Reflexión

Link github:[TC1031-Portafolio_Final-/bosques.cpp at main · ArturoL0709/TC1031-Portafolio_Final- (github.com)](https://github.com/ArturoL0709/TC1031-Portafolio_Final-/blob/main/bosques.cpp)

Al desarrollar este programa para contar árboles y bellotas en un bosque, se han aprendido y reforzado varios conceptos clave en programación y estructuras de datos. A continuación, se destacan algunas lecciones aprendidas y áreas de mejora identificadas:

El programa utiliza eficazmente estructuras de datos como mapas y conjuntos para representar las conexiones entre vértices y realizar un seguimiento de los nodos visitados. Esto destaca la importancia de elegir las estructuras de datos adecuadas para resolver un problema específico.

Se ha experimentado con la entrada de datos y la necesidad de garantizar que el programa pueda manejar de manera efectiva situaciones de entrada específicas, como la terminación de la entrada con un marcador especial ("****"). La importancia de la validación de entrada se ha vuelto evidente durante el desarrollo.

La implementación de un recorrido en profundidad (DFS) para identificar componentes conectados en el bosque ha sido una lección valiosa. Este enfoque proporciona una forma efectiva de explorar la conectividad entre los vértices.

Se ha reconocido la necesidad de mejorar la claridad del código mediante la inclusión de comentarios explicativos. Esto facilitará la comprensión y el mantenimiento del código, especialmente para otros desarrolladores que puedan revisarlo.

Podría considerarse la implementación de una validación más robusta de la entrada del usuario para garantizar que solo se ingresen datos válidos, reduciendo así posibles errores durante la ejecución.

El programa aborda de manera efectiva el problema de contar árboles y bellotas en un bosque, proporcionando resultados precisos.

La modularidad del código, con funciones dedicadas a la construcción del bosque y al recorrido para contar, hace que el programa sea más legible y fácil de mantener.

La implementación de un algoritmo de búsqueda en profundidad (DFS) destaca la versatilidad de este enfoque para resolver problemas de conectividad en grafos.

En conclusión, el desarrollo de este programa ha permitido explorar y aplicar varios conceptos fundamentales de programación y estructuras de datos. Las mejoras propuestas y las lecciones aprendidas contribuyen a fortalecer las habilidades de diseño de programas y fomentan la adopción de prácticas de programación más efectivas. Este proyecto no solo ha proporcionado una solución funcional, sino que también ha abierto la puerta a futuras optimizaciones y refinamientos.