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

PROYECTO 1

201801075 – Cinthia Avex Nim Quiñonez Alvarez

Resumen

A continuación, se describe cómo se maneja un archivo XML y se almacenan los datos en una matriz creada a base de clases y objetos a través de diferentes métodos que se necesiten en los cuales se pueden hacer diferentes operaciones, para lograr todo ello se usa una herramienta para poder manejar ficheros XML el cual es la librería minidom, y asi usando sus diferentes métodos se logra obtener los datos para que luego se creen tipo de datos abstractos (TDAs) los cuales almacenaran los datos del fichero que en este programa se crea una matriz ortogonal y ya almacenado los datos se puede realizar la operación solicitada la cual son mostrar una gráfica con los datos a través del programa graphviz y encontrar el recorrido en donde la suma sea la menor, luego mostrar el recorrido de menor suma en un archivo de salida el cual es un fichero XML.

Palabras clave

Python, Matriz, Lista, Cabeceras, Nodo, Clases, Objetos, Tipo de dato abstracto.

Abstract

The following describes how an XML file is handled and the data is stored in an array created based on classes and objects through different methods that are needed in which different operations can be done, To achieve this, a tool is used to manage XML files, which is the minidom library, and using its different methods, the data is obtained so that later abstract data types (TDAs) are created which will store the data of the file that in this program an orthogonal matrix is created and already stored the data can perform the requested operation which are to show a graph with the data through the graphviz program and find the path where the sum is the youngest, then show the least sum path in an output file which is an XML file.

Keywords

Python, Matrix, List, Headstones, Nodes, Clases, Objects, Abstract Dato Type.

Introducción

En el siguiente trabajo se presenta la solución al problema planteado, en el cual se usará el lenguaje de programación Python y el paradigma de programación orientado objetos (POO), para una forma correcta del uso de la memoria se usará tipo de dato abstracto (TDA), enfocado en la lista ortogonal y lista simple.

Los datos del archivo se almacenan en una lista ortogonal y la realización de esta lista es con el manejo de memoria dinámica usando tipo de dato abstracto (TDA) y creando clases para el uso de programación orientada a objetos (POO) para tener un manejo de información más eficiente, eficaz y código ordenado.

Para este programa se realizaron clases con métodos y atributos, estas se usan de una forma lógica para poder realizar las operaciones que solicitan.

Desarrollo del tema

El problema que se nos presento es el siguiente:

"Recorrer una matriz con valores numéricos empezando con la posición inicial que ya nos da el problema para llegar a la posición final también ya dada y encontrar el recorrido en donde la suma de cada casilla sea la menor"

Para encontrar la solución se realizó un diagrama para una mejor comprensión del problema y así de una manera más eficiente encontrar la solución.

Para la creación de la matriz se planteó la siguiente solución desarrollada en un diagrama y así poder comprender como es el camino que toma cada dato para almacenarse:

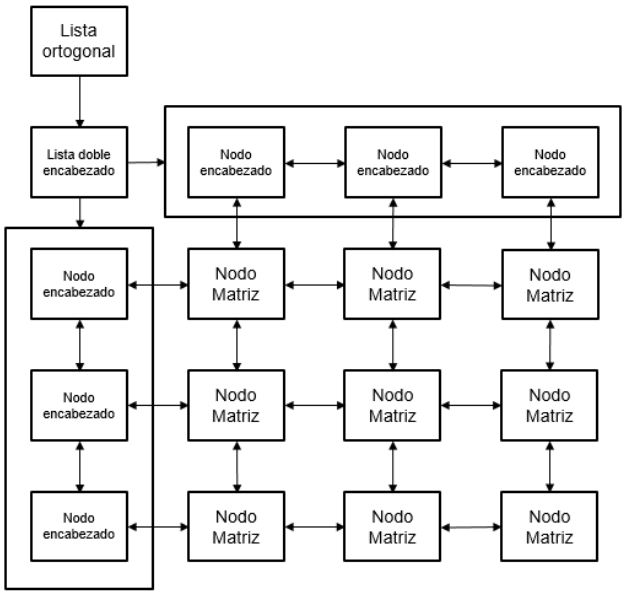

Figura 1. Diagrama

Fuente: Elaboración propia

Se puede observar en el diagrama que la matriz almacena varios nodos donde está alojada la información que en este problema es un número.

Las opciones que va encontrar en el menú del programa son:

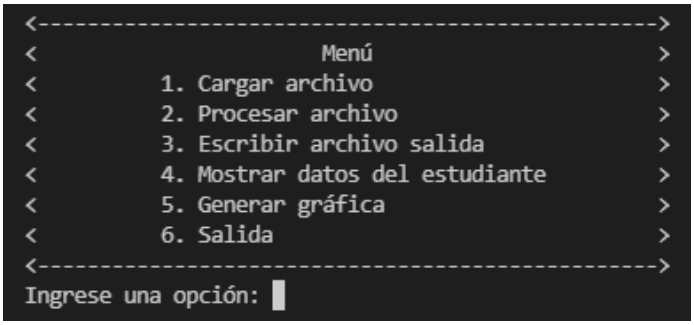

Figura 2. Menú

Fuente: Elaboración propia

En la opción cargar archivo va a pedir una ruta del archivo .xml va a almacenar los datos y va mostrar los datos en pantalla.

