ANALIZADOR DE MAPAS PARA DETERMINAR EL MENOR CONSUMO DE COMBUSTIBLE.

Resumen (Summary).

We request the creation of a map analyzer that indicates the lowest fuel consumption, displays it on the console and generates reports either in pdf or jpg format to be able to show it graphically.

During the creation of the project positions were developed along the development, from using json structures to changing all the code for the reading of the file, I would like to say that the project was finished, however it would be incorrect that statement because it only reached the point of creating single lists to store the land and double lists to store the positions, thus creating a list of lists, the main objective was to use Dijkstra's model to obtain the route with the lowest possible cost, at the present time the implementation of a finalized program that wishes to use this type of algorithms is very useful (in countries where the road education is correct), since it could be implemented not only to robots in specific as the one requested in the project but also to particular vehicles (although at the present time already there are some of these in use).

Palabras Clave (Keywords).

Terrenos (land).
Posición (position).
Optimización de combustible (Fuel optimization).
Mapa con el menor consumo de combustible (Map with the lowest fuel consumptinon).
Grafica (imagen del terreno analizado) (Graph (image of the analyzed terrain)).

Introducción (Abstract).

Creación de software capaz de recibir varios mapas para su análisis a través de la estructura de xml, nombrados con el atributo terrenos, estos deberán ser insertados en listas y nodos, para ser procesados y poder analizar el mejor recorrido siendo este en donde se encuentre el menor gasto de combustible, a su vez este reporte debe ser generado ya sea en formato de pdf o jpg indicando el nombre del terreno, las coordenadas a seguir y el gasto de combustible generado durante el trayecto.

Cabe mencionar que los archivos xml traen predefinidas las etiquetas de terreno, dimensión, posicioninicio, posicionfin, y el listado con las coordenadas y el gasto de combustible, para el procesamiento del mejor camino se buscó utilizar el método de Dijkstra para poder encontrarlo, posterior a ser encontrado ser almacenado dentro de otra lista y esto enviarlo a un método utilizando la función graphviz para realizar el mapeo de forma visual.

Desarrollo del tema.

1.- Lectura de archivos xml.

Para esto se utilizó ElementTree que es un método implementado por Python para la lectura de los archivos con la extensión xml, junto con ciclos for anidados para luego introducirlos a las listas.

2.- Listas y nodos.

Se utilizaron diferentes métodos para almacenar la información recolectada del archivo xml, esto con el objetivo de poder realizar el análisis del mejor recorrido.

3.- Entre las principales posturas que se tomarían para lograr obtener el camino mas eficiente se encontraba la realización del modelo Dijkstra o el rowmajor colmajor.

4.- Para la visualización grafica del camino mas optimo se optaría por la implementación del proceso graphviz el cual puede ser implementado tanto para crear solo las imágenes de los nodos como para generar pdf’s, entre varias otras opciones.

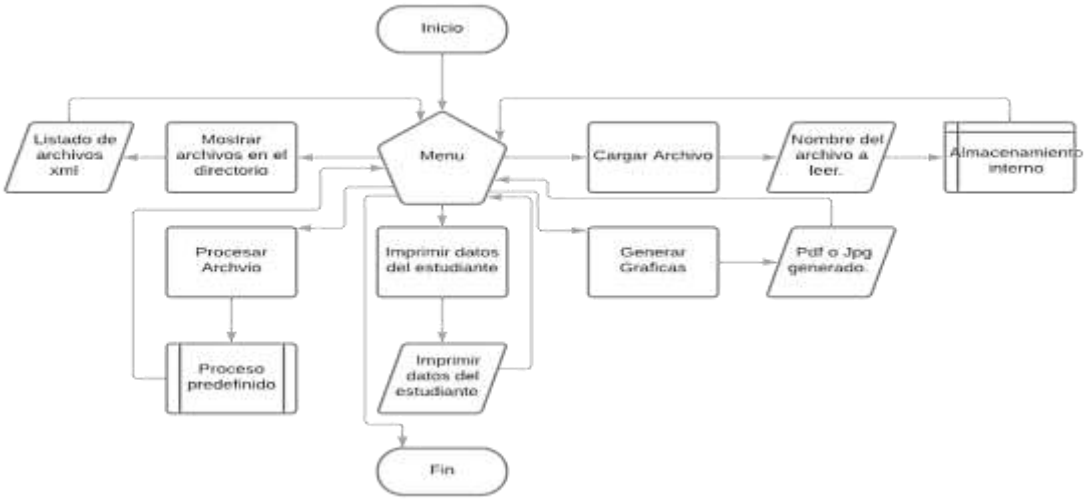

Figura 1. Diagrama de flujo, idea de cómo funcionaria el programa.
Fuente: Elaboración propia.

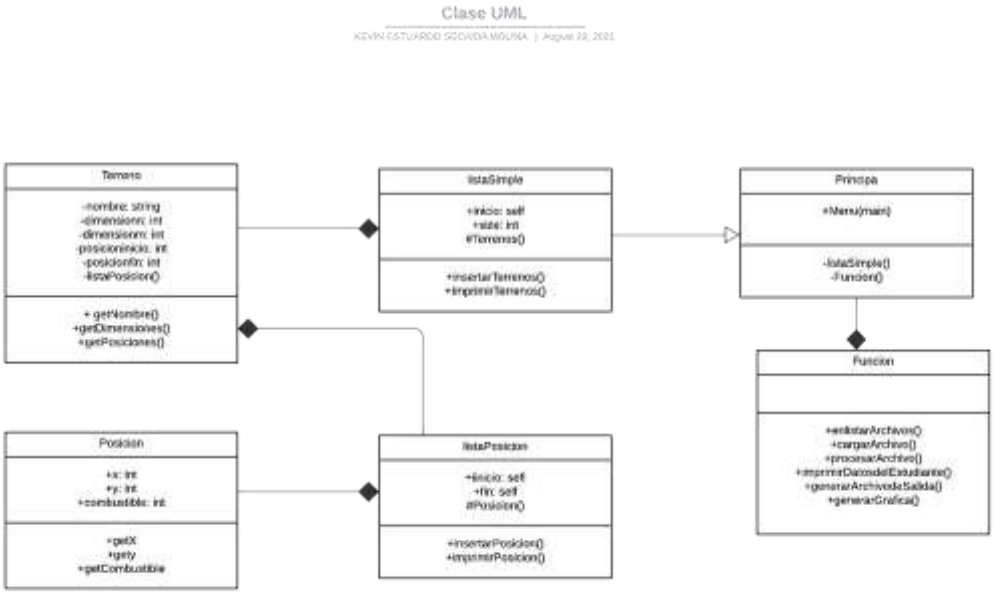

Figura 2: Diagrama de clases, bosquejo de la idea del proyecto.
Fuente: Elaboración propia.

Conclusiones.

La implementación de TDA’s al momento de trabajar con listas simples y dobles facilitan su comprensión e implementación al momento de ejecución.

El uso de listas dobles, dentro de listas simples es una manera más rápida entorno a la velocidad con la que el compilador puede analizar lo solicitado.