MANUAL TECNICO

HOJA DE CALCULO EN EXCEL

INDICE

Arquitectura del software

La estructura general de nuestro proyecto se divide en 3 campos

Los controladores:

Sirve para accionar el funcionamiento de la tabla hash y nuestra Hoja de Cálculo de Excel

Por lo que contienen tipos de validaciones para el correcto funcionamiento en caso agreguemos un 0 o no se pueda realizar alguna expresión, también tiene funcionamiento para realizar la creación de las hojas en este caso por defecto es 10*10,

private void crearNuevaHoja Al crear la nueva hoja tendremos filas y columnas el cual se ira enlazando con las hojas creadas y esta conectado con nuestro libro de excel (el conjunto de todo).

Tambien en la clase private void procesarFormula tendremos nuestro funcionamiento básico de las formulas para que pueda convertir la formula a mayúsculas con sus respectivas validaciones.

private String evaluarFormula realiza una validación para ver si es correcto el formato que ingreso el usuario para poder realizar los cálculos de las fórmulas por lo que si es necesario modificar para poder realizar distintas combinaciones para hacer distinto el programa si en caso es necesario.

private String evaluarFuncion analiza una fórmula para identificar si está en el formato correcto FUNCION (parametros). Si lo está, extrae la función y los parámetros, y luego llama a un método específico para evaluar la función. Si la función no es reconocida o la fórmula no coincide con el formato esperado, retorna un mensaje de error.

private String evaluarSUMA toma una lista de parámetros separados por comas, transforma cada parámetro en una expresión evaluable, evalúa cada expresión, y suma los resultados. Finalmente, retorna la suma como una cadena. Este método permite que la función SUMA en una fórmula de hoja de cálculo sume correctamente los valores especificados en los parámetros.

private String transformarFormula toma una fórmula que contiene referencias de celdas y las reemplaza con los valores actuales de esas celdas, ya sea en la misma hoja o en otra hoja del libro. Este método permite que las fórmulas con referencias de celdas sean evaluadas correctamente utilizando los valores de las celdas referenciadas.

private double evaluarExpresion evalúa una expresión matemática en formato posfijo. Utiliza una pila para almacenar operandos y el resultado intermedio de las

operaciones. Itera sobre cada token en la expresión posfija, realizando operaciones según el tipo de token (número u operador). Finalmente, retorna el resultado final de la evaluación de la expresión.

TABLAHASH

Constructor y Método inicializar:

El constructor ControladorHashTable recibe una instancia de HojaCalculo y la asigna al campo vista.

En el método inicializar, se agrega un ActionListener al menú de la tabla hash en la interfaz gráfica. Cuando se selecciona este menú, se llama al método solicitarDatosIniciales.

Método solicitarDatosIniciales:

Este método muestra dos cuadros de diálogo (JOptionPane) para solicitar al usuario el tamaño de la tabla hash y la cantidad de claves que desea ingresar.

Verifica que los valores ingresados sean válidos (números positivos) y crea una nueva instancia de TablaHash con el tamaño especificado.

Luego, llama al método mostrarTablaHash.

Método mostrarTablaHash:

Este método crea una nueva ventana (JFrame) para mostrar la tabla hash.

Define un panel con un diseño de cuadrícula (GridLayout) que contiene campos de texto para ingresar las claves y los valores de la tabla hash.

Crea un botón "Insertar" que, al hacer clic, llama al método insertarClavesYValores para ingresar las claves y los valores en la tabla hash.

Método insertarClavesYValores:

Este método recorre los campos de texto para las claves y los valores.

Para cada par de clave-valor no vacío, intenta convertir la clave a un entero y la inserta en la tabla hash utilizando el método insertar de la clase TablaHash.

Si hay algún error al convertir la clave a entero, muestra un mensaje de error.

este controlador maneja la lógica de interacción entre la interfaz gráfica y la tabla hash. Solicita datos al usuario, muestra la tabla hash en una ventana separada y permite insertar claves y valores en la tabla hash a través de la interfaz gráfica.

Modelos:

Atributos de la Celda:
fila: Representa el índice de fila de la celda.

columna: Representa el índice de columna de la celda.

valor: Almacena el valor contenido en la celda, inicializado como una cadena vacía "".

arriba, abajo, izquierda, derecha: Son referencias a las celdas adyacentes (arriba, abajo, izquierda, derecha) en la estructura de datos.

Constructor:
El constructor Celda(int fila, int columna) inicializa los atributos fila y columna de la celda con los valores proporcionados al crear una nueva instancia de Celda. El atributo valor se inicializa como una cadena vacía por defecto.

Métodos Getters y Setters:
Los métodos get y set se utilizan para acceder y modificar los valores de los atributos de la celda (fila, columna, valor, arriba, abajo, izquierda, derecha).

