Trabajo Práctico K1031

Algoritmos y estructura de datos

Fecha de entrega: 28/10/2021

Profesor: Ing. Pablo Mendez

Alumnos: Micaela Andrea Paredes, Agustin Emanuel Teresin, Gissbel Castillo

Legajos: 176.128-6, 175.436-1, 1466124

Correos: miparedes@frba.utn.edu.ar, ateresin@frba.utn.edu.ar, mori@frba.utn.edu.ar

Librerías que fueron utilizadas

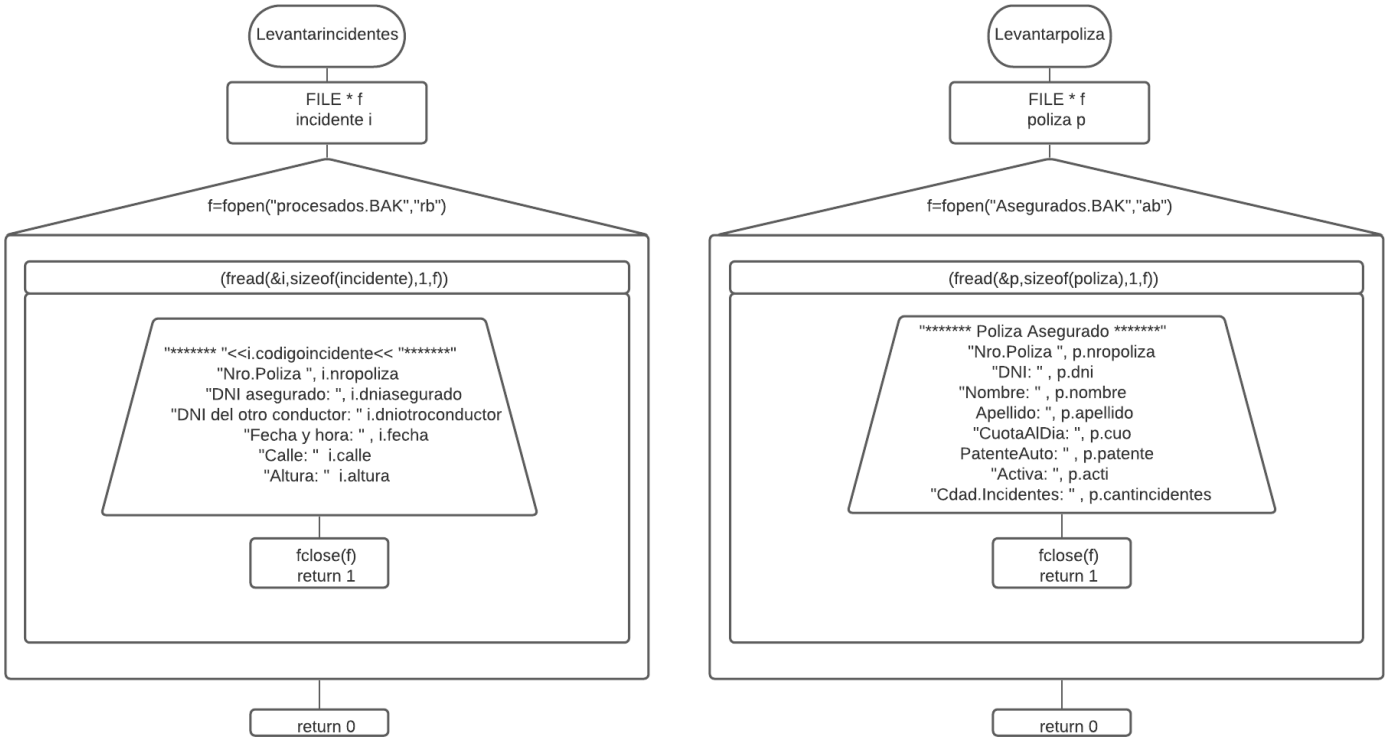
Para los algoritmos de buscar, levantar, cargar, actualizar, desactivar y menú, fueron usadas las librerías de <iostream> <stdio.h>, <stdlib.h>, <string.h>, <conio.h>, <cstring>.

Para el algoritmo de ordenar y crear html se hizo uso de la librería <fstream>.

Estructuras y diagramas de Lindsay

Levantar

La estructura *levantar póliza* es lo primero que realiza el programa para poder mostrarnos las pólizas que ya fueron ingresadas anteriormente. Este algoritmo, lo que hace es abrir el archivo binario de "Asegurados.BAK" con el modo de abertura "rb" (modo lectura) y mostrarnos el contenido del archivo hasta que termine de leerlo. El algoritmo *levantar incidentes* funciona de la misma forma y se implementa a la hora de procesar un lote de incidentes.



