



Instituto Politécnico Nacional

Escuela Superior de ingeniería Mecánica y Eléctrica

Programación Avanzada

Profesor: Anzaldo Bustos Jorge Alejandro

ORGANIZADOR:

Caballero Trejo Joan Oziel 2025300069

Grupo: 2CV14

ÍNDICE

Planteamiento del problema	3
Objetivos (General y particular)	4
Justificación	5
Puntos a destacar como avances técnicos.....	7
Funcionalidades ya implementadas	8
1. Gestión de presupuesto	8
2. Selección de aeropuerto	8
3. Visualización de rutas disponibles (mapas gráficos).....	8
4. Menú de destinos por aeropuerto	8
5. Asignación de rutas de vuelo personalizadas	8
6. Visualización gráfica de itinerarios.....	8
7. Soporte para vuelos rápidos.....	8
8. Mensajes informativos y explicaciones	9
9. Registro en consola	9
Herramientas y tecnologías utilizadas	9

Planteamiento del problema

En la actualidad, el turismo y los viajes aéreos han experimentado un crecimiento significativo, impulsado por la globalización y la accesibilidad a destinos nacionales e internacionales. Sin embargo, uno de los principales obstáculos que enfrentan los viajeros al planificar un vuelo es la restricción presupuestaria, la cual limita las opciones disponibles y puede afectar la experiencia de viaje. Aunque en algunos casos los usuarios cuentan con un presupuesto suficiente que les permite priorizar la comodidad, en otros, las limitaciones económicas obligan a buscar alternativas más asequibles, lo que puede implicar sacrificios en términos de calidad, conveniencia o servicios adicionales. Esta dualidad en las capacidades económicas de los viajeros representa un desafío para las agencias de viajes y aerolíneas, que deben ofrecer opciones flexibles que se adapten a diferentes perfiles de clientes.

A su vez, otro factor crítico que influye en la experiencia del viajero es el tiempo de vuelo. Los vuelos de larga duración, o aquellos con múltiples escalas, pueden resultar agotadores, afectando tanto el bienestar físico como emocional de los pasajeros. Este aspecto cobra especial relevancia en un contexto donde la optimización del tiempo es un valor creciente, y los viajeros buscan alternativas que minimicen la fatiga asociada con los desplazamientos. La combinación de restricciones presupuestarias y la necesidad de reducir el impacto del tiempo de vuelo plantea la necesidad de desarrollar soluciones personalizadas que equilibren costos, comodidad y duración del trayecto.

En este sentido, el presente proyecto busca abordar estas problemáticas mediante la propuesta de un modelo de planificación de vuelos que contemple dos enfoques basados en la variable “presupuesto”: uno orientado a viajeros con recursos limitados, priorizando la economía, y otro dirigido a aquellos que pueden invertir en mayor comodidad y eficiencia. Además, se considerará la variable “tiempo de vuelo” para ofrecer alternativas que reduzcan la fatiga y mejoren la experiencia del usuario. Con esto, se pretende brindar a los clientes opciones claras y adaptadas a sus necesidades, permitiéndoles tomar decisiones informadas que optimicen su experiencia de viaje.

Objetivos (General y particular)

General:

Diseñar y desarrollar un programa que genere planes de vuelo personalizados basados en la variable principal “presupuesto” del usuario, integrando factores adicionales como comodidad, facilidad de acceso a la unidad, tiempo de vuelo, y preferencias de horarios. El sistema ofrecerá opciones optimizadas que se ajusten a la capacidad económica del usuario, priorizando una experiencia de viaje eficiente y satisfactoria pero siempre con un vuelo garantizado.

Específicos

- ✚ Analizar el presupuesto con el que cuenta el usuario.
- ✚ Usar archivos json para el control de vuelos que se estarán modificando cada 3 horas.
- ✚ Implementación de grafos para la localización de destinos ligados al punto de origen.
- ✚ Implementación de algoritmos de búsqueda para los destinos más cercanos desde el punto de partida.
- ✚ Diseñar y presentar dos paquetes de vuelo (económico y premium) basados en los resultados de búsqueda.
- ✚ Optimización del código.

Justificación

El turismo aéreo ha crecido notablemente por la globalización y la accesibilidad, pero los viajeros enfrentan restricciones presupuestarias que limitan opciones y afectan la calidad del viaje, mientras que vuelos largos o con escalas generan fatiga en un contexto donde el tiempo es clave. Las plataformas actuales no integran comodidad, accesibilidad ni preferencias de horario, dejando un vacío en soluciones personalizadas.

Este proyecto propone un programa que genere planes de vuelo basados en el presupuesto, considerando comodidad, tiempo de vuelo y accesibilidad, usando archivos JSON para datos dinámicos, grafos para rutas, algoritmos de búsqueda para destinos óptimos y paquetes económico(a) y premium(b). La optimización del código asegurará eficiencia. Aunque el planteamiento es claro, técnico y bien estructurado, podría detallar el tipo de herramienta (web o móvil), incluir datos específicos (como estadísticas de la IATA), suavizar transiciones entre ideas y destacar más el impacto en la industria para reforzar la propuesta.

