PROYECTO 2 – IPC2

Estudiante

201115455 – José Ernesto Pajoc Raymundo.

Resumen

El proyecto está enfocado al análisis de matrices, manejo de archivos XML y tipo de datos abstracto como lo son matrices ortogonales, todo ello utilizando el lenguaje de alto nivel Python para la creación del software, donde se aplica el paradigma de la programación orientada a objetos, modular y la funcional, además se crea grafos por medio de archivos con extensión DOT para representar las matrices.

El usuario es el encargado de seleccionar el archivo para cargar las matrices, esta debe cumplir con la estructura definida para el funcionamiento adecuado del programa, si la información es correcta el programa mostrará en pantalla la imagen que posee las matrices y a través del entorno gráfico podrá realizar operaciones sobre una matriz o ambas, para finalizar con un reporte en HTML.

Palabras clave

- Nodo
- Lista
- Grafo
- Matriz
- Apuntador

Abstract

The project is focused on the analysis of matrices, handling of XML files and abstract data types such as orthogonal matrices, all using the high-level language Python for the creation of the software, where the paradigm of object-oriented programming is applied. , modular and functional, in addition graphs are created by means of files with the DOT extension to represent the matrices.

The user is in charge of selecting the file to load the matrices, this must comply with the structure defined for the proper functioning of the program, if the information is correct the program will display the image that has the matrices on the screen and through the graphic environment you can perform operations on a matrix or both, to end with an HTML report.

Keywords

- *Node*
- *List*
- *Graph*
- *Matrix*
- *Prompter*

Introducción

El programa que corresponde al proyecto 2 es una aplicación desarrollada en el lenguaje de programación Python, el cual cuenta con un menú principal manejado a través de un entorno gráfico, en el se puede cargar un archivo con extensión XML, luego serán mostradas las matrices en imágenes con formato PNG en sus respectivos visores, luego de cargar las matrices se podrá realizar diversas operaciones sobre una o dos matrices según los requiera el usuario, cada operación realizada será registrada en el reporte HTML.

Desarrollo del software

El lenguaje utilizado para desarrollar el software es Pyhton en su versión 3.8.1 y el editor de texto utilizado es Visual Studio Code 1.54.1

La finalidad del programa es crear una herramienta que permita cargar un archivo XML a través de la programación orientada a objetos y los tipos de datos abstractos, para que esto pueda ser representado de manera gráfica y se facilite la edición de las matrices, esto se logra a través del uso de una matriz ortogonal, la cual su base fundamental es un nodo que posee sus cuatro apuntadores y el dato a almacenar.

Para crear el software se utilizo diversos archivos con extensión ".py" identificados como main, nodo, listaVertical, listaHorizontal, nodoListaCabeCol, nodoListaCabeFil, matrizOrtogonal, nodoMatrix. Los paradigmas utilizados son modular, funcional, estructurado y orientado a objetos, el tipo de dato abstracto que maneja el control de la información es una matriz ortogonal formada por listas doblemente enlazadas y el programa encargado de generar el grafo de las matrices y sus respectivas imágenes es Graphviz.

El flujo de datos debe ser el siguiente:

1. Cargar el archivo XML con la estructura adecuada.
2. Seleccionar el tipo de operación para matrices, este puede ser para una o dos.
3. Seleccionar la operación que se desea aplicar.
4. Generar reporte.

A continuación, se muestra los módulos importados y funciones utilizadas en cada archivo que conforma el proyecto:

Tabla I.

clase utilizada en nodo.py

NOMBRE	FINALIDAD
nodo	Crear un nuevo nodo para la matriz ortogonal, este posee la información del dato, columna, fila y sus apuntadores arriba, abajo, izquierda, derecha.

Fuente: elaboración propia.

Tabla II.

