Logotipo, nombre de la empresa

Descripción generada automáticamente

Materia: Algoritmos y estructura de datos – K1031 2021

Informe de trabajo practico integrador N° 2

Profesor: Damián Méndez

Grupo:

Diego Tola [dtolaquispe@frba.utn.edu.ar](mailto:dtolaquispe@frba.utn.edu.ar)

Kevin Quiñones [kquinones@frba.utn.edu.ar](mailto:kquinones@frba.utn.edu.ar)

FECHA DE PRESENTACIÓN: 28/10/2021 FIRMA PROFESOR:

FECHA DE DEVOLUCIÓN: CALIFICACIÓN:

**División de tareas**

Para poder dividir las tareas primero se planteó cuáles son las cualidades de cada integrante del grupo y de esa forma poder asignarle la tarea que mejor pueda desarrollar tratando de evitar que el trabajo a realizar por cada integrante no suponga una carga o genere disgusto en el mismo y eso repercuta en el equipo. Dicho esto, se concluyó todos los integrantes se encontraban cómodos realizando cualquiera de los deberes que se le asignara por ello se dividió el trabajo de la siguiente forma. A Diego Tola se le asignaron los puntos 6,7,8,9,10 y Kevin Quiñones los restantes.

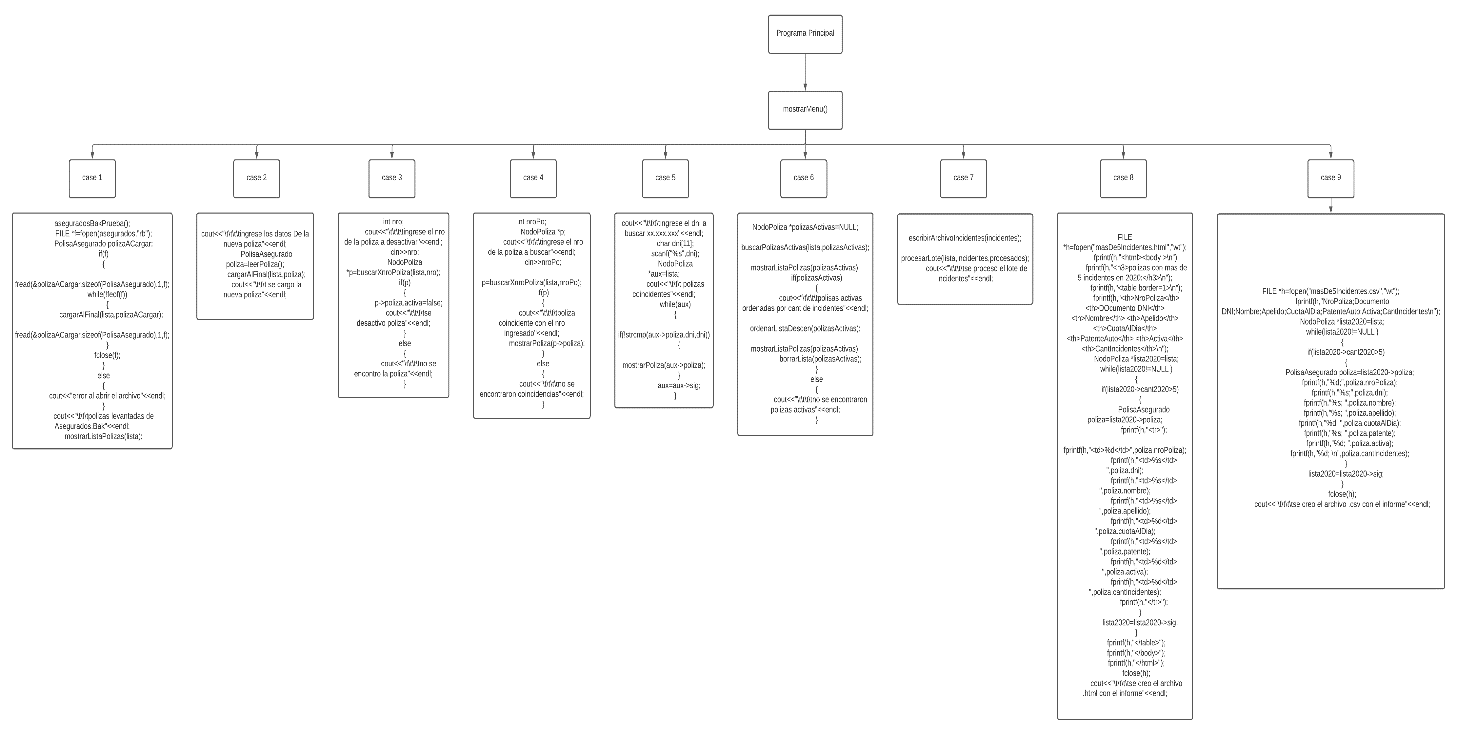
**Resolución de los puntos del trabajo**

Observación: no se hace énfasis en algunos subprogramas del tipo *cargarAlfinal*, *obtenerUltimonodo*, *longLista* y todas las que forman farte del manejo de una *listaSE* ya que se consideran un tanto triviales para la resolución de los problemas.

