[](https://camo.githubusercontent.com/0eee3fd599da51af8542e0508bd35a001a031a2e35e66a0659181d06e3dd9c42/68747470733a2f2f75706c6f61642e77696b696d656469612e6f72672f77696b6970656469612f636f6d6d6f6e732f662f66632f5550435f6c6f676f5f7472616e73706172656e74652e706e67)

**Universidad Peruana de Ciencias Aplicadas**

Ingeniería de Software

Ciclo 2023-02

**Complejidad Algoritmica**

**TP REPORT**

**Sección:** WX73

**Profesor**: Luis Martin Canaval Sanchez

**Producto**: AutoGo

**Team Members:**

| **Member** | **Code** |
| --- | --- |
| Rojas Ccama, Carlos Andres | U202114657 |
| Pozo Campos, Rodrigo Jair | U20181E187 |
| Gamio Upiachihua, Brenda Lucía | U202120344 |

Agosto del 2023

**Contenido**

**Tabla de Contenidos**

[**Contenido**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#contenido)

[**Student Outcome**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#student-outcome-1)

[**Hito I: Descripción del problema**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#1-descripci%C3%B3n-del-problema)

[**Hito II: Descripción y vizualición del conjunto de datos**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#2-descripci%C3%B3n-y-vizualici%C3%B3n-del-conjunto-de-datos)

[**Hito III: Propuesta**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#3-propuesta)

[**Hito IV: Diseño del Aplicativo**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#4-dise%C3%B1o-del-aplicativo)

[**Hito V: Validación de Resultados y Pruebas**](https://github.com/BGamio/Complejidad-Algoritmica/blob/main/Hito%203.md#5-validaci%C3%B3n-de-resultados-y-pruebas)

**Student Outcome**

**ABET - EAC - Student Outcome 4**

**Criterio:** La capacidad de reconocer responsabilidades éticas y profesionales en situaciones de ingeniería y hacer juicios informados, que deben considerar el impacto de las soluciones de ingeniería en contextos globales, económicos, ambientales y sociales.

En el siguiente cuadro se describen las acciones realizadas y enunciados de conclusiones por parte del grupo, que permiten sustentar el haber alcanzado el logro del ABET - EAC - Student Outcome 4.

| **Criterio específico** | **Acciones realizadas** | **Conclusiones** |
| --- | --- | --- |
| Demuestra ética profesional en el ejercicio de la ingeniería de software. | Carlos Rojas Ccama, Propuesta  Rodrigo Pozo Campos, Descripción del problema  Brenda Gamio Upiachihua, Descripción y visualización del conjunto de datos. | La colaboración en equipo, guiada por prácticas de desarrollo Agile, fue fundamental para fortalecer nuestra cohesión como grupo. La metodología Agile, junto con nuestra colaboración, desempeñó un papel crucial en nuestro logro. Descubrimos la relevancia de trabajar juntos de manera ágil en el ámbito de la ingeniería de software. |
| Demuestra Responsabilidad profesional para el logro de los objetivos. | Carlos Rojas Ccama, Propuesta  Rodrigo Pozo Campos, Descripción del problema  Brenda Gamio Upiachihua, Descripción y visualización del conjunto de datos. | El compromiso en la elaboración de la propuesta es evidencia de la dedicación para alcanzar metas en mi campo de trabajo. El desarrollo continuo en equipo demostró responsabilidad. |
| Emite juicios considerando el impacto de las soluciones de ingeniería de software en el contexto global, impacto social, ambiental y económico. | Carlos Rojas Ccama, Propuesta  Rodrigo Pozo Campos, Descripción del problema  Brenda Gamio Upiachihua, Descripción y visualización del conjunto de datos. | El equipo exhibe una sólida ética profesional y adopta un enfoque integral hacia la ingeniería de software, considerando no solo los aspectos técnicos, sino también su impacto en un contexto más amplio. |

