Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

GENERADOR Y EDITOR DE IMÁGENES

201901772 – Daniel Reginaldo Dubón Rodríguez

Resumen

El ensayo presenta el desarrollo de una aplicación que consiste en representar imágenes utilizando listas ortogonales (utilizada para representar matrices) y que permite realizar operaciones sobre estas mismas. Se diseño y se implementó una lista ortogonal para la lógica de manejo y almacenamiento de los datos que utiliza la aplicación, los datos que se le proveen a la aplicación para generar las imágenes vienen contenidos en un archivo con estructura XML, para la generación de las imágenes se utilizó el conjunto de herramientas de software para el diseño de diagramas (Graphviz). La solución e implementación de lo antes mencionado fue a través del uso de un lenguaje de programación (Python) para el manejo de la lógica y así crear los distintos algoritmos y estructuras requeridos para el funcionamiento, también se utilizó para la construcción de la interfaz gráfica que permite interactuar de una forma más agradable e intuitiva con la aplicación.

Palabras clave

Python, Lista ortogonal, Lenguaje de Marcado Extensible (XML), Programación Orientada a Objetos (POO), Graphviz.

Abstract

The essay presents the development of an application that consists of representing images using orthogonal lists (used to represent matrices) and that allows operations on them. An orthogonal list was designed and implemented for the logic of handling and storage of the data used by the application, the data that is provided to the application to generate the images is contained in a file with an XML structure, for the generation of the images. images was used the set of software tools for the design of diagrams (Graphviz). The solution and implementation of the aforementioned was through the use of a programming language (Python) to handle the logic and thus create the different algorithms and structures required for operation, it was also used for the construction of the graphical interface that allows you to interact in a more pleasant and intuitive way with the application.

Keywords

Python, Orthogonal list, Extensible Markup Language (XML), Objetct Oriented Programming (OOP), Graphviz.

Introducción

Se realizo una aplicación que consiste en representar imágenes utilizando listas ortogonales y que permite realizar operaciones sobre estas mismas. Se realizó a través del lenguaje de programación Python para construir los diferentes algoritmos y estructuras para poder desarrollar la aplicación, también se implementó una interfaz gráfica para una poder interactuar de una mejor forma con la aplicación, así mismo se hizo uso del paradigma de programación orientada a objetos para abstraer el problema y lograr plasmarlo en el lenguaje de programación, también se usaron tipos de datos abstractos para crear una estructura de datos en este caso particular so opto por una lista ortogonal, en la cual se almacena los datos a procesar así mismo se hizo uso de la herramienta graphviz para crear el diseño y generar una las distintas imágenes en el cual se pueden visualizar los datos obtenidos de un archivo con estructura XML, también se hizo uso de HTML y CSS para genera un reporte en el cual se pueden visualizar todas las interacciones realizadas en la aplicación.

Desarrollo del tema

La aplicación consiste en una forma de representar imágenes utilizando listas ortogonales y permitir realizar operaciones sobre estas imágenes.

Se ha optado el uso del lenguaje de programación Phyton debido a que es multiplataforma y por su facilidad de uso, además también es soporta varios paradigmas de programación.

Para la elaboración de la solución se provee un archivo con extensión y estructura XML, el cual provee información y modelo de las matrices que luego se procesaran y generaran una imagen, este archivo tiene la estructura de la siguiente forma:

```
<?xml version="1.0" encoding="UTF-8"?>
<matrices>
    <matriz>
        <nombre>Matriz_1</nombre>
        <filas>10</filas>
        <columnas>10</columnas>
        <imagen>
            ----------
            -***-***--
            --*--***--
            --*--*----
            -***-*----
            ----------
            -***-***--
            -*-----*--
            -***-***--
            ----------
        </imagen>
    </matriz>
</matrices>
```

Figura 1. Estructura del archivo XML dado.

Fuente: Elaboración propia.

donde:

matrices = es la etiqueta padre.

matriz = es la etiqueta que indica una nueva matriz creada para el respectivo análisis.

nombre = es el identificador de la matriz.

filas = indica las filas que conformara la matriz.

columnas = indica las columnas que conformara la matriz.

imagen = esta etiqueta contendrá los valores respectivos a cada celda de la matriz, consiste en usar cadenas de caracteres; donde el carácter “*” representa una celda con datos y el cambio de fila en la matriz se identificará mediante el carácter salto de línea (\n). Además, se utilizará el carácter “-” para representar los espacios en blanco.

Obtenido ese formato se necesitaba de una estructura en la cual se guardarían la información del archivo dado, por lo cual se llevó al análisis del uso de un tipo

de dato abstracto ya que permite el uso de memoria dinámica ya que el programa lo requiere debido a que no se sabe con certeza la cantidad de datos que puede contener.

