**UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS**

Ingeniería de Software

Ciencias de la Computación

Complejidad Algorítmica - CC184

**Informe del Trabajo Parcial**

Sección: CC184

Profesor: Luis Canaval Sánchez

**Integrantes:**

|  |  |  |
| --- | --- | --- |
| **N°** | **NOMBRES:** | **CÓDIGO:** |
| **1.** | Marcelo Guerrero Iparraguirre | u202012258 |
| **2.** | Sofia Gabriel Miranda Cardenas | u20191c439 |

**2023-01**

# Índice

Contenido

[Índice 1](#_Toc139027892)

[Descripción del problema: 3](#_Toc139027893)

[Descripción del Conjunto de datos 3](#_Toc139027894)

[Propuesta 4](#_Toc139027895)

[Diseño de Aplicativo 7](#_Toc139027896)

[Análisis del Algoritmo 9](#_Toc139027897)

[Validación de resultados y pruebas 9](#_Toc139027898)

[Enlace del repositorio de trabajo 12](#_Toc139027899)

[Conclusiones 12](#_Toc139027900)

[Referencias 12](#_Toc139027901)

[Tabla de Ilustraciones 13](#_Toc139027902)

# Descripción del problema:

La problemática que se desarrollará en este proyecto es el encontrar las rutas más rápidas y menos propensas a accidentes dentro de Madrid.

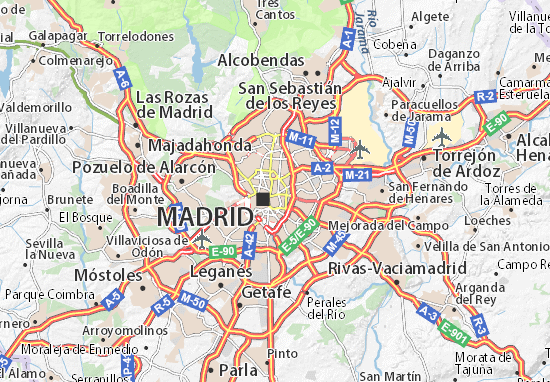


Ilustración 1.Imagen referencial de la zona.

Como podemos apreciar, hay múltiples rutas a considerar para alcanzar a cada vértice del grafo formado, así mismo, hay múltiples algoritmos que nos ayudarán a determinar los recorridos mediante diferentes técnicas de programación, llámese Backtracking o Dijkstra. Además, se hará uso de una interfaz visual para el mapa y la visualización del camino óptimo el cual se va a diferenciar con un color diferente al resto.

# Descripción del Conjunto de datos

Para el presente trabajo se eligió el dataset “Accidentes en Madrid”, que proporciona información sobre accidentes vehiculares en Madrid que se originan en la vía pública y son informados a la policía.

Este dataset cuenta con valores únicos y tiene 32 columnas que albergan datos de número de vehículos, número de bajas, longitud, latitud, gravedad del accidente, fecha, día de la semana, tiempo, etc. Datos que usaremos para desarrollar una solución para la problemática planteada.

Para el desarrollo del programa, se hará uso de las variables index\_accidente, lugar, nodo1, nodo2, peso, Accident\_severity,date,time,speed\_limit y weather\_condition.

