Universidad de Santiago de Chile
Facultad de Ingeniería
Departamento de Ingeniería en Informática

Informe 3 de Laboratorio: Paradigma Orientado a Objetos

Nombre: Leonardo Espinoza Ortiz
Profesor: Miguel Truffa
Asignatura: Paradigmas de Programación
Fecha de entrega: 03-12-2022

Tabla de contenidos:

Introducción:

En el presente informe se presentará un laboratorio acerca de uno de los Paradigmas vistos en clases de Paradigmas de programación, el Paradigma orientado a Objetos.
Este Paradigma será practicado mediante el lenguaje de programación JAVA a través de un IDE a elección del estudiantado.
Este trabajo tendrá una estructura en donde se dará a conocer el problema a resolver, un análisis, diseños de la solución y finalmente una conclusión.

Descripción breve del problema:

Se pide implementar en el Lenguaje JAVA una aplicación de edición de imágenes similar a herramientas conocidas como GIMP y Adobe Photoshop.

Un software de edición o manipulación de imágenes digitales es aquel que permite a un usuario realizar distintas operaciones sobre éstas.
Existen distintos tipos de operaciones tales como rotar una imagen, invertirla, rotarla, retocarla, transformarla y redimensionarla, entre otras.
Este software se concentrará en trabajar en imágenes RGBD.

Descripción breve del paradigma:

El paradigma orientado a objetos, este se basa en la definición de objetos, es decir, abstracciones de entidades que tienen ciertas características como Atributos y Métodos.
Estos objetos pueden interactuar entre sí mediante métodos, etc

Uno de los lenguajes más utilizados en la actualidad bajo este paradigma es Java.
Este paradigma posee características importantes a la hora de programar, estos son los siguientes:
-Clases: Es la definición de las características que posee un objeto, las cuales pueden ser atributos y métodos. Es equivalente a un TDA.
-Objetos: Son representaciones de una clase.
-Atributos: Corresponde a las características de una clase(tal como dice su nombre).
-Métodos: Son las acciones que puede realizar un objeto. Es una implementación similar a las funciones.

Todo objeto debe tener un constructor, para crear dicho objeto.

Existen tipos de representaciones que facilitan la lectura y análisis de un código. Uno de los diagramas más utilizados son los Diagramas UML

Análisis del problema:

De forma particular, se consideran 3 tipos de representaciones para la implementación de las imágenes.
-bitmaps-d para imágenes donde cada pixel o pixbit puede tomar el valor 0 o 1
-pixmap-d para imágenes donde cada pixel o pixbit es una combinación de los valores para los canales R, G y B
-hexmaps-d para imágenes donde cada píxel o pixhex expresa la información del color del píxel a través de un valor único hexadecimal de 6 valores.
Los requerimientos para lograr completar la tarea son los siguientes:
Requerimientos obligatorios:
-Se debe realizar la solución de este problema bajo el paradigma orientado a objetos, en este caso se utilizará Java para la resolución de este.
-Todas las interacciones se deben realizar mediante consola.
-Se debe documentar el código indicando descripción de cada clase, atributos y métodos utilizados.
-El código debe estar organizado y seguir el Mantra: Bajo acoplamiento y Alta Cohesión.
-Se debe crear un Diagrama de Análisis (antes de desarrollar el problema) y un Diagrama de Diseño(una vez realizada la tarea) con sus respectivas relaciones entre clases, etc
Tras este análisis, se puede proceder a una estructura en el diseño de la solución que se espera crear.
-Se espera un buen desarrollo del código y uso de la herramienta Git para este.
Requerimientos funcionales:
-Se espera implementar de forma correcta clases y estructuras en el programa.
-Se debe crear un menú interactivo en el programa que permita interactuar con todas las funcionalidades del programa.