Clase y funciones utilizadas en menu.py

NOMBRE Y TIPO	FINALIDAD
Clase: listaHorizontal	Crear una lista doble enlazada para simular las filas de la matriz ortogonal.
Función: verVacio-Horizontal	Verificar que el apuntador inicio no sea nulo.
Función: insertar	Verificar el valor de la columna del nodo para saber si se agrega al inicio, en medio o al final de la lista

	doble enlazada llamando al respectivo método.
Función: insertarInicio	Agrega el nodo al inicio de la lista doble enlazada.
Función: insertarFinal	Agrega el nodo al final de la lista doble enlazada
Función: insertarMedio	Recorre la lista doble enlazada para comparar la columna del nodo con los que ya existen y así posicionarlo en el lugar que le corresponde.
Función: buscarColumna	Busca un nodo de la lista doble enlazada a través del valor de la columna que se desea.

Fuente: elaboración propia.

El archivo "listaVertical.py" posee las mismas funciones solo que aplicado a las columnas que simulará la matriz ortogonal.

El archivo "nodoListaCabeCol.py" es importante dentro de la matriz ortogonal ya que este simulará el índice de las columnas y a su vez poseerá un atributo de tipo listaVertical, en otras palabras posee una lista doble enlazada como propiedad, este archivo se conforma por dos clases, la primera para crear un nodo cabecera para las columnas y la segunda para crear la lista doble enlazada para las cabeceras de las columnas.

Tabla III.

Clases y funciones utilizados en nodoListaCabeCol.py

NOMBRE Y TIPO	FINALIDAD
Clase: nodoCabecera-Columnas	Crear un nodo cabecera para las columnas, con el número de columna, apuntadores arriba, abajo y una lista doble enlazada de tipo listaVertical.
Clase: listaCabecera-Vertical	Crear una lista doble enlazada para almacenar los nodos cabecera.
Función verVacioLista-CabeceraVertical	Verificar que el apuntador inicio no sea nulo.
Función: insertar	Verificar el valor de la columna del nodo para saber si se agrega al inicio, en medio o al final de la lista doble enlazada llamando al respectivo método.
Función: insertarInicio	Agrega el nodo al inicio de la lista doble enlazada.
Función: insertarFinal	Agrega el nodo al final de la lista doble enlazada
Función: insertarMedio	Recorre la lista doble enlazada para comparar la columna del nodo con los que ya existen y así posicionarlo en el lugar que le corresponde.
Función: buscarCabecera-Vertical	Busca un nodo de la lista doble enlazada a través del valor de la columna que se desea.

Fuente: elaboración propia.

El archivo "nodoListaCabeFil.py" posee las mismas funciones solo que aplicado a la fila cabecera que simulará el índice de las filas para la matriz ortogonal.

El archivo "matrizOrtogonal.py" es uno de los más importantes ya que realiza la unión de las listas verticales y horizontales con las listas cabeceras de las columnas y las filas, de esa manera se logra obtener la matriz ortogonal, con los nodos que poseen sus cuatro apuntadores, en este archivo también se encuentran las funciones necesarias para realizar las operaciones sobre una y dos matrices cargadas previamente.

Tabla IV.

Funciones utilizadas en matrizOrtogonal.py

NOMBRE	FINALIDAD
insertar	Crear un nuevo nodo ortogonal y lo insertar en la lista vertical y la lista horizontal haciendo uso de las listas cabeceras para ubicarlo en la posición deseada.
llenado	Recorre una cadena de texto la cual es la imagen extraída del archivo XML, transformada en una sola línea, cada vez que recorre un carácter de la cadena de texto llama al método insertar.
llenadoVacio	Crear una matriz con las mismas dimensiones de la matriz más grande cargada en el archivo XML, esta es llenada con guiones los cuales representa un espacio en blanco en la imagen.
llenadoRotacion-Horizontal	Llama al método insertar con la diferencia de que las filas van disminuyendo de índice mientras que las columnas van aumentando, de esa forma se logra la rotación
llenadoRotacion-Vertical	Llama al método insertar con la característica de que las filas van aumentando de índice mientras que las columnas van disminuyendo.
devolverCadena	Recorre la matriz ortogonal y extra el dato que posee cada nodo, todo esto se concatena en una sola cadena de texto la cual es retornada por la función.
crearGrafo	Crea el archivo DOT necesario para Graphviz, formando una tabla donde las filas y columnas coinciden con los valores de cada nodo en la matriz ortogonal, luego renderiza el archivo para crear la imagen en formato PNG.
buscarNodo	Permite buscar un nodo puntual de la matriz ortogonal utilizando sus coordenadas para ser localizado y retornar el nodo como tal.
buscarNodo-SustituirDato	Permite buscar un nodo puntual de la matriz ortogonal utilizando sus coordenadas para ser localizado y así modificar el dato que posee.