**1. Descripción del problema**

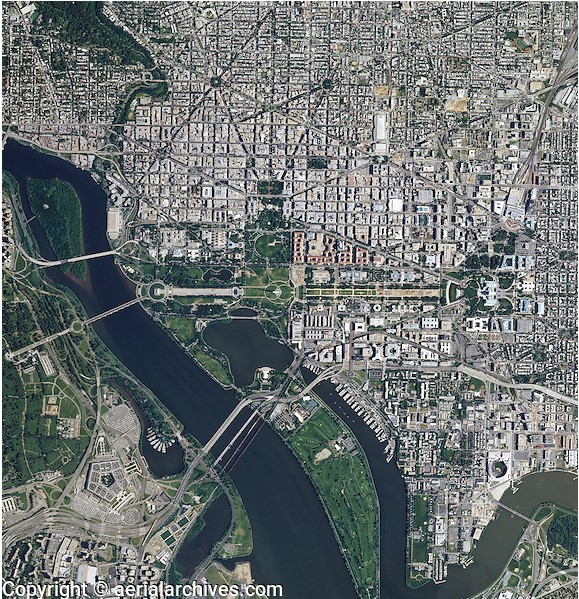
En el contexto del curso de complejidad algorítmica, se exploran temas que ofrecen soluciones a desafíos actuales. Estos desafíos incluyen cuestiones como la gestión vehicular y el estresante tráfico que ocurre alrededor del mundo. Los algoritmos aplicados resultan sorprendentemente prácticos y se encuentran presentes en nuestras actividades cotidianas a través de diversas aplicaciones. Es relevante destacar que uno de los más destacados es el algoritmo Dijkstra, cuyo propósito principal es "definir criterios para alcanzar metas específicas y construir un modelo con una jerarquía que tome en cuenta diversos factores o atributos para guiar la toma de decisiones" (Pavas, A. 2016). Este algoritmo desempeñará un papel central en nuestro proyecto.

Un ejemplo representativo es la aplicación Waze, líder a nivel global, que se dedica a optimizar el tiempo y los recursos en los desplazamientos. Esta aplicación recomienda rutas alternativas basándose en la información recopilada de sus usuarios. En este contexto, es importante señalar lo que la Universidad de Georgetown afirma: "En la era digital actual, los algoritmos adquieren un rol cada vez más preponderante en los procesos tecnológicos y, por consiguiente, en nuestra vida cotidiana". En parte lo que nos comenta la Universidad de Georgetown es cierto ya que cada vez vivimos en un mundo que está estrechamente ligado con tecnologías de cualquier índole.

Decidimos que será en esta ciudad ya que según Gizmodo, blog acerca de ciencia, tecnología y cultura digital menciona que: La ciudad de Washington DC es la segunda ciudad mejor planificada del mundo. Por lo que, esta ciudad conocida por su relevancia en la política y la cultura de los Estados Unidos, se convierte en el foco de nuestra investigación.

Nuestro objetivo en el curso es desarrollar una aplicación similar a Waze, que se especialice en encontrar la ruta más eficiente entre dos puntos de una ciudad seleccionada. Esta se representa mediante un grafo el cual representa los detalles de la ciudad.

Para abordar esta problemática, utilizaremos conjuntos de datos (datasets) que incluyen "edges" (tramos de carretera), "nodes" (nodos de intersección y coordenadas) y "streets" (nombres de calles), en ese mismo orden. También elegimos esta ciudad ya que presenta distintas avenidas diagonales con grandes intersecciones, tal como podemos ver en la **imagen** 1.

[](https://camo.githubusercontent.com/5a7adaa11dd64db34ccaad1e6591ba9bf3c811b388cfb0631b8d559e5dae05a4/68747470733a2f2f63646e2e646973636f72646170702e636f6d2f6174746163686d656e74732f313135353632343436333136313839363937302f313135353632343436373036363739303036382f666f746f5f61657265612e6a7067)

**Imagen 1. Washington DC.** Recuperado de <https://aerialarchives.photoshelter.com/image/I0000xtaFsXx.VLU>

**2. Descripción y vizualición del conjunto de datos**

Para llevar a cabo este proyecto, adquirimos dos conjuntos de datos con el propósito de abordar la cuestión planteada en la introducción. Estos conjuntos de datos contienen un total de 178 167 registros, de las que usarán 1500 de estos de acuerdo con las indicaciones de la guía. Nuestra intención es utilizar esta información para construir una solución efectiva y eficiente mediante el uso de gráficos y varios algoritmos.

