****

**FACULTAD DE INGENIERÍA DE SOFTWARE**

**CURSO:**

Complejidad Algorítmica

**PROFESOR:**

Luis Martín Canaval Sánchez

**SECCIÓN:**

WS6A

**INTEGRANTES:**

**NOVIEMBRE 2022-2**

**ÌNDICE**

1. Presentación…………………………………………………………………..1

2. Descripción del problema……………………………………………………3

3. Descripción del conjunto de datos (dataset)............................................5

4. Propuesta……………………………………………………………………...6

5. Diseño del aplicativo…………………………………………………………7

6. Validación de resultados y pruebas………………………………………...11

7. Conclusiones………………………………………………………………….12

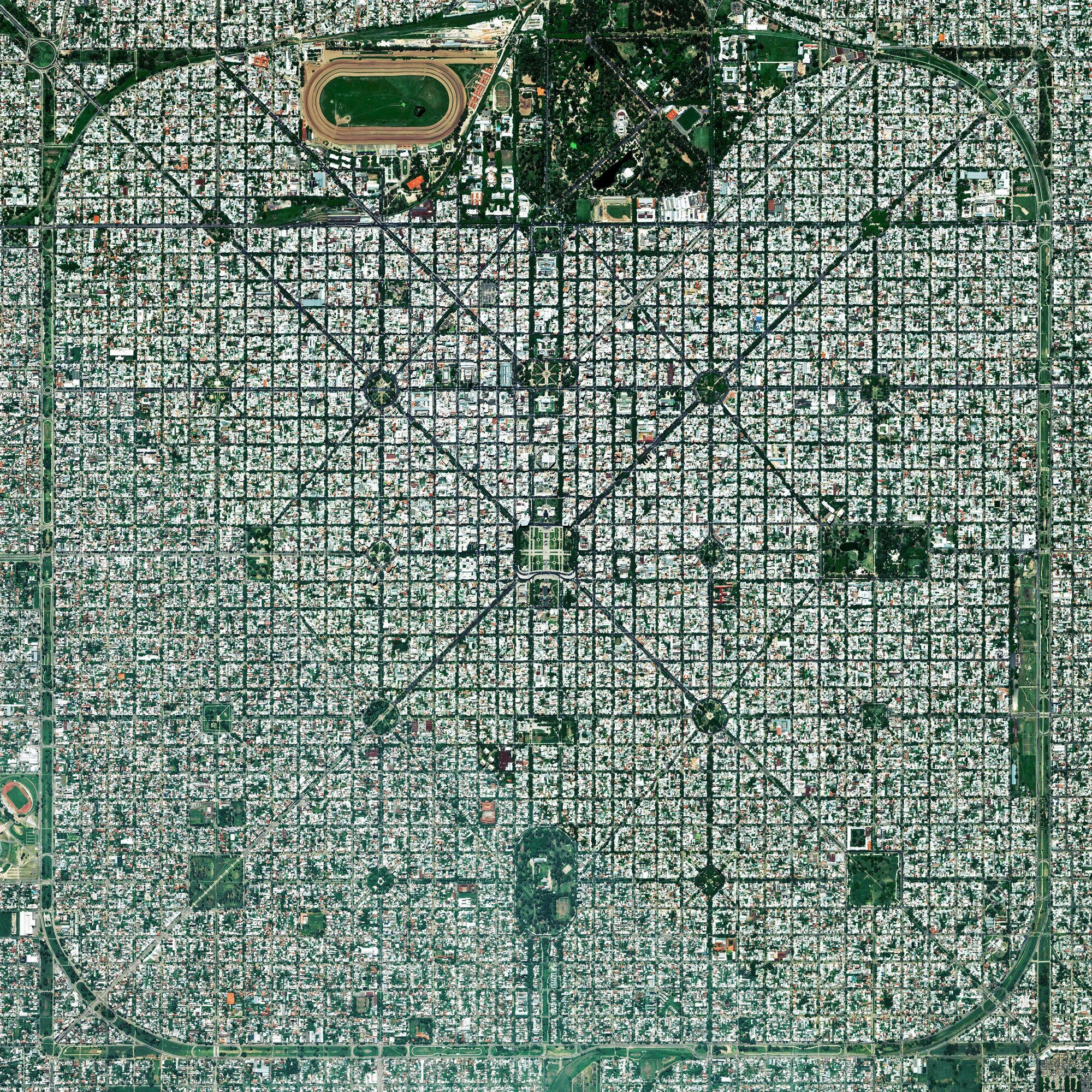
8. Referencias bibliográficas…………………………………………………...13

1. **Descripción del problema.**

En el curso de complejidad algorítmica se presentan temas que dan soluciones a diversas problemáticas de la actualidad. Existen varias soluciones a muchos problemas, como el problema del tráfico. Dichos algoritmos implementados demuestran ser útiles e inesperadamente los utilizamos a diario a través de aplicaciones. Cabe resaltar que uno de los más utilizados es Dijkstra, tal que cumple el objetivo principal de “establecer los criterios para alcanzar el objetivo planteado y desarrollar un modelo con una estructura jerárquica utilizando diversos factores o atributos que contribuyan para la toma de decisiones” (Pavas, A. 2016). Cabe resaltar que dicho algoritmo será uno de los principales a utilizar en este proyecto.