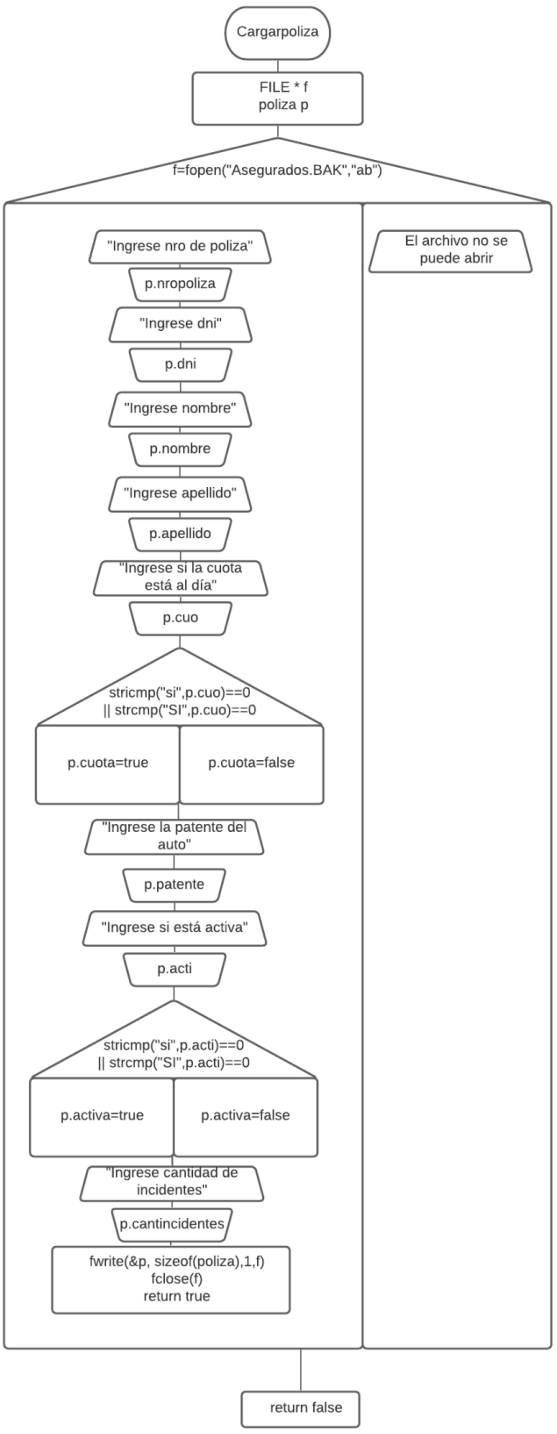
Cargar

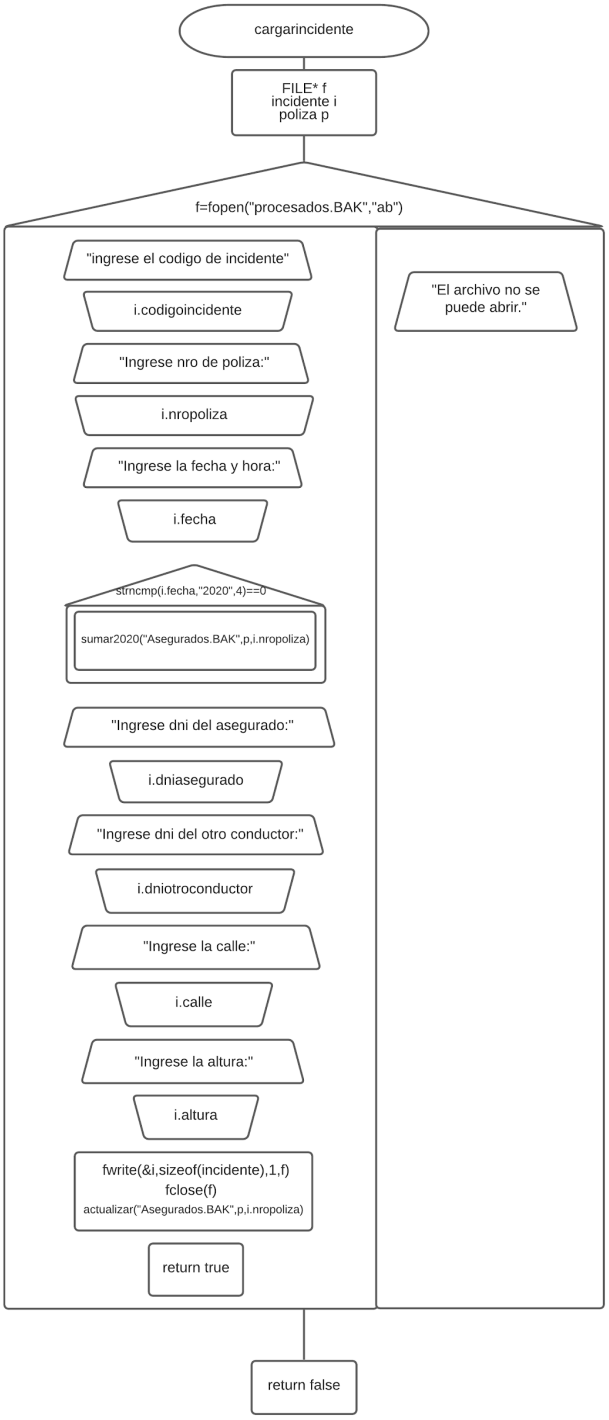
Los siguientes algoritmos son de tipo bool y funcionan de tal manera que, al comenzar el procedimiento se declaran un flujo con el archivo utilizando un puntero a FILE*, luego se entra a un condicional el cual pregunta si se pudo abrir el archivo, en caso negativo, se muestra por pantalla un mensaje que informa que hubo error al abrirlo y se retorna el valor false.

En caso positivo, se comienza a cargar los datos con el struct asociado, utilizando salida por pantalla para indicar qué dato debe cargarse y en qué orden.

En el caso de preguntar si la cuota está al día y si la patente está activa, se utiliza una variable booleana, la cual según lo que ingrese el usuario está puede adoptar los valores de "true" o "false".

Finalmente, al completar los datos necesarios se cierra el archivo y se devuelve el valor true.