Por ejemplo:

getFila() y setFila(int fila) permiten obtener y establecer el valor de la fila de la celda.

getValor() y setValor(String valor) permiten obtener y establecer el valor contenido en la celda.

Estos métodos get y set son comunes en programación orientada a objetos y facilitan el encapsulamiento de los atributos de la clase, permitiendo un acceso controlado a los mismos desde otras clases.

la clase Celda proporciona una estructura básica para representar celdas en una cuadrícula o matriz, con la capacidad de almacenar un valor y mantener referencias a las celdas adyacentes.

Atributos de la Clase Hoja:
celdas: Una matriz de objetos Celda que representa las celdas de la hoja.

filas, columnas: Números enteros que representan la cantidad de filas y columnas en la hoja.

nombre: El nombre de la hoja de cálculo.

Constructor Hoja(int filas, int columnas):

Crea una nueva instancia de Hoja con una cantidad específica de filas y columnas.

Inicializa la matriz de celdas (celdas) llamando a los métodos inicializarCeldas() y conectarCeldas().

Métodos Getters y Setters:
getCeldas(), setCeldas(Celda[][] celdas): Para acceder y modificar la matriz de celdas.

getFilas(), setFilas(int filas): Para acceder y modificar el número de filas.

getColumnas(), setColumnas(int columnas): Para acceder y modificar el número de columnas.

getNombre(), setNombre(String nombre): Para acceder y modificar el nombre de la hoja.

Métodos para Inicializar y Conectar Celdas:

inicializarCeldas(): Crea y asigna una nueva instancia de Celda a cada posición en la matriz de celdas.

conectarCeldas(): Establece las referencias a las celdas adyacentes (arriba, abajo, izquierda, derecha) para cada celda en la matriz.

Métodos para Acceder y Modificar Celdas:

getCelda(int fila, int columna): Obtiene una celda específica en la posición (fila, columna) de la hoja.

setValorCelda(int fila, int columna, String valor): Establece el valor de una celda específica.

getValorCelda(int fila, int columna): Obtiene el valor de una celda específica.

Estos métodos permiten acceder y manipular las celdas dentro de la hoja de cálculo, proporcionando una interfaz para trabajar con la información contenida en la hoja.

la clase Hoja encapsula la lógica de una hoja de cálculo, incluyendo la gestión de celdas y sus relaciones, la inicialización de la estructura de la hoja y la manipulación de valores en las celdas.

Atributos de la Clase Libro:

hojas: Una lista enlazada (LinkedList) que almacena las hojas del libro.

Constructor Libro():

Inicializa la lista de hojas al crear una nueva instancia de Libro.

Método agregarHoja(Hoja hoja):

Agrega una hoja al final de la lista de hojas del libro.

Método Getter getHojas():

Permite obtener la lista de hojas del libro.

Método getNumeroHojas():

Retorna el número de hojas que hay en el libro, basándose en el tamaño de la lista de hojas.

Método getHoja(int i):

Permite obtener una hoja específica del libro por su índice en la lista de hojas.

Verifica que el índice proporcionado sea válido antes de intentar acceder a la hoja correspondiente.

Método getHojaPorNombre(String nombre):

Busca una hoja por su nombre dentro del libro.

Recorre la lista de hojas y compara el nombre de cada hoja con el nombre proporcionado.

Si encuentra una coincidencia, retorna esa hoja. Si no encuentra ninguna coincidencia, retorna null.

Este conjunto de métodos proporciona funcionalidades básicas para manejar hojas dentro de un libro de cálculo. Permite agregar hojas, obtener el número total de hojas, acceder a hojas específicas por su índice o por su nombre, y obtener la lista completa de hojas.

El uso de una lista enlazada (LinkedList) para almacenar las hojas permite una flexibilidad en la manipulación de las hojas, ya que se pueden agregar, eliminar y acceder a las hojas de manera eficiente en términos de rendimiento.

Package Modelo:

El código está dentro de un paquete llamado Modelo. Los paquetes en Java se utilizan para organizar y estructurar el código en módulos lógicos.

Clase NodoHash:

Esta clase representa un nodo en la tabla hash.

Contiene tres atributos:

clave: Un entero que representa la clave asociada con este nodo.

valor: Una cadena de texto (String) que representa el valor asociado con este nodo.

siguiente: Una referencia al siguiente nodo en la cadena, utilizado para manejar colisiones en la tabla hash.

Constructor NodoHash(int clave, String valor):

Este es el constructor de la clase NodoHash.

Toma dos parámetros: la clave (clave) y el valor (valor) asociados con el nodo.

Inicializa la clave y el valor del nodo con los valores proporcionados.

