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.

PROYECTO 1 – TDA's

201800678 – Wilmer Estuardo Vásquez Raxón

Resumen

Consiste en analizar un archivo XML con una estructura definida, almacenando en memoria el contenido de este para su posterior análisis y procesamiento. Este archivo será el que se procese posteriormente y el que definirá las imágenes que se podrán operar. El programa cuenta con diferentes funcionalidades que permitirán realizar las operaciones sobre las imágenes

Es posible modicar las imágenes a discreción del usuario permitiéndole hacer modificaciones a estas y además operaciones entre 2 imágenes las cuales podrán guardar si desean.

Se utilizaron librerías como ElementTree y graphviz las cuales facilitan la lectura y construcción de los xml y las imágenes como grafos.

El usuario podrá generar el archivo xml cuantas veces quiera siempre que ingrese un archivo de entrada y las imágenes se pueden modificar cuantas veces se quiera.

Palabras clave

XML: archivo con extensión .xml el cual se estructura con etiquetas anidadas las cuales pueden tener atributos

Grafo: figuras referentes a mapas de interconexión de datos.

Librerías: conjunto de archivos que se utilizan para facilitar la programación, a los que llamamos al principio de la página

Abstract

It consists of analyzing an XML file with a defined structure, storing its contents in memory for later analysis and processing. This file will be the one that will be processed later and the one that will define the images that can be modify. The program has different functionalities that will allow to perform operations on the images.

It is possible to modify the images at the user's discretion allowing them to make modifications to the images and also to operate between 2 images which they can save if they wish.

Libraries such as ElementTree and graphviz were used to facilitate the reading and construction of xml and images as graphz.

The user can generate the xml file as many times as he/she wants as long as he/she enters an input file and the images can be modified as many times as he/she wants.

Keywords

XML: file with .xml extension which is structured with nested tags which may have attributes.

Graph: figures referring to data interconnection maps.

Libraries: set of files used to facilitate programming, which we call at the top of the page.

Introducción

Se busca disminuir el espacio en memoria que consumen los datos haciendo que los datos se almacenen en una matriz ortogonal para el óptimo manejo de datos

Se intenta dejar claro y de forma simple el manejo de diferentes datos para su manejo optimo de las posiciones, y su operación simple para modificación de los patrones dentro de la imagen.

Se implementan librerías para procesar y escribir archivos XML y PNG para los reportes respectivos los cuales se abren automáticamente.

Además se utilizo la librería Tkinter para la realización de la interfaz grafica, la cual aunque es una librería fácil de usar tiene ciertas complicaciones para operar imágenes y el manejo de métodos para los botones.

Desarrollo del tema

Se implementaron TDA's los cuales almacenan todo el contenido obtenido de los archivos, se manejaron ciclos que permiten que el programa siga en ejecución después de realizar cualquier acción.

Se implemento la matriz con TDA's anidados para evitar el uso de vectores, definiendo otro TDA el cual contiene las matrices implementadas.

Para evitar la complejidad de las clases se implementaron varios métodos y funciones en cada TDA para hacerlos más simples de operar y de acceder a ellos, además de implementar varios métodos dinámicos para que la operación de las imágenes sea mucho más óptimo.

Se accedió al contenido del xml con la librería ElementTree y se obtuvo cada matriz y se itero en ella para obtener cada valor y sus atributos.

Se implemento un método el cual crea una matriz de $n_x m$ la cual se fue llenando buscando la posición x,y de cada dato en el TDA matriz esto si el dato que se analizaba correspondía a un carácter indicador de información en la imagen, esto evita que haya datos innecesarios en cada matriz, además de verificar que no tenga una posición fuera del rango de la matriz.

Se uso comparación matrices simples para la correcta operación de matrices, esto ayuda a simplificar la operación de matrices ortogonales, adicionalmente a esto evito que la aplicación sufra de errores correspondientes a apuntadores nulos.

Cada método retorna una matriz ortogonal la cual se reemplazara en la matriz operada, esto evita tener que operar la matriz ortogonal solamente sustituyendo por la matriz operada evitando así que se pueda corromper algún nodo o apuntado r de esta.

Para la creación del HTML se usó la librería de escritura incluida en el entorno virtual de Python, la cual nos facilita la escritura de este, además de poseer métodos específicos para la estructura correcta del archivo escrito de otra manera se escribiría como texto en el archivo, haciendo la comprensión de este demasiado difícil y poco ortodoxa.

En la creación del grafo de la matriz que se seleccione se utilizó la librería de graphviz la cual facilita la creación de nodos y de el correcto manejo de las celdas o posiciones de la imagen, esto se implemento de manera simple pero legible esta es la que nos proporciona la vista grafica de las matrices en tiempo real dentro de la aplicacion.