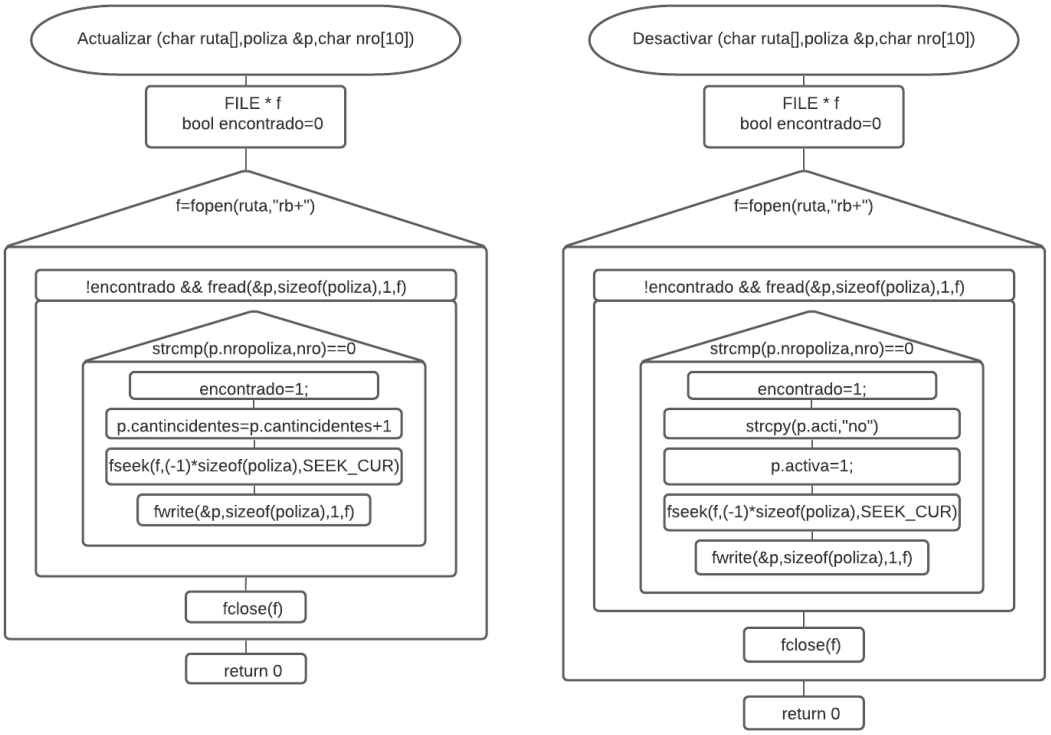




Cargarincidente se utiliza para cumplir el propósito del punto número 6, que pedía procesar un lote de incidentes. Su funcionamiento es similar al de cargarpoliza en la forma de ingresar los datos, pero se diferencia en que en cargarincidente se utiliza otro struct que es el de incidentes y se implementan los subprogramas actualizar y cargar2020 los cuales se encargan de aumentar la cantidad de incidentes relacionado con la póliza y cargar las pólizas de esos usuarios que tengan más de 5 incidentes en el año 2020.

Desactivar y actualizar

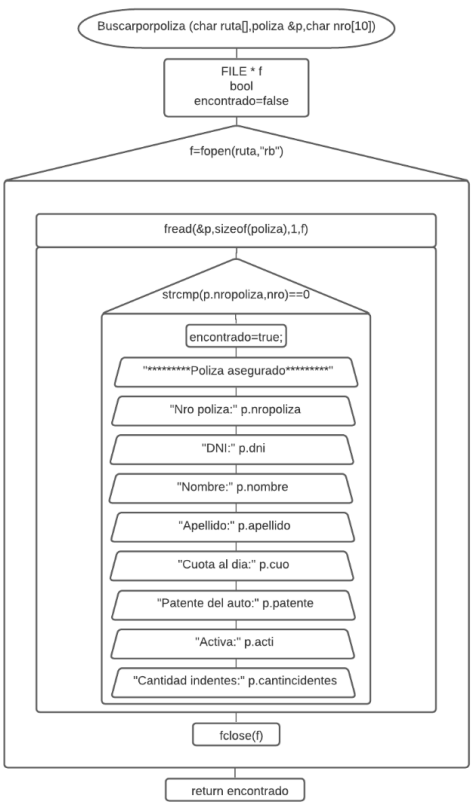
El procedimiento *desactivar* lo que hace es recibir por parámetros la ruta de un archivo binario, el struct que se utilizará y la información a buscar. Primero abre el archivo el modo lectura y escritura, y con la implementación de un while busca por el archivo si existe alguna coincidencia entre el parámetro que ingresó el usuario y la información del archivo. Cuando lo encuentra cambia p.activa para indicar que ya no está activo y con el uso de *"fseek(f,(-1)*sizeof(poliza),SEEK_CUR)"* y *"fwrite(&p,sizeof(poliza),1,f)"* modifica esa información reescribiendo el archivo en esa sección. De la misma forma, el algoritmo de actualizar busca el número de póliza al que se agregó un lote de incidentes y le suma 1 a la cantidad de incidentes que la póliza tenía (p.cantincidentes).

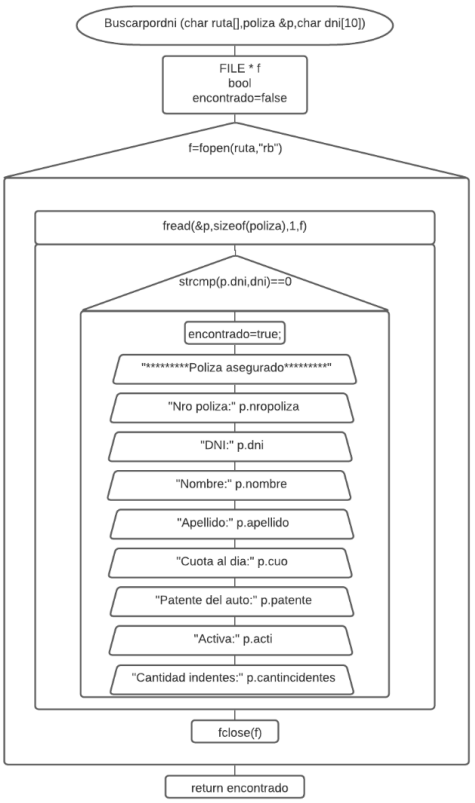


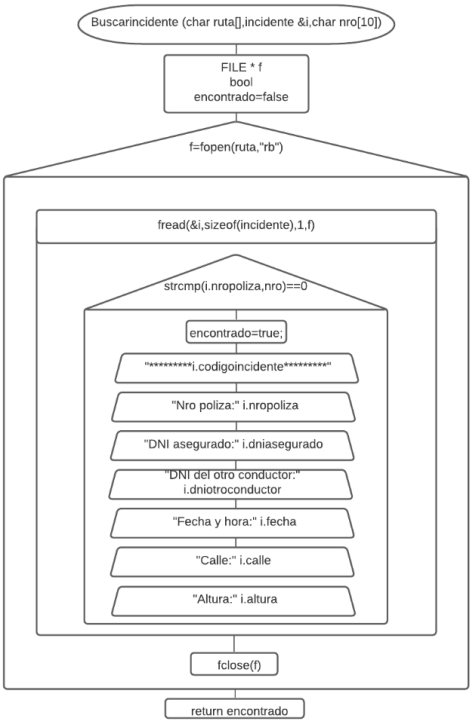
Algoritmos de búsqueda

Estos tres procedimientos son similares, su única diferencia son los parámetros recibidos. En los tres se recibe la ruta del archivo, en los casos de "buscarpoliza" y "buscarpordni" se recibe además, el struct "póliza", mientras que en "Buscarincidente" el struct es el de incidentes.