Un gran ejemplo de muestra es Waze. La aplicación más grande en el mundo que se encarga de evitar gastar tiempo y dinero al dirigirse de un lugar a otro, las APP recomiendan rutas alternas de llegada al destino, en base a los datos que recopila de todos sus usuarios. Por lo tanto, es útil mencionar lo que la Universidad de Chile afirma que “con el avance de la era de la información, los algoritmos se apropian cada vez más de los procesos tecnológicos y, por lo tanto, de nuestra vida diaria.” En sí es bastante lógico porque cada vez vivimos más ligados a la tecnología.

Para lograr cumplir con los objetivos del curso se pretende elaborar una aplicación similar a Waze, la cual se encarga de encontrar la ruta más corta entre dos puntos de una ciudad seleccionada. La ciudad se pretende mostrar mediante un grafo que represente detalles del presente trabajo práctico. Para dar solución al caso mostrado se utilizan (datasets) tanto como: “edges”, “nodes” y “streets”, el cual en dicho orden representan calles, intersecciones y nombre de las calles. Nuestra problemática identificada es la ciudad de la Plata, Argentina, es una ciudad comúnmente conocida por sus avenidas diagonales tal como se observa en la **imagen 1.**



**Imagen 1 Ciudad de La Plata, Argentina.**  Recuperado de: <https://www.archdaily.pe/pe/948980/trazas-urbanas-17-ciudades-vistas-desde-arriba/5f77388b63c017ae22000958-trazas-urbanas-17-ciudades-vistas-desde-arriba-imagen>

Como podemos evidenciar en la imagen 1 se presenta la estructura y modelo de las calles de la ciudad de la plata, comúnmente conocida por tener una construcción multitudinaria de calles. Hecho por el cual se presenta el problema del tráfico, la congestión de vehículos en las principales avenidas en su mayoría es un factor que altera la tranquilidad de los transeúntes y vecinos que habitan la ciudad. Uno de los grandes detalles que en su mayoría es útil de hacer recalcar lo que Agenda de Reflexión en Arquitectura, Diseño y Urbanismo menciona “La Plata es última ciudad argentina planificada antes de los automóviles”.

En vista a ello, se pretende dar una gran alternativa de solución a los transeúntes con el diseño de **SmartGo** una aplicación que se encarga de calcular el tiempo, tránsito y la distancia mínima que se emplea en trasladarse desde un lugar de la ciudad a otro, calculando vías alternas y posibles caminos a seguir.

1. **Descripción del conjunto de datos (dataset).**

Para el desarrollo de este trabajo, conseguimos los tres archivos de datos que se mencionaron previamente para abarcar la problemática planteada en la introducción del proyecto, con un total de 1500 datos como lo especifica la guía. Esperamos con la inclusión de esta información mediante grafos y diversos algoritmos poder estructurar una solución eficaz y eficiente.

**nodes.txt:** este archivo de texto posee la información referencial a los nodos de nuestro futuro grafo, los cuales representarán las intersecciones entre dos calles. El formato del archivo es de cuatro columnas, las dos primeras son las coordenadas del lugar en específico, mientras que las dos últimas columnas de datos vienen a ser los identificadores de las calles que se interceptan en dicho nodo.

Ejemplo: -34.93315482 -57.97989967 (coordenadas) 8 (id de la primera calle de la intercepción) 11 (id de la segunda calle de la intercepción)

**edges.txt:** el siguiente archivo de texto contiene la información de los bordes o en este caso las aristas que conformarán el grafo. El formato del archivo posee únicamente dos columnas, las cuales hacen referencia a los indicadores de las dos calles entre las cuales se encuentra dicha arista.

Ejemplo: 10 (id de la primera calle de donde inicia la arista) 11 (id de la segunda calle por donde termina la arista)

**streets.txt:** este último archivo de texto es el más sencillo de todos, en él se encuentran los nombres de cada una de las distintas calles que se encuentran en la zona que elegimos. Dichas calles son las que conforman los otros dos archivos de texto presentados.

Ejemplo: Avenida Antártida Argentina (nombre de una de las calles que se encuentran en los datasets nodes.txt y edges.txt)

Con la información recolectada empezaremos a trabajar el proyecto para nuestra primera presentación, en la cual indicaremos la estructura que ideamos para almacenar todos estos datos en un grafo.

**¿Cómo se va a utilizar en el aplicativo?**

Los datos recopilados serán empleados en la elaboración de nuestro programa, utilizándolos para la elaboración del mapa utilizando los algoritmos necesarios, en el cual se realizarán las búsquedas de caminos.

Con la información del archivo nodes y edjes, iremos creando nuestro grafo el cual representará y se mostrará en la aplicación como el mapa el cual abarca la zona elegida. También en la App mostraremos una lista de las calles que tenemos en nuestra data, siendo las calles que componen el alcance de nuestro mapa.

1. **Propuesta.**

Nuestra propuesta como grupo acerca del trabajo final es cumplir con el objetivo de crear una aplicación con el cual un usuario pueda seleccionar las rutas por donde normalmente transita. Lo que hará nuestra App será mostrarle cuál de todas las rutas posibles es la más corta y eficiente, considerando distintos tipos de factores que influyen a la elección de esta.

El objetivo principal del programa implementado es dirigir al usuario desde un cierto punto de la ciudad a otro evitando coincidir con el tráfico y de esa manera ahorrar tiempo de llegada.