1. Para este punto se aplicó la primera resolución planteada. Esta solución consiste en desarrollar la funcionalidad pedida dentro del MAIN ya que no se usa en otras ocasiones y es relativamente pequeña, por lo tanto, no amerita un subprograma. Como se puede ver en el código se abre el archivo y a medida que se lee, se inserta en una *listaSE* porque no se sabe la cantidad de registros que puede llegar a tener el archivo.
2. Para este ítem en principio se pensó hacer todo en un subprograma y llamar a la misma dentro del ***MAIN***, pero al avanzar en el trabajo se vio que no era necesario y se redefinió dentro del programa principal haciendo llamada a *leerPoliza* y *cargarPoliza* .
3. Este ejercicio se dividió en dos partes, en primera instancia se busca en la lista de pólizas la que se desea desactivar mediante la invocación del subprograma *buscarXnroPoliza*, luego una vez encontrada, se modifica el valor de la misma.  
   La función invocada, como lo indica su nombre, no hace más que devolver la póliza que se le requiere por parámetro. Se desarrollo como subprograma porque se pudo detectar que esa funcionalidad sería necesaria para resolver otras tareas.
4. Aquí en principio se pensó en resolver las dos funcionalidades de manera idéntica, pero luego leyendo mejor las consignas esto se volvió un caos.  
   La solución que se adopto fue poner en un ítem del MENU la funcionalidad *buscarXnroPoliza* para satisfacer uno de los pedidos, y en otro ítem del MENU se desarrolló el código correspondiente al segundo pedido y de esa manera resolver el problema en su totalidad.
5. Teniendo en cuenta los recursos provistos por en el aula del curso, se hizo uso de la funcionalidad para ordenar una *listaSE* pero adaptándola a lo que el caso requería  
   Se dividió la consigna en dos partes, la primera consiste en armar una *listaSE* con todas las pólizas activas y la segunda, en caso de poder construir dicha lista, es ordenarla haciendo uso del subprograma mencionado.
6. Debido a que en la consigna no queda muy claro, si los incidentes se ingresan por solo por teclado o si son leídos directamente desde un archivo, por ello se pide que el usuario ingrese por teclado los incidentes del lote uno por uno y estos luego son almacenados en un archivo de incidentes. Lo primero que se pensó fue que por cada registro leído desde el archivo de incidentes se escriba *procesados.BAK* previa actualización de la lista pólizas nuevamente aquí fue necesario *buscarXnroPoliza* confirmando la correcta decisión de llevarla implementada como subprograma. También se debe aclarar que en ningún caso se vio la necesidad de guardar los incidentes en alguna estructura como una *listasSE*.
7. Se aplico la única solución que invadió al equipo. Esta consiste en poner una variable en la estructura de *NodoPoliza*, la cual cuente solo la cantidad de incidentes que tuvo cada póliza durante el 2020, siendo actualizado este contador cada vez que se procesa un incidente. También se requirió el uso de la función *compararAnio* para obtener el año del incidente y poder compararlo. Teniendo prácticamente resuelto el problema, se muestran en el informe las pólizas según el valor al que llego el contador de incidentes del 2020.
8. Este ejercicio fue solucionado del mismo modo que el anterior habiendo hecho las correspondientes modificaciones para mostrar el informe.
9. Al ser tan simple no se presentaron diferentes hipótesis más que la aplicada como solución. Cabe marcar que una vez que se finaliza la jornada se borran las *listaSE* que contenían las pólizas liberando de este modo el espacio reservado al inicio.
10. Para poder probar el programa se crearon dos subprogramas *loteIncidentesPrueba()* y *aseguradosBakPrueba()* que cargan archivos de polizas e incidentes con instancias que se determinan en los mismos. Adicional a eso el usuario puede ingresar, si así lo desea, los incidentes uno por uno, sin embargo el archivo de pólizas se considera que existe siempre.

**Diagrama de Bloques**

Debido a la poca legibilidad que presenta el diagrama en este formato se adjunta el enlace donde se puede ver claramente: <https://lucid.app/documents/view/71d33eaa-ced0-4a4d-987b-3f30718ae891>

****

**Diagramas de Lindsay**

Se presentan los diagramas de los subprogramas más relevantes para resolver los problemas.