Inicializa la referencia al siguiente nodo (siguiente) como null, ya que al crear un nuevo nodo no hay ningún nodo siguiente en la cadena de colisiones.

La clase representa un nodo básico en una estructura de tabla hash, que es una técnica utilizada para almacenar y buscar datos de manera eficiente mediante el uso de claves asociadas a valores y el manejo de colisiones mediante listas enlazadas.

Package Modelo:
Al igual que en el código anterior, este código está dentro del paquete Modelo, que probablemente contiene las clases relacionadas con la lógica del modelo en una aplicación más grande.

Clase TablaHash:
Esta clase representa una tabla hash con encadenamiento para manejar colisiones.

Contiene dos atributos:

tabla: Un arreglo de NodoHash que almacena los pares clave-valor y las listas enlazadas para manejar colisiones.

tamano: El tamaño de la tabla hash.

Constructor TablaHash(int tamano):
El constructor inicializa la tabla hash con el tamaño especificado y crea el arreglo de NodoHash para almacenar los datos.

Método funcionHash(int clave):
Este método calcula el valor hash para una clave dada utilizando una función simple (en este caso, calculando el residuo de la división de la clave por el tamaño de la tabla).

Método insertar(int clave, String valor):
Inserta un nuevo par clave-valor en la tabla hash.

Calcula el índice utilizando la función hash y luego maneja las colisiones agregando el nuevo nodo al final de la lista enlazada correspondiente.

Método obtener(int clave):
Recupera el valor asociado con una clave dada.

Utiliza la función hash para encontrar el índice y luego busca el nodo con la clave especificada en la lista enlazada correspondiente.

Método eliminar(int clave):
Elimina el par clave-valor asociado con una clave dada de la tabla hash.

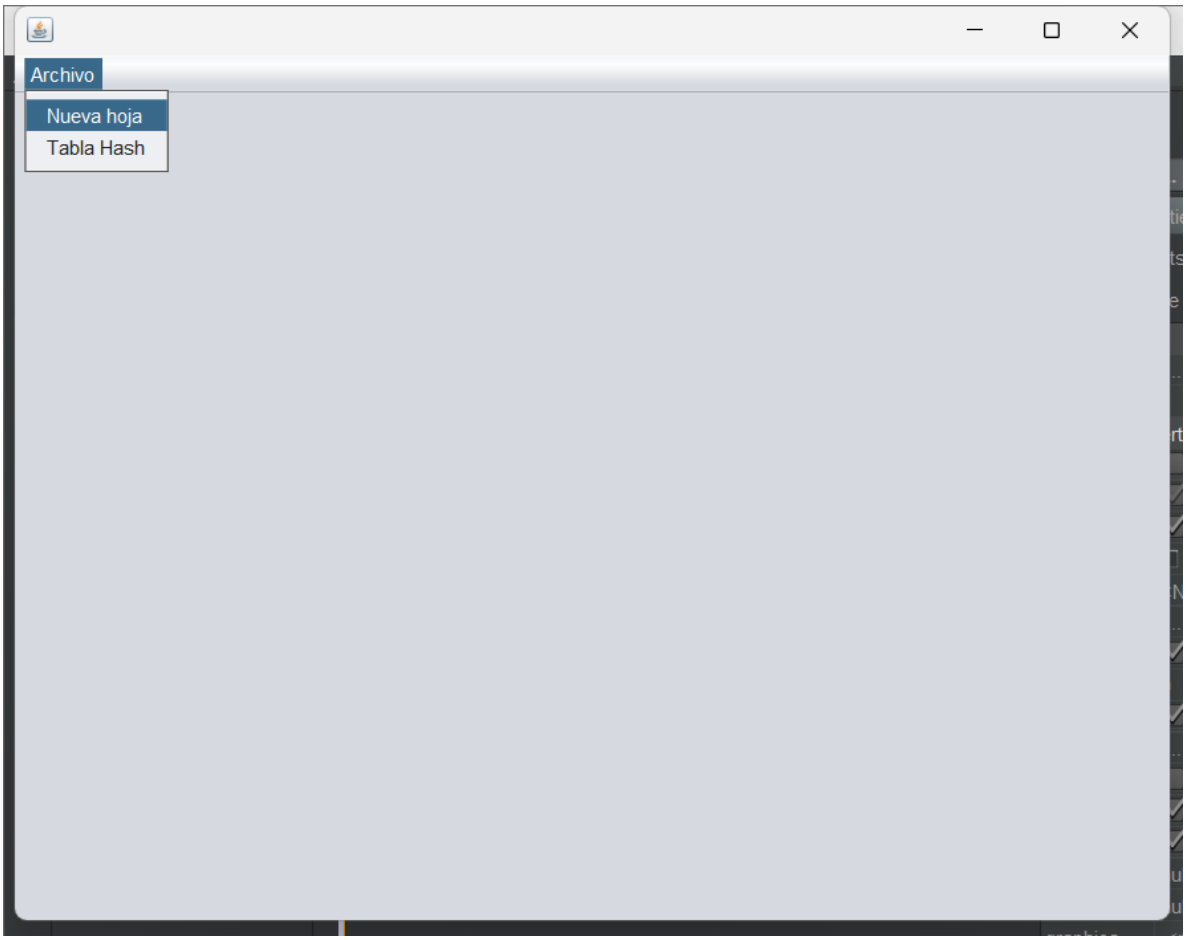
Utiliza la función hash para encontrar el índice y luego busca y elimina el nodo correspondiente en la lista enlazada.