Las técnicas que utilizaremos para el desarrollo de esta App serán las mismas que hemos estado estudiando durante el ciclo actual, el uso de Backtracking, Dijkstra y otros algoritmos que sean necesarios, además se investigará a los distintos algoritmos de tipo topológico que nos ayude a realizar las acciones que hará nuestra aplicación.

Los algoritmos de búsqueda de ruta más corta serán los principales actores de nuestra app. Algoritmos como Djikstra será uno de los principales a utilizar, ya que este se encarga de encontrar el camino más corto en nodos los cuales tienen un peso en específico, estos pesos vendrían a ser la longitud de cada calle en cualquier ciudad, esto nos ayudará a calcular la ruta más corta entre dos puntos que sean puestos por el usuario. Además de la aplicación de diversos algoritmos para el funcionamiento del aplicativo, este también debe ser actualizado constantemente y con respuestas alternativas para el resultado final de la ruta más corta, ya que en la vida real pueden existir distintos factores que dificultan el cálculo de esta, como por ejemplo cierre de calles, accidentes u otros problemas que pueden ocurrir en las pistas.

**Metodología utilizada**

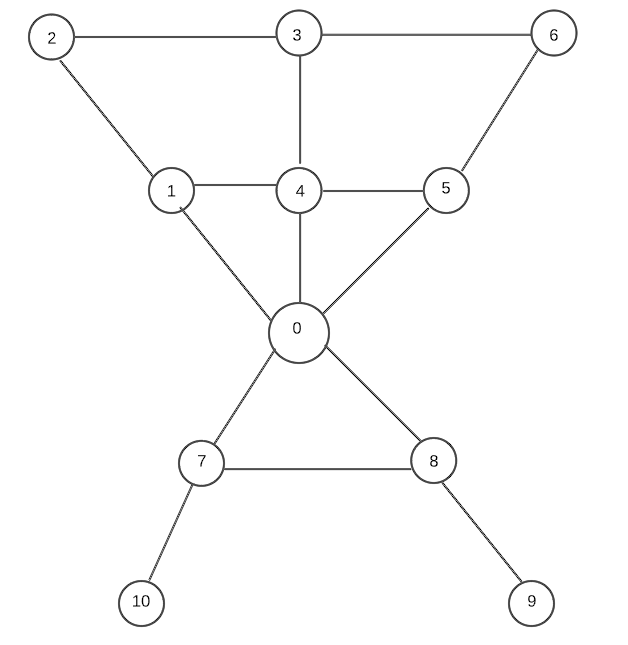
La metodología que utilizaremos será la de investigar y plantearnos cuales pueden ser los algoritmos más eficientes para nuestra aplicación, durante todo nuestro aprendizaje del curso hemos visto distintos tipos de algoritmos que realizan alguna función diferente, en nuestro caso será ver cuáles son los que sirven principalmente para la búsqueda de caminos más cortos en menos tiempo. Estos serán descritos a lo largo de este informe los cuales serán descritos y explicados de manera independiente.

Estos algoritmos nos ayudarán en la realización de nuestra aplicación a lo que también se suman todos los conocimientos que hemos adquirido a lo largo de nuestro curso. Esto además sumado a lo que hayamos estado investigando en foros o videos sobre distintos métodos para realizar la aplicación como son los algoritmos que estos conllevan o la forma en que se armara la aplicación, tanto la forma visual como la matemática.

1. **Diseño del aplicativo.**

Los procesos del diseño del aplicativo que seguiremos para la realización de nuestro proyecto serán el análisis de algoritmos que implementemos a nuestra aplicación. Decidimos el análisis de estos ya que en una aplicación como la nuestra donde lo más importante es que los algoritmos funcionen correctamente, estos deben ser monitoreados constantemente por si hubiera alguna falla o error en el código. Además de las correcciones necesarias que tendrá nuestra aplicación a lo largo de toda su desarrollo.

Como nuestro tema de aplicación se concentra en las calles de ciudades, estas pueden ser interpretadas como grafos con peso, cada calle sería interpretada como una arista de un grafo y los óvalos como vértices del mismo. Esto hace que sea más fácil poder realizar algoritmos y cálculos en base a grafos como explicamos anteriormente, además nuestra aplicación al calcular la ruta más corta entre distintos puntos, hace que utilizar e interpretar las calles como grafos conectados con un peso en específico sea lo mejor para la realización de esta.

En la siguiente imagen podemos ver un ejemplo de un grafo que representa varias calles con distintos puntos de una ciudad, al ser solo un ejemplo faltaría los pesos de cada aristas, los cuales vendrían a ser la longitud de cada calle entre los distintos puntos en que puedan estar conectados.

Para esta imagen hemos tomado como referencia a la **imagen 1** que se explica en la descripción del problema. Este grafo es una forma simple de ver cómo es que están interconectadas las calles de una ciudad, facilitando el entendimiento de la misma.

**Para el desarrollo del programa se utilizarán los siguientes algoritmos a modo de solución.**

**Dijkstra:** Es un algoritmo el cual sirve para determinar el camino más corto entre un vértice de origen hacia otros vértices conectados mediantes aristas con un peso específico. Este fue inventado por Edsger Dijkstra, científico de la computación, de los países bajos.

