**Universidad Tecnológica Nacional**

Facultad Regional Buenos Aires

Algoritmos y Estructuras de Datos

Imagen que contiene nombre de la empresa

Descripción generada automáticamente

Curso K1031

Ing. Pablo D. Mendez

Trabajo práctico grupal

**Integrantes:**

|  |  |  |
| --- | --- | --- |
| Nombre y Apellido | Legajo | Correo institucional |
| Alex Brian Tolaba | 1743934 | [aletolaba@frba.utn.edu.ar](mailto:a%6c%65%74%6fl%61ba@%66%72%62a%2e%75t%6e%2ee%64%75.%61%72) |
| Federico Fanara Maldonado | 1770846 | [ffanaramaldonado@frba.utn.edu.ar](mailto:%66%66a%6ear%61%6daldo%6e%61d%6f@%66rb%61.u%74n%2e%65%64u%2e%61r) |
| Maia Camila Velazquez Yarbe | 1744665 | mvelazquezyarbe@frba.utn.edu.ar |
| Marianela Ivanna Sayago | 1775420 | [msayago@frba.utn.edu.ar](mailto:%6ds%61y%61g%6f@fr%62a.%75tn.%65d%75%2e%61r) |

**Descripción de las funciones:**

CargarPoliza y CargarLoteInc:

Estas funciones tienen el objetivo de cargar la cantidad de pólizas o el lote de incidentes que el usuario desee ingresar, según de que función se trate. Como hipótesis, suponemos que el usuario no ingresará datos distintos con el mismo número de póliza o código de incidente como, así también, datos erróneos.  
Se considera que cuando se crea el archivo, el número de incidentes es igual 0, ya que no se cargó incidentes hasta el momento. También consideramos que, al ingresar el estado de actividad y el estado de la cuota, lo va a hacer en mayúsculas, como se indica.

La función CargarPoliza guarda los datos en el archivo Asegurados.BAK. Tiene la opción de crear el archivo y cargar pólizas, como así también, de seguir agregándolas al final del archivo.

La función CargarLoteInc guarda los datos en el archivo Procesados.BAK. Tiene la opción de crear el archivo y cargar los incidentes, como así también, de seguir agregándolos al final del archivo.

En ambas funciones, contemplamos la cantidad de pólizas o incidentes que el usuario quiera cargar.

MostPoliza y MostInc:

Estas funciones llaman a sus respectivos archivos para mostrar los datos cargados. MostPoliza lee el archivo Asegurados.BAK y MostInc lee el archivo Procesados.BAK.

DesPoliza, BuscarPorDni y BuscarPoliza:

Esta función abre el archivo Asegurados.BAK en modo lectura y escritura en la primera función y sólo lectura en la segunda.

En un principio, le pedimos al usuario que ingrese el número de póliza a desactivar o el dni a buscar o el número de póliza a buscar, según de qué función se trate. Este dato lo guardamos en una variable. Luego, recorremos el archivo buscando la presencia de este.

Si se quiere desactivar una póliza y hay presencia, se le cambia el valor a la variable activa. Esto se ve reflejado en el archivo ya que luego nos ubicamos donde estaba tal número de póliza para escribir el nuevo valor de la variable.

Si se busca un dni y hay presencia, se muestra por pantalla la información correspondiente.

Si se busca un número de póliza y hay presencia, se muestra por pantalla la información correspondiente.

En el caso de no haber presencia, se le avisa al usuario que no se ha encontrado el número de póliza o el dni.

ActArch:

Esta función llama a los archivos Asegurados.BAK y Procesados.BAK. Se creo un archivo auxiliar con el objetivo de escribir las pólizas activas y sus cantidades de incidentes actualizadas. Al finalizar la carga de este, se elimina definitivamente el archivo Asegurados.BAK y se renombra al archivo auxiliar como Asegurados.BAK.

Recorremos el archivo Asegurados.BAK buscando que la variable activa tenga como valor “V”. Si cumple con ello, procedemos a recorrer el archivo Procesados.BAK buscando el número de póliza que leímos.