Método mostrarTabla():
Muestra el contenido de la tabla hash imprimiendo cada índice y la lista enlazada de nodos asociada.

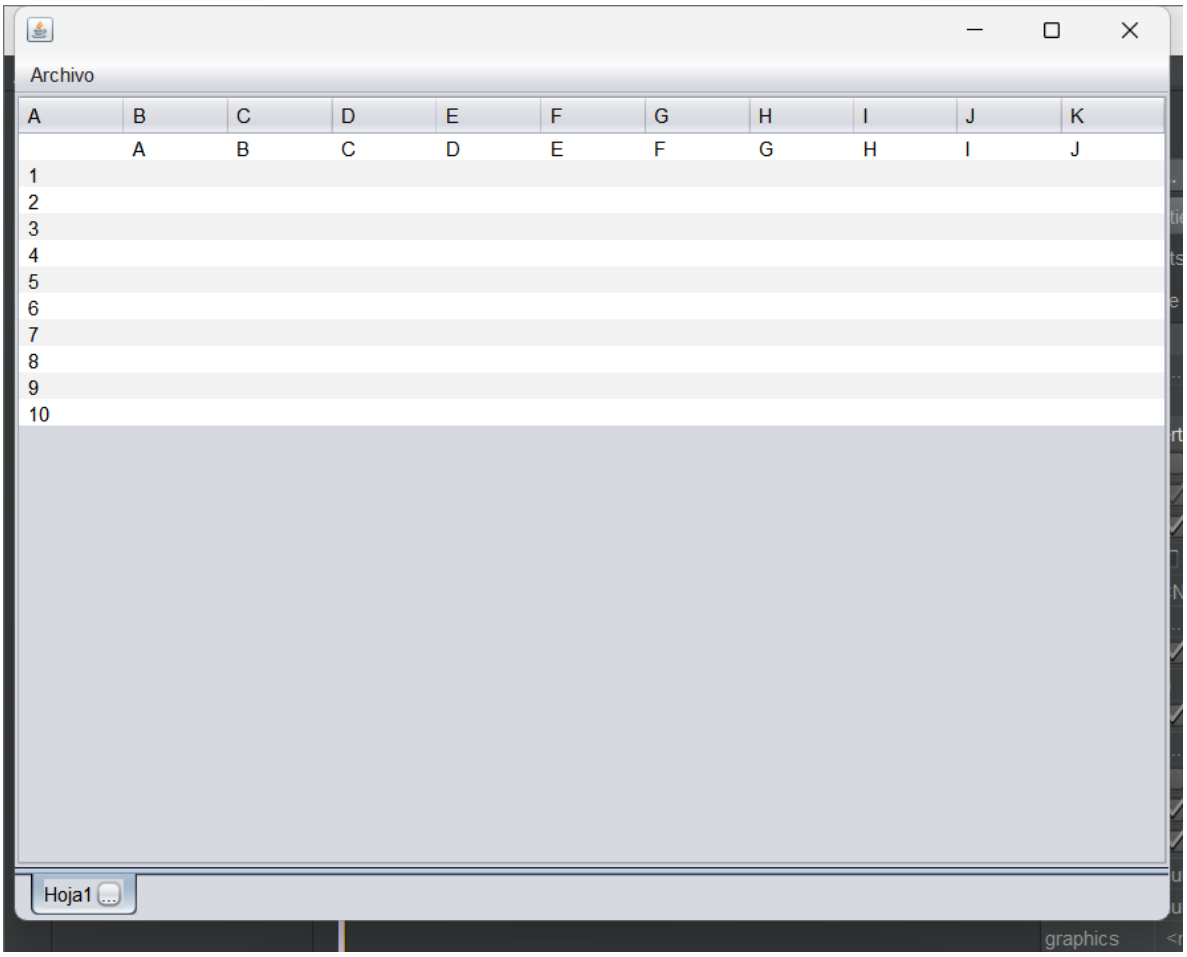
VISTA

En esta clase tendremos todo para el correcto funcionamiento del programa de manera grafica para realizar la interacción con el usuario:

