Programación de Computadores

Primera Entrega Proyecto Final

Profesor: Edgar Ruiz

Nicolás Barragán Sánchez

John Sebastián Rubio

Pontificia Universidad Javeriana

2020-1

1. **Análisis del problema.**

Las empresas hoy en día necesitan herramientas para tener informes detallados de la situación financiera de la empresa y poder tomar decisiones acertadas para mejorar la situación financiera de la compañía. Estos softwares pueden ser genéricos o hechos a la medida dependiendo de las necesidades de la empresa.

La compañía Súper Cálculo S.A desea reducir sus costos de producción dado que las utilidades trimestrales no son suficientes. También desean detectar qué procesos se deben administrar. El departamento financiero no ha logrado cumplir esta meta dado que siempre les queda faltando algún dato.

Por esto, el programa buscará realizar el procesamiento de datos en hojas de cálculo para no perder información a lo largo del proceso y finalmente generar reportes automáticos a cada uno de los usuarios de la información financiera seleccionados, todos bajo un formato establecido.

1. **Funcionalidad.**

El programa recibirá una o varias hojas de cálculo con los datos a procesar. Estas se encontrarán en un archivo que será leído desde el disco duro, a partir de estas, se realizarán los cálculos necesarios para finalmente tener las hojas de cálculo mencionadas con los datos totalmente procesados.

Una vez todos los datos de las hojas hayan sido procesados se generará un nuevo archivo con la información. De este nuevo archivo se generará un reporte automático que informará a uno o varios destinatarios la situación financiera de la empresa. Estos reportes tendrán el nombre del destinatario y se generará uno para cada uno. Los destinatarios serán entregados al programa por medio de un archivo.

***Los siguientes ejemplos muestran el formato de cómo quedará el archivo de texto, la consola solo representa ese formato.***

Si el archivo tiene solo una hoja de cálculo, el reporte se verá de la siguiente manera:

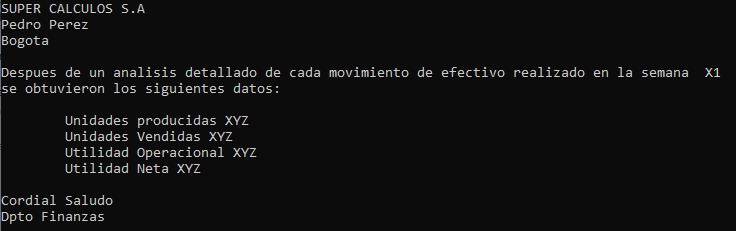


Figura 1.

Si el archivo contiene más de una hoja de cálculo, dentro de un mismo archivo por usuario se mostrará el reporte para cada una, de la siguiente manera:

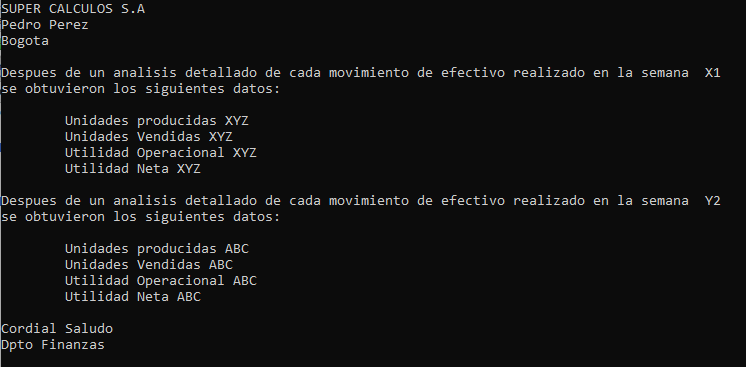


Figura 2.

Este continuará mostrando los reportes para cada hoja de cálculo por semana.

(En el ejemplo figura 2. Semana: X1, Y2…)

Finalmente, los reportes generados se almacenarán en la misma ubicación del disco duro de donde fue leída la información inicialmente.