El algoritmo funciona de tal manera que, se abre el archivo binario en modo lectura y luego con un ciclo de repetición while se corrobora si existen coincidencias entre los datos almacenados en el archivo y lo ingresado previamente por el usuario, sea el número de póliza, el dni o el número de incidentes. Si existe coincidencia, se muestra por pantalla todos los datos del usuario en cuestión, finalmente, se cierra el archivo y se devuelve el valor de la variable "encontrado".

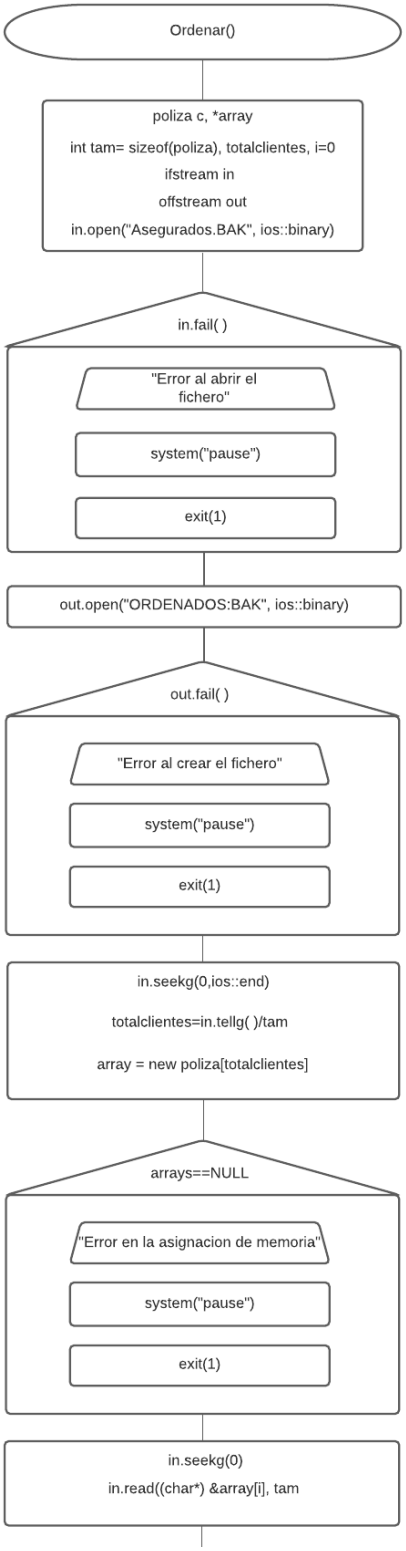


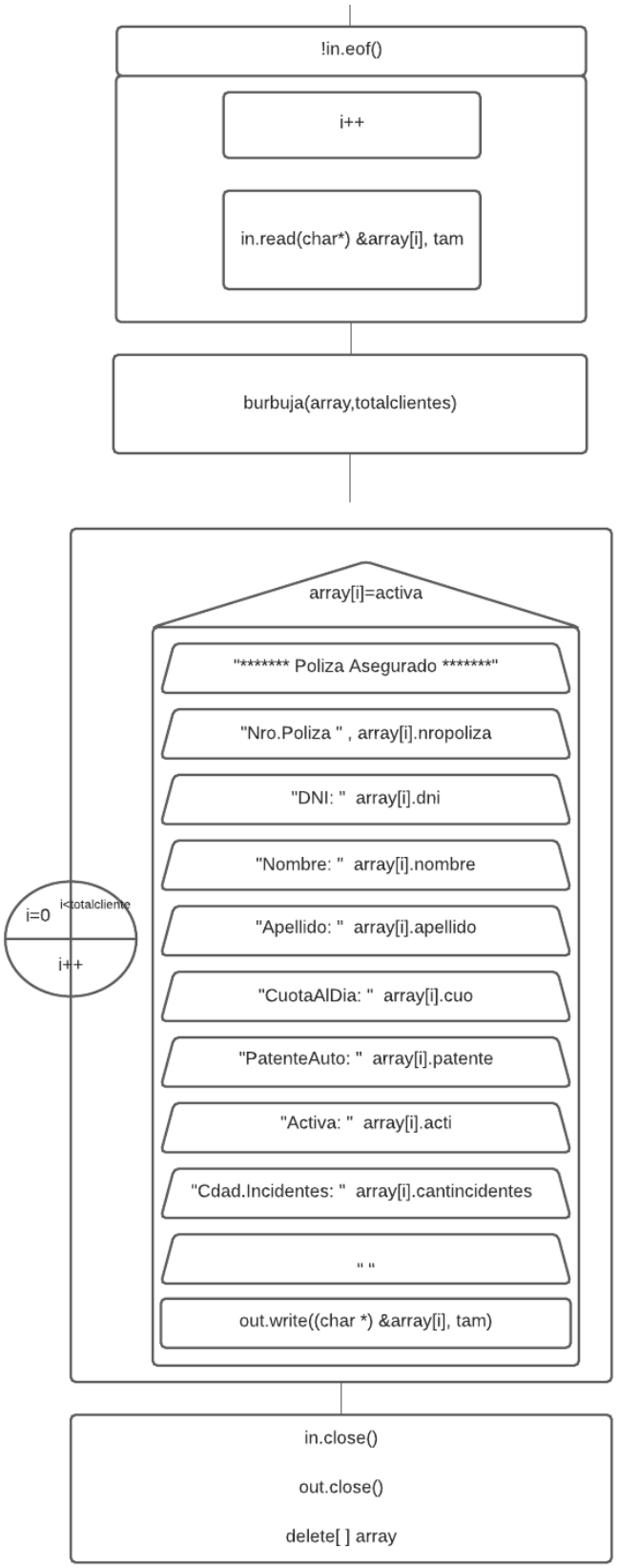


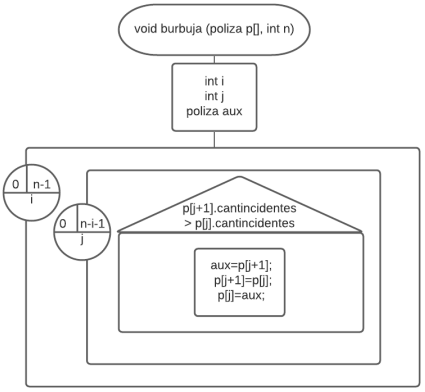


ORDENAR

Para este algoritmo se hizo uso de la librería <fstream>. Lo que hace este algoritmo es pasar los datos del archivo binario "Asegurados.BAK" a un array, ordenar el array a través de la función burbuja y luego pasar la información ordenada a un nuevo archivo. Para lograrlo primero abre los dos archivos (el original y el que tendrá la información ya ordenada) utilizando "*in.open("Asegurados.BAK", ios::binary)*" para indicar que son archivos binarios. Una vez abiertos se mide el tamaño de la información del archivo para luego crear el array con el mismo tamaño y con un while lee la información de "Asegurados.BAK" y la transcribe al arreglo. Finalmente se usa el subprograma "*burbuja*" para ordenar el array según la cantidad de incidentes y luego se escriben todos los datos ya ordenados en el archivo de "ORDENADOS.BAK".