-Se debe crear un constructor de imagen utilizando una de las estructuras del Paradigma, Interface principalmente.
-También se debe implementar un constructor para los pixeles.
-Se deben crear 3 métodos que verifiquen cada tipo de imagen.
-Se debe crear un método que permita saber si una imagen se encuentra comprimida.
-Se debe crear un método que permita invertir una imagen de forma horizontal, y otro método que permita invertir de forma vertical.
-Se debe poder recortar la imagen.
-Se debe poder transformar la representación de una imagen RGB a una imagen HEX
-Se debe crear una representación de una imagen vía Histograma.
-Se debe poder rotar la imagen 90 grados hacia la derecha.
-Se debe poder comprimir una imagen.
-Se debe poder reemplazar un pixel en una imagen.
-Se debe poder crear una imagen e invertir los colores dentro de esta.
-Se debe poder invertir los colores RGB de una imagen RGB
-Se debe crear una imagen y representarla en String
-Se debe poder separar una imagen y dividirla en capas según su profundidad.
-Se debe poder descomprimir una imagen.

Diseño de la solución

Para construir un diseño para esta solución, se implementarán TDAs(tipos de datos abstractos) que deben poseer la estructura de Representación, Constructores, Predicados de Pertenencia, Selectores, Modificadores y Otros predicados.
Estos TDA servirán para una correcta y ordenada estructura de esta solución.

Estos son los TDA más importantes a considerar:
También considerar un TDA pixel para así organizar de mejor forma los pixeles.