Si se encuentra, se suma la cantidad de incidentes. Al terminar de leer Procesados.BAK, escribimos en el archivo auxiliar la información correspondiente a la póliza activa con su respectiva cantidad de incidentes actualizados.

OrdDes:

La función OrdDes recorre el archivo Asegurados.BAK. A medida que va leyendo cada póliza, verifica que el estado de actividad sea “V”.

Si la póliza está activa, en un vector se guardan la cantidad de incidentes correspondientes a la misma. Así sucesivamente con las pólizas que cumplan la condición.

Teniendo cargado el vector, ordenamos descendentemente los incidentes.

Con el vector ordenado, recorremos el archivo Asegurados.BAK. Mostramos la información correspondiente a la cantidad de incidente si coincide con el que vamos leyendo del archivo. De esta forma, se logra listar todas las pólizas activas ordenadas por saldo descendente.

Reporte, CantInc, comprararPol, escribirReporteHTML, escribirReporteCSV:

El reporte llama a los archivos Asegurados.BAK y Procesados.BAK. De Asegurados.BAK obtiene la cantidad de pólizas ingresadas.

Cuando recorremos Procesados.BAK, verificamos que la fecha leída tenga como año 2020 creando un vector que guarde los primeros cuatro caracteres de esta. Luego, llamamos a la función CantInc pasandole el valor número de póliza leído y el archivo Procesados.BAK.

La función CantInc recorre el archivo Procesados.BAK de la misma forma, pero con el objetivo de contabilizar todos los incidentes que tenga el número de póliza que le pasamos por valor.

Si el resultado devuelto por la función CantInc es mayor a 5, procedemos a llamar a la función compararPol, pasándole el número de póliza leído, un vector que contiene a todos los números de pólizas que tengan más de 5 incidentes durante todo el 2020 y una variable que delimita el tamaño del vector a medida que se sigan sumando pólizas que cumplan con la condición. Esta última variable es igual a cero al principio.

La función compararPol con una variable tipo bool inicializada en false va a recorrer el vector que contiene a todos los números de pólizas que tengan más de 5 incidentes durante todo el 2020, verificando si el número de póliza pasado está en el mismo. Si está, la función devuelve true.

Si el resultado devuelto por la función compararPol es true, se prosigue a leer el siguiente registro ya que la cantidad total de incidentes con ese número de póliza ya fueron almacenados en el vector. En caso de ser false, se guarda el número de póliza en el vector pol\_may5 y se aumenta la capacidad de este sumándole uno a la variable contar.

La función Reporte también recibió una variable código que indica el tipo de reporte que quiere el usuario. Entonces bien, a medida que vayamos cargando nuevos números de pólizas que cumplan con la condición iremos escribiendo en el archivo correspondiente. A dichos procedimientos les pasamos el numeró de póliza que cumpla las condiciones y el archivo Asegurados.BAK.

Los procedimientos escribirReporteHTML y escribirReporteCSV utilizan dos archivos cada uno. Ambos utilizan el archivo Asegurados.BAK para mostrar los datos correspondientes cuando, a medida que se recorra el archivo, se verifique que se trata del mismo número de póliza pasado. La única diferencia entre estos procedimientos es que escribirReporteHTML crea un archivo .html y escribirReporteCSV crea un archivo .CSV al que le van cargando los registros correspondientes.

**Dinámica de trabajo:**

Para la dinámica de grupo, optamos por definir un horario, en el cual esté la mayoría de los integrantes disponibles. En ese horario, ingresábamos a una sala en donde uno de nosotros, se encargaba de la escritura del código, y a la vez, mientras compartía pantalla al resto, todos en conjunto íbamos ideando y dictando la resolución de las consignas.  
Además, cuando surgía un inconveniente o problema el cual no nos dejaba avanzar, al finalizar el meet, cada uno tenía que pensar posibles soluciones, para en la siguiente reunión ponerlas en práctica.  
Para la resolución de la entrega, nos dividimos en equipos conformados por dos personas, en el cual, uno se encargaba de la realización del informe y el otro grupo, elaboraba del diagrama de subprogramas.