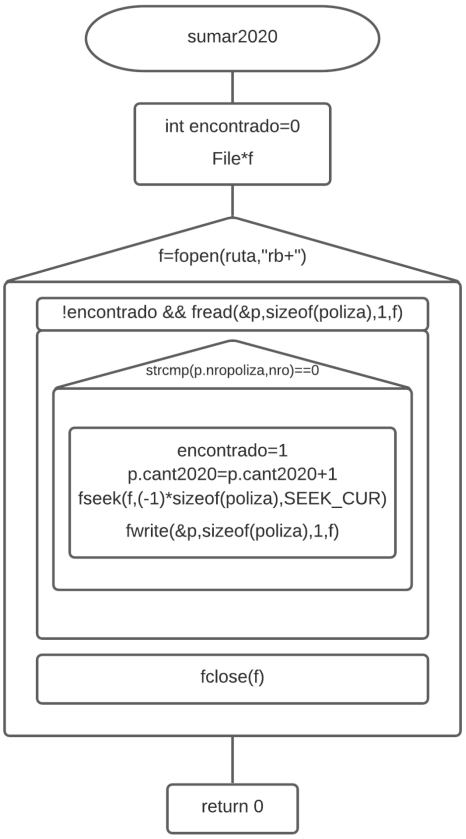


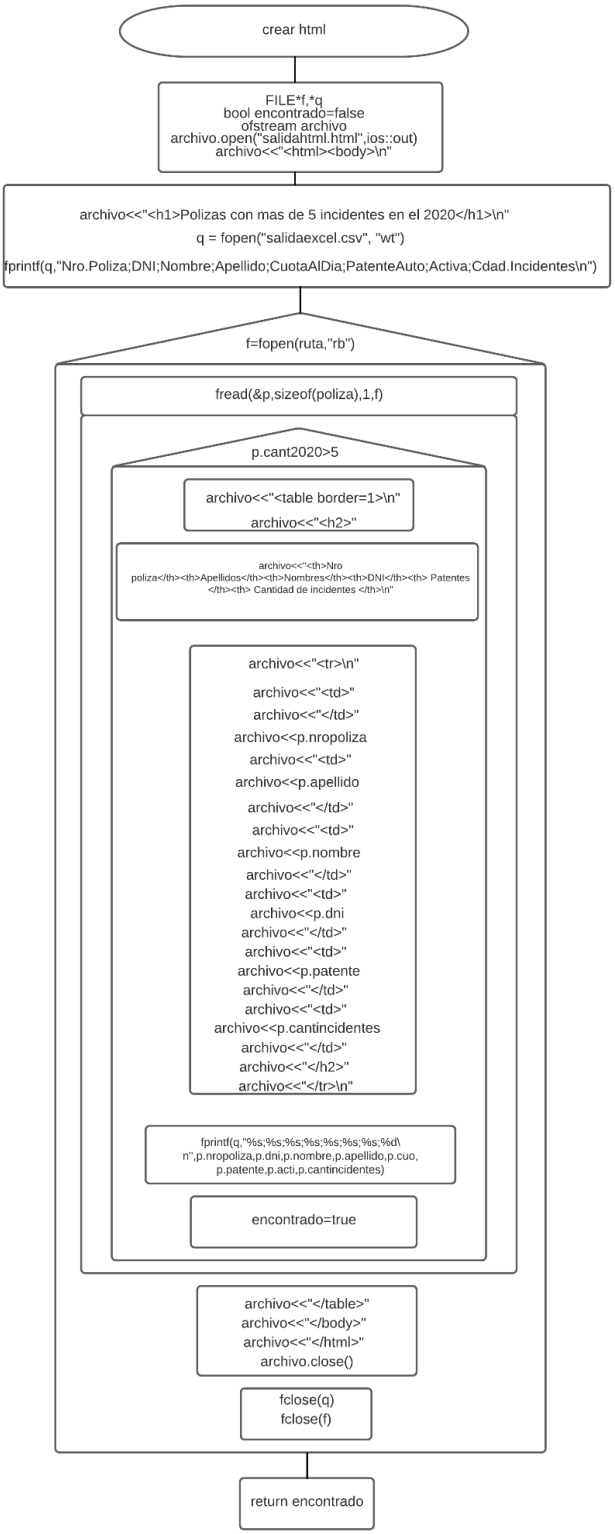

!in.eof()
i++
in.read(char*) &array[i], tam
burbuja(array,totalclientes)
array[i]=activa
"******* Poliza Asegurado *******"
"Nro.Poliza " , array[i].nropoliza
"DNI: " array[i].dni
"Nombre: " array[i].nombre
"Apellido: " array[i].apellido
"CuotaAlDia: " array[i].cuo
"PatenteAuto: " array[i].patente
"Activa: " array[i].acti
"Cdad.Incidentes: " array[i].cantincidentes
" "
out.write((char *) &array[i], tam)
i=0
i<totalcliente
i++
in.close()
out.close()
delete[ ] array



