

Paradigmas de la Programación

Informe Laboratorio nro. 1

Paradigma Funcional

*Profesor: Gonzalo Martínez V.*

*Alumna: Karina Bustamante H.*

30 de junio de 2023

Tabla de contenido

[Introducción 2](#_Toc134312314)

[Descripción del problema 2](#_Toc134312315)

[Descripción del paradigma 3](#_Toc134312316)

[Análisis del problema 3](#_Toc134312317)

[Diseño de la solución 6](#_Toc134312318)

[Consideraciones de implementación 6](#_Toc134312319)

[Instrucciones de uso 7](#_Toc134312320)

[Resultados y evaluación 8](#_Toc134312321)

[Conclusiones 9](#_Toc134312322)

[Referencias 10](#_Toc134312323)

# Introducción

En el presente trabajo, se aborda el desarrollo del laboratorio Nº1 de la asignatura de Paradigmas de la Programación. En este proyecto, se nos ha encomendado la tarea de implementar un algoritmo en el lenguaje de programación Scheme, el cual se basa en la programación declarativa - funcional. El objetivo principal consiste en crear un simulador de sistema operativo que se centre específicamente en un sistema de archivos simplificado y los comandos necesarios para operar en él.

Para llevar a cabo esta implementación, hemos utilizado el lenguaje de programación Scheme a través de la aplicación DrRacket en su versión 8.8. Con esta herramienta, buscamos proporcionar una solución integral a los requerimientos funcionales planteados. Nuestra propuesta de solución permitirá al usuario realizar diversas acciones sobre este sistema de archivos ficticio, como crear, buscar, listar, entre otras.

El desarrollo de este proyecto no solo nos permitirá adquirir conocimientos prácticos sobre la programación en Scheme, sino también explorar los fundamentos de la programación declarativa - funcional y su aplicación en el contexto de un sistema operativo simplificado. A través de este trabajo, esperamos demostrar nuestra capacidad para diseñar algoritmos eficientes y crear soluciones robustas en el ámbito de la programación paradigmática.

A continuación se presenta en detalle el enunciado del problema, la metodología utilizada, la estructura del programa implementado y los resultados obtenidos.

# Descripción del paradigma

El paradigma funcional se basa en la composición de funciones que se fundamentan en principios de funciones matemáticas puras. En este enfoque, una función se considera determinista, lo que implica que siempre producirá el mismo resultado cuando se le proporcionen los mismos argumentos. Además, se enfoca en la evaluación de expresiones y la transformación de datos, evitando efectos secundarios y dependencia del estado global del programa. En su lugar, una función debe depender únicamente de sus entradas y generar una salida basada en ellas.

Podríamos establecer una analogía entre el paradigma funcional y el álgebra, donde las ecuaciones pueden ser reemplazadas por sus resultados, lo que se conoce como transparencia referencial. En lugar de utilizar bucles, se recurre a la definición de funciones recursivas, que se refieren a una función expresada en función de sí misma. Este enfoque permite resolver problemas dividiéndolos en subproblemas más pequeños.

La inmutabilidad de los datos es otra característica fundamental de la programación funcional. Esto implica que una vez que se ha creado un valor, no se puede modificar. En lugar de modificar los datos existentes, se generan nuevas estructuras de datos a partir de las existentes mediante operaciones de transformación.

Dentro de las técnicas de programación funcional, se destacan conceptos como el cálculo lambda, que utiliza la notación prefija donde el operador precede a los operandos. Asimismo, se emplean funciones de alto orden, que permiten que las funciones se utilicen como argumentos de entrada o se devuelvan como resultados. La currificación es otra técnica relevante, que consiste en transformar una función de varios argumentos en una secuencia de funciones de un solo argumento, lo que facilita la evaluación de la función paso a paso.

# 

# Descripción del problema

El objetivo de este proyecto es desarrollar la simulación de un sistema de archivos utilizando el lenguaje de programación Scheme. Un sistema de archivos es una estructura jerárquica utilizada para

# Análisis del problema

El problema principal se centra en el diseño y la implementación de las reglas y hechos necesarios para simular un sistema de archivos eficiente y funcional en Prolog. Se deben tener en cuenta las siguientes consideraciones clave durante el análisis del problema:

***Jerarquía de directorios:*** El sistema de archivos debe permitir la creación y gestión de una estructura jerárquica de directorios. Cada directorio puede contener archivos y subdirectorios, y el sistema debe permitir la navegación fluida a través de esta estructura jerárquica.

