Algoritmos y Estructuras de Datos Trabajo practico K1024

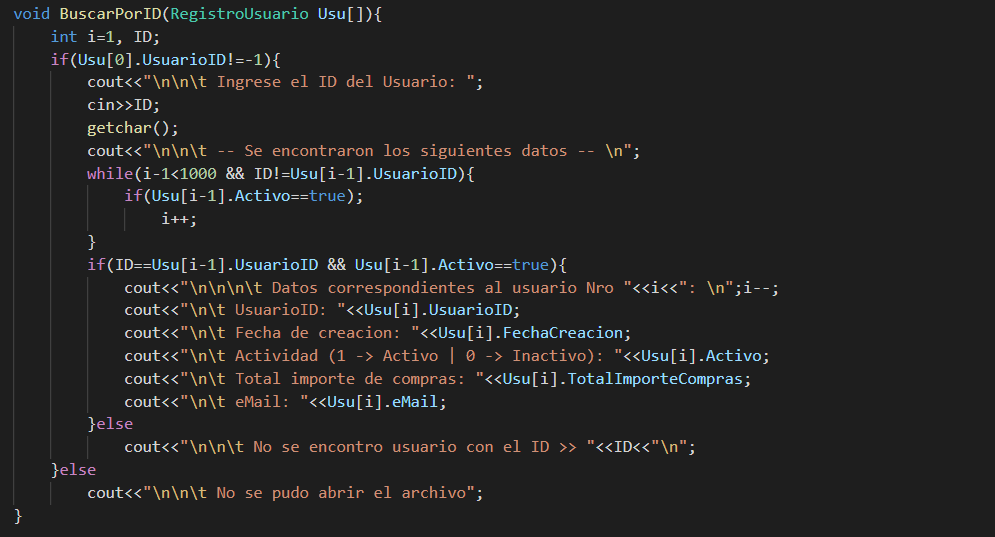
|  |  |  |
| --- | --- | --- |
|  | | |
|  | | |
| **Fernández**, Nicolás | 1782393 | nicolasfernandez@frba.utn.edu.ar |
| **Scotta**, Ezequiel Santiago | LEGAJO | escotta@frba.utn.edu.ar |
| **Araoz**, Santiago | 203373-2 | saraoz@frba.utn.edu.ar |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FECHA DE PRESENTACIÓN: | 07/10/2021 |  | FIRMA PROFESOR |  |
|  |  |  |  |  |
| FECHA DE DEVOLUCIÓN: |  |  | CALIFICACIÓN |  |

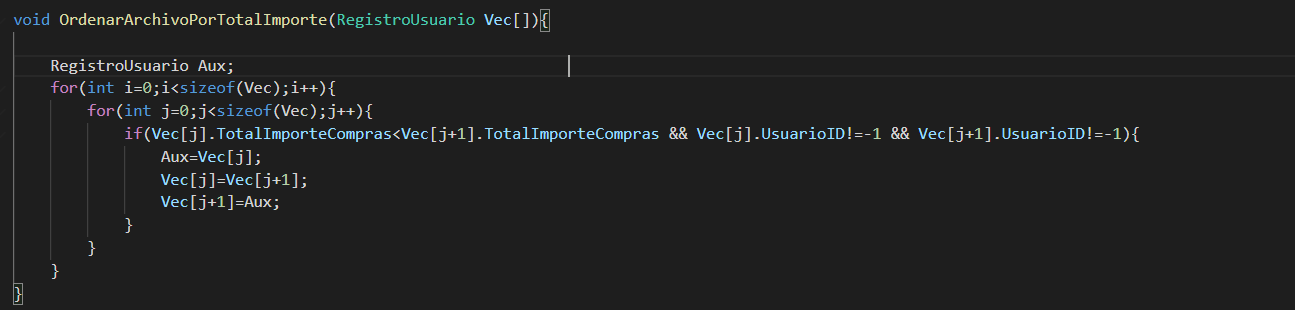
Sacamos la conclusión de que usamos un solo vector ya que pensamos que era más rápido que solo manejar el archivo solo.