El programa tendrá un pequeño menú dónde únicamente se dará inicio al procesamiento de datos del archivo. Se verá así:

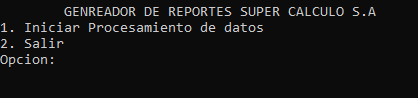


Figura 3.

Al finalizar se mostrará el resultado en pantalla y se dará la opción (si/no) de generar el reporte para cada uno de los usuarios. Se verá así:

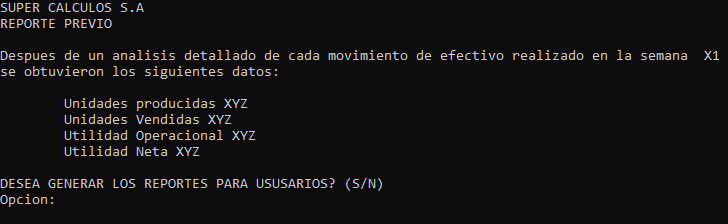


Figura 4.

Como adicional, al iniciar el procesamiento de datos el nombre del archivo será ingresado a través de la consola, de esta manera el programa será más amigable con la persona que lo usa al no tener que relacionarse con el código para colocar el nombre del archivo que se desea abrir.

1. **Previsión de riesgos y/o problemas que se puedan presentar.**

* Si la cantidad de reportes a generar exceden con la capacidad de memoria del dispositivo estos no se generarían por completo.
* El que se ingrese un dato no valido en la hoja de cálculo representará el mal funcionamiento del programa, pues este podrá interferir con cálculos y con el tipo de dato que inicialmente iría referenciado.
* Los valores decimales no son aceptados por el programa, este únicamente trabaja con enteros, así se especificó por parte de **SUPER CALCULO S.A.**
* Si el archivo a leer no existe o la dirección y/o nombre del archivo es diferente a la que se ingrese este no podrá acceder a la información.
* Si el archivo a leer no es del tipo .txt el programa no será capaz de abrirlo.
* Si la información dentro del .txt no está en el orden especificado por **SUPER CALCULO S.A** el programa posiblemente será deficiente o en su defecto no funcionará.
* La letra ‘Ñ’ no será válida en el programa, pues hará que este se comporte de manera no deseada.

1. **Estructuras de datos**

***Nombre***: hojaN – Hoja Nueva

* **numHojas**: Almacena el total de hojas de cálculo que trae el archivo, este dato será igual para todas las hojas de cálculo del mismo archivo. Se guarda para facilitar el procesamiento de datos al final.
* **filas**: Almacena la cantidad de filas de la hoja de cálculo.
* **Columnas**: Almacena la cantidad de columnas de la hoja de cálculo.
* \*\***hojaN**: Matriz dinámica que almacena la hoja de cálculo.

***Finalidad:***  Esta almacenará las hojas de cálculo recién leídas del archivo, es decir, con las formulas sin calcular.

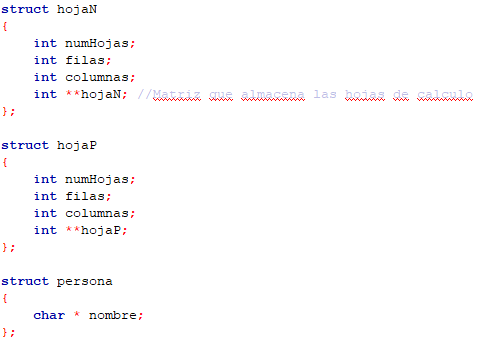
****

Figura 5.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Nombre;*** hojaP – Hoja Procesada

* **numHojas**: Almacena el total de hojas de cálculo que trae el archivo, este dato será igual para todas las hojas de cálculo del mismo archivo. Se guarda para facilitar el procesamiento de datos al final.
* **filas**: Almacena la cantidad de filas de la hoja de cálculo.
* **Columnas**: Almacena la cantidad de columnas de la hoja de cálculo.
* \*\***hojaN**: Matriz dinámica que almacena la hoja de cálculo.