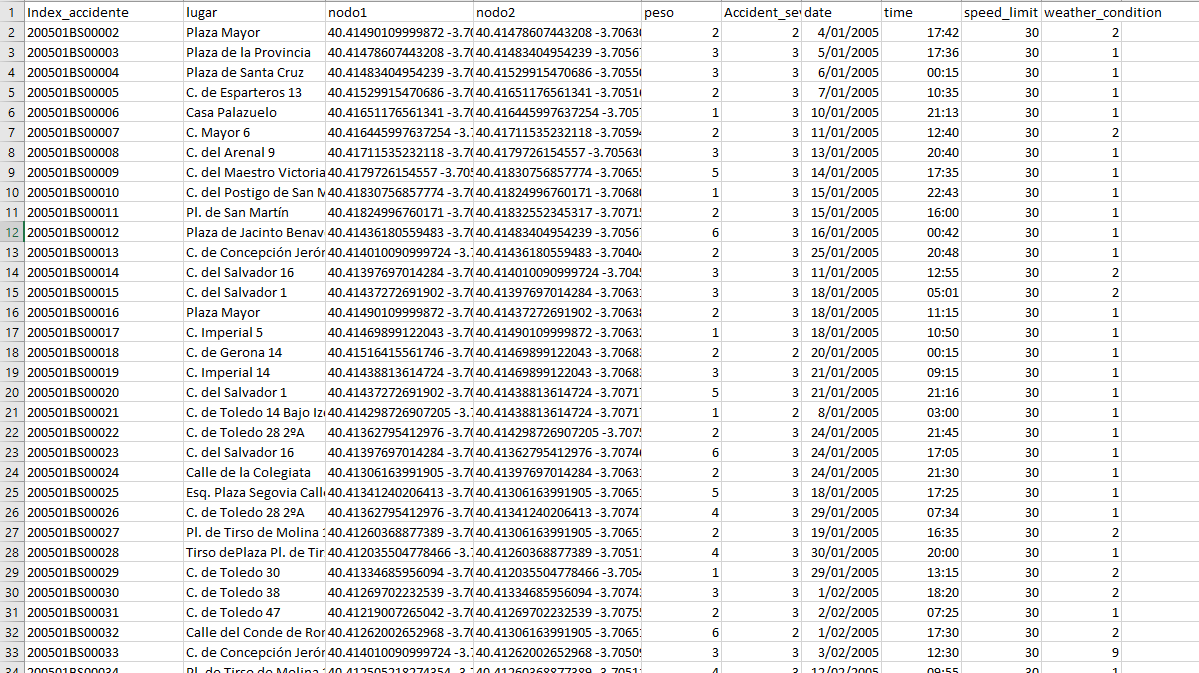


Ilustración 2.Dataset modificado.

# Propuesta

Vamos a desarrollar un aplicativo que en base a la data recolectada encuentre las rutas más rápidas y menos propensas a que ocurran accidentes de tránsito en Madrid, que desarrollaremos haciendo uso de los algoritmos de Djkstra y Ford-Fulkerson, haciendo uso del lenguaje de programación Python.

En la imagen 3, se visualiza que el grafo recorre cada calle de Madrid tomando en cuenta las intercepciones de las calles.