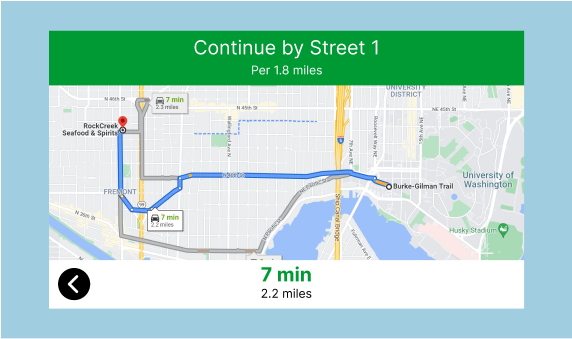
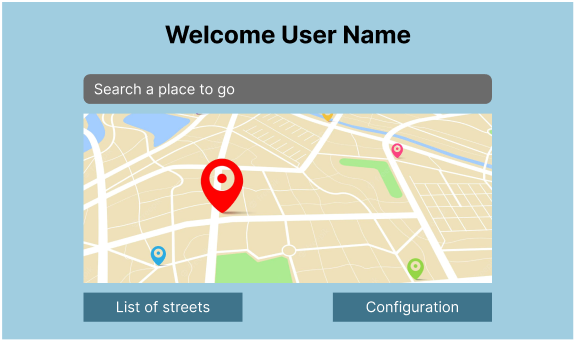
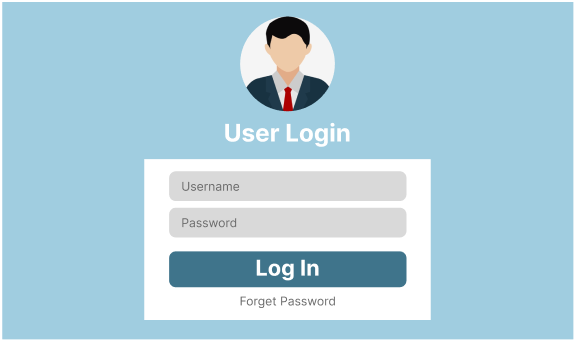
Nosotros pensamos en utilizar este algoritmo para nuestro proyecto ya que al ser un aplicación que calcula la ruta más corta entre las calles de distintos puntos del país, nos será más fácil calcularla ya que cada calle tiene una longitud específica el cual será tomado como un peso y los puntos entre los que se quiere saber la ruta serán tomados como los vértices de un grafo.

**Algoritmo de Bellman-Ford:** Es un algoritmo el cual se busca el camino más corto en un grafo que esté ponderado, es decir un grafo en el que el peso de una de sus aristas puede tener un valor negativo. Una de las principales diferencias de este algoritmo con el de *Djisktra* es que este se puede utilizar en grafos que tengan como peso en sus aristas valores negativos.

Nosotros pensamos en implementar este algoritmo a nuestro proyecto para poder probar distintos métodos de generar la ruta más corta, ya que no queremos sobrecargar a un algoritmo con todas las rutas que se vayan a trazar en la aplicación.

