

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 3](#_Toc134312318)

[Consideraciones de implementación 4](#_Toc134312319)

[Instrucciones de uso 4](#_Toc134312320)

[Resultados y evaluación 5](#_Toc134312321)

[Conclusiones 6](#_Toc134312322)

[Referencias 6](#_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, en donde se solicita realizar una implementación basándonos en la programación declarativa – funcional, de un algoritmo para resolver el problema que corresponde al siguiente enunciado:

***“Sistema para la creación, despliegue y administración de chatbots simplificado.”***

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

El propósito fundamental de este proyecto consiste en crear una simulación de un sistema de chatbot empleando el enfoque de la programación funcional y el lenguaje de programación Scheme. Este chatbot pertenece a la categoría de los ITR, lo que significa que está diseñado para ofrecer respuestas estructuradas a través de acciones específicas. Las opciones de interacción pueden variar desde el uso de palabras clave con sinónimos hasta una alternativa más simplificada mediante números o letras. Esto permite a los usuarios seleccionar sus preguntas de una lista predefinida.

# Análisis del problema

El problema que se intenta resolver corresponde a la creación de un sistema de chatbot del tipo ITR que ofrezca respuestas estructuradas mediante acciones específicas, que permita a los usuarios interactuar de manera estructurada, utilizando palabras clave, sinónimos, números o letras. Esta solución se limitará a una lista de preguntas que se podrá generar a través del mismo sistema.

# Diseño de la solución

Para el diseño de la solución se debe identificar los objetivos del chatbot, que tipo de preguntas responderá y cuales son las acciones especificas que debe realizar. Dentro de las acciones clave estará:

* ***Implementación en Scheme***: para implementar la lógica del chatbot. Se definirán las funciones y estructuras de datos necesarias para procesar las preguntas y proporcionar respuestas adecuadas.
* ***Manejo de Preguntas y Respuestas***: Se deberá desarrollar un mecanismo para que el chatbot entienda y procese las preguntas, creando estructuras de datos para mapear preguntas a respuestas o acciones específicas.
* ***Gestión palabras claves***: deberá realizar el reconocimiento exacto de palabras claves o números de opciones para realizar operaciones.

En cuanto al registro de la hora y fecha de creación, se ha utilizado el Unix Timestamp, que cuenta los segundos que han transcurridos desde el 1 de enero de 1970 desde las 00.00 hrs. que corresponde al nacimiento de Unix, este será el formato que utilizaremos para representar la fecha (current-seconds) en la solución.

# 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 | 0,75 |  |  |  | Autoincremental |
| 3 | TDA Flow - Constructor | 1 |  |  |  |  |
| 4 | TDA Flow-Modificador | 0,75 |  |  |  | Verificar duplicidad |
| 5 | TDA Chatbot - Constructor | 0,5 |  |  |  | Id incremental, verificar duplicidad |
| 6 | TDA Chatbot - Modificador | 0,5 |  |  |  | Verificar duplicidad |
| 7 | TDA System - Constructor | 1 |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |

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

# Conclusiones

# Referencias