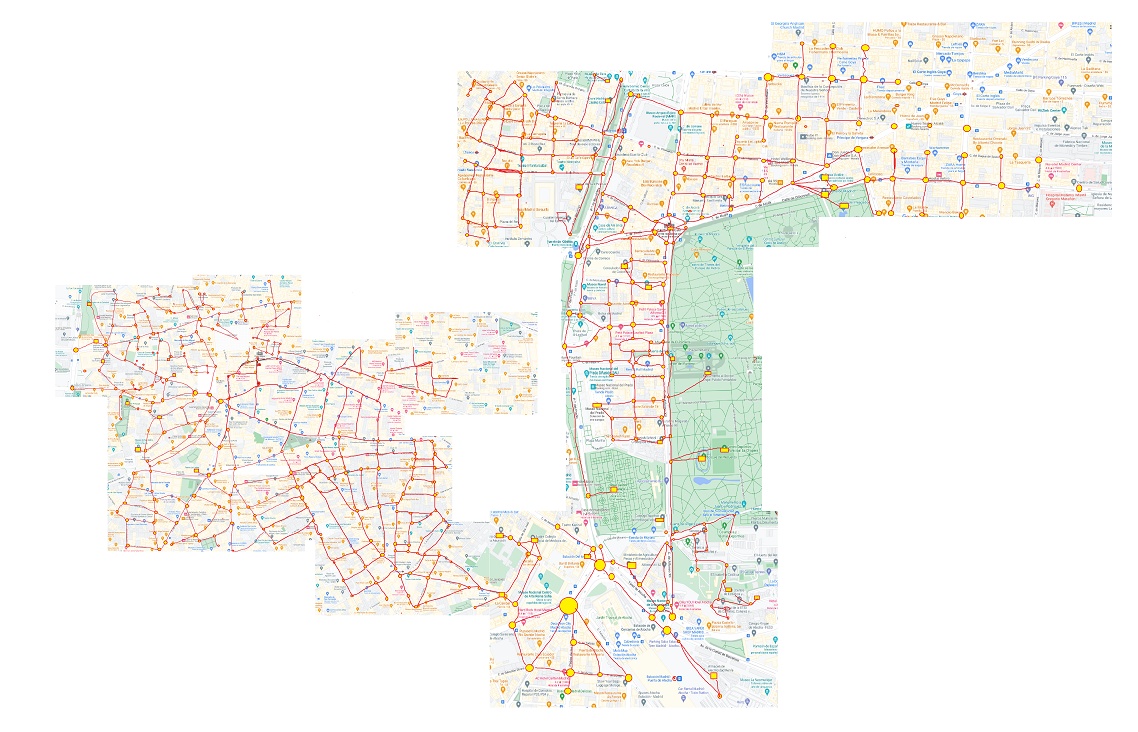
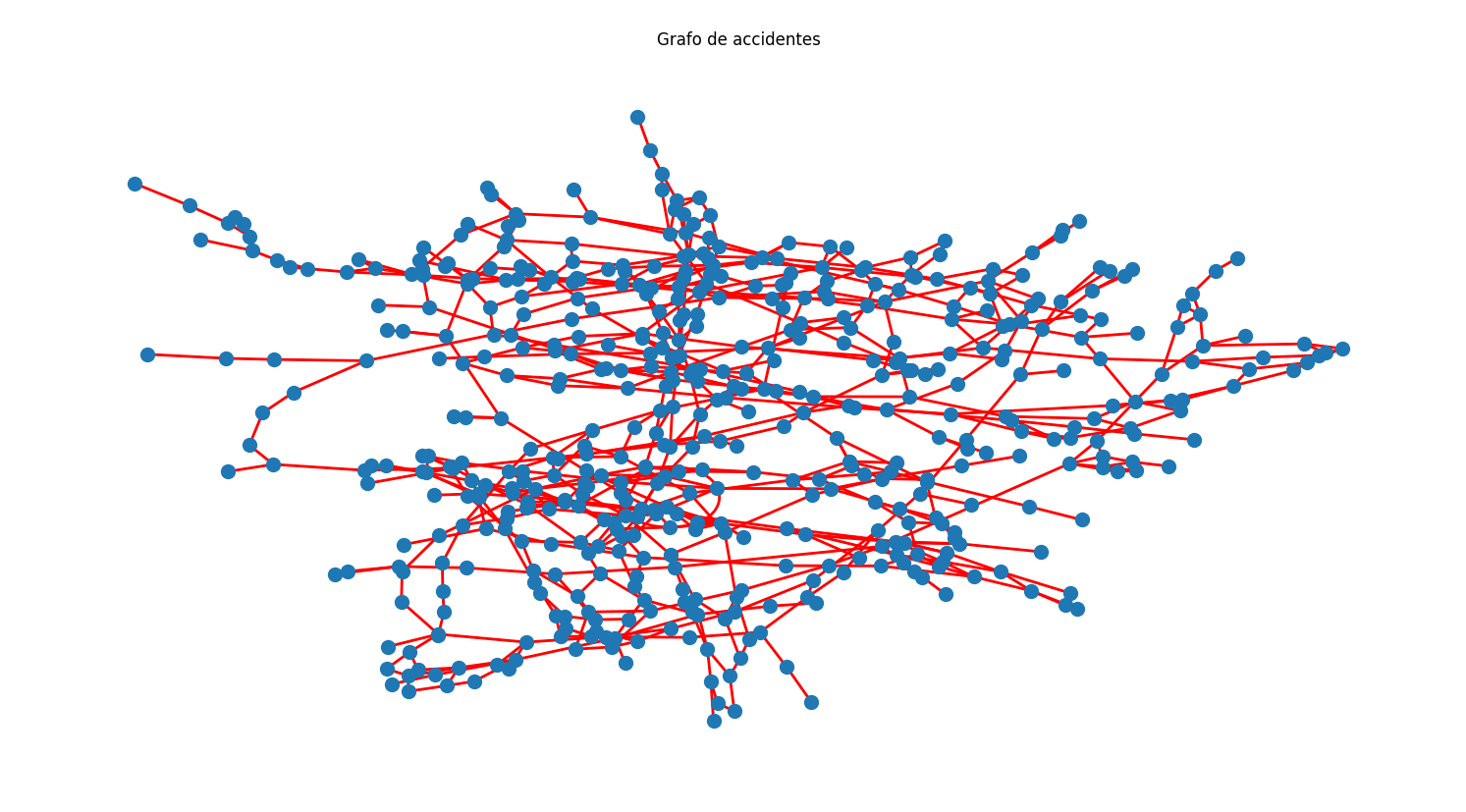


Ilustración 3. Intercepciones del dataset.