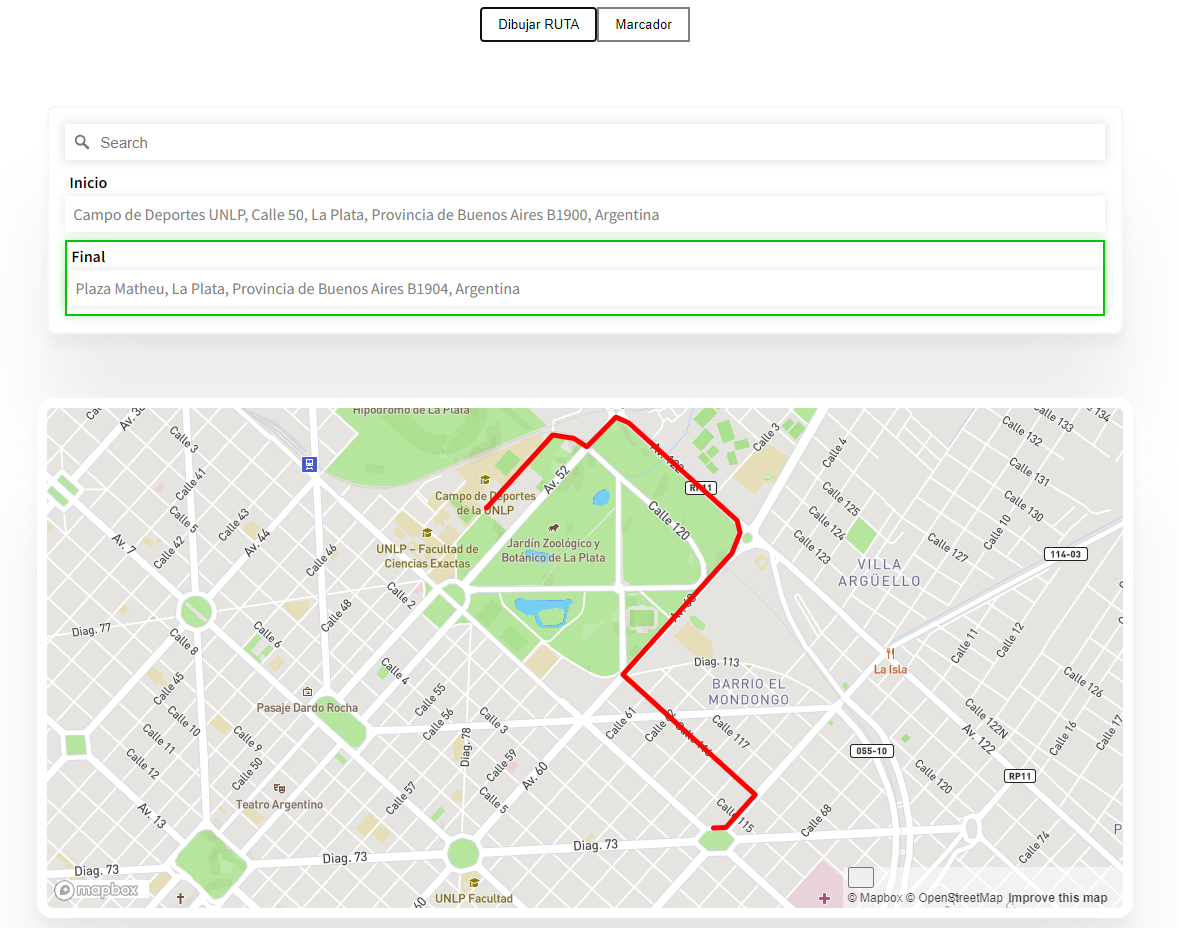
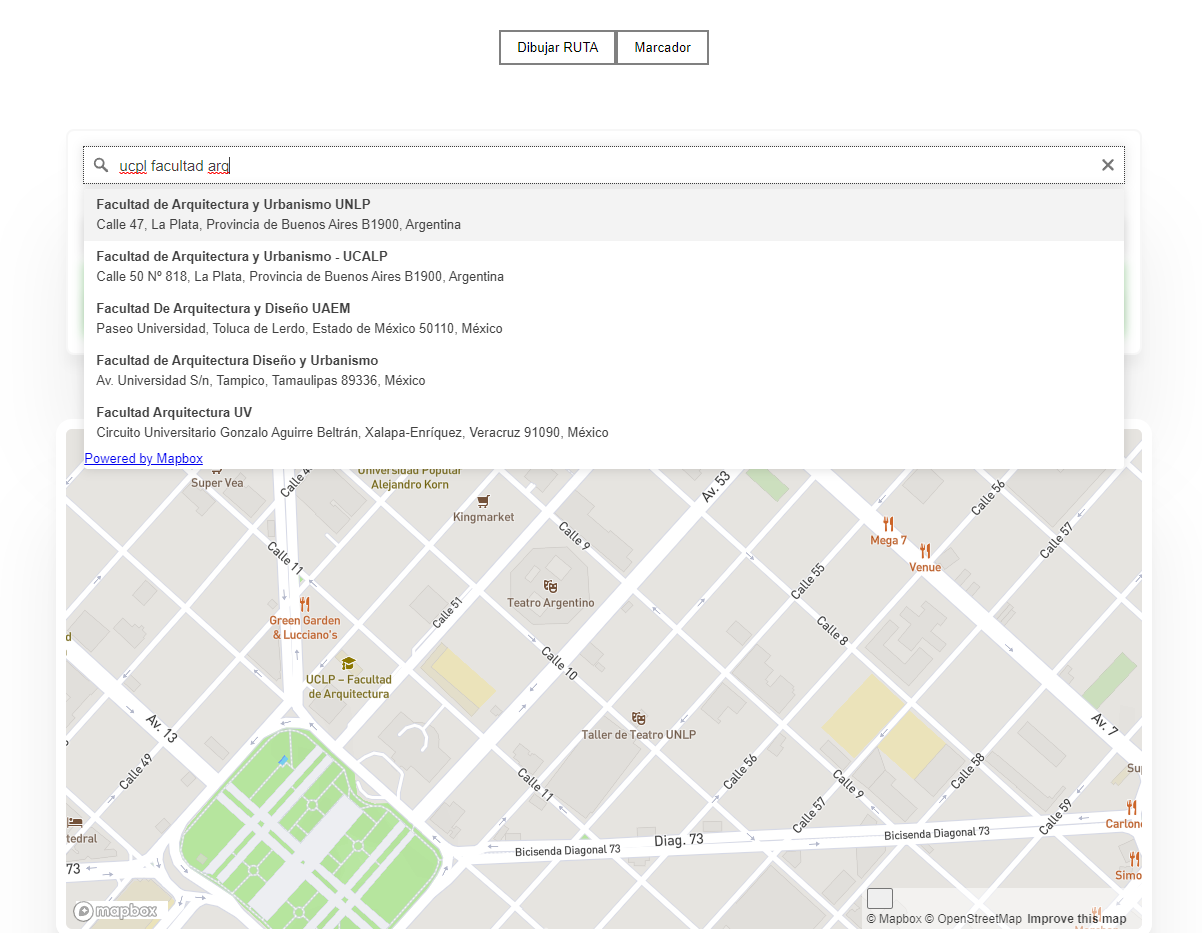
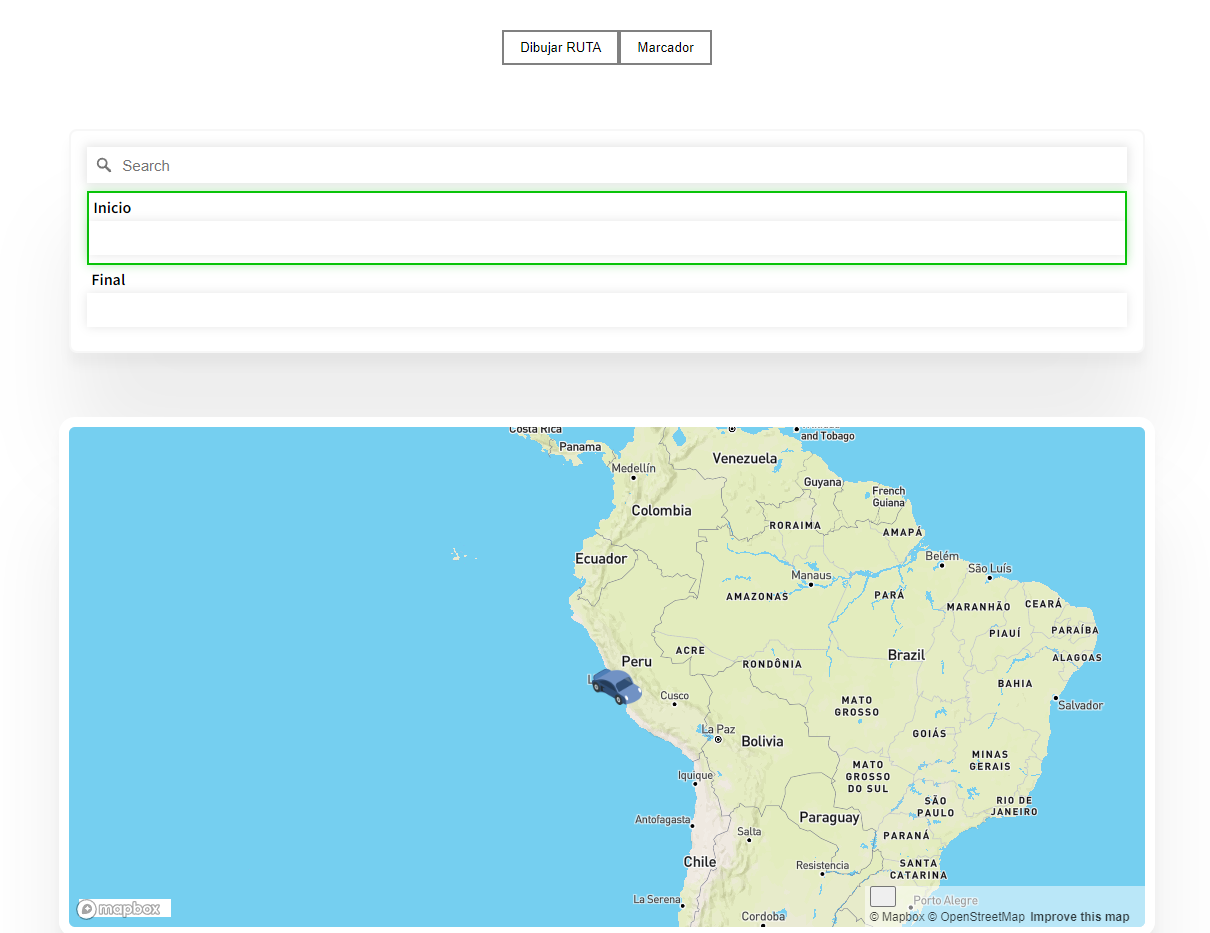
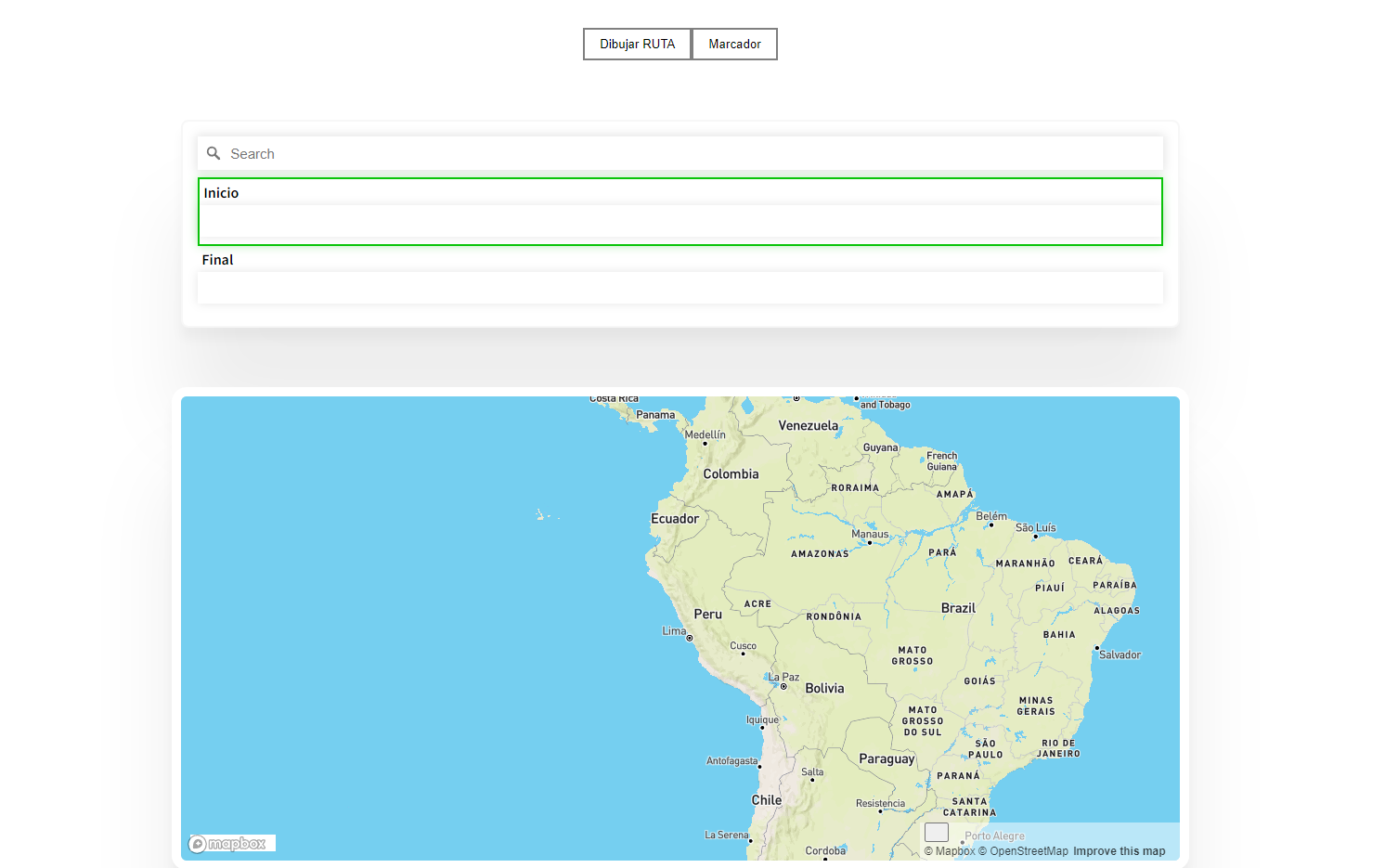
**Algoritmo de Johnson:** Es un algoritmo el cual busca conseguir las rutas más cortas entre los pares de vértices dentro de un grafo dirigido disperso. Una peculiaridad es que permite que se tengan pesos negativos en las aristas. Utiliza el algoritmo Bellman-Ford para eliminar las aristas de peso negativo para después usar Dijkstra en dicho grafo resultante.