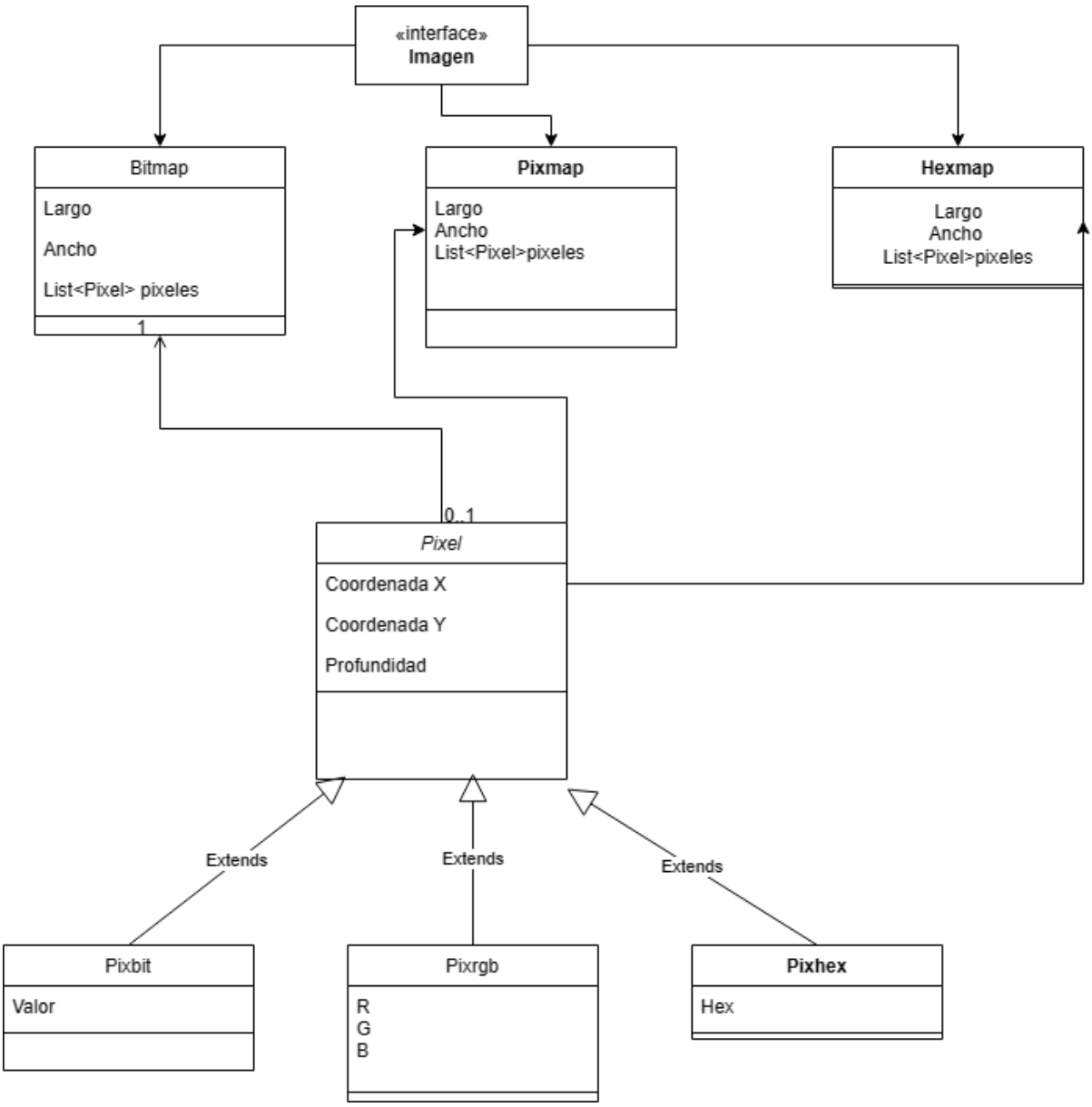

Figura 1: Diagrama de análisis de clases

En el diagrama de análisis, se tenía como idea utilizar una interface con funcionalidades en común e implementarlas en cada tipo de Imagen (Bitmap, Pixmap, Pixmap)

Pero después de desarrollar el código, se presentaron problemas de implementación y exceso de código.

Y al final se decidió implementar de forma más simplificada y eficaz el TDA Image, que se verá en el siguiente diagrama.