En la imagen 4, se visualiza el grafo con los datos que se encuentran dentro del dataset dentro del dataset, cada nodo representa un lugar específico o una intercepción de una calle con otra.



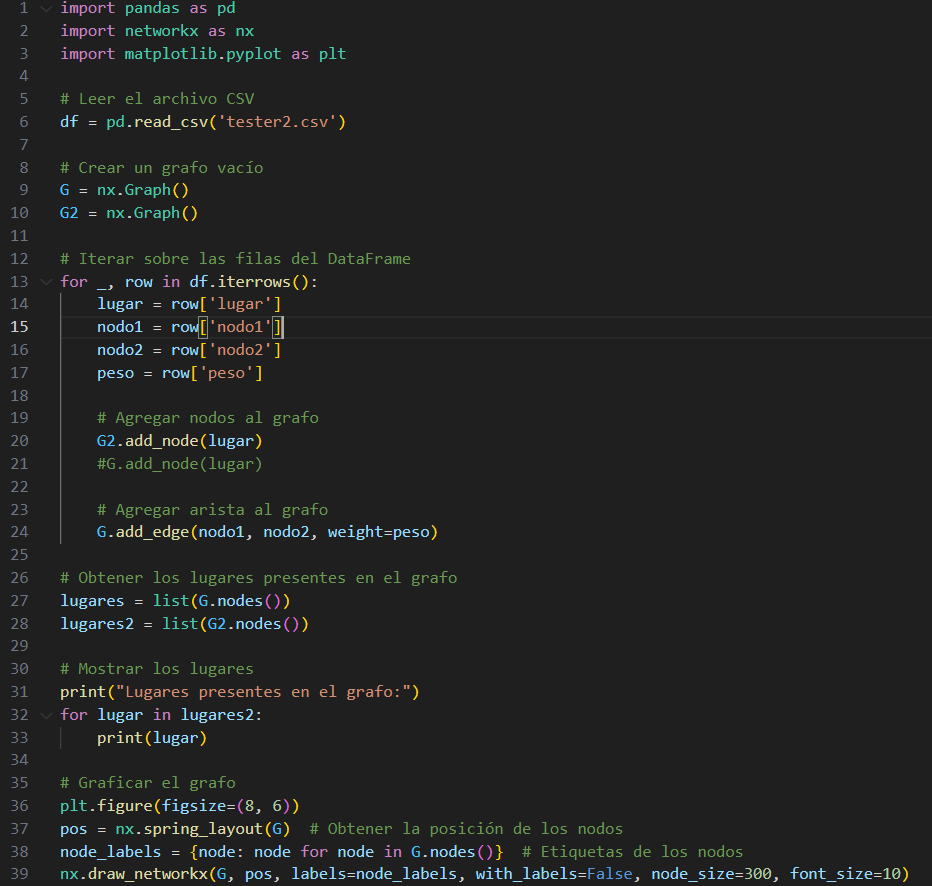
Ilustración 5.Visualización del grafo con los datos probados.

Ilustración 4. código del grafo.

# Diseño de Aplicativo

Para el desarrollo de este proyecto, se esta haciendo uso del lenguaje de programación Python.

Python:  Lenguaje de programación dinámico, presenta una sintaxis legible y fácil de usar, presenta un entorno interactivo que nos permitirá realizar diversas pruebas en nuestro proyecto, posee una variedad de estructura de datos que se pueden manejar de forma sencilla (Marzal & Gracia,2009).



Ilustración 6.Python.

Se implementará una interfaz gráfica, para que se visualice las funciones del programa y que nuestros usuarios puedan hacer uso de ellas. Por esa razón, se hará uso de las librerías:

pandas (pd): pandas es una biblioteca de análisis de datos que proporciona estructuras de datos y funciones para manipular y analizar conjuntos de datos. Es muy útil para trabajar con datos tabulares, como hojas de cálculo o bases de datos.

networkx (nx): networkx es una biblioteca de Python que se utiliza para crear, manipular y estudiar la estructura, dinámica y funciones de redes complejas. Proporciona herramientas para la creación de grafos, cálculo de métricas de red, algoritmos de búsqueda, visualización de redes, entre otros.

