**INFORME DE LABORATORIO 3: SIMULACIÓN DE UN PROGRAMA PARA REALIZAR UN TRATAMIENTO DE IMÁGENES SIMPLIFICADO**

Nombre: Aracely Castro V.

Profesor: Roberto Gonzales I.

Asignatura: Paradigmas de Programación (2/2022)

Índice

[1. INTRODUCCIÓN 2](#_Toc119155547)

[**1.1** **DESCRIPCIÓN DEL PROBLEMA** 2](#_Toc119155548)

[**1.2** **DESCRIPCIÓN DEL PARADIGMA** 3](#_Toc119155549)

[2. DESARROLLO 3](#_Toc119155550)

[**2.1** **ANÁLISIS DEL PROBLEMA** 3](#_Toc119155551)

[**2.2** **DISEÑO DE LA SOLUCIÓN** 3](#_Toc119155552)

[**2.3** **ASPECTOS DE IMPLEMENTACIÓN** 3](#_Toc119155553)

[**2.3.1** **EJEMPLOS DE USO** 3](#_Toc119155554)

[**2.3.2** **RESULTADOS ESPERADOS** 3](#_Toc119155555)

[**2.3.3** **POSIBLES ERRORES** 3](#_Toc119155556)

[**2.4** **RESULTADOS Y AUTOEVALUACIÓN** 3](#_Toc119155557)

[**2.4.1** **RESULTADOS** 3](#_Toc119155558)

[**2.4.2 AUTOEVALUACIÓN** 4](#_Toc119155559)

[3. CONCLUSIÓN 4](#_Toc119155560)

[4. BIBLIOGRAFÍA Y REFERENCIAS 4](#_Toc119155561)

[5. ANEXOS 4](#_Toc119155562)

# **INTRODUCCIÓN**

Con el objetivo de aplicar conceptos del paradigma de programación lógico usando el lenguaje de programación Prolog en la resolución de un tratamiento de imágenes simple, a continuación, se dará detalle sobre el proceso de desarrollo de la solución al problema a través del compilador SWI-Prolog versión 8.4.3. El informe constará de una breve introducción de cómo surgió el problema, una descripción del paradigma utilizado, el análisis del problema y como fue el diseño de las soluciones para algunos predicados, sus aspectos de implementación, las instrucciones necesarias para compilar el archivo con el script de pruebas junto con ejemplos de algunas consultas a los predicados, los resultados y la autoevaluación. Finalmente, se dará una conclusión respecto a todo lo anterior.

## **DESCRIPCIÓN DEL PROBLEMA**

Al igual que los laboratorios anteriores se busca desarrollar la simulación de un programa para el tratamiento de imágenes de forma simplificada como GIMP y Adobe Photoshop. Este simulador permitiría crear imágenes con distintos formatos de píxeles sobre los cuales se pueden aplicar distintas operaciones como, por ejemplo, recortar una imagen, invertir una imagen, obtener histograma de la imagen, entre otros a través de la consola. Para implementar esto, se debe de tener en cuenta que:

Una imagen pixmap-d es una imagen donde cada uno de sus píxeles contiene información de espacio de colores y profundidad (R)ed, (G)reen, (B)lue y (D)epth. Cada color cubre valores entre 0 y 255 formando en conjunto un color del espectro RGB. La profundidad, en cambio, ofrece información más detallada de la imagen como espacio tridimensional.

Una imagen bitmap-d es una imagen donde sus píxeles tiene asociado su color por un bit, negro como 0 y blanco como 1. Cada píxel también tendría una profundidad asociada.

Una imagen hexmap-d es similar a una imagen pixmap-d con la diferencia que los valores RGB son representados en hexadecimal con formato “#RRGGBB”.

## **DESCRIPCIÓN DEL PARADIGMA**

# **DESARROLLO**

## **ANÁLISIS DEL PROBLEMA**

## **DISEÑO DE LA SOLUCIÓN**

## **ASPECTOS DE IMPLEMENTACIÓN**

### **EJEMPLOS DE USO**

### **RESULTADOS ESPERADOS**

### **POSIBLES ERRORES**

## **RESULTADOS Y AUTOEVALUACIÓN**

### **RESULTADOS**

### **2.4.2 AUTOEVALUACIÓN**

La Autoevaluación se realiza de la siguiente forma: 0: No realizado – 0.25: Funciona 25% de las veces – 0.5: Funciona 50% de las veces 0.75: Funciona 75% de las veces – 1: Funciona 100% de las veces. Para ver la tabla de Autoevaluación, XXX

# **CONCLUSIÓN**

# **BIBLIOGRAFÍA Y REFERENCIAS**

# **ANEXOS**