Los tipos de datos abstractos implementados para la solución fueron, lista simplemente enlazada, lista doblemente enlazada y una lista ortogonal.

Se utilizó la lista simple enlazada se utilizó para guardar información de las matrices con su respectiva matriz en sí y también para almacenar información sobre los logs generados en la aplicación.

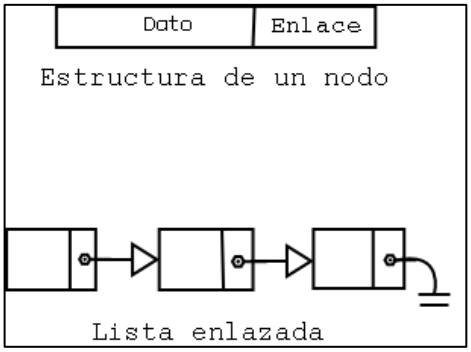

Figura 2. Diagrama de una lista enlazada simple.

Fuente: Diana Molina.

Se hizo uso de la lista doblemente enlazada para utilizarla como cabeceras en la lista ortogonal, ya que estas cabeceras dan orientación acceder e insertar un nodo y también para establecer los nodos que contendrá la lista ortogonal.

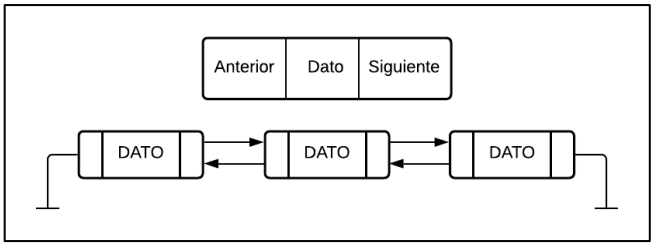

Figura 3. Diagrama de una lista doblemente enlazada.

Fuente: Elaboración propia.

Y también se hizo uso de una lista ortogonal utilizada para representar matrices y almacenar los datos de las matrices que contiene el archivo de entrada, Los nodos utilizados en la lista ortogonal contienen cuatro apuntadores. Uno para apuntar al nodo izquierdo, otro para apuntar al derecho, otro al nodo de abajo y por último un apuntador al nodo de arriba.

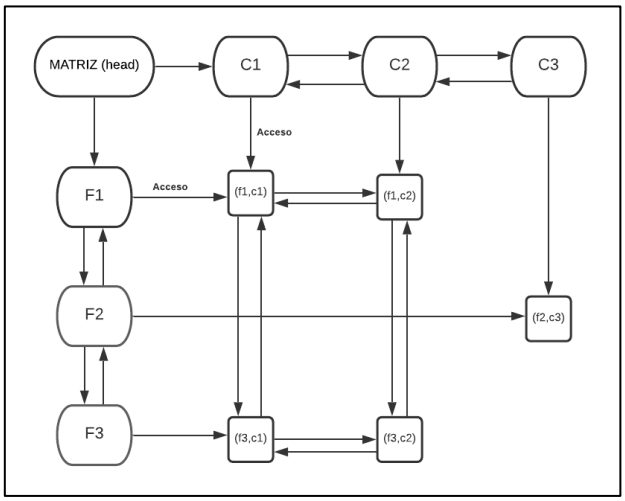

Figura 4. Diagrama de una lista ortogonal.

Fuente: Elaboración propia.

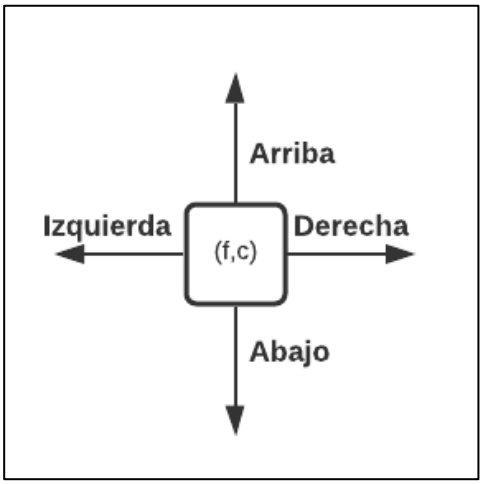

Figura 5. Diagrama de un nodo de la lista ortogonal.

Fuente: Elaboración propia.

Con el paradigma de programación orientada a objetos se emulo los diferentes tipos de datos para almacenarlos en la estructura de datos correspondientes, esto abrió la posibilidad de trabajar más eficientemente y optimización de memoria del programa ya que se tienen varios datos y se necesitan agruparlos entre ellos para su manipulación.

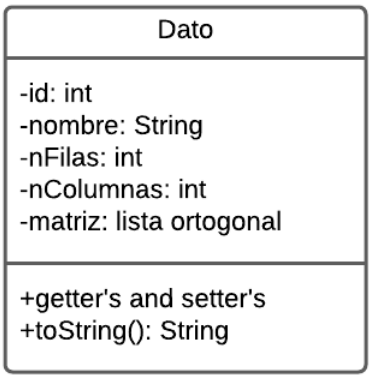

Figura 6. Modelo del tipo de dato Dato.