* Líneas centrales de la calle 2013 con 34 059 registros. De estos registros se podría sacar:
  + **Nodos.txt:** El presente archivo de texto contiene contiene los nodos que conformarán el grafo. Estos nodos poseen únicamente una columna, la cual hace referencia al “StreetSegid” en los registros.
  + **Bordes.txt:** Este archivo de texto contiene las aristas que conformarán el grafo. Estas aristas poseen únicamente una columna, la cual hace referencia al “Quadcode” en los registros.
  + **Calles.txt:** Este último archivo de texto contiene los nombres de cada una de las calles que contiene la zona elegida.

| **STREETSEGID** | **QUADCODE** | **ST\_NAME** |
| --- | --- | --- |
| 1 | 3 | Bangor St |
| 5 | 2 | Florida Ave |
| 11 | 1 | O St |
| 12 | 1 | T St |
| 13 | 1 | Whittier St |
| 15 | 1 | 19th St |
| 16 | 4 | Maryland Ave |
| 17 | 1 | 8th St |
| 19 | 3 | K St |
| 22 | 2 | Kenilworth Ave |
| 24 | 1 | Chesapeake St |
| 26 | 2 | H St |
| 27 | 2 | C St |
| 29 | 2 | 4th St |

* Puntos de direcciones con 144 108 registros. De estos registros se podría sacar:
  + **Nodos.txt:** El presente archivo de texto contiene datos de referencia sobre los nodos que conformarán nuestro grafo en el futuro. Estos nodos representarán las intersecciones entre dos calles. El formato del archivo consta de cuatro columnas, donde las cuatro corresponden a las coordenadas geográficas precisas (Coordenada X, Coordenada Y, Latitud y Longitud).
  + **Bordes.txt:** Este archivo de texto contiene las aristas que conformarán el grafo. Estas aristas poseen únicamente una columna, la cual hace referencia al número de dirección.
  + **Calles.txt:** Este último archivo de texto contiene los nombres de cada una de las calles que contiene la zona elegida.

| **ADDRESS\_NUMBER** | **STREET\_NAME** | **Y\_COORDINATE** | **X\_COORDINATE** | **LATITUDE** | **LONGITUD** |
| --- | --- | --- | --- | --- | --- |
| 7428 | GEORGIA | 145855.59 | 397663.49 | 38.98062369 | -77.02696622 |
| 2 | REV LLOYD YOUNG | 136378.23 | 401734.22 | 38.89525024 | -76.98000891 |
| 4 | REV LLOYD YOUNG | 136385.15 | 401734.75 | 38.89531258 | -76.98000278 |
| 6 | REV LLOYD YOUNG | 136391.27 | 401734.22 | 38.89536771 | -76.98000888 |
| 8 | REV LLOYD YOUNG | 136397.39 | 401733.95 | 38.89542284 | -76.98001197 |
| 10 | REV LLOYD YOUNG | 136403.24 | 401733.95 | 38.89547554 | -76.98001196 |
| 12 | REV LLOYD YOUNG | 136408.83 | 401733.95 | 38.8955259 | -76.98001194 |
| 14 | REV LLOYD YOUNG | 136415.21 | 401733.68 | 38.89558337 | -76.98001504 |
| 16 | REV LLOYD YOUNG | 136421.87 | 401733.42 | 38.89564337 | -76.98001802 |
| 2007 | TRENTON | 130814.77 | 402173.11 | 38.8451317 | -76.97496722 |
| 1302 | TRINIDAD | 137308.68 | 401194.41 | 38.9036333 | -76.98622991 |
| 1327 | UNDERWOOD | 144587.65 | 397316.83 | 38.96920114 | -77.03096214 |
| 511 | BRUMMEL | 145786.18 | 398130.87 | 38.97999988 | -77.02157188 |
| 4712 | ALTON | 142116.05 | 391904.38 | 38.94690312 | -77.09338929 |

