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 2 – INTRODUCCION A LA PROGRAMACION Y COMPUTO 2

201700375 – Bryan Steve Montepeque Santos

Resumen

Se requiere que el programador desarrolle una solución de Software diseñada para funcionar en dispositivos de bajos recursos computacionales, debido a esto se necesita de una solución de software que sea altamente eficiente con el manejo de memoria para que pueda funcionar de manera óptima en la mayor cantidad de dispositivos posibles permitiendo así un amplio acceso a nuestra aplicación aumentando nuestras ventas de ella, la aplicación en sí consiste en un procesador de imágenes que se trabajarán con matrices, debido a la alta eficiencia de memoria que se requiere estas matrices deben de ser matrices ortogonales, es decir, que un nodo solo exista si contiene información, la aplicación debe de contar con una Interfaz Gráfica que le facilite al usuario trabajar las matrices, las matrices deberán de ser ingresadas con un archivo XML, el usuario elegirá la operación que desea realizar, el programa trabajará en las matrices y las mostrará en la Interfaz al usuario.

Palabras clave

1. Matriz Ortogonal
2. Interfaz Gráfica
3. Barra de Menú
4. Matriz Transpuesta
5. Tipo de Dato Abstracto

Abstract

We require the programmer to develop a software solution designed to work in devices with very low computational resources, because of this the solution must be very efficient with memory usage so that it can have the optimal performance on the widest range of devices possible allowing the public to have greater access to our application and thus, rising the sales on it, the application itself consists in an image processor, each image will be a matrix, due to the high memory efficiency required, this matrixes must be orthogonal matrixes, in other words, every node on the matrix must only exist if it has data on it, the application must have a Graphic User Interface that allows the user to easily work with the matrixes, each matrix will be entered in an XML file, the user will then choose the operation they want to perform, the program will then proceed to work on the matrixes and finally it will show the result on the GUI to the User.

Keywords

1. Orthogonal Matrix
2. Graphic User Interface
3. Menu Bar
4. Transposed Matrix
5. Abstract Data Type

Introducción

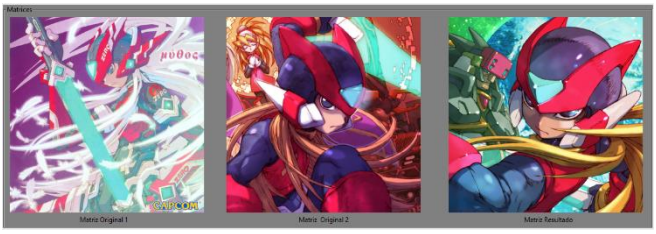
Se ha solicitado un Procesador de Imágenes utilizando Matrices cargadas mediante un archivo XML de forma tan eficiente con los recursos como se pueda, debido a esto se utilizarán Matrices Ortogonales, es decir, Matrices en las cuales las únicas casillas guardadas en memoria son aquellas que no estén vacías, algo que es posible usando nodos y apuntadores de una forma más avanzada que en una Matriz normal, una vez que estas matrices estén almacenadas en el programa se deben mostrar al cliente en una Interfaz Gráfica donde se contará con una Barra de Menú donde el usuario podrá elegir la operación que desee y el programa le mostrará el resultado en la Interfaz, pero, ¿Cuál es la forma en la que se crea una Matriz Ortogonal? y ¿Es más complicado trabajar con ellas? Estas preguntas se responden a continuación.

Desarrollo del tema

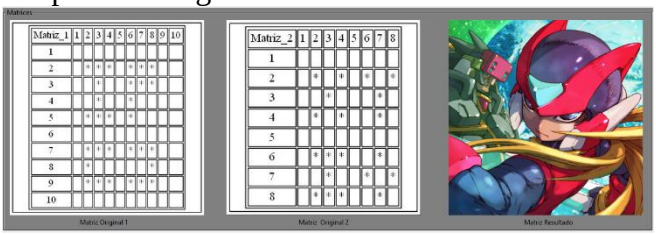
Para poder desarrollar este proyecto, fue necesario dividirlo en varias partes más pequeñas:

1. **Crear la Interfaz Gráfica:**
   Para crear la Interfaz Gráfica fue necesario utilizar una librería que nos permitiera generar componentes gráficos en Python, la librería utilizada fue tkinter ya que ya está incluida en Python, se utilizó una MenuBar para mostrar las diferentes opciones que puede realizar el usuario.

