Trabajo Práctico N°2

Asignatura: Algoritmos y Estructuras de Datos

Curso: K1024

Profesor: Ing. Pablo Damián Mendez

Integrantes:

Legajos	Nombres	Apellidos	Correos
2040517	Bruno Salvador	Sapoznik	bsapoznik@frba.utn.edu.ar
2027148	Chabela María	Lamas	clamas@frba.utn.edu.ar
2026715	Federico Matías	Cassina Carbia	fcassinacarbia@frba.utn.edu.ar
2037476	Germán María	Justo	gjusto@frba.utn.edu.ar

Fecha de entrega: 23/10

Descripción de la solución

El sistema informático propuesto cuenta con dos archivos principales: “clientes.bin” (archivo que contiene a los usuarios) y “procesados.bin” (archivo que guarda las compras realizadas en el día), que se cargan cada vez que inicia el programa. Asimismo, el programa dispone de dos structs que representan a:

1. El usuario. Este struct engloba la siguiente información:
   - int ID
   - int FechaCreacion (fecha de creación)
   - bool activo (estado de actividad, que comienza en true)
   - float TotalImporteCompras (importe de compras)
   - char eMail [26] (vector de caracteres del email)

2. Las compras. Este struct precisa la siguiente información:
   - int CompraID
   - char FechaHora[14] (vector de caracteres de Fecha y Hora)
   - float Monto
   - int UsuarioID
   - int NroArticulo (número de artículo)

Teniendo en cuenta lo mencionado anteriormente, el struct de usuarios y de compras se encuentran almacenados en dos distintas listas. Asimismo, éstos están precargados en el sistema, a modo de prueba.
Por lo que refiere al armado del proyecto, se comenzó armando un menú de opciones con los requisitos que se encuentran plasmados en la consigna. Luego, se diseñó un diagrama de bloques de subprogramas para determinar qué estructuras son necesarias para cada cumplir con cada requerimiento definido (véase en el anexo).
De la misma manera, una vez finalizada la etapa de diseño, se construyó el código. Se agregaron funciones de escribir, listar y borrar listas para testear el funcionamiento de cada subprograma y se expusieron por pantalla las direcciones de los punteros en cada ejecución (así se podía identificar los errores con precisión).
Por último, para la primera ejecución del programa, se agregaron clientes con sus importes a partir de la función “Agregar usuarios”, que fueron ordenados y luego adjuntados en el archivo “clientes.bin”.
A continuación, se encuentra el menú de opciones utilizado en el programa, con una breve descripción de cada punto.

Menú:
1---Cargar Usuarios
2---Listar Usuarios
3---Escribir Lista Archivo Clientes
4---Agregar nuevo usuario
5---Desactivar usuario existente
6---Buscar cliente por ID o Mail
7---Listar clientes activos
8---Procesar lote de compras

1- Cargar usuarios

El objetivo del subprograma es cargar los usuarios del archivo usuarios.bin en una lista. Para ello, se abre el archivo en modo lectura. Cuando la apertura sale exitosa, se determina el tamaño de la lista por medio de la función fseek (lleva el puntero al final del archivo) y la función ftell (calcula la cantidad de bytes ocupados por el archivo). Por consiguiente, dividiendo el resultado del ftell por la cantidad de bytes que ocupa cada struct (sizeof(struct)), se conoce cuántos elementos hay en el archivo.
Si no se encuentran elementos en el archivo, se muestra por pantalla el siguiente mensaje: “lista vacía agregue elementos”.
Caso contrario, con el tamaño de los elementos del archivo se hace un loop, leyendo los nodos y almacenándolos en una variable. De esta manera, con la función InsertarOrdenado, se agrega cada nodo a una lista, teniendo en cuenta los valores de las variables creadas.

2-Listar usuarios

Luego de cargar los usuarios, el objetivo de este subprograma es mostrar la lista de éstos por pantalla. Un aspecto tenido en cuenta para su correcta ejecución, fue ir recorriendo las listas secuencialmente.

3-Escribir lista archivo clientes

Este subprograma se implementa para guardar, en la primera ejecución, los clientes en el archivo “clientes.bin”.