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

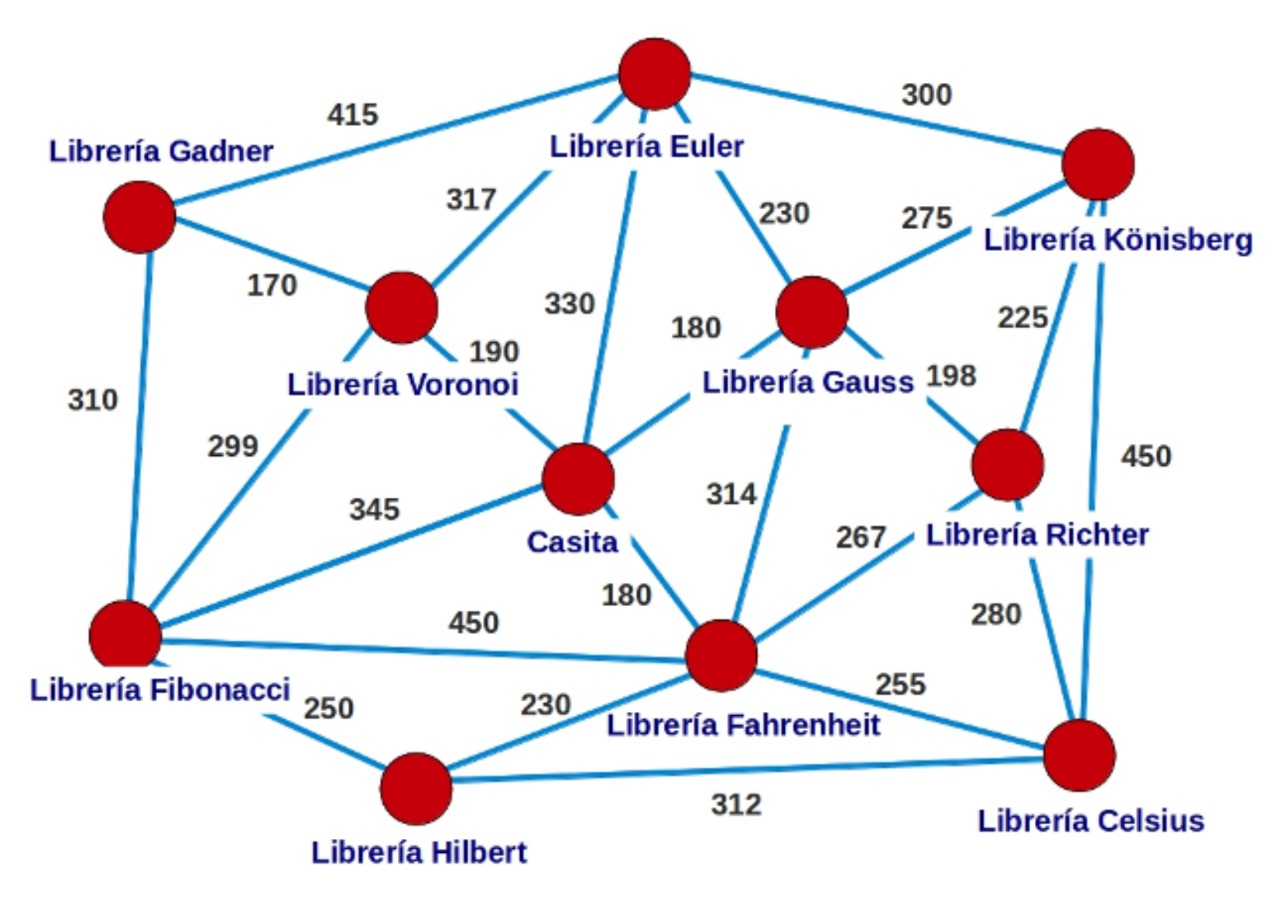
Los datos recopilados serán empleados en la elaboración de nuestro programa, para la elaboración del mapa utilizaremos los algoritmos necesarios, en el cual se realizarán las búsquedas de caminos.

Con la información del archivo nodes, bordes y calles, 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.

**3. Propuesta**

El objetivo principal de esta propuesta es desarrollar un sistema de navegación eficiente que permita a los usuarios encontrar la ruta más corta entre dos puntos dentro de una ciudad. Este sistema debe ser fácil de usar, preciso y rápido, y deberá tener en cuenta varios factores como el tráfico en tiempo real, las preferencias del usuario y otros elementos relevantes.

El objetivo del programa que vayamos a realizar 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.