PySimpleGUI (sg): PySimpleGUI es una biblioteca de interfaz de usuario (UI) de Python que proporciona una forma sencilla de crear interfaces gráficas de usuario para tus programas. Utiliza una sintaxis fácil de entender y ofrece una variedad de elementos de UI para crear ventanas, botones, campos de entrada, entre otros.

matplotlib.pyplot (plt): matplotlib es una biblioteca de visualización de datos en Python. pyplot es un módulo de matplotlib que proporciona una interfaz similar a MATLAB para crear gráficos y visualizaciones. Se utiliza para generar gráficos, diagramas, histogramas, dispersión y otros tipos de visualizaciones.

matplotlib.backends.backend\_tkagg.FigureCanvasTkAgg: Esta parte importa una clase específica llamada FigureCanvasTkAgg del módulo matplotlib.backends.backend\_tkagg. Esta clase se utiliza para integrar las visualizaciones de matplotlib en interfaces gráficas creadas con la biblioteca Tkinter.

El usuario podrá ingresar al lugar a donde quiere dirigirse, desplegándose los lugares en donde se puede desplazar dentro de Madrid, tendrá que insertar el punto de origen y el punto de destino, y una vez hecho esto, el programa le mostrará la ruta más optima, tomando en cuenta los lugares donde se producen más accidentes y mostrando la ruta más segura, como se visualiza en la imagen 7.

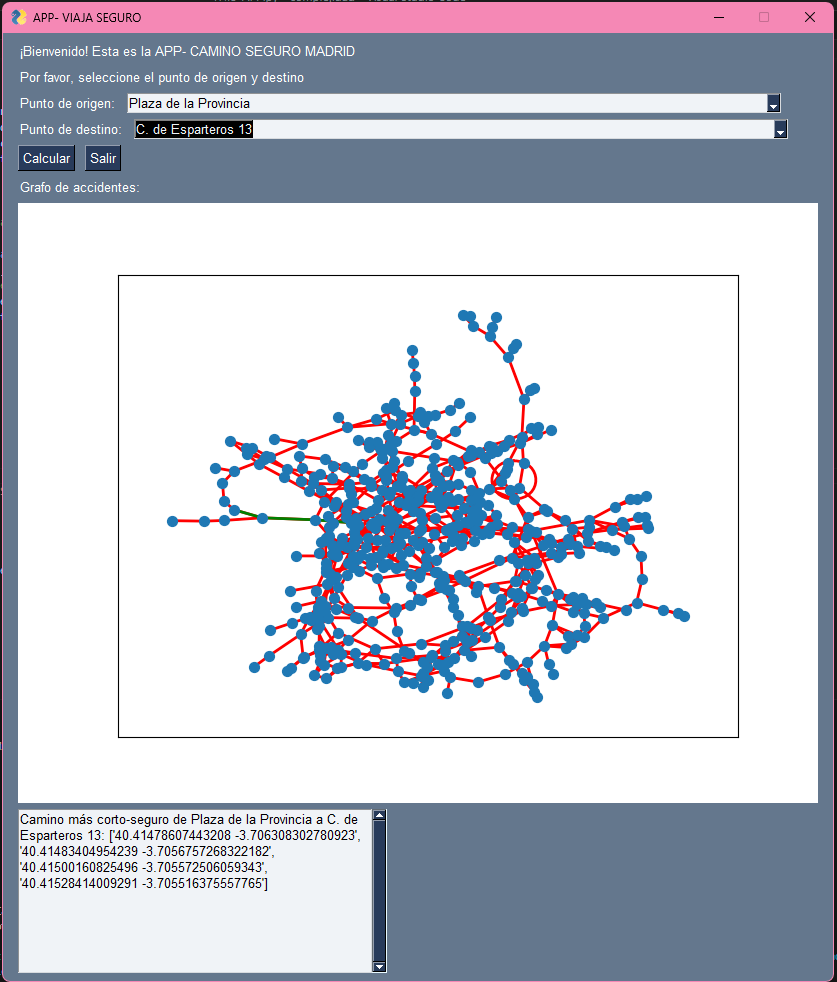


Ilustración 7.Codigo en ejecución, busca el camino más corto de un punto A a un punto B.

# Análisis del Algoritmo

Algoritmo de Dijkstra

Este es uno de los algoritmos más usados para encontrar el camino más corto en un grafo ponderado desde un nodo de origen hacia todos los demás nodos. Es un grafo eficiente dentro de los grafos con pesos no negativos.