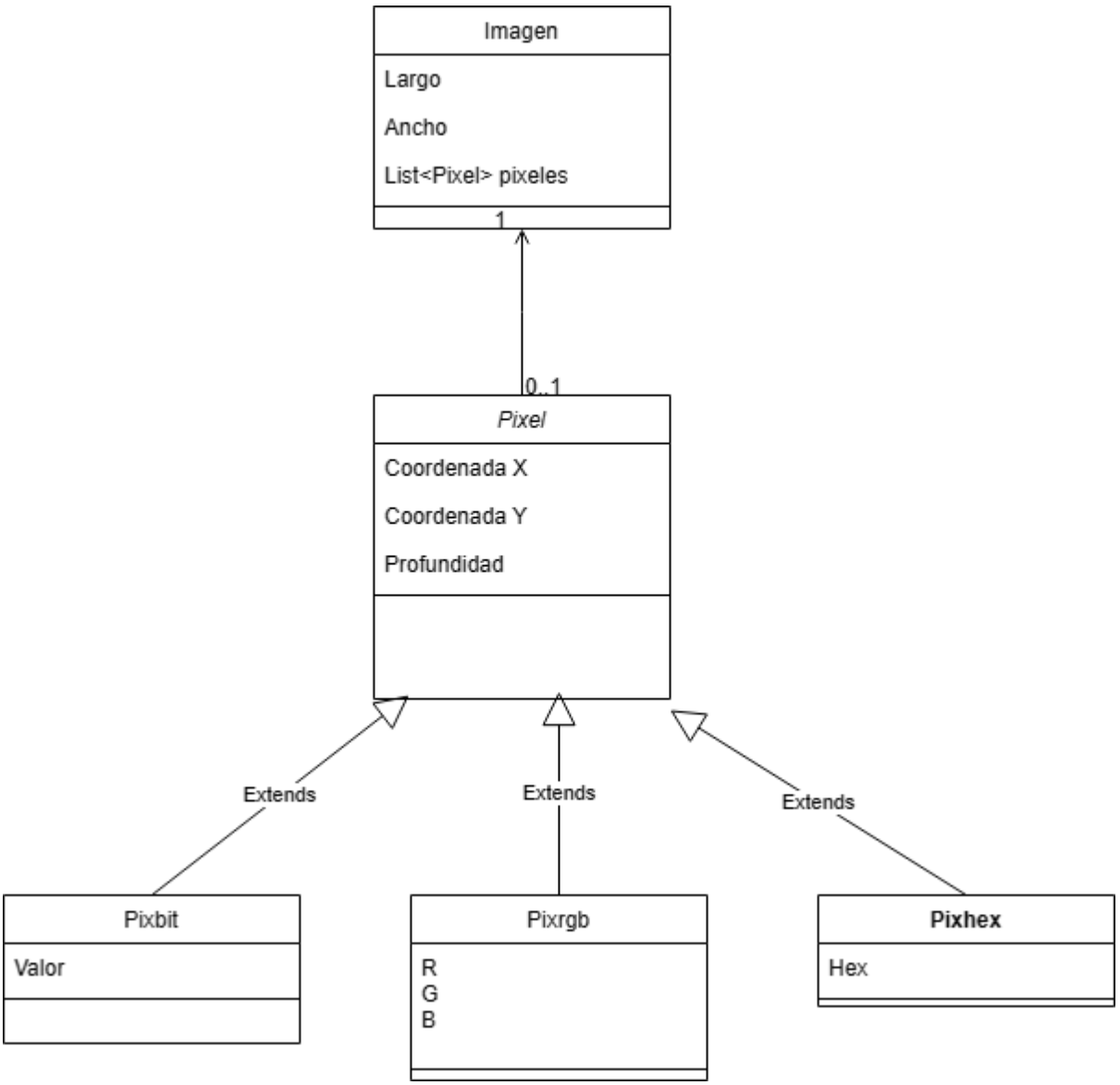

Figura 2: Diagrama de diseño

El diagrama de diseño es el final presentado para el código.

Este diagrama simplifica el uso de la clase Imagen, ya que se asume que Imagen y todas las imágenes pueden contener una lista de píxeles. Así que toda imagen comparte los mismos atributos.

Clases Implementadas:

-Imagen: Corresponde a la base del programa, es una clase que tiene como atributos Ancho, Largo y una lista de Pixeles.

En esta clase se implementan en su mayoría todas las funcionalidades.

-Pixel: Clase correspondiente a los pixeles. Esta clase posee Herencia, que se extiende a 3 clases más.
Pixbit, Pixrgb, Pixhex.
Pixel contiene la información general de un Pixel(Coordenadas, profundidad)
-Pixbit se añade el atributo de Valor donde puede ser 1 o 0
-Pixrgb se añaden los atributos de Red, Green y Blue, que incluye los valores de los 3 canales.
-Pixhex se añade el atributo String Hex que incluye un String.

Luego la clase Main unifica estas clases e interactúa con ellas.
Se crea un menú interactivo y una lista de Imágenes, en donde puedes almacenar las imágenes creadas, modificar alguna imagen con alguna funcionalidad y visualizar las imágenes ya almacenadas en la lista.

Aspectos de implementación

Para aspectos de implementación, se utilizó un IDE de Java, en este caso fue IntelliJ IDEA.

En el caso del JDK se utilizó la versión 11 para la compilación del programa.

El programa fue testeado y compilado en el Sistema Operativo de Windows 11.

Instrucciones de uso

Para utilizar el programa, se debe ejecutar el archivo Script.bat situado junto a los archivos del programa.
También se puede ejecutar vía CMD del sistema.
Una vez ejecutado, se abre el menú interactivo y se siguen las indicaciones que el usuario prefiera.

Resultados esperados

Se espera poder emular de forma consistente la representación de imágenes y el manipulador de estas.

Tras haber realizado el proyecto, se puede decir que si se lograron los resultados esperados, si bien faltaron implementar algunas funcionalidades, se logró implementar la interfaz del programa, crear la imagen (lo más importante), una estructura de los TDA y del proyecto y una buena navegación por el menú interactivo.
-Cada funcionalidad implementada se testeó con los 3 tipos de imágenes, también está implementada una imagen para hacer testeo de ella con las funcionalidades implementadas.
Si bien al inicio, hubo problemas de implementación de estrategias con la estructura de la imagen, se logró unificar todo gracias a una sola clase y resolver este problema.

Conclusiones del trabajo

Tras realizar este trabajo, se puede concluir que el paradigma orientado a objetos es una forma muy amigable a la hora de programar.
Estas preferencias se pueden ver ya que Java es uno de los lenguajes más usados en todo el mundo.
A diferencia del paradigma lógico, este paradigma OPP es mucho más intuitivo y se acerca más al "lenguaje humano".
En lo personal fue el paradigma que más me gustó y el más fácil de abordar a comparación de los otros lenguajes que se vieron en este proyecto.
Se logró una buena implementación y uso de este Lenguaje y Paradigma así que se puede concluir que se aprendió un lenguaje muy útil e interesante para su uso en el futuro como programador.

Bibliografía

Gonzales, R. (2022). "Proyecto Semestral de Laboratorio". Paradigmas de programación. Recuperado de:
2022_02 Laboratorio 3 - Documentos de Google