[](https://camo.githubusercontent.com/e14392e97f2c2f359a69f43e97bee431de23f1bf188c13f928e873aac7d67834/68747470733a2f2f63646e2e646973636f72646170702e636f6d2f6174746163686d656e74732f313137323339383232353231393635373831382f313137323635303435343837343836313632382f496d6167656e5f64655f57686174734170705f323032332d31312d31305f615f6c61735f31352e31362e35335f31303463633634302e6a70673f65783d36353631313664372669733d363534656131643726686d3d3161633630356637633632323466663635313861663565663439356263623562616539373031373632373939653331353938613437316566383665303961376126)

Los algoritmos de búsqueda de la ruta más eficiente desempeñarán un papel fundamental en nuestra aplicación. Uno de los algoritmos clave que utilizaremos es el algoritmo de Dijkstra, el cual se encarga de identificar la ruta más óptima en nodos que tienen asignados valores específicos, que en nuestro caso representan la longitud de las calles en una ciudad. Esto nos permitirá determinar la ruta más corta entre dos ubicaciones especificadas por el usuario. Además de la implementación de varios algoritmos para el funcionamiento de la aplicación, es esencial mantenerla actualizada de forma constante y proporcionar alternativas en caso de que surjan obstáculos en el cálculo de la ruta más eficiente en la vida real, como cierres de calles, accidentes u otros problemas que puedan afectar las condiciones de la vía.

* **Recopilación de datos:** Obtener datos cartográficos detallados de la ciudad, que incluyan información sobre calles, carreteras,  
  intersecciones, señales de tráfico
* **Algoritmo de enrutamiento** Implementar un algoritmo de enrutamiento eficiente, como el algoritmo Dijkstra, que calcule la ruta más corta entre dos puntos.
* **Interfaz de usuario** Desarrollar una interfaz de usuario intuitiva para la aplicación, que permita a los usuarios ingresar sus puntos de inicio y destino. Mostrar la ruta recomendada en el mapa, junto con información relevante, estimado de llegada, el tráfico y las instrucciones paso a paso.

**4. Diseño del Aplicativo**

En el proceso de diseño de nuestro proyecto de aplicación, nos enfocaremos en analizar los algoritmos que implementaremos en la aplicación. Esta elección se debe a la importancia de garantizar el correcto funcionamiento de los algoritmos en nuestra aplicación, ya que es crucial monitorearlos constantemente en busca de posibles fallas o errores en el código. Asimismo, se realizarán las correcciones necesarias a lo largo de todo el desarrollo de la aplicación.

Dado que nuestra aplicación se centra en las calles de las ciudades, podemos interpretarlas como grafos ponderados, donde cada calle se convierte en una arista y los cruces en vértices. Esta representación facilita la aplicación de algoritmos y cálculos basados en grafos, especialmente al calcular la ruta más corta entre diferentes puntos. Por lo tanto, utilizar y entender las calles como grafos conectados con pesos específicos resulta óptimo para la ejecución de nuestra aplicación.

Para la elaboración del programa, hemos decidido emplear diversos algoritmos como solución.

**En primer lugar, consideramos la implementación del algoritmo de Dijkstra.**

Este método, ideado por Edsger Dijkstra, un destacado científico de la computación neerlandés, se utiliza para determinar la ruta más corta entre un vértice de origen y otros vértices conectados por aristas con pesos específicos. En nuestro proyecto, que se centra en calcular la ruta más eficiente entre las calles de diferentes ubicaciones en el país, encontramos que este algoritmo es idóneo. Cada calle se considera como una arista con un peso correspondiente a su longitud, y los puntos entre los cuales se desea calcular la ruta se interpretan como los vértices de un grafo.

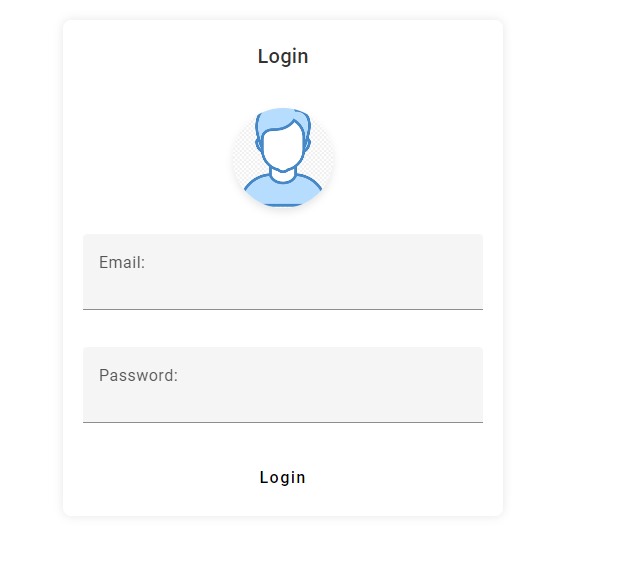
**Otro algoritmo que contemplamos es el de Bellman-Ford.**