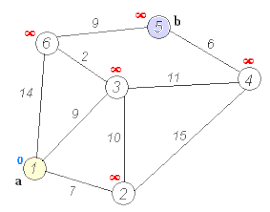


Ilustración 8.ejemplo de desarrollo del algoritmo de Dijkstra.

Algoritmo de Ford-Fulkerson

Este algoritmo es usado para resolver problemas de flujo máximo en una red de flujo.

El algoritmo de Ford-Fulkerson utiliza la idea de caminos aumentantes para encontrar iterativamente un flujo máximo en una red. Un camino aumentante es un camino desde el nodo fuente hasta el nodo sumidero en la red residual, donde la capacidad residual de al menos un arco es mayor que cero.

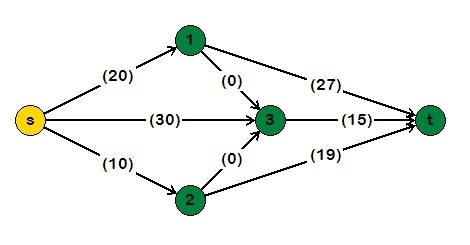


Ilustración 9. Ejemplo resuelto del algoritmo de Ford-Fulkerson.

# Validación de resultados y pruebas

El usuario al ingresar a la aplicación podrá elegir a que parte de Madrid quiere ir, siempre y cuando este dato ingresado este guardado dentro del grafo.



Ilustración 10, punto de origen y destino.

Y esto podrá verlo, desplegando la barra de opciones que hay en la aplicación.

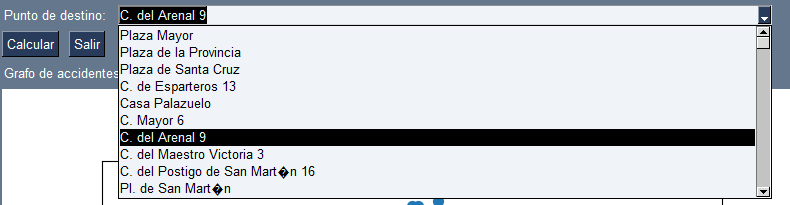


Ilustración 11, lugares de Madrid.

Y podrá visualizar el los nodos contenidos en el grafo.

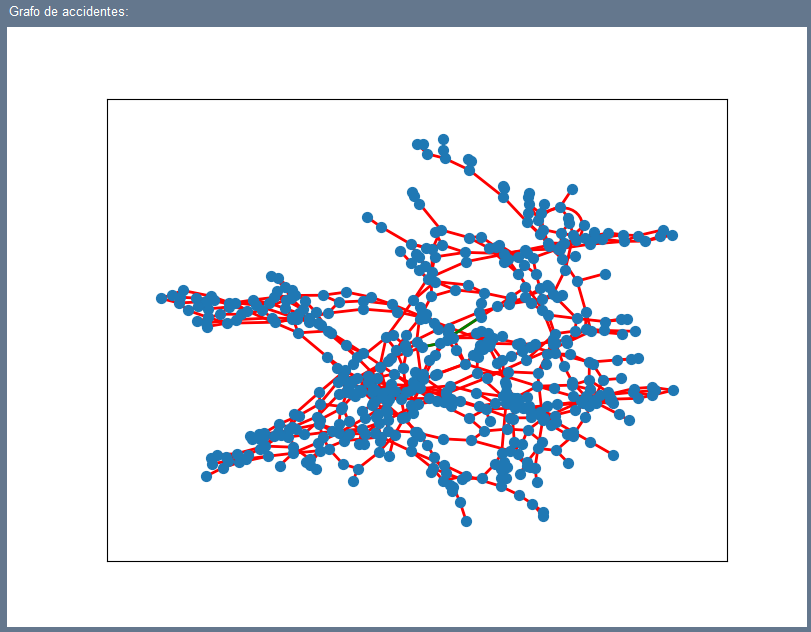


Ilustración 12, grafo de accidentes.

Y luego de elegir a donde quiere ir, solo tiene que dar click en calcular y el programa ejecutara el código que se encarga de buscar el camino más seguro en base a la cantidad de accidentes que se originan en cada nodo.