MOSTRAR TODAS LAS PÓLIZAS QUE TENGAN MÁS DE 5 INCIDENTES DURANTE TODO EL 2020

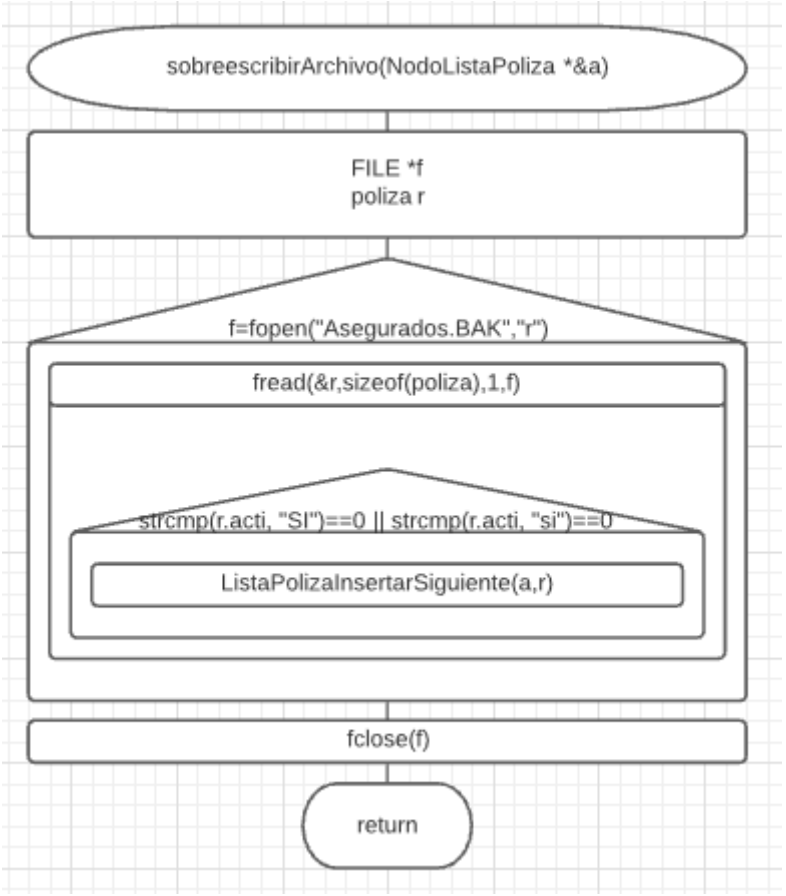
Para este algoritmo se utiliza la librería <fstream>. Lo primero que se hace es abrir un archivo ofstream (archivo de salida) y se escribe en él las sentencias para crear un archivo html. Al mismo tiempo se crea un archivo csv y escribe en éste los encabezados de las celdas. Luego abre el archivo "Asegurados.BAK" y mientras lee el archivo busca a las pólizas que tienen una cantidad de incidentes en el año 2020 mayor que cinco. Las pólizas que encuentra las escribe en el archivo html dentro de una tabla y realiza lo mismo en el archivo csv.




crear html
FILE*f,*q
bool encontrado=false
ofstream archivo
archivo.open("salidahtml.html",ios::out)
archivo<<"<html><body>\n"
archivo<<"<h1>Polizas con mas de 5 incidentes en el 2020</h1>\n"
q = fopen("salidaexcel.csv", "wt")
fprintf(q,"Nro.Poliza;DNI;Nombre;Apellido;CuotaAlDia;PatenteAuto;Activa;Cdad.Incidentes\n")
f=fopen(ruta,"rb")
fread(&p,sizeof(poliza),1,f)
p.cant2020>5
archivo<<"<table border=1>\n"
archivo<<"<h2>"
archivo<<"<th>Nro poliza</th><th>Apellidos</th><th>Nombres</th><th>DNI</th><th> Patentes </th><th> Cantidad de incidentes </th>\n"
archivo<<"<tr>\n"
archivo<<"<td>"
archivo<<"</td>"
archivo<<p.nropoliza
archivo<<"<td>"
archivo<<p.apellido
archivo<<"</td>"
archivo<<"<td>"
archivo<<p.nombre
archivo<<"</td>"
archivo<<"<td>"
archivo<<p.dni
archivo<<"</td>"
archivo<<"<td>"
archivo<<p.patente
archivo<<"</td>"
archivo<<"<td>"
archivo<<p.cantincidentes
archivo<<"</td>"
archivo<<"</h2>"
archivo<<"</tr>\n"
fprintf(q,"%s;%s;%s;%s;%s;%s;%s;%d\n",p.nropoliza,p.dni,p.nombre,p.apellido,p.cuo,p.patente,p.acti,p.cantincidentes)
encontrado=true
archivo<<"</table>"
archivo<<"</body>"
archivo<<"</html>"
archivo.close()
fclose(q)
fclose(f)
return encontrado