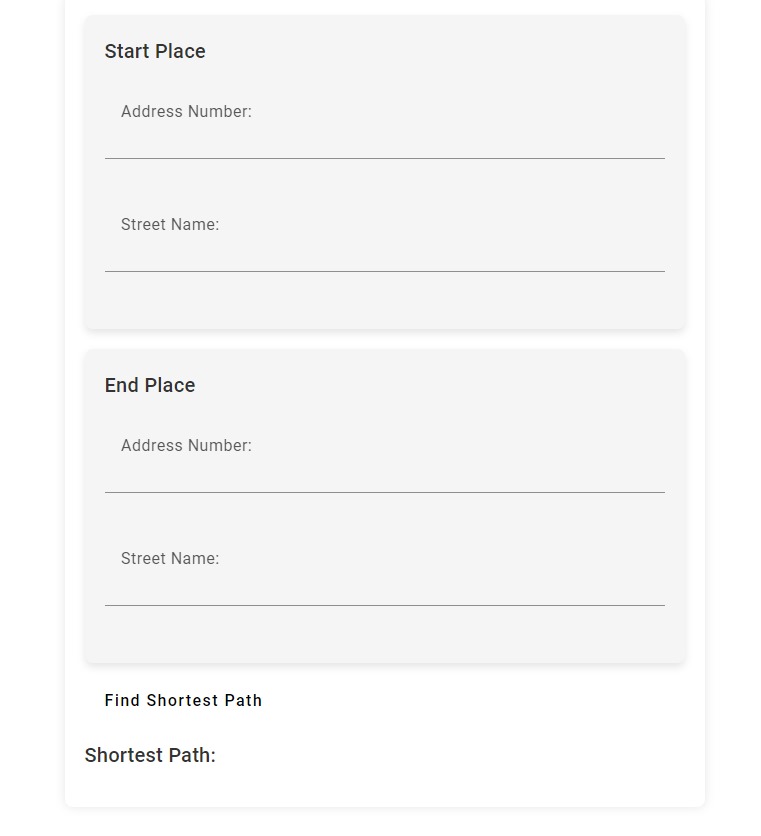
Este enfoque busca la ruta más corta en un grafo ponderado, permitiendo incluso aristas con valores negativos. A diferencia del algoritmo de Dijkstra, el de Bellman-Ford puede utilizarse en grafos con aristas de peso negativo. En nuestro proyecto, planeamos implementar este algoritmo para explorar diferentes métodos de generación de rutas eficientes, evitando sobrecargar un único algoritmo con todas las rutas que la aplicación pueda trazar.

**Finalmente, evaluamos la posibilidad de incorporar el algoritmo de Johnson.**

Este método busca las rutas más cortas entre pares de vértices en un grafo dirigido disperso, permitiendo pesos negativos en las aristas. Utiliza el algoritmo de Bellman-Ford para eliminar aristas con peso negativo, y posteriormente aplica el algoritmo de Dijkstra en el grafo resultante.

Algunas capturas de la aplicacion funcionando y en donde se podran realizar las solicitudes:

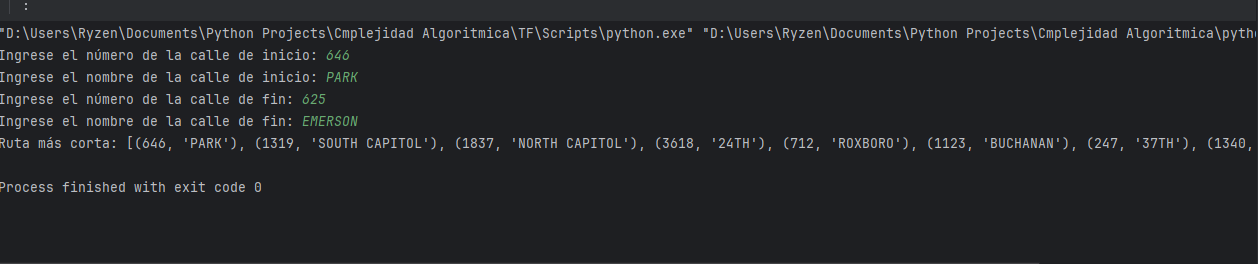
[](https://camo.githubusercontent.com/f958bd10c3174c49bd573d4fb1443e3ec60f55df1d680a2340703defd33def00/68747470733a2f2f63646e2e646973636f72646170702e636f6d2f6174746163686d656e74732f313137323339383232353231393635373831382f313137323632303733393435313535353838302f496d6167656e5f64655f57686174734170705f323032332d31312d31305f615f6c61735f31302e30372e34305f31366139366433302e6a70673f65783d36353630666232612669733d363534653836326126686d3d3266383965373138613862386164613233623666616665336637386665623132663363336335653430383936363466346162393765663733633362386130303526)