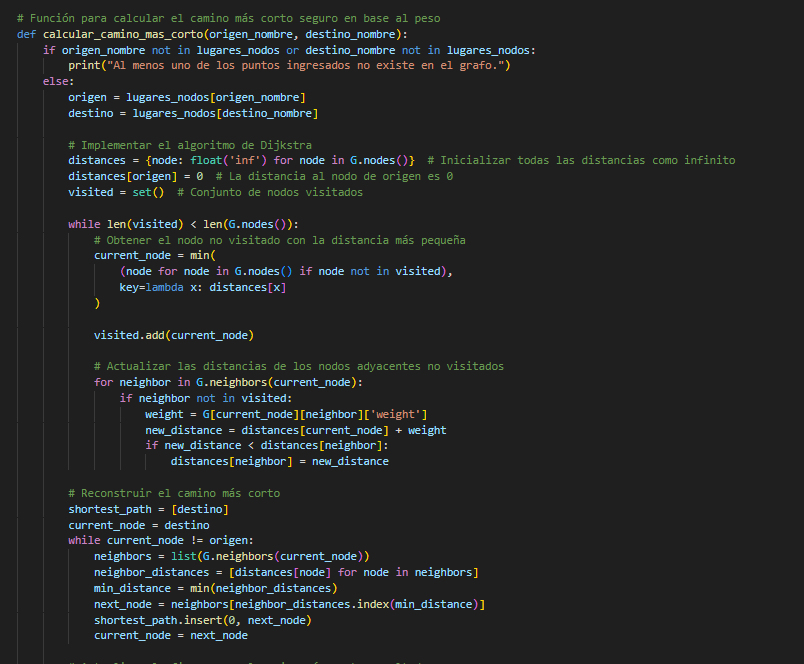


Ilustración 13, código.

Y se podrá ver la ruta a seguir y en el grafo saldrá la ruta marcada de color verde.

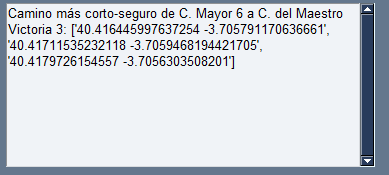


Ilustración 14, camino seguro.

Representando ['40.416445997637254 -3.705791170636661', '40.41711535232118 -3.7059468194421705', '40.4179726154557 -3.7056303508201'], cada lugar albergado en cada nodo.

# Enlace del repositorio de trabajo

|  |
| --- |
| https://github.com/jisso123/COMPLEJIDAD-TRABAJO.git |

# Conclusiones

En conclusión, el algoritmo de Dijkstra es un método eficiente y ampliamente utilizado para encontrar el camino más corto entre dos nodos en un grafo ponderado. Al utilizar una estrategia de búsqueda exhaustiva y actualizar las distancias de manera iterativa, el algoritmo encuentra de manera confiable el camino óptimo.

El algoritmo de Dijkstra es especialmente útil en problemas de optimización de rutas, como la navegación en mapas, el enrutamiento de paquetes en redes de comunicación y la planificación de rutas en logística y transporte.

# Referencias

1. Marzal Varó, A., & Gracia Luengo, I. (2009). Introducción a la programación con Python. Universitat Jaume I.

# Tabla de Ilustraciones

[Ilustración 1.Imagen referencial de la zona. 3](#_Toc139027927)

[Ilustración 2.Dataset modificado. 4](#_Toc139027928)

[Ilustración 3. Intercepciones del dataset. 5](#_Toc139027929)

[Ilustración 5.Visualización del grafo con los datos probados. 6](#_Toc139027930)

[Ilustración 4. código del grafo. 6](file:///C:\Users\user\Desktop\INFORME_COMPLEJIDAD_HITO%202.docx#_Toc139027931)

[Ilustración 6.Python. 7](#_Toc139027932)

[Ilustración 7.Codigo en ejecución, busca el camino más corto de un punto A a un punto B. 8](#_Toc139027933)

[Ilustración 8.ejemplo de desarrollo del algoritmo de Dijkstra. 9](#_Toc139027934)

[Ilustración 9. Ejemplo resuelto del algoritmo de Ford-Fulkerson. 9](#_Toc139027935)

[Ilustración 10, punto de origen y destino. 10](#_Toc139027936)

[Ilustración 11, lugares de Madrid. 10](#_Toc139027937)

[Ilustración 12, grafo de accidentes. 10](#_Toc139027938)

[Ilustración 13, código. 11](#_Toc139027939)

[Ilustración 14, camino seguro. 11](#_Toc139027940)