4-Agregar nuevo usuario

Como se menciona en el título, el objetivo de este subprograma es añadir un nuevo usuario a la lista. El nuevo cliente se agrega en el primer nodo de la lista al contar con un monto igualado a cero (de esta manera ésta se mantiene ordenada por importes).

5-Desactivar usuario existente

Para desactivar un usuario existente, se hace un bucle while hasta encontrar el ID buscado y cambiar el valor de la propiedad de su struct de activo a false. Cabe aclarar que, las listas se actualizan teniendo en cuenta el bool de activo, y de ese momento en adelante se listarán únicamente los clientes que no se encuentran desactivados.

6-Buscar cliente por ID o Mail

El objetivo de este subprograma es mostrar por pantalla los datos de un usuario. Por consiguiente, esta función comienza preguntando cómo se desea realizar la búsqueda, ya sea ingresando el ID o el Mail del usuario. A partir de ello, el algoritmo recorre la lista de usuarios con el dato colocado (esto se realiza con la función correspondiente: BuscarClientePorID o BuscarClientePorMail) hasta encontrarlo. Si llega al final de la misma y no localiza al usuario solicitado, señala por medio de un mensaje que la búsqueda no ha sido exitosa. Caso contrario, se muestran por pantalla los datos correspondientes.

7-Listar clientes activos

El propósito de esta función es recorrer la lista y mostrar por pantalla los usuarios que cuentan con la propiedad de activo en true.

8-Procesar lote de compras

Este subprograma busca agregar a los usuarios correspondientes, la suma de los importes de sus distintas compras. Para ello, primero escribe el lote dentro de “procesados.bin” y luego recorre, por cada compra, la lista de usuarios. De esta manera, si llegan a concordar los datos de los clientes, se modifica dentro de los datos de ese usuario, su importe con la suma de las compras efectuadas.

9-Listar compras de un usuario dado

La meta de esta función es mostrar por pantalla las compras efectuadas por un usuario específico. En efecto, se solicita un ID y después se recorre la lista hasta encontrar el usuario que corresponda. En caso de que el ID no sea el de ningún usuario, se señala por medio de un mensaje que la búsqueda no ha sido exitosa.

10-Escribir reporte HTML

Este subprograma permite realizar un reporte en una página web, con los datos de las compras. Cabe aclarar que, los datos expuestos son aquellos que se encuentran ejecutados dentro de un período de tiempo determinado. De esta manera, la función comienza pidiendo estas dos fechas límites. A partir de que los datos sean ingresados, por medio de un comparador de strings y de los caracteres particulares de HTML, se arma una tabla. Por último, luego de la compilación, se puede ver el resultado en una página web.

11-Escribir reporte CSV

Este punto busca realizar un reporte como el punto 10 pero por medio de un Excel. Para ello se piden las dos fechas límites y, a partir de un comparador de strings y del código particular del CSV, se diseña la tabla. Finalmente, después de la compilación, se puede observar el resultado en un Excel.

12- Borrar lista de usuarios

Esta función se utiliza como prueba para ejecutar el borrado de lista de usuarios.

13-Finalizar Jornada

El objetivo de este subprograma es actualizar el archivo “clientes.bin" con los usuarios que se encuentren activos, y borrar las listas creadas en el programa (listas de procesados, listas de clientes y listas de usuario). De esta manera, no habrá fugas de memoria.

División de tareas

El desarrollo del programa se dividió en distintas partes: lectura y escritura de archivos, listado y modificación de listas, procesado de lotes y creación de reportes en los formatos pedidos (HTML y CSV). Asimismo, se realizaron reuniones semanales con el objetivo de revisar y corregir el programa, en grupo, para lograr obtener su correcto funcionamiento.
Se utilizó como medio de trabajo, la aplicación GitHub Desktop para mantener un seguimiento del programa en la cual cada integrante subía su avance al repositorio. Luego, se revisó el programa por completo para pulir los últimos detalles, comprobar que no hubiese errores y que compile apropiadamente. Finalmente, el informe fue completado en grupo teniendo en cuenta los requisitos de la consigna.

ANEXO:
Diagrama de Bloques