**Mockups de modelo de la Aplicación**

****

**Link del figma:** <https://www.figma.com/file/XRAkgpqFRgI95xFFDUThLj/COMPLEGIDAD-DISE%C3%91O?node-id=0%3A1>

**Diseño final de la aplicación**

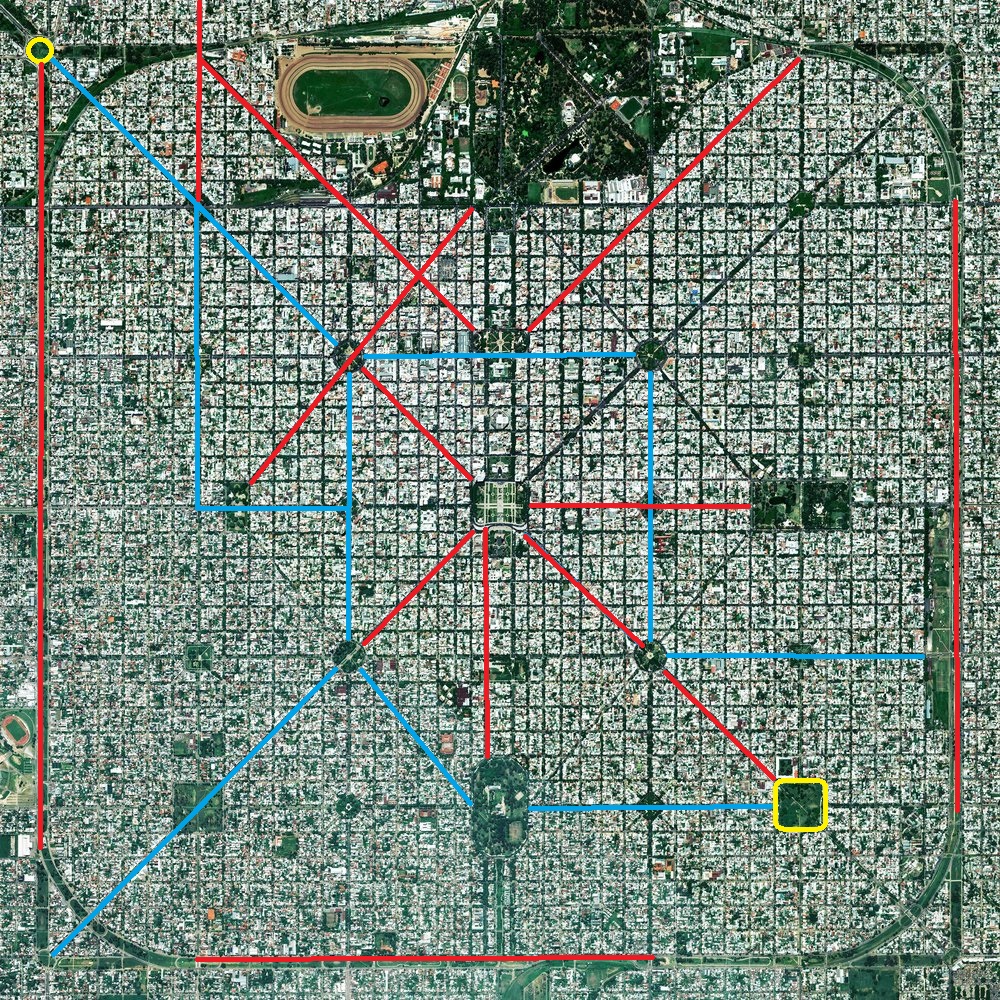
****

1. **Validación de resultados y pruebas.**

Para esta primera entrega, la única validación en cuanto a resultados y pruebas sería la presentación de nuestros dataset con los que trabajaremos y el planteamiento de nuestro grafo.

