

Paradigmas de la Programación

Informe Laboratorio nro. 1

Paradigma Funcional

*Profesor: Gonzalo Martínez.*

*Alumna: Karina Bustamante H.*

10 octubre de 2023

[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

La asignatura Paradigmas de la Programación busca enseñar los diferentes paradigmas que existen frente a forma de programar. El presente informe, correspondiente al laboratorio Nº1 de la asignatura, solicita realizar una implementación basándonos en la programación declarativa – funcional, de un algoritmo para resolver el problema que corresponde al siguiente enunciado:

***“Crear un simulador de chatbots simplificado y los comandos que permitan operar sobre éste.”***

En base a lo solicitado y aplicando conceptos del paradigma de programación funcional, usaremos el lenguaje de programación Scheme en la resolución de este problema acotado.

# Descripción del paradigma

La programación funcional corresponde al paradigma donde la programación se asemeja al álgebra ya que podemos reemplazar las ecuaciones por sus resultados, es decir, si llamamos a una función con los mismos argumentos siempre obtendremos los mismos resultados, esto se conoce como ***Referential*** ***Transparency***. La programación funcional se centra en el uso de funciones y la composición de funciones para resolver problemas evitando los cambios de estados. Otros conceptos importantes de la programación funcional son:

* ***Inmutabilidad***: Los datos son inmutables, una vez creados, no se pueden modificar, se crean nuevas estructuras de datos cuando se realizan operaciones.
* ***Funciones Puras***: Una función crea otro estado basado en la estructura que recibe como parámetro, no existen estados intermedios, las salidas solo dependen delas entradas.
* ***Recursividad***: Se fomenta el uso de recursión
* ***Cálculo Lambda***: Es un símbolo que toma argumentos y que retorna una salida, donde estas salidas solo dependen de las entradas.

# Descripción del problema

# Análisis del problema

# Diseño de la solución

Unix Timestamp, cuenta los segundos que han transcurridos desde el 1 de enero de 1970 00.00 hrs. Que es el nacimiento de Unix -> año 0

;Fecha (current-seconds)

# Consideraciones de implementación

# Instrucciones de uso

# 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 | TDA’s |  |  |  |  |  |
| 2 | TDA Option - Constructor |  |  |  |  | Autoincremental |
| 3 | TDA Flow - Constructor |  |  |  |  |  |
| 4 | TDA Flow-Modificador |  |  |  |  | Verificar duplicidad |
| 5 | TDA Chatbot - Constructor |  |  |  |  | Id incremental, verificar duplicidad |
| 6 | TDA Chatbot - Modificador |  |  |  |  |  |
| 7 | TDA System - Constructor |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |

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

# 

# Conclusiones

# Referencias