Primero para manejar los usuarios decidimos crear

Los primeros 3 puntos fueron fáciles de hacer ya que consistía en levantar los clientes del archivo, para el punto 2 que consistía en un registro usuario, no fue complicado. Era más que nada algoritmos para comprobar que la información escrita sea válida. lo que se puede destacar fue que desarrollamos una función bool llamada buscarUsuarioID que te permite comprobar si un usuario está en el archivo. Esta la usamos ahora para que no se sobrescriban los Id de los usuarios. Tiene más aplicaciones a lo largo del código. Para el punto 3 también hicimos una función bool que activase o desactivase los clientes según lo pidieran.

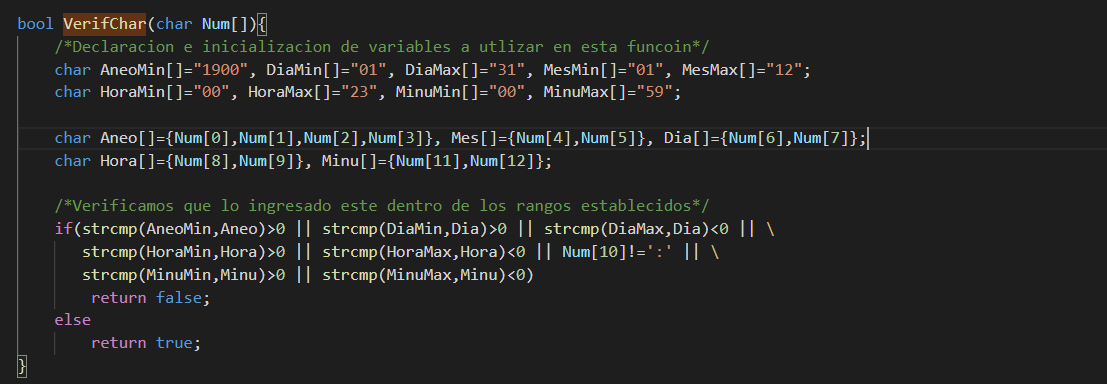


En esta función BuscarPorId () se pide que pase por parámetro la estructura registro usuario, la cual se va a estar almacenándose en el vector Usu. Mientras que la posición del vector sea menor a 1000 y la Id ingresada sea distinta del archivo se va a seguir buscando, cuando coincida se va a preguntar si el usuario en cuestión está activo. A partir de eso se traen los siguientes datos. La Id, la fecha de registro y demás datos del usuario. En caso contrario no se abra encontrado el usuario o no se podrá haber abierto el archivo.