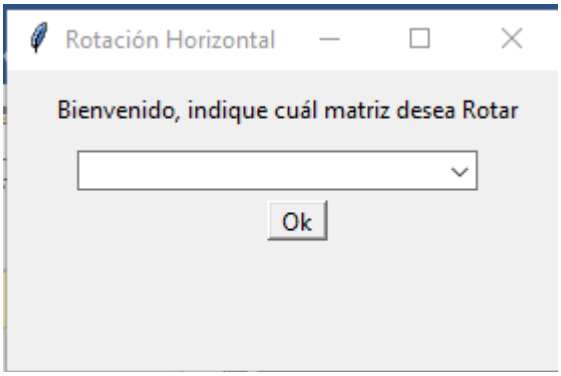
   Después de esto se crearon los Paneles donde se van a mostrar las matrices, debido a que original mente las matrices no están cargadas no hay nada que mostrar en ellos, para evitar dejarlos en blanco se optó por asignarles imágenes predeterminadas al iniciar el programa, pero una vez se cargan las matrices estos paneles cambian para mostrar las matrices cargadas, estas matrices fueron Graficadas con GraphViz como se verá en un inciso posterior, los Paneles vacíos se ven así:



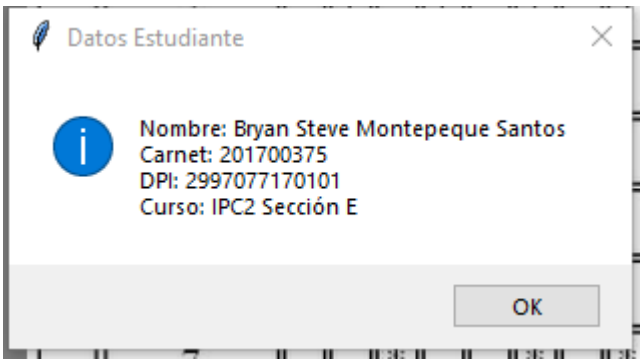
   Después de cargar un archivo se ven así:



   Una vez teniendo la estructura básica de la Interfaz Gráfica, hablaremos de los componentes gráficos menores de nuestro proyecto, tales como las ventanas donde podemos darle a cada operación la información que necesita para poder trabajar adecuadamente, cada operación tiene su propia ventana que pide información diferente según sea la operación, por ejemplo, para la Rotación Horizontal, no debería de pedir información pero debido a que tenemos dos matrices en el programa, el usuario debe de elegir cuál de las dos matrices es la que desea rotar, para pedir estos datos se utilizó un frame diferente el cual se ve así:

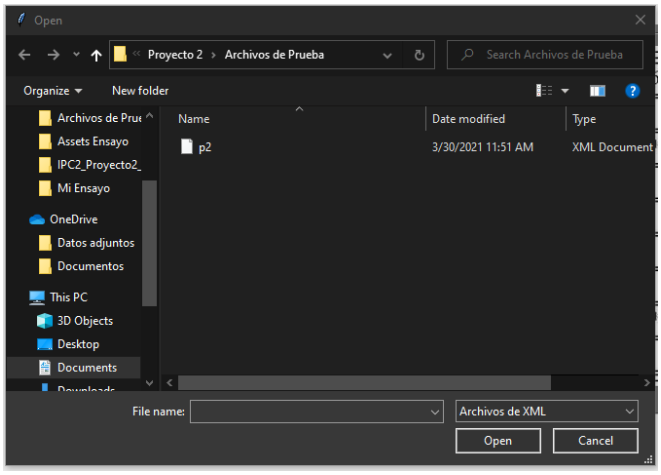


Finalmente, se requiere mostrar los datos del creador del programa, esto se muestra en un MessageBox:



2. Cargar Archivo:

Para cargar el archivo se utilizó un FileChooser, es decir, una ventana que nos permite abrir un archivo en cualquier dirección de nuestra computadora;

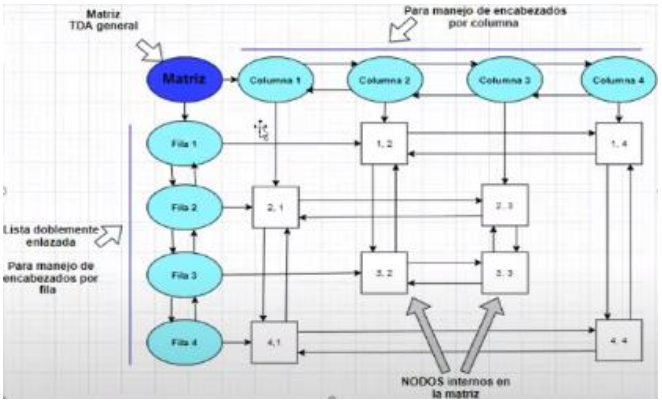


Este FileChooser tiene filtros para que solo reconozca archivos XML, aunque con el fin de facilitarle al usuario el orientarse mientras navega entre las carpetas también se puede quitar el filtro, pero el programa solo funciona con archivos XML.