Para futuras entregas, nuestra validación se realizaría en el desempeño de nuestra aplicación, la cual está planificada para el conseguimiento de información relevante al trazar rutas. Para ello, las entradas que estimamos serían los dos puntos (nodos) mediante los cuales queremos trazar una ruta. Mientras que nuestras salidas serían, primero la ruta más corta entre ambos puntos, segundo la distancia calculada para la ruta (serie de aristas), y el tiempo aproximado que debería tomarnos culminar dicha ruta.

Un ejemplo de validación de resultado esperado es plantear la siguiente ruta, la cuál nuestro algoritmo se va a encargar de resolver.



Esta **imagen 2,** es una representación de las rutas trazadas de tráfico. Donde las calles señaladas con color rojo representan alto trafico de automoviles, mientras que que las calles pintadas de color celestes muestran las avenidas libres de trafico. Si evidenciamos que el objetivo de este problema, es validar la llegada desde el circulo amarillo hasta el cuadrado amarillo, entonces el algoritmo realizará su recorrido de una manera proactiva, evitando ocupar mayor tiempo.

1. **Conclusiones.**

En conclusión, los algoritmos son de fundamental importancia, pues permiten estudiar las necesidades del problema para realizar un conjunto de posibles soluciones y aplicarlas a posterioridad, hasta alcanzar la solución deseada.

Un punto importante a tomar en cuenta es que, los algoritmos utilizados para dar solución el problema, se aprendieron en clase y se implementará acorde a las necesidades que especifique el caso, para dar un mayor realce se realizará el análisis de complejidad de los algoritmos utilizados así como una breve presentación de los aspectos que se incluyan en el proyecto.

Por último, el dataset utilizado pretende simular el nombre y calles de la Región delimitada el cual se pretende utilizar, cabe mencionar que los datos son generados manualmente, los datos de prueba cumplen con las características requeridas por el proyecto y se espera a cumplir con los objetivos esperados de aprendizaje del curso.

Además de lo comentado anteriormente, en este entregable pudimos darnos cuenta cuales son los algoritmos que vamos a poder utilizar en la aplicación, esto como resultado de comprender sobre lo que tratará nuestra app, así como las pautas que seguiremos en la realización de la misma.

El previo diseño realizado, los wireframes serán de fundamental importancia, pues es una noción para lograr tener en cuenta un punto de partida y lograr empezar el proyecto con una noción, dichas plantillas se presentarán los resultados obtenidos de nuestro proyecto desarrollado.

El desarrollo de la documentación del proyecto será importante para detallar nuestros avances y descubrimientos, asimismo resumirá los distintos entregables que presentaremos en las distintas presentaciones.

Además de lo comentado anteriormente, en este entregable pudimos darnos cuenta cuales son los algoritmos que vamos a poder utilizar en la aplicación, esto como resultado de comprender sobre lo que tratará nuestra app, así como las pautas que seguiremos en la realización de la misma.

1. **Referencias bibliográficas.**

*Galería de Trazas urbanas: 17 ciudades vistas desde arriba - 7*. (s/f). ArchDaily Perú. Recuperado el 27 de septiembre de 2022, de <https://www.archdaily.pe/pe/948980/trazas-urbanas-17-ciudades-vistas-desde-arriba/5f77388b63c017ae22000958-trazas-urbanas-17-ciudades-vistas-desde-arriba-imagen>

*La Plata: la última ciudad argentina planificada antes del automóvil*. (2019, febrero 12). AREA - Agenda de Reflexión en Arquitectura, Diseño y Urbanismo. <https://area.fadu.uba.ar/area-23/munoz23/>

*La omnipresencia de los algoritmos en la vida cotidiana y en los grandes saltos de la tecnología*. (2017, April 21). Uchile.cl. <https://www.uchile.cl/noticias/132454/la-presencia-de-los-algoritmos-en-la-vida-cotidiana->

Pavas, A. (2016). Visibilidad de la Revista Ingeniería e Investigación. *Ingeniería e Investigación*, *36*(3), 3. <https://doi.org/10.15446/ing.investig.v36n3.61596>

*Difference between Breadth Search (BFS) and Deep Search (DFS)*. (2020, 25 mayo). Encora. <https://www.encora.com/es/blog/dfs-vs-bfs>

LINK DEL REPOSITORIO GITHUB: <https://github.com/AntoniMendoza/TFComplejidadAlgoritmica>

LINK DEL VIDEO GRUPAL: <https://drive.google.com/file/d/1eYE472LGFFF-t4MwOA7_VsW63CyhkmMd/view?usp=sharing>