Fuente: elaboración propia.

Para poder tener varias matrices ortogonales se crea una nueva lista simple enlazada, la cual almacenara las características principales de cada matriz, el nodo de esta lista posee el nombre, dimensiones, fecha de creación, una matriz ortogonal y su apuntador siguiente.

Tabla V.

Clases y funciones utilizados en nodoMatrix.py

NOMBRE	FINALIDAD
Clase: nodoMatriz	Nodo el cual poseerá las características principales de cada matriz ortogonal.
Clase: listaMatriz-Ortogonal	Es una lista simple enlazada para almacenar cada nodoMatriz.
verVacioLista-Matriz-Ortogonal	Verificar si el apuntador inicio es igual nulo.
insertarFinal	Agrega un nuevo nodo al final de la lista simple.

buscarNombre-Matriz	Permite buscar una matriz ortogonal dentro de la lista simple enlazada a través de su nombre
buscarPosicion-Matriz	Permite buscar una matriz ortogonal dentro de la lista simple enlazada a través del índice.
crearReporte	Crea dos archivos para generar el reporte, el primero es una hoja de estilos css y el segundo crea un archivo HTML el cual utiliza la información de todos los nodos que se encuentren dentro de la lista simple enlazada y sus respectivas imágenes.

Fuente: elaboración propia.

Finalmente, en el archivo main.py se encuentra los elementos que conforman el entorno gráfico y las operaciones que se pueden realizar sobre dos matrices.

Tabla VI.

Funciones utilizadas en main.py

NOMBRE	FINALIDAD
obtenerFecha	Toma la fecha y hora actual del sistema y lo retorna en una variable de tipo texto con el formato deseado en el reporte.
cargarXML	Convierte el archivo cargado en uno legible por elementTree.
procesarXML	Extrae la imagen que posee el XML y lo envía a una matriz ortogonal, a su vez llama al método para crear su grafo e imagen y así mostrarlo en los visores de la ventana principal.
mostrarImagenes	Muestra en los visores las imágenes creadas a partir de los archivos DOT que representan las matrices del archivo XML.
rotacionHorizontal-Matriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para realizar la rotación horizontal, crear archivo DOT y general la imagen.
rotacionVertical-Matriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para realizar la rotación vertical, crear archivo DOT y generar la imagen.
transpuestaMatriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para realizar la transpuesta de la matriz, crear archivo DOT y generar la imagen.
limpiarZona-Imagen	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para limpiar la zona que se desea, crea archivo DOT y genera la imagen.
agregarLinea-HorizontalMatriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para agregar una línea horizontal en el lugar que se desea, crea archivo DOT y genera la imagen.
agregarLinea-VerticalMatriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para agregar una línea vertical en el lugar que se desea, crea archivo DOT y genera la imagen.
agregarRectangulo-Matriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para agregar un rectángulo en el lugar que se