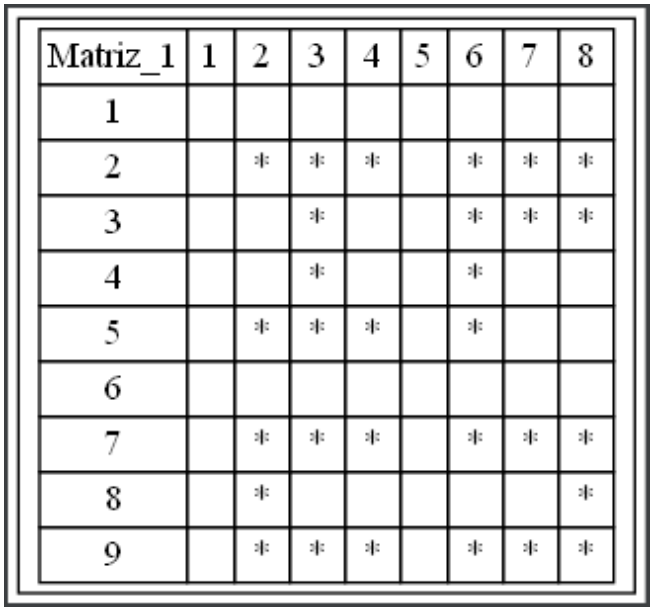

Figura 1. Grafo de matriz seleccionada.

Fuente: elaboración propia.

Se realizo el grafo de forma simple debido a la estructura en la que se trabajo cada lista y cada nodo, por lo cual se debe mencionar que se podría estructurar de mejor forma la matriz si se utilizara algún tipo de estructura mas optima para el manejo de los nodos

Se opto por realizar la mayoría de los métodos en clases separadas y dinámicas lo cual facilito en gran medida la complejidad de operaciones en las imágenes. Esto evito que surgieran problemas de referenciación de nodos y confusión entre las imágenes a operar.

Se utilizo programación orientada a objetos para facilitar el almacenamiento de datos, y para simplificar los bloques de código.

Conclusiones

Se intento crear un programa simple, con una forma ligera para el procesamiento de las imágenes procurando una correcta realización se ideo una metodología de programación orientada a objetos.

Se opto por realizar gran parte de los métodos en clases especificas, y se procuró realizar un código limpio en el cual se validaron las excepciones de forma simple y general.

Se evidencio que existen muchos problemas al no referenciar los nodos de forma correcta haciendo que estos sean un problema al realizar ciertos ciclos dentro de los métodos, además de problemas al momento de referenciar a widgets de Tkinter, dificultando el hecho de no recibir errores tan precisos los cuales son un tanto difíciles de identificar si no se conoce el funcionamiento correcto de estos.

Se recomienda utilizar mensajes de traza frecuentemente para obtener referencias de ejecución de ciertos valores y verificar que lo que se este operando sea un valor y no una dirección de memoria.

Adicionalmente se recomienda que se busquen alternativas para cierto tipo de problemas siendo uno de estos un factor importante para la correcta verificación de funcionamiento de los módulos, siendo un claro ejemplo la correcta creación y visualización de las imágenes las cuales no se muestran debido a problemas de referenciación en el widget.

Se espera que se verifiquen primeramente sitios con documentación oficial y en el idioma de origen para obtener información detallada y precisa del funcionamiento de librerías y del lenguaje mismo antes de buscar en sitios externos.

Se recomienda realizar pruebas constantes para tener un punto de referencia para posibles fallas durante la construcción de la solución, esto evitara la complejidad de búsqueda de errores, además evitara el uso excesivo de herramientas como el debugger del IDE.

Se espera que el código sea implementado correctamente, también deberá estar correctamente versionado y comentado para no confundirdiferentes segmentos de código que puedan ser similares.

Siendo la complejidad del problema un tanto elevada se buscó siempre la forma más simple de solucionarlo.

Aunque siempre puede ser mejor y optimizado de mejor forma se implementó lo que se creyó en su momento los adecuado y lo necesario, pero siempre existe la posibilidad de hacerlo mejor y óptimo.

Se recomienda hacer documentación que ayude al fácil desarrollo de la solución, tales como diagramas de flujo, diagramas de clase, diagramas de entidades, etc.

Se utilizo el software de control de versiones GIT, el cual además se complementó con la integración de github en el IDE pycharm el cual facilito en gran medida la realización de commits los cuales tuvieron como objetivo mejorar el desarrollo de la solución.

Adicionalmente en github se manejo el control delos releases para la publicación de versiones funcionales del programa, esto también ayuda a la comunidad y el propio desarrollador o desarrolladores los cuales pueden tener retroalimentación de comentarios de los que han revisado el programa.

Referencias bibliográficas

Graphviz. (s. f.). *The DOT Language*. graphviz.org. Recuperado 5 de abril de 2021, de https://www.graphviz.org/doc/info/lang.html

pypi. (2020, 24 diciembre). *Graphviz*. Recuperado 5 de abril de 2021, de, https://pypi.org/project/graphviz/#description

python.org. (s. f.). *xml.etree.ElementTree — The ElementTree XML API — Python 3.9.2 documentation*. python. Recuperado 5 de abril de 2021, de https://docs.python.org/3/library/xml.etree.elementtree.html

Python. (s. f.). *tkinter — Python interface to Tcl/Tk*. Python.org. Recuperado 5 de abril de 2021, de https://docs.python.org/3/library/tkinter.html