Documento de Proceso - Práctica de Grafos en Python

1. Organización y Planificación

El trabajo se abordó en equipo, dividiendo las tareas principales entre los cuatro integrantes para aprovechar las fortalezas de cada uno y fomentar la colaboración. Se establecieron reuniones iniciales para definir los objetivos, la estructura del proyecto y los entregables esperados.

2. Análisis del Problema

El primer paso fue comprender los conceptos fundamentales de los grafos y los requisitos de la práctica. Se revisaron los tipos de grafos (dirigidos y no dirigidos), las operaciones básicas (agregar vértices y aristas), y los algoritmos de recorrido (BFS y DFS). También se investigaron ejemplos y recursos para asegurar una base teórica sólida.

3. Diseño de la Solución

Se decidió implementar una clase `Grafo` en Python, capaz de representar tanto grafos dirigidos como no dirigidos. Se diseñó la interfaz de la clase para que fuera intuitiva y flexible, permitiendo agregar vértices y aristas, consultar vecinos, verificar conectividad y buscar caminos entre nodos.

El diseño modular facilitó la colaboración, permitiendo que cada integrante se enfocara en una parte específica:

- Dennis: Implementación de la estructura básica del grafo y métodos para agregar vértices/aristas.
- Farid: Desarrollo de los algoritmos de recorrido BFS y DFS.
- Emilio: Funciones de consulta (vecinos, aristas, conectividad).
- Nahum: Ejemplos de uso, pruebas.

4. Implementación

La implementación se realizó en el archivo definiciones.py, siguiendo buenas prácticas de programación y documentación. Se crearon métodos claros y comentados para cada funcionalidad. Se utilizó un diccionario para representar la lista de adyacencia del grafo, lo que permitió una gestión eficiente de los nodos y sus conexiones.

Se desarrolló un archivo ejemplo_grafo.py con ejemplos prácticos y pruebas para validar el correcto funcionamiento de la clase `Grafo` y sus métodos.

6. Documentación

Se elaboró un README.md completo, con instrucciones de uso, ejemplos de código y recursos útiles para el aprendizaje. Además, se documentó cada función dentro del código para facilitar la comprensión y el mantenimiento.

7. Revisión y Entrega

Finalmente, se realizó una revisión grupal del proyecto, corrigiendo detalles y asegurando la calidad del código y la documentación. Se agradeció la colaboración de todos los integrantes y se preparó el repositorio para su entrega.

Conclusión:

El trabajo en equipo, la planificación y la comunicación constante fueron claves para el éxito de la práctica. La implementación modular y la documentación detallada permitieron crear una solución funcional, didáctica y fácil de mantener.