	desea, crea archivo DOT y genera la imagen.
agregarTrianRec-Matriz	Crea una nueva matriz ortogonal en la lista simple enlazada y envía la información necesaria para agregar un triángulo rectángulo en el lugar que se desea, crea archivo DOT y genera la imagen.
unionMatrices	Crea dos nuevas matrices ortogonales en la lista simple enlazada y envía la información necesaria para hacer la unión de la imagen A con la imagen B, crea archivo DOT y genera la imagen.
Intersección-Matrices	Crea dos nuevas matrices ortogonales en la lista simple enlazada y envía la información necesaria para hacer la intersección de la imagen A con la imagen B, crea archivo DOT y genera la imagen.
diferencia-Simetrica-Matrices	Crea dos nuevas matrices ortogonales en la lista simple enlazada y envía la información necesaria para hacer la diferencia simétrica, crea archivo DOT y genera la imagen.
diferencia-Matrices	Crea dos nuevas matrices ortogonales en la lista simple enlazada y envía la información necesaria para hacer la diferencia de la imagen A con la imagen B, crea archivo DOT y genera la imagen.
mostrar-Operaciones1	Muestra en la ventana principal las operaciones que se pueden realizar sobre una matriz.
Mostrar-Operaciones2	Muestra en la ventana principal las operaciones que se pueden realizar sobre dos matrices.
reporteMatrices	Llama al método que permite crear el reporte en HTML y CSS.
buscarXML	Permite abrir una ventada de dialogo en la cual el usuario puede seleccionar el archivo XML.
abrirPDF	Permite abrir la documentación del programa.

Fuente: elaboración propia.

Con lo anterior se logra cargar, procesar, exportar imágenes y reportar toda la información de las matrices.

El sistema operativo que se utilizo es Windows 10 y el instalador de paquetes de Python fue pip para poder hacer uso de Graphviz, tkinter, PIL, datetime, webbrowser, elementTree, io.

Conclusiones

El lenguaje de Python es muy sencillo de utilizar y entender comparado con otros lenguajes de programación de alto nivel, lo cual permite enfocarse en la solución del problema.

Existe muchas fuentes de información que facilitan el uso del lenguaje Python, pero respecto a tipos de datos abstractos la mayor parte de fuentes se encuentran en inglés y no es mucha, ya que la mayoría utiliza la memoria dinámica que gestiona el mismo lenguaje, reduciendo las fuentes de información.

Al hacer uso de los tipos de datos abstractos se utiliza únicamente la memoria necesaria en cada matriz y no se desperdicia hardware lo cual permite mayor rendimiento en el equipo.

Referencias bibliográficas

(s.f.). Obtenido de http://facundoq.github.io/courses/images/res/03_numpy.html

Programador clic. (s.f.). Obtenido de https://programmerclick.com/article/8282763126/

Severance, C. (2009). *Python para informáticos.* Michigan.

Anexos

Diagrama de clases

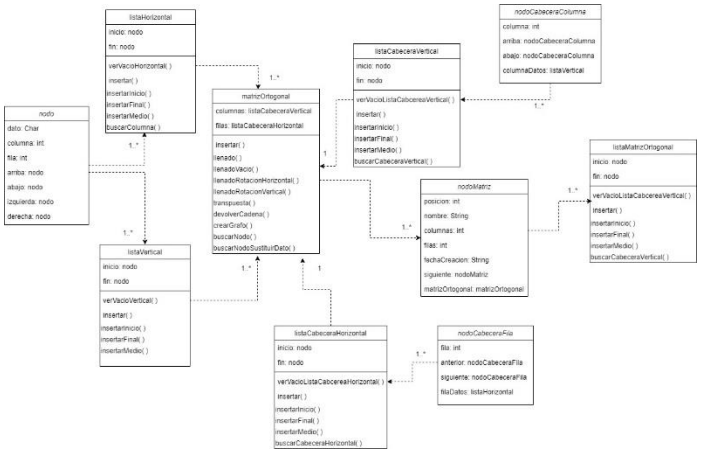
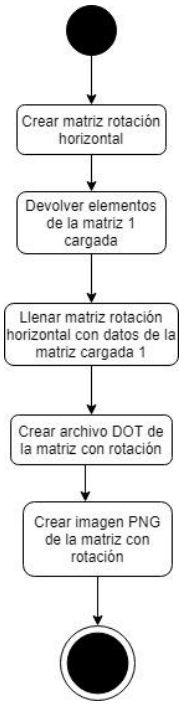


Diagrama de actividades: rotación horizontal de una matriz.



Insertar un nuevo nodo en la matriz ortogonal.