3. Crear la Matriz Ortogonal:

Bien, esta es la parte más complicada del programa, ya que no existe ninguna posición que esté vacía esto dificulta mucho hasta las tareas más sencillas como recorrer la matriz o graficarla, pero empezaremos con lo básico, ¿Cómo se hace una Matriz Ortogonal? Bueno, se inicia creando dos listas ortogonales de "Cabeceras", estas Cabeceras son nuestras Filas y nuestras Columnas, pero no son las que guardan información, estas sólo indican qué número de Fila o qué número de Columna es la que se está trabajando en determinado momento, para esto es como crear una Lista Doblemente Enlazada normal, una para Filas y una para Columnas, solo que a estos nodos se les debe de agregar el id de la Fila o de la Columna que se va a ingresar puesto que aquí no hay cabeceras consecutivas, solo hay nodos que guardan el id de la cabecera entonces ¿Cómo se forma la estructura de la matriz de forma ordenada? Haciendo un Bubble Sort utilizando el id de cada nodo, esto se hace cada vez que se ingresa un nodo a la Lista de Filas o de Columnas, ahora bien, estas Listas no son las matrices solo representan los índices de las verdaderas Filas y Columnas, para crear las verdaderas Filas y Columnas se necesita de un TDA bastante complejo en donde cada nodo debe tener tanto la información del nodo como su fila, su columna y su dato, así como los apuntadores para todas 4 direcciones, Arriba, Derecha, Abajo e Izquierda, esto para poder navegar bien en la matriz y poder recorrerla usando tanto sólo las filas cómo sólo las columnas, entonces, cada vez que insertamos un nodo a la matriz también se inserta un nodo en la Lista de Filas y en la Lista

de Columnas, si es que no existe un nodo con este id previamente, si ya existe un nodo con este id entonces solo agrega el nodo al final de la lista y se hace un Bubble Sort para poder ordenar qué nodo va de primero, pero aún nos falta una parte importante, la conexión entre las Listas de Cabeceras y la matriz en sí, bien, cada nodo de estas listas no solo contiene un id y los apuntadores para anterior y siguiente, también contiene un atributo llamado "accesoNodo" que guarda un puntador hacia al primer nodo que se encuentre en determinada Fila o Columna, ya que toda esta estructura está realizada, solo falta unir ambas Listas de Cabeceras en una sola Matriz, para esto se crea un objeto Matriz que lo único que hace es apuntar hacia el primer nodo de la Lista de Filas y la Lista de Columnas y listo, ya tenemos nuestra Matriz Ortogonal, he aquí un diagrama de esta explicación:



4. Graficar las matrices en GraphViz:

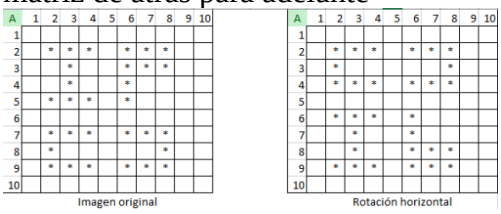
Para Graficar las matrices en GraphViz se complica bastante porque las Matrices Ortogonales no tienen otros nodos fuera de aquellos que contienen los datos, entonces, ¿cómo hacer una tabla en GraphViz? Con un for desde 0 hasta la cantidad de Filas/Columnas de la matriz, básicamente como se recorre una matriz de las que no se hacen con TDA pero usando un método que devuelva cada nodo existente de la matriz y si no existe el nodo que se solicita con el id de Fila y Columna entonces devuelva None, y verificando con un if si lo que devuelve la función es None entonces agrega un espacio vacío a la matriz en esa posición y si el Nodo sí existe que tome el valor del nodo devuelto y si este es igual a "*" agregue un "*" a la matriz en esa posición, luego lo volvemos una imagen y se lo ponemos al Label correspondiente. Así es cómo se hace:

```
for j in range(1, int(ColumnasMO2) + 1):
    if MatrizOrtogonal2.MostrarNodoMatriz(i, j) == None:
        CodigoGraphViz = CodigoGraphViz + "<TD border=\"1\"> </TD>\n"
    elif MatrizOrtogonal2.MostrarNodoMatriz(i, j).valor == "*":
        CodigoGraphViz = CodigoGraphViz + "<TD border=\"1\">*</TD>\n"
```

5. Operaciones:

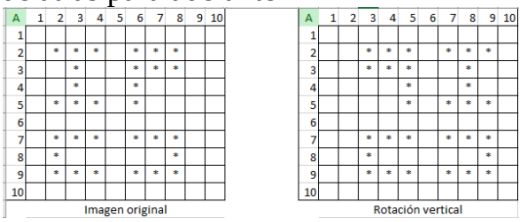
a. Rotación Horizontal:

Este método se resuelve solo recorriendo las Columnas de la matriz de atrás para adelante