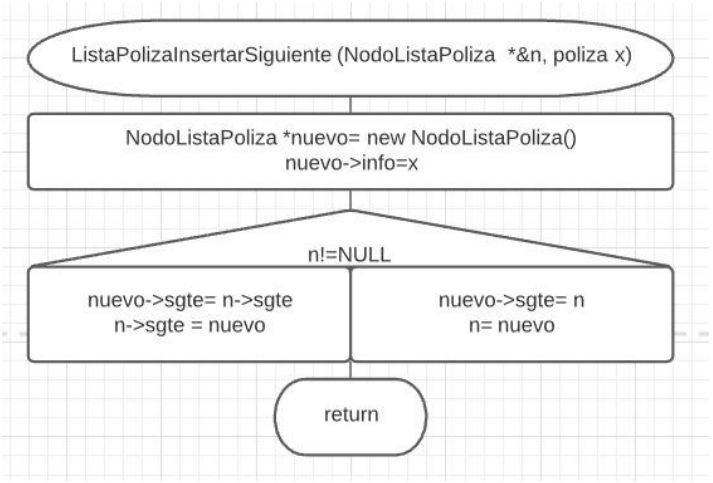
FINALIZAR JORNADA

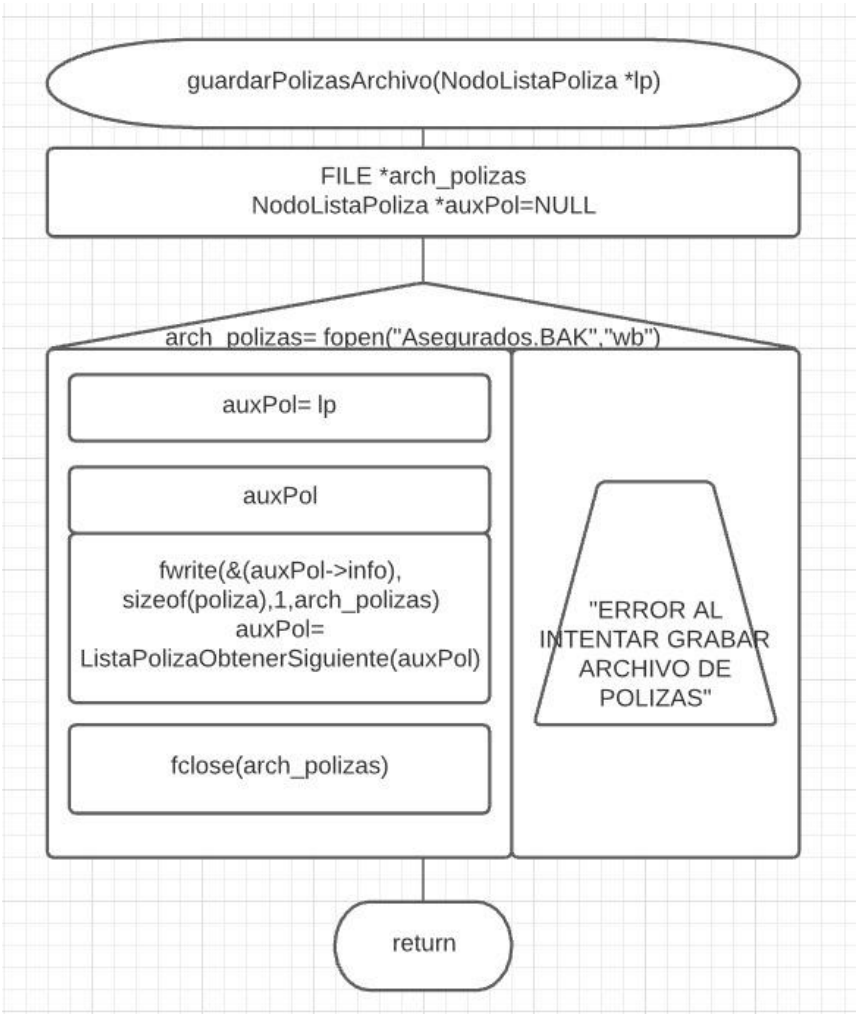
El procedimiento *sobreescribirArchivo* toma por parámetro la lista de las pólizas. Se abre el archivo "Asegurados.BAK" para leerlo desde el principio y mientras recorre la lista escribe en el archivo aquella información que tenga las pólizas activas.



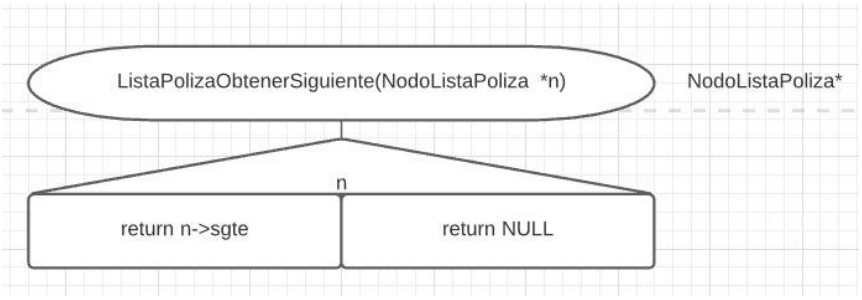
Para poder realizar el procedimiento anterior se empleó el procedimiento *ListaPolizaInsertarSiguiente*, el cual toma por parámetro la lista y la 'póliza'.
Se crea un nuevo nodo del tipo NodoListaPoliza y se guarda la información de la póliza, mientras la lista no esté vacía se le asigna al nodo nuevo->sgte la dirección del nodo de la lista (n->sgte) y lo que apuntaba a éste ahora pasa a apuntar al nuevo nodo.
Caso contrario a nuevo->sgte se le asigna el valor de la lista, que en este caso es NULL, y la lista ahora pasa a apuntar al nuevo nodo.

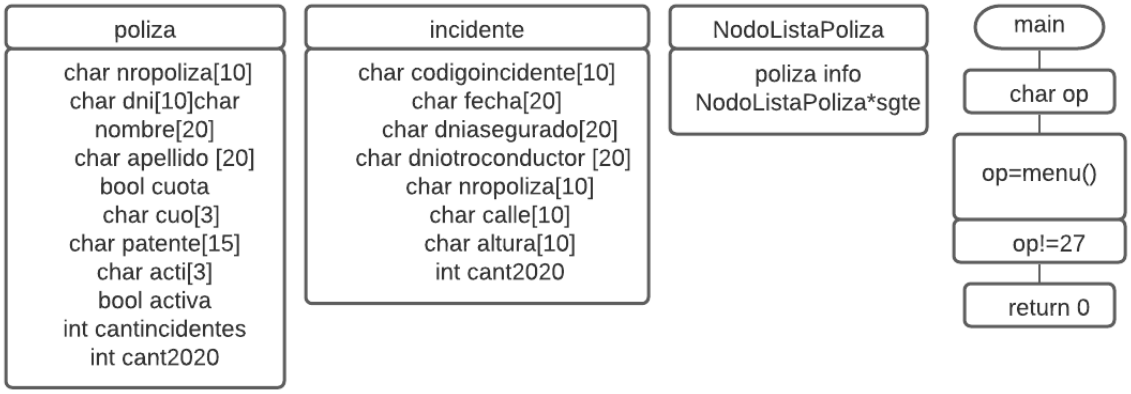
El procedimiento *guardarPolizasArchivo* recibe por parámetro la lista de las pólizas.
Se abre el archivo "Asegurados.BAK" en modo escritura y con la ayuda de un nodo auxiliar recorre la lista y mientras ésta no esté vacía escribe en el archivo la información de la lista.