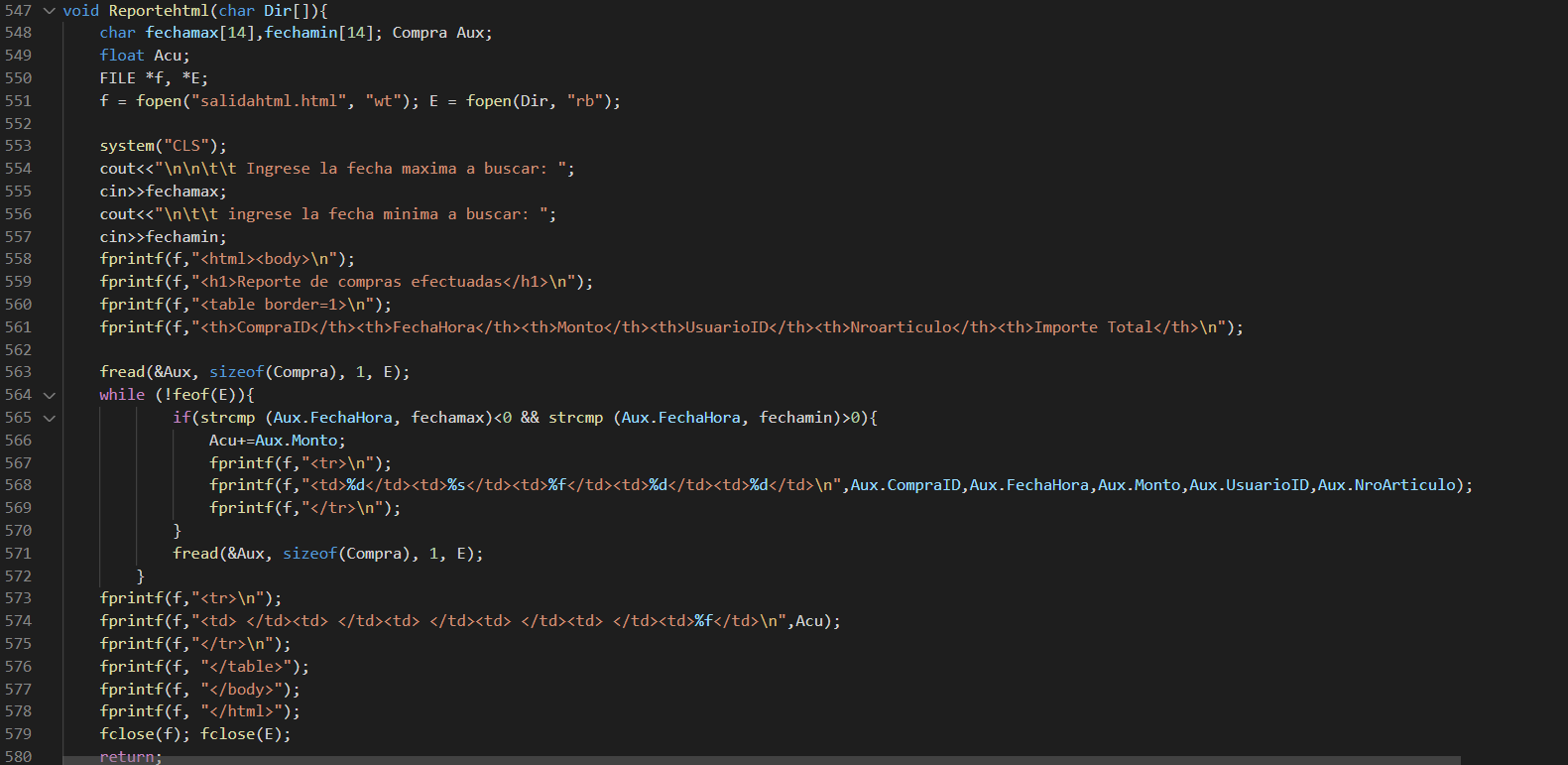
Void Ordenar archivo por total de importe. Recibe por parámetros la estructura de registro y la mete en un vector.

Lo que hace esta función es ordenar los usuarios por total de monto a través de un algoritmo de burbujeo. En el if lo que pregunta es si el monto de compras de un tal usuario es menor al del usuario siguiente, y ambas posiciones de vectores esta ocupadas (se comprueba esto a través de la ID si es -1 significa que el espacio no fue usado) se procede a ordenar. La variable AUX almacena al usuario en cuestión, el espacio donde estaba el usuario se ocupa por el que estaba una posición adelante. Y el usuario en cuestión, toma el lugar del usuario siguiente.



Esta función la queremos destacar ya que sirve para verificar que la hora ingresada en el lote de compras sea válida, para esto lo que hicimos primero setear una fecha de año, mes, día, hora, minutos mínima y máxima.

Después separamos el vector original en distintos mini vectores de año, mes, día, hora y numero. Y a partir de lo seteado anteriormente y los vectores con los datos separados los comprobamos para que no estén mal. Que el año este dentro del año correcto, que el mes sea un valor entre 1 y 12 y así con todos los datos. Si no se encontraba ningún error significaba que la fecha estaba bien, caso contrario devolvía un false.



Esta función la hicimos con el objetivo de mostrar los datos en un HTML.

La función se basó en crear un archivo HTML y hacer muchos fprintf con etiquetas HTML y con los datos que queríamos mostrar.

