Universidad del Rosario
Matemáticas aplicadas y ciencias de la computación
Complejidad Pilas y Colas
Tatiana Cabrera
María Valeriano
2024.1

UNIVERSIDAD DEL ROSARIO

COMPLEJIDAD PILAS Y COLAS

1. Entiende el Problema:

El problema consiste en gestionar las vacaciones de una familia, lo que incluye:

Agregar destinos a una lista general.
Permitir que los miembros de la familia planifiquen viajes a destinos específicos.
Consultar los destinos planificados por un miembro de la familia.
Encontrar a los miembros de la familia que planean visitar un destino específico.

2. Identifica las Partes Críticas:

Las partes críticas del código son las funciones que realizan las operaciones principales, como:

agregarDestino: Agrega un nuevo destino a la lista general.
planificarViaje: Permite que un miembro de la familia planifique un viaje a un destino.
consultarDestinosPlanificados: Muestra los destinos planificados por un miembro de la familia.
encontrarMiembrosPorDestino: Encuentra a los miembros de la familia que planean visitar un destino específico.

3. Conteo de Operaciones Básicas:

La cantidad de operaciones básicas que se realizan en las partes críticas depende de la cantidad de datos y las acciones específicas que se realicen. Sin embargo, en general, se pueden identificar las siguientes operaciones básicas:

Comparaciones: Se realizan comparaciones para verificar la igualdad de destinos, nombres, etc.
Asignaciones: Se asignan valores a variables, como agregar destinos a colas o pilas.
Accesos a memoria: Se accede a la memoria para leer y escribir datos, como obtener información de destinos o miembros de la familia.

4. Notación O Grande (O):

La complejidad en el peor de los casos para cada función crítica puede expresarse en notación O grande de la siguiente manera:

agregarDestino: O(1), ya que agregar un nuevo elemento a una pila o cola es una operación constante.
planificarViaje: O(n), donde n es la cantidad de destinos disponibles, ya que en el peor de los casos se debe buscar en toda la lista de destinos.

consultarDestinosPlanificados: O(n), donde n es la cantidad de destinos planificados para el miembro de la familia, ya que se recorre la cola de destinos planificados.
encontrarMiembrosPorDestino: O(n * m), donde n es la cantidad de miembros de la familia y m es la cantidad de destinos planificados para cada miembro, ya que se recorren las colas de destinos planificados de cada miembro.

5. Análisis de Bucles:

El código utiliza bucles for para recorrer listas, colas y pilas. La complejidad de estos bucles se refleja en la complejidad de las funciones que los contienen.

6. Considera Funciones y Recursión:

El código no utiliza llamadas a funciones recursivas.

7. Evalúa Algoritmos Específicos:

Los algoritmos utilizados en el código son principalmente búsquedas lineales en listas, colas y pilas. Estos algoritmos tienen una complejidad lineal en el peor de los casos.

8. Prueba con Diferentes Tamaños de Entrada:

El rendimiento del código puede variar dependiendo del tamaño de los datos que se procesan. Se espera que el tiempo de ejecución aumente linealmente con el tamaño de la entrada para las operaciones que involucran bucles.

9. Comparación con Escalabilidad:

El código es escalable en general, ya que las operaciones básicas y los algoritmos utilizados tienen una complejidad lineal. Sin embargo, para conjuntos de datos muy grandes, se podrían considerar optimizaciones, como el uso de estructuras de datos más eficientes o algoritmos de búsqueda más rápidos.

10. Documenta tus Conclusiones:

La complejidad del código en el peor de los casos es O(n * m), donde n es la cantidad de miembros de la familia y m es la cantidad de destinos planificados para cada miembro. El código es escalable en general, pero se podrían considerar optimizaciones para conjuntos de datos muy grandes.

Recomendaciones:

Se podría optimizar la función planificarViaje utilizando una estructura de datos más eficiente para buscar destinos, como un árbol de búsqueda binaria.
Se podría optimizar la función encontrarMiembrosPorDestino utilizando un índice hash para almacenar los destinos planificados por cada miembro de la familia.