[](https://camo.githubusercontent.com/a02056a583bd07b1e47c9373bdeb0b380f674443a4af75be2a83926c5f18b637/68747470733a2f2f63646e2e646973636f72646170702e636f6d2f6174746163686d656e74732f313137323339383232353231393635373831382f313137323632303736373537383536323736322f496d6167656e5f64655f57686174734170705f323032332d31312d31305f615f6c61735f31302e30372e34305f31623964386232652e6a70673f65783d36353630666233312669733d363534653836333126686d3d3963623036656262356465306262656163613930383865333934636633393336303065383763626533306238326236316233633233353761353736653132666126)

**5. Validación de resultados y pruebas**

Para la realización de esta tuvimos que tener una aplicación realizada en python, es aquí donde desarrollamos todo el trabajo en sí. De esta forma el programa nos ayudará a obtener una ruta eficaz que conecte los dos puntos que hemos realizado.

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 serán dos, la primera el número y nombre de la calle de inicio y una de final. De esta manera el resultado que esperamos será una lista de las calles con su respectivo nombre y número. Como se puede observar en la siguiente imagen:

[](https://camo.githubusercontent.com/c1cfdf3b1557991a5717783ba3952ef8029a0c1d93e8ddfd34e18ab5654b82e5/68747470733a2f2f63646e2e646973636f72646170702e636f6d2f6174746163686d656e74732f313137323339383232353231393635373831382f313137363031383034363630333737313936362f696d6167652e706e673f65783d36353664353732372669733d363535616532323726686d3d6437386531613035323935613262643564636239386662393131626431633363346232333134663766323439623436346466626432316536663339613430363626)

**6. Conclusiones**

En conclusión este hito, se centra en el desarrollo de un sistema de navegación eficiente que utiliza algoritmos como Dijkstra, Backtracking y posiblemente otros algoritmos mas para calcular rutas óptimas. Esto refleja una gran eficiencia en la elección de algoritmos adaptados a diferentes situaciones, permitiendo flexibilidad en el manejo de distintos escenarios y condiciones de tráfico.

En conclusión, este hito tambien destacara la representación de las calles de la ciudad como grafos ponderados, donde cada calle se convierte en una arista y los cruces en vértices. Esta elección facilita la aplicación de algoritmos basados en grafos para calcular la ruta más corta entre diferentes puntos. La interpretación de la ciudad como un grafo conectado con pesos específicos resalta la atención a la estructura y la topología de la red vial, lo cual es esencial para el diseño eficiente de la aplicación proporcionando una experiencia de navegación más precisa y eficiente.

**6. Bibliografía**

Washington DC: una ciudad moldeada por y para el poder, una concentración de poder. (s. f.). <https://geoimage.cnes.fr/fr/geoimage/washington-dc-una-ciudad-moldeada-por-y-para-el-poder-una-concentracion-de-poder>

Parábola Estudio. (2018, 15 mayo). Vistas desde el cielo de 10 sorprendentes ciudades planificadas. Medium. <https://medium.com/@parabola.redes/vistas-desde-el-cielo-de-10-sorprendentes-ciudades-planificadas-2c0a3def7368>

Address points. (s. f.). <https://opendata.dc.gov/datasets/DCGIS::address-points/explore?showTable=true>

Street Centerlines 2013. (s. f.). <https://opendata.dc.gov/datasets/8655869366064bdca04bb6793cc02a54_3/explore?location=38.894894%2C-77.025952%2C13.00&showTable=true>