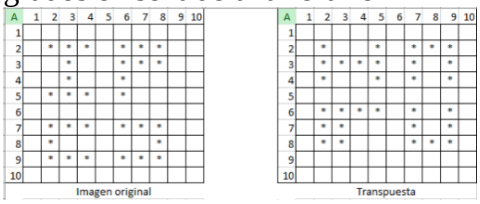


b. Rotación Vertical:

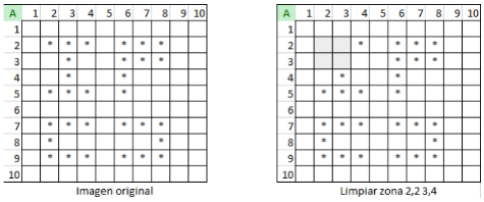
Solo se recorren las Filas de la Matriz de atrás para adelante



c. Transpuesta:
Primero se hace una Rotación Horizontal y luego se rota todo 90 grados en sentido antihorario

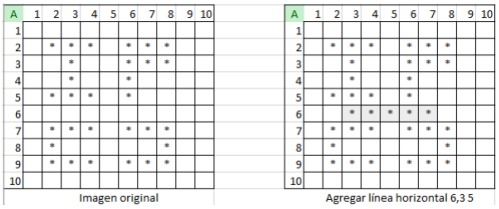


d. Limpiar Área:
Solo se recorre la matriz y se eliminan todos los apuntadores hacia los nodos en el área indicada

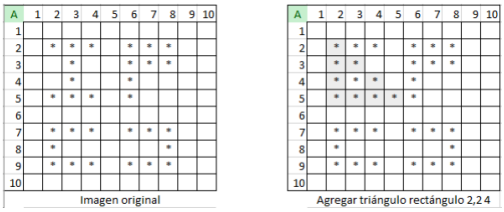


e. Insertar Figuras Varias:
Solo se recorre la Matriz con unos fors insertando los nodos en las casillas indicadas y según se necesite

i. Línea Horizontal

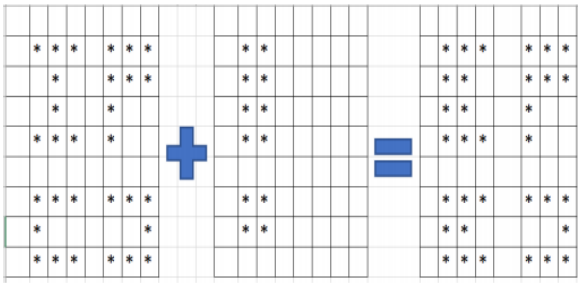


ii. Triángulo Rectángulo

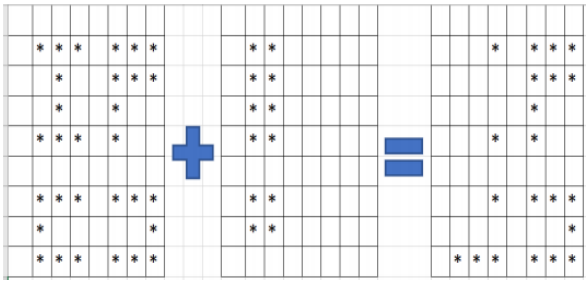


f. Operaciones con Dos Matrices:
Se compara la información de las dos matrices y se van insertando los nodos o eliminando los apuntadores hacia los nodos según se requiera por la operación

i. Unión



ii. Diferencia



Conclusiones

El programa fue dividido en varios “Mini Programas” o módulos para poder simplificarnos la tarea de programarlo y también para poder repartirnos el trabajo entre varias personas si fuera necesario hacerlo así, esta dedicación y atención al detalle se ve reflejada a lo largo de todo el programa, como al enfocarnos en asegurarnos que cada matriz al ser graficada incluyera su nombre en la esquina superior izquierda se la matriz, y cómo puede apreciarse, trabajar con matrices ortogonales es muy complicado por lo que lo más recomendado es que solo se usen si es completamente necesario.

Referencias bibliográficas

1. marc.gisbert@matricesydeterminantes.com. (2021, February 17). *Matriz Ortogonal*. matrices y determinantes.

2. https://www.matricesydeterminantes.com/matrices/matriz-inversa/matriz-ortogonal-ejemplos-propiedades-2x2-3x3/
3. colaboradores de Wikipedia. (2021, April 6). *Interfaz gráfica de usuario*. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Interfaz_gr%C3%A1fica_de_usuario

4. *Matriz transpuesta – Matemáticas fáciles*. (2017, August 16). Mates Fácil. https://blogs.ua.es/matesfacil/bachillerato/algebra-matricial/matriz-transpuesta/

5. R. (2017, March 13). *Tipos de datos abstractos (ADT).* FundamentosPOO. http://fundamentospoorrr.blogspot.com/2017/03/tipos-de-datos-abstractos-adt.html

6. *Tk menubar - Menus in Tkinter*. (2015, May 5). Pythonspot. https://pythonspot.com/tk-menubar/