***Finalidad;*** Esta almacenará las hojas de cálculo después de haber procesado todas las formulas exitosamente.

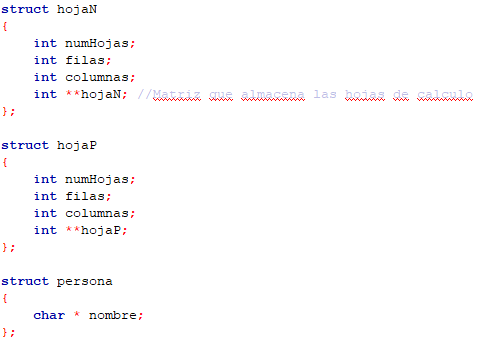


Figura 6.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(***Son estructuras iguales pero su finalidad es facilitar el desarrollo del programa y la lectura del código pudiendo distinguir donde se está manejando las no procesadas y las ya procesadas***)

***Nombre:*** persona.

* **nombre:** Almacena el nombre del destinatario.

***Finalidad:*** Es una estructura a simple vista innecesaria, pero facilitará el trabajo de manejar los nombres de los destinatarios a la hora de escribirlos en un archivo de texto para generar los reportes.

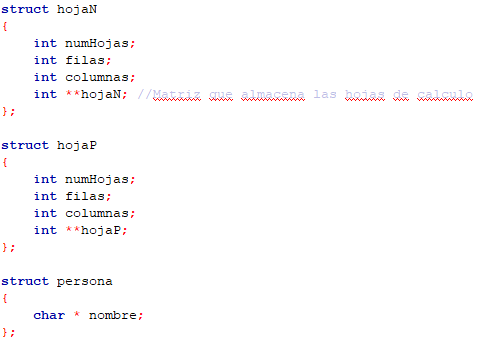


Figura 7.

***Nombre:*** Reporte

* **Semana:** Valor de la hoja de cálculo basado en un formato que corresponde a la semana.
* **UnidadesProducidas:** Valor de la hoja de cálculo basado en un formato que corresponde a las unidades Producidas.
* **UnidadesVendidas:** Valor de la hoja de cálculo basado en un formato que corresponde a las unidade vendidas.
* **UtilidadOperacional:** Valor de la hoja de cálculo basado en un formato que corresponde a la utilidad operacional.
* **UtilidadNeta:** Valor de la hoja de cálculo basado en un formato que corresponde a la utilidad neta.

***Finalidad:*** Almacenar los datos pertinentes de cada hoja de cálculo necesarios para poder realizar el llenado del formato para finalmente generar los reportes para cada uno de los destinatarios del archivo.

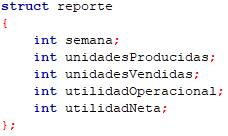


Figura 8.

1. **Prototipo de funciones. (Documentadas)**

* **void proceso(struct hojaN \*hojasNuevas, struct hojaP \*hojasProcesadas)**

**Nombre**: Proceso.

**Entradas**: Arreglo de tipo hojaN, arreglo de tipo hojaP.

**Salidas:** Arreglo de tipo hojaN lleno, arreglo de tipo hojaP lleno. Nuevos archivos de texto para cada uno de los destinatarios de la información financiera.

**Descripción**: La función proceso es la función madre que llevará toda la lógica de negocio. Esta se encargará de llevar el curso del proceso y de llamar a las demás funciones pertinentes para conseguir el resultado esperado.

**Funciones relacionadas:** buscarArch, leeryProcesarHojas, imprimirReporte, generarReportes, calculasFormula.

* **void buscarArch(char \*nombre);**

**Nombre:** buscarArch o Buscar Archivo.

**Entradas:** un array con el nombre del archivo que se desea buscar y abrir.

**Salidas:** Ninguna.

**Descripción:** La función pide al usuario que ingrese por teclado el nombre del archivo que se desea buscar, en seguida lo busca y lo abre. Al momento de abrirlo hace el llamado a leeryProcesarHojas.