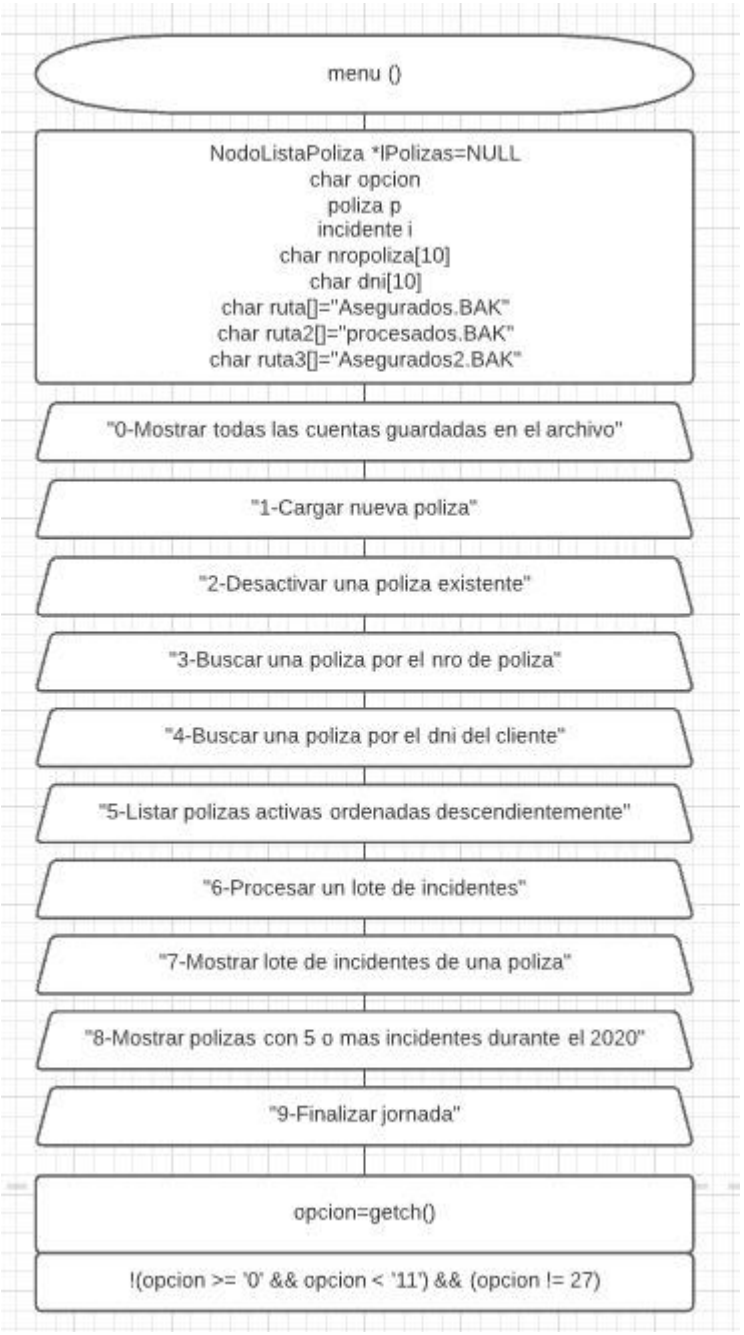

ListaPolizaInsertarSiguiente (NodoListaPoliza *&n, poliza x)
NodoListaPoliza *nuevo= new NodoListaPoliza()
nuevo->info=x
n!=NULL
nuevo->sgte= n->sgte
n->sgte = nuevo
nuevo->sgte= n
n= nuevo
return

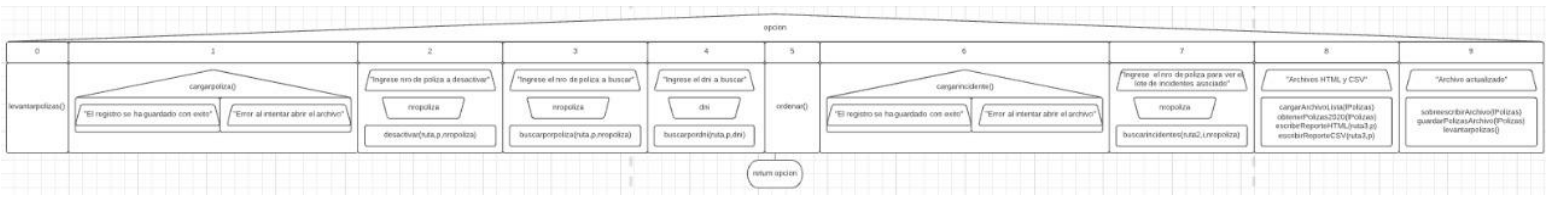

guardarPolizasArchivo(NodoListaPoliza *lp)
FILE *arch_polizas
NodoListaPoliza *auxPol=NULL
arch_polizas= fopen("Asegurados.BAK","wb")
auxPol= lp
auxPol
fwrite(&(auxPol->info),
sizeof(poliza),1,arch_polizas)
auxPol=
ListaPolizaObtenerSiguiente(auxPol)
fclose(arch_polizas)
"ERROR AL INTENTAR GRABAR ARCHIVO DE POLIZAS"
return

La función empleada en el procedimiento anterior es *ListaPolizaObtenerSiguiente,* que recibe como parámetro un nodo y devuelve el siguiente si es que éste no está vació, de lo contrario retornará null.









(diagrama del menú adjuntado aparte)

División de tareas

Algoritmos del programa

Teresin: encargado de los algoritmos de "cargarpoliza", "cargarincidente", "buscarpordni", "buscarporpoliza", "buscarincidente".

Castillo: encargada de los algoritmos de "guardarPolizasArchivo", "sobreescribirArchivo", "ListaPolizaObtenerSiguiente", "ListaPolizaInsertarSiguiente", "crearhtml".

Paredes: encargada de los algoritmos de "levantarpoliza", "levantarincidentes", "actualizar", "desactivar","sumar2020", "ordenar", "burbuja", "crearhtml".

Diagramas del programa

Castillo: encargada de los algoritmos de "guardarPolizasArchivo", "sobreescribirArchivo", "ListaPolizaObtenerSiguiente", "ListaPolizaInsertarSiguiente", "menu"

Teresin: encargado de los algoritmos de "cargarincidente", "sumar2020", "crearhtml", "ordenar".

Paredes: encargada de los algoritmos de "cargarpoliza", "buscarpordni", "buscarporpoliza", "buscarincidente", "levantarpoliza", "levantarincidente", "actualizar", "desactivar", "burbuja".