En la opción procesar archivo va a solicitar un nombre del terreno ya almacenado y buscara el camino con menor suma.

En la opción escribir archivo salida muestra los datos de la matriz que se procesó en un archivo .xml.

En la opción mostrar datos del estudiante se observan los datos del estudiante.

En la opción generar grafica va a solicitar un nombre del terreno ya almacenado y mostrara la gráfica.

Para la realización de este problema se hizo uso de programación orientada a objetos (POO) y tipo de dato abstracto (TDA) donde se realizaron clases y objetos.

Para una mejor comprensión de cómo se realizó la estructura del programa, a continuación, un diagrama de clases de cómo se desarrolló en Python:

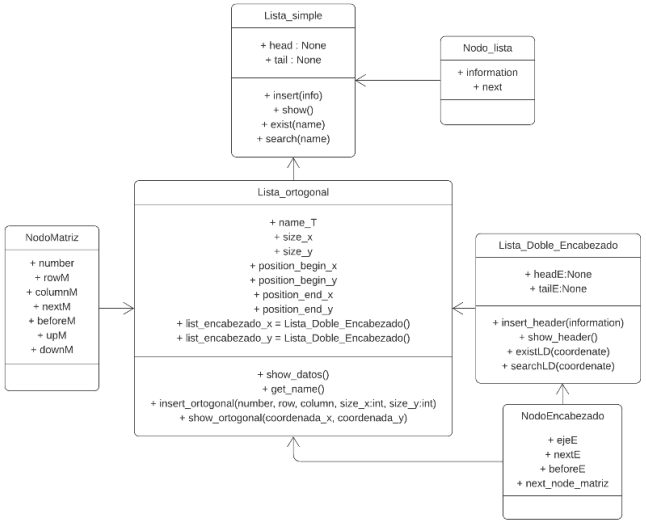

Figura 3. Diagrama de clases

Fuente: Elaboración propia

La estructura realizada consta de dos listas cabeceras una es la cabecera de x que representan las filas y la otra es la cabecera de y que representan las columnas para formar la matriz, ambas listas cabeceras acceden a los nodos de la matriz.

Para insertar un nuevo nodo a la matriz se verifica si existe con el método “existLD()” en las cabeceras de ‘x’ y de ‘y’, enlazando la cabecera con la matriz y se inserta el nuevo nodo en la matriz, ya si no existe el nodo cabecera se crea el nuevo nodo cabecera y se enlaza la cabecera con la matriz para inserta el nuevo nodo a la matriz.

Para recorrer y mostrar la matriz se buscan con el método “searchLD(coordenada_x)” las cabeceras de x y se enlazan a las filas de la matriz después se buscan con el método “searchLD(coordenada_y)” las cabeceras de ‘y’ y se enlazan a las columnas de la matriz, luego se igualan las filas y columnas para después recorrer y mostrar los valores de la matriz almacenados.

La otra estructura es la lista simple la cual almacena los nombres, dimensiones, posición inicio, posición final y matriz con el método “insert(info)” y para mostrar los datos con el método “show()”.

Otro método que se encuentra en la clase Lista_simple es el método “search(name)” el cual nos sirve en el programa para buscar el nombre de un archivo y mostrar la matriz almacenada con dicho nombre.

Para poder visualizar la matriz se genera una gráfica con graphviz donde solicitando el nombre de la matriz que el usuario desee ver y se mostrara la matriz por medio de una imagen.

Los datos se obtienen de un archivo XML para manejar este tipo de archivo se utilizó la librería minidom.

Todo este programa se realizó en Python y se muestran todas las operaciones en consola.

Definición:

Un Tipo de Dato Abstracto (TDA) es un modelo que define valores y las operaciones que se pueden realizan sobre ellos. Y se denomina abstracto ya que la intención es que quien lo utiliza, no necesita conocer los detalles de la representación interna o bien el cómo están implementadas las operaciones.

Conclusiones

Se concluye que un buen manejo de memoria dinámica y un buen uso de tipo de dato abstracto (TDA) puede hacer que el manejo de datos sea de una forma más eficiente además que para el uso de un usuario es más fácil y rápido el encontrar un dato que necesite.

Se comprueba que el uso de programación orientada a objetos es muy fundamental e importante para poder realizar cualquier programa, ya que se puede tener de una forma ordenada el código y el manejo de cada variable es más eficiente y eficaz.

Referencias bibliográficas

Graphviz, https://graphviz.org/
Python, https://www.python.org/
Tipo de Dato Abstracto, https://sites.google.com/site/programacioniiuno/temario/unidad-2---tipo-abstracto-de-dato/tipo-de-dato-abstracto