**Funciones relacionadas:** leeryProcesarHojas.

* **void leeryProcesarHojas(struct hojaN \*hojasNuevas, struct hojaP \*hojasProcesadas);**

**Nombre:**  Leer y Procesar Hojas.

**Entradas:** Arreglo de tipo hojaN, arreglo de tipo hojaP. (Recibidos como parámetros por referencia para que puedan ser modificados)

**Salidas:** Arreglo de tipo hojaN lleno, arreglo de tipo hojaP lleno.

**Descripción:** Lee los datos del archivo y los almacena directamente en el array hojasNuevas de tipo hojaN; Una vez guardados procede a calcular las fórmulas que existan dentro de la(s) hoja(s) (**Llamado a calcularFormula()**). Cada vez que una hoja es calculada es almacenada en el array hojasProcesadas. Cuando la única o ultima hoja de cálculo fue procesada retorna la secuencia a la función proceso().

**Funciones relacionadas:** calcularFormula();

* **void imprimirReporte();**

**Nombre:** Imprimir Reporte

**Entradas:** Ninguna

**Salidas:** Ninguna

**Descripción:** Basado en un formato, imprime en pantalla un demo de cómo se vería el reporte para los destinatarios.

**Funciones relacionadas:** Ninguna

* **void generarReportes(struct hojaP hojasProcesadas);**

**Nombre:** Generar Reportes

**Entradas:** Arreglo hojas Procesadas de tipo hojaP

**Salidas:** Archivos de texto para cada uno de los destinatarios, este archivo lleva el nombre del destinatario.

**Descripción:** Toma las hojas de cálculo y obtiene cada uno de los valores necesarios para llenar el formato de reporte al destinatario, a medida que va obteniendo datos va llenando una colección de tipo Reporte, para finalmente recorrer esta colección n veces dependiendo de la cantidad de destinatarios y generar los reportes para cada uno de ellos.

**Funciones relacionadas:** Ninguna.

* **int calcularFormula(struct hojaN &aProcesar, char\*formula);**

**Nombre:** Calcular Formulas.

**Entradas:** Estructura del tipo hojaN, aProcesar y la formula encontrada.

**Salidas:** un valor entero.

**Descripción:** Cada vez que es llamada recibe la hoja de cálculo sobre la que se está trabajando y la formula a calcular. La función calcula cada uno de los valores hasta que ya no existan más signos de suma en la cadena. En este momento realiza el ultimo calculo y retorna el valor esperado. Si la formula requiere la solución de otra fórmula para poder ser calculada, la función se llamará a si misma para conseguirlo.

**Funciones relacionadas:** calcularFormula().

1. **Resultados esperados.**

El programa realizará los cálculos de manera acertada y los colocará a manera de reporte, uno para cada hoja de cálculo. Luego de validar que estos sean correctos el programa generará reportes de todas las hojas de cálculo para cada uno de los usuarios de la información, este será un archivo individual con el nombre del destinatario.

El programa resolverá los problemas de cálculos financieros que tiene hoy en día **SUPER CALCULO S.A,** la solución estará hecha a la medida y con cada una de las especificaciones dadas. Además, tendrá las funcionalidades del menú que harán del uso de la herramienta algo más sencillo, incluyendo el mostrar el reporte al departamento de finanzas antes de generar uno para cada uno de los destinatarios.

1. **Pruebas a realizar al programa.**

El programa se verá sometido a diferentes pruebas para ver cómo se comportará frente a ciertas situaciones. Por ejemplo:

* Validar antes de procesar que el archivo no venga vacío, y si lo está, genere un mensaje al usuario.
* El contenido del archivo no está en el formato definido, en este caso se dará un mensaje de error en el archivo.
* En caso de que no existan usuarios en el archivo de texto que contiene los destinatarios genere un mensaje.
* Si alguna de las formulas contiene un dato no valido genere un mensaje de error antes de generar un archivo defectuoso. Por ejemplo, una hoja de cálculo de 2x2 que tenga una fórmula: H9 + A1. H9 es una posición no válida para esta hoja de cálculo.
* Los datos ingresados en el menú sean incorrectos, por ejemplo, cuando se da la opción para ingresar sí o no (S/N) ingresen una J.
* Corroborar el funcionamiento del programa en un escenario donde todos los datos y archivos son correctos.