elementos que hayan sido eliminados en la papelera de reciclaje, entre otras funciones que se mencionarán a lo largo del informe.

Por otra parte, hay requisitos que se deben cumplir a lo largo del trabajo, siendo algunos obligatorios para tener una evaluación mayor a la nota mínima, y otros opcionales. Un ejemplo de estos requerimientos obligatorios es el de realizar una autoevaluación de los requerimientos funcionales que son solicitados, usar una versión de SWI-prolog 8.4 o superior, entre otras.

3. Descripción Paradigma

Como se hizo mención en apartados anteriores, el paradigma que se debe implementar en este laboratorio es el de programación lógica. La programación lógica estudia el uso de la lógica tanto para plantear problemas, como para el control sobre las reglas que permiten concluir con una solución aritmética. Esta programación, al igual que la funcional, forman parte de la Programación Declarativa, es decir, resolver problemas mediante sentencias. En la Programación Lógica se ejerce de manera descriptiva, estableciendo relaciones entre entidades para indicar el que se debe hacer.

En el Lenguaje de Programación Prolog el código se compone de las siguientes partes:

1. Una Base de Conocimientos.
2. Términos que conforman el lenguaje.
3. Cláusulas que relacionan términos.
4. Hechos que permiten declarar valores verdaderos.
5. Reglas, las cuales son cláusulas que definen una verdad.

4. Análisis del Problema

Los requisitos solicitados en esta ocasión se pueden fragmentar en dos tipos:

1. Requerimientos No Funcionales.
2. Requerimientos Funcionales

En los Requerimientos No Funcionales hay 4 apartados que son obligatorios y 5 apartados opcionales. En los Requerimientos No Funcionales obligatorios tenemos:

1. Autoevaluación: como su nombre dice, se debe realizar una autoevaluación de los requerimientos funcionales solicitados.
2. Lenguaje: Se pide usar el lenguaje de programación Prolog en base a una programación mayormente declarativa.
3. Versión: Usar SWI-prolog versión 8.4 o superior.

4. Standard: No usar bibliotecas externas.

En cambio, en los Requerimientos No Funcionales opcionales se poseen:

1. Documentacion: Realizar comentarios en el codigo que se realice.
2. Organizacion: Realizar los TDA en archivos independientes.
3. Historial: Commits en Github subidos de manera constante.
4. Script de pruebas: Archivo que documente el funcionamiento del programa.

Por otra parte, en los requerimientos funcionales se tienen:

1. TDAs: Realizar una debida implementacion de los TDA pedidos, respetando la estructura dada.

Seguido de ello se muestran todos los TDAs que deben ser implementados, cada uno con su nombre de predicado, sus prerrequisitos y requisitos, su dominio y un ejemplo de como usarlo.

5. Diseño de la Solución

Antes de iniciar la resolucion del problema, se decie hacer una revision para estar mas preparado a la hora de afrontar el laboratorio, por lo que se hizo un estudio del paradigma logico con los recursos que contiene uvirtual, y una lectura tanto de el tutorial de swi-prolog online y algunos enlaces externos que contiene.

Cuando ya se habia realizado dicho repaso, se da inicio a la realizacion del proyecto. Lo primero que se decide hacer fue el escribir una base para todos los TDAs que se piden, escribiendo su dominio, recorrido, metas y una clausula que solo hace referencia al predicado que se debe implementar.

6. Aspectos de Implementación

7. Instrucciones de Uso

8. Resultado y Autoevaluación

9. Conclusiones del Trabajo