***Creación y eliminación de archivos y directorios***: El sistema debe proporcionar la capacidad de crear y eliminar tanto archivos como directorios. Al crear un archivo, se debe especificar su nombre y ubicación en el sistema de archivos. Al eliminar un archivo o directorio, se deben eliminar todos sus contenidos y actualizar la estructura del sistema de archivos en consecuencia.

# Diseño de la solución

# El sistema de archivos para este laboratorio se diseñará e implementará utilizando una combinación de hechos y reglas en Prolog. A continuación, se detallan los principales elementos del diseño y la implementación:

# Consideraciones de implementación

# La implementación de este laboratorio fue realizada a través del entorno virtual de prolog <https://swish.swi-prolog.org/> por ser desarrollado en un equipo computacional con sistema operativo macOS Monterrey. Por lo anterior se recomienda la utilización de este entorno virtual para realizar las pruebas de funcionamiento del código entregado.

# Instrucciones de uso

Para la ejecución de los algoritmos propuestos, debe abrir el archivo lab2\_13452929\_Bustamante.pl, seleccionar todo el contenido, copiar y pegar en el editor virtual de prolog, <https://swish.swi-prolog.org/>

Para hacer las pruebas de funcionamiento debe ingresar en el apartado para las consultas del intérprete de prolog, lo que se encuentra en el contenido del archivo, pruebas\_13452929\_Bustamante.pl

Para crear unidades de disco se debe considerar escribir la letra del disco junto con “:”, para una mejor visualización.

# Resultados y evaluación

A continuación se muestra la autoevaluación de los requerimientos funcionales requeridos por el laboratorio, y el grado de logro alcanzado para cada uno de ellos.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RF Nº | Requerimiento funcional | Grado de alcance | Tipo o cantidad de pruebas | Éxitos | Fracasos | Razones de fallo |
| 1 | TDAs | 1 |  |  |  |  |
| 2 | system - constructor | 1 | 2 | 2 | 0 |  |
| 3 | system - addDrive | 1 | 2, uni. creadas | 2 | 0 |  |
| 4 | system - register | 1 | 3, si no existe usuario, se crea | 3 | 0 |  |
| 5 | system - login | 1 | 2 | 1 | 1 | Usuario no existe en lista de usuarios |
| 6 | system - logout | 1 | 2 | 2 |  |  |
| 7 | system - switch-drive | 1 | 2 | 1 | 1 | Unidad no existe en lista de unidades |
| 8 | system - mkdir | 0,75 | 4 | 1 | 3 | No asignaba bien los predicados. |
| 9 | system- cd |  |  |  |  |  |
| 10 | system - add-file | 0,05 | 3 | 1 | 2 | No logra mostrar agregar el archivo a pesar que en la ejecución paso a paso se ve que los agrega a la lista. |
| 11 | system - del |  |  |  |  |  |
| 12 | system - copy |  |  |  |  |  |
| 13 | system - move |  |  |  |  |  |
| 14 | system - ren |  |  |  |  |  |

**Tabla 1.-** Muestra los requerimientos funcionales y el nivel de logro.

# 

# Conclusiones

En este proyecto de laboratorio 2, de la asignatura de paradigmas de la programación, se ha logrado desarrollar con éxito parte de la simulación de un sistema de archivos utilizando el lenguaje de programación Prolog. A través del diseño y la implementación de reglas y hechos, se ha creado una herramienta para gestionar la organización, manipulación de archivos, unidades de disco y directorios.

En resumen, el paradigma funcional se basa en la composición de funciones matemáticas puras, evitando efectos secundarios y dependencia del estado global. Promueve la transparencia referencial, la inmutabilidad de los datos y el uso de técnicas como el cálculo lambda, funciones de alto orden y la currificación. Al adoptar este enfoque, los programadores pueden crear sistemas más seguros, modulares y expresivos, permitiendo una mayor facilidad para razonar sobre el comportamiento del programa.

# 

# Referencias

* Swi-prolog, Reference manual. https://www.swi-prolog.org/pldoc/doc\_for?object=manual
* Campus Virtual, (2023). Paradigmas de Programación. https://uvirtual.usach.cl/moodle/