UNIVERSIDAD ICESI

Departamento de TIC
Algoritmos y Estructuras de Datos
Tarea Integradora 1

Objetivos

Unidad 1: Análisis de Algoritmos

OE1.1. Calcular la complejidad temporal de algoritmos iterativos.
OE1.2. Calcular la complejidad espacial de algoritmos iterativos.
OE1.3. Caracterizar la entrada de un algoritmo iterativo con el fin de calcular la complejidad para el mejor y peor caso.
OE1.4. Analizar algoritmos independiente de una implementación concreta (no dependiente del lenguaje de programación).
OE1.5. Utilizar notación asintótica para describir la complejidad de algoritmos.
OE1.6. Evaluar varios algoritmos que resuelven el mismo problema en términos de sus complejidades computacionales.
OE1.7. Comprender la importancia del Modelo RAM en el proceso de análisis de algoritmos.

Unidad 2: Diseño y Construcción de Estructuras de Datos

OE2.1. Proponer y justificar un diseño para implementar una estructura de datos, siguiendo una metodología y considerando la flexibilidad en los tipos de datos y la complejidad temporal de las operaciones.
OE2.2. Implementar estructuras de datos extensibles y generales utilizando interfaces, herencia y tipos de datos genéricos.
OE2.3. Escribir el invariante de una clase e implementar los métodos necesarios para su verificación utilizando los elementos apropiados del lenguaje.
OE2.4. Diseñar, adaptar y utilizar estructuras de datos de acceso directo por llave, las cuales están basadas en la capacidad de las funciones de hashing para localizar una posición física a partir de una llave lógica.
OE2.5. Utilizar estructuras lineales FIFO, LIFO y diccionarios como parte de la solución de un problema.
OE2.7. Diseñar y construir las pruebas unitarias de cada una de las estructuras de datos lineales implementadas.

Enunciado

Un gran banco desea desarrollar un software que modele el funcionamiento de una de sus sedes con mayor flujo de personas. Para ello, lo ha contratado a usted y a su equipo de trabajo con el objetivo de construir un programa capaz de solventar todas la necesidades del cliente, entre las cuales se encuentran el proceso de turnos al momento del ingreso (fila para clientes y fila para personas con diferentes prioridades), manejo de tablas de datos, entre otros.

Con el fin de llevar a cabo procesos estadísticos y de agilizar el servicio de atención, esta sede bancaria registra el nombre y la cédula de todos los usuarios que ingresan al establecimiento a la hora de obtener su turno. Con ello no sólo se busca en cuál de las dos filas ubicar a la persona, sino que permite que la persona encargada de la atención de dicho usuario pueda buscarlo de manera eficiente en la base de datos y obtener toda su información antes de que este llegue a su despacho. Los datos que encontrará el encargado serán: nombre, cédula, cuenta bancaria, tarjetas de débito/crédito, fecha de pago de la tarjeta de crédito y fecha en que se incorporó al banco. Dicha información deberá mostrarse en pantalla.

Una vez sea el turno de atender al usuario, éste podrá realizar una o más de las siguientes operaciones:

a) **Retiro/consignación:** el usuario podrá modificar el monto de su cuenta de ahorros al solicitar un retiro o consignación.

b) **Cancelación de cuenta:** borra sus datos de la base de datos de clientes y los incorpora a una exclusiva para aquellos que desertan de dicho banco. Asimismo, se guardará tanto la fecha como el motivo de cancelación.

c) **Pago de tarjeta:** el usuario podrá pagar el monto utilizado con la tarjeta de crédito hasta el momento. Puede realizar el pago en efectivo o a través de su cuenta de ahorros.

La información de todos los usuarios presentes en la sede bancaria deberá visualizarse en una tabla tipo hoja de cálculo y podrá ser organizada respecto a cuatro (4) parámetros de su escogencia. Con el objetivo de encontrar el algoritmo más óptimo para el problema, el gerente le ha solicitado implementar un método de ordenamiento de su parecer por cada uno de los parámetro escogidos, con la restricción de que solamente uno (1) de ellos puede tener complejidad temporal promedio de (n^2).

Teniendo en cuenta los errores humanos que se introducen por parte de los cajeros ya sea por equivocaciones propias al digitar la solicitud del cliente, se le solicita al equipo de ingenieros agregar una funcionalidad de *undo* para poder deshacer las equivocaciones, incluso después de haberlas guardado.

El programa esperado por el banco debe implementar:

1. Una interfaz gráfica de usuario donde se pueda visualizar el estado actual de las 2 filas de espera.
2. Una interfaz gráfica de usuario que le permita visualizar y ordenar una tabla tipo hoja de cálculo por los siguientes criterios:
    a. Nombre del cliente
    b. Cédula del cliente
    c. Tiempo de vinculación del cliente
    d. Monto

    De los 4 algoritmos de ordenamiento solo uno puede ser de complejidad temporal promedio de $O(n^2)$, los 3 restantes deben ser de mejor complejidad.

3. Una interfaz gráfica de usuario que le permita buscar la información bancaria de un cliente a partir de su cédula y realizar las siguientes acciones:
    a. Retiro: Aumentar el monto de la cuenta de ahorros
    b. Consignación
    c. Cancelación de cuenta
    d. Pago de la tarjeta
    e. *Undo* de una de las anteriores acciones

UNIVERSIDAD ICESI

Departamento de TIC
Algoritmos y Estructuras de Datos
Tarea Integradora 1

Usted debe utilizar el método de la ingeniería para resolver este problema y dejar evidencia en su informe de los resultados de cada fase. Por ejemplo, en la fase 1 deben identificar claramente el problema, justificarlo y especificar los requerimientos funcionales. Recuerde revisar el Resumen del Método de la Ingeniería y el ejemplo del Método de la Ingeniería aplicado a un problema.

Entregables.

1. Informe PSP0. Cada estudiante debe entregar el informe de su desarrollo.
2. Entrega de informe del método de la ingeniería.
3. Análisis de complejidad temporal de cada uno de sus algoritmos
4. Análisis de complejidad espacial de cada uno de sus algoritmos
5. Especificación de Requerimientos y Diseño.
6. Diseño del TAD para cada estructura de datos requerida.
7. Diseño del diagrama de clases desacoplado y utilizando generics.
8. Diseño de los casos de prueba. Adicionalmente debe explicar cómo se resuelven dos casos, paso a paso (con dibujos, si es necesario), por cada estructura de datos diseñada e implementada.
9. Diseño del diagrama de clases de pruebas unitarias automáticas.
10. Todos los archivos deben estar almacenados en GitHub y debe evidenciarse su uso desde el inicio del proyecto.
11. Implementación de las pruebas unitarias automáticas.