Fuente: Elaboración propia

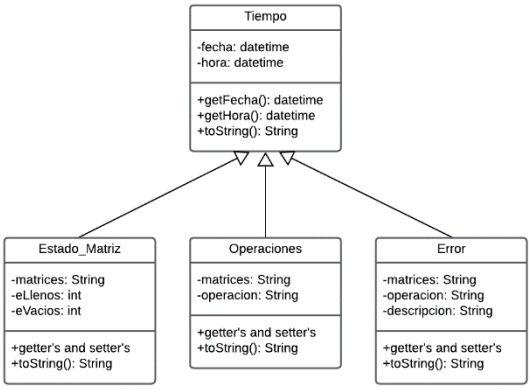

Figura 7. Modelo para los objetos que almacenan información de los logs.

Fuente: Elaboración propia

Con las estructuras ya planteadas, se realizaron los siguientes algoritmos:

a. Extraer datos: este algoritmo obtiene toda la información que contiene el archivo XML proporcionado, identificando los nombres de las matrices junto a sus dimensiones, también extrae la información de la matriz y esta se guarda en una lista ortogonal y el conjunto de todos los datos antes mencionados se guarda en un objeto tipo Dato y este objeto a la vez se agrega a una lista simple para luego poder acceder a ella y extraer la información contenida.

b. Generar imagen: este algoritmo permite graficar una matriz con la herramienta graphviz para luego poder visualizarla en la interfaz gráfica.

c. Reportar: este genera un reporte con todos los logs generados durante la ejecución del programa.

d. Iterar listas: este algoritmo es importante ya que con este puedo recorrer toda la lista y acceder a cada dato contenida en ella y a la vez poder modificar sus valores.

e. Append: este algoritmo permite agregar un nuevo dato a una lista.

f. Buscar: esta función permite buscar un dato específicamente en una lista y extraer toda su información.

g. Eliminar: este algoritmo me permite eliminar un dato en específico contenida en la lista.

Con esos algoritmos basicos implementados en las listas, puede generar las operaciones requeridas por la aplicación, como por ejemplo, rotar horizontal, rotar vertical, transpuesta, agregar linea horizontal, etc. Ya que para generar las distintas operaciones solo se hacen uso de eso algoritmos agregando, modificando, eliminando ciertos datos.

La aplicación utiliza un paquete nativo que permite generar una interfaz gráfica y poder generar una aplicación de escritorio.

Conclusiones

Conocer e implementar bien las estructuras de datos permiten manejar de forma muy rápida y eficiente los datos almacenados.

El desarrollo de esta aplicación da un ejemplo claro de cómo es que trabajan a nivel interno la generación y edición de imágenes.

Se debe conocer a profundidad el lenguaje con el que se trabajara la solución ya que de estos dependen cuan accesible puede ser construir los algoritmos que determinaran la solución al problema, también es necesario conocer que paradigmas se pueden utilizar y en base a esto apegarse a una o varias para hacer lo menos tedioso posible el desarrollo de este.

Apéndice

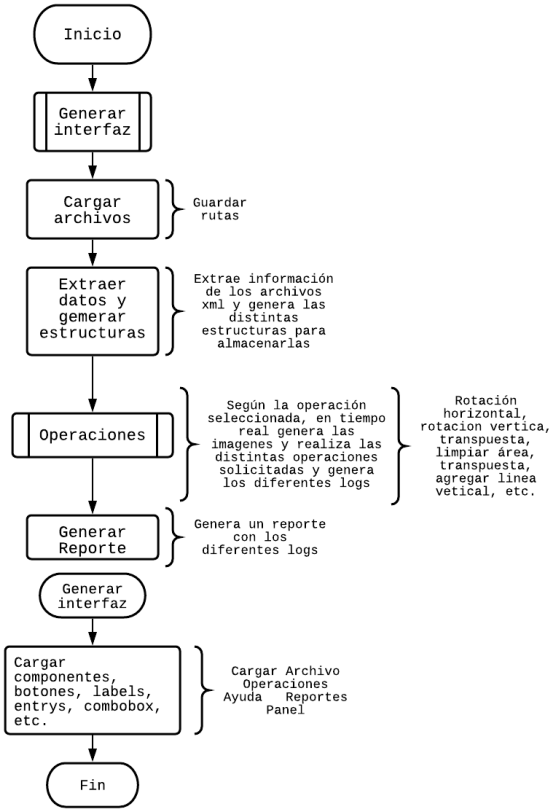

Figura 7. Diagrama general del flujo de la aplicación.

Fuente: Elaboración propia