Avances técnicos logrados hasta el momento:

1. Uso de librerías externas

- `tkinter`: Es utilizado para crear interfaces gráficas de usuario (GUI). Aunque en el código proporcionado aún no está habilitado para su uso ya esta establecido como es que actuara su funcionamiento.
- `networkx`: Sirve para la creación, manipulación y análisis de gráficos y redes complejas. El código hace uso intensivo de esta librería para representar rutas entre aeropuertos y destinos mediante grafos.
- `matplotlib.pyplot`: Se utiliza para la visualización gráfica de los grafos creados con `networkx`. Esto permite que las conexiones entre nodos (destinos) sean visualizadas de forma clara.

2. Diseño de clases

- Clase Volando:
 - Estructura modular con métodos bien definidos para manejar diferentes funcionalidades.
 - Uso del constructor `__init__` para inicializar variables de instancia como `g` (grafo dirigido) y otras propiedades que almacenan datos relacionados con aeropuertos y vuelos.
 - Métodos específicos para manejar el flujo del programa, como presupuesto, aeropuertos, y asignación de vuelos (`asignacionAIFA`, `asignacionAICM`).

3. Gestión de datos mediante grafos

- Uso de grafos dirigidos (`DiGraph`) para modelar rutas de vuelo entre aeropuertos y destinos.
- Métodos para agregar nodos y aristas con atributos como peso (`weight`), lo que permite incluir información sobre la duración o costo de las rutas.
- Visualización de grafos mediante `networkx.draw`, permitiendo analizar las conexiones entre diferentes puntos de forma gráfica.

4. Interacción con el usuario

- Implementación de entradas interactivas (`input`) para obtener información como el presupuesto del usuario y sus preferencias de destino.
- Validación de entradas para asegurar que los valores ingresados sean correctos (e.g., valores numéricos para el presupuesto y opciones válidas para la selección de destinos).

5. Representación gráfica de rutas

- Creación de mapas de rutas predefinidas para los aeropuertos AIFA y AICM.
- Generación de visualizaciones dinámicas basadas en las elecciones del usuario.

6. Estructura y lógica

- Uso de estructuras de control como ciclos while y condicionales if-elif-else para guiar el flujo del programa.
- Métodos reutilizables como menu, menu_rapido, y otros, que contribuyen a la modularidad del código.

7. Mejores prácticas

- Mensajes informativos y explicativos para el usuario, facilitando la comprensión del sistema.
- Separación lógica de las funcionalidades en métodos independientes.

Puntos a destacar como avances técnicos

1. Uso de librerías avanzadas para gráficos y visualización: Integración de networkx y matplotlib para gestionar y representar rutas de vuelo.
2. Modularidad: Separación de responsabilidades en métodos, lo que facilita la comprensión, mantenimiento y ampliación del código.
3. Gestión dinámica de rutas: Implementación de grafos dirigidos con pesos que simulan el costo o tiempo de los vuelos.
4. Interactividad con el usuario: Manejo de entradas dinámicas y validación de datos.
5. Visualización gráfica: Generación de mapas interactivos para representar conexiones entre nodos, mejorando la experiencia del usuario.

- Funcionalidades ya implementadas.

Funcionalidades ya implementadas.

1. Gestión de presupuesto

- El sistema permite al usuario ingresar su presupuesto y, en función de este, determinar si puede acceder a vuelos regulares o rápidos.
- Validación de entradas numéricas para evitar errores por caracteres no válidos.

2. Selección de aeropuerto

- Ofrece al usuario la posibilidad de elegir entre el Aeropuerto Internacional Felipe Ángeles (AIFA) y el Aeropuerto Internacional de la Ciudad de México (AICM).
- Valida la selección para asegurarse de que se elija una opción válida (1 o 2).

3. Visualización de rutas disponibles (mapas gráficos)

- Genera y muestra un grafo de las rutas posibles desde ambos aeropuertos utilizando la librería networkx y matplotlib.
- Diferencia los grafos por aeropuerto, proporcionando una representación clara de las rutas disponibles.

4. Menú de destinos por aeropuerto

- Ofrece un menú de destinos dependiendo del aeropuerto seleccionado.
- Valida las entradas del usuario asegurándose de que elija opciones válidas.

5. Asignación de rutas de vuelo personalizadas

- Según el destino seleccionado, el programa crea un grafo dirigido que representa el itinerario del vuelo.
- Permite incluir escalas y asignar pesos (por ejemplo, tiempo o distancia) a las rutas.

6. Visualización gráfica de itinerarios

- Muestra un grafo dirigido personalizado con las rutas de vuelo seleccionadas, utilizando colores para nodos y aristas.
- Titula las gráficas y las adapta para una mejor presentación visual.

7. Soporte para vuelos rápidos

- Incluye lógica inicial para vuelos rápidos, permitiendo elegir destinos que puedan ser añadidos al grafo correspondiente.

8. Mensajes informativos y explicaciones








- Ofrece descripciones detalladas de las opciones disponibles, explicando al usuario las diferencias entre las opciones y el funcionamiento del programa.
- Proporciona mensajes claros en caso de errores de entrada.

9. Registro en consola

- Imprime en consola mensajes como la creación de objetos y los pasos de cada proceso, lo que ayuda en la depuración.

Estas funcionalidades permiten una experiencia básica pero funcional para la simulación de la selección de vuelos y la visualización de rutas disponibles desde dos aeropuertos principales.

Herramientas y tecnologías utilizadas.

-  Python 3
-  NetworkX
-  Matplotlib
-  Tkinter
-  JSON (no implementado)
-  Algoritmo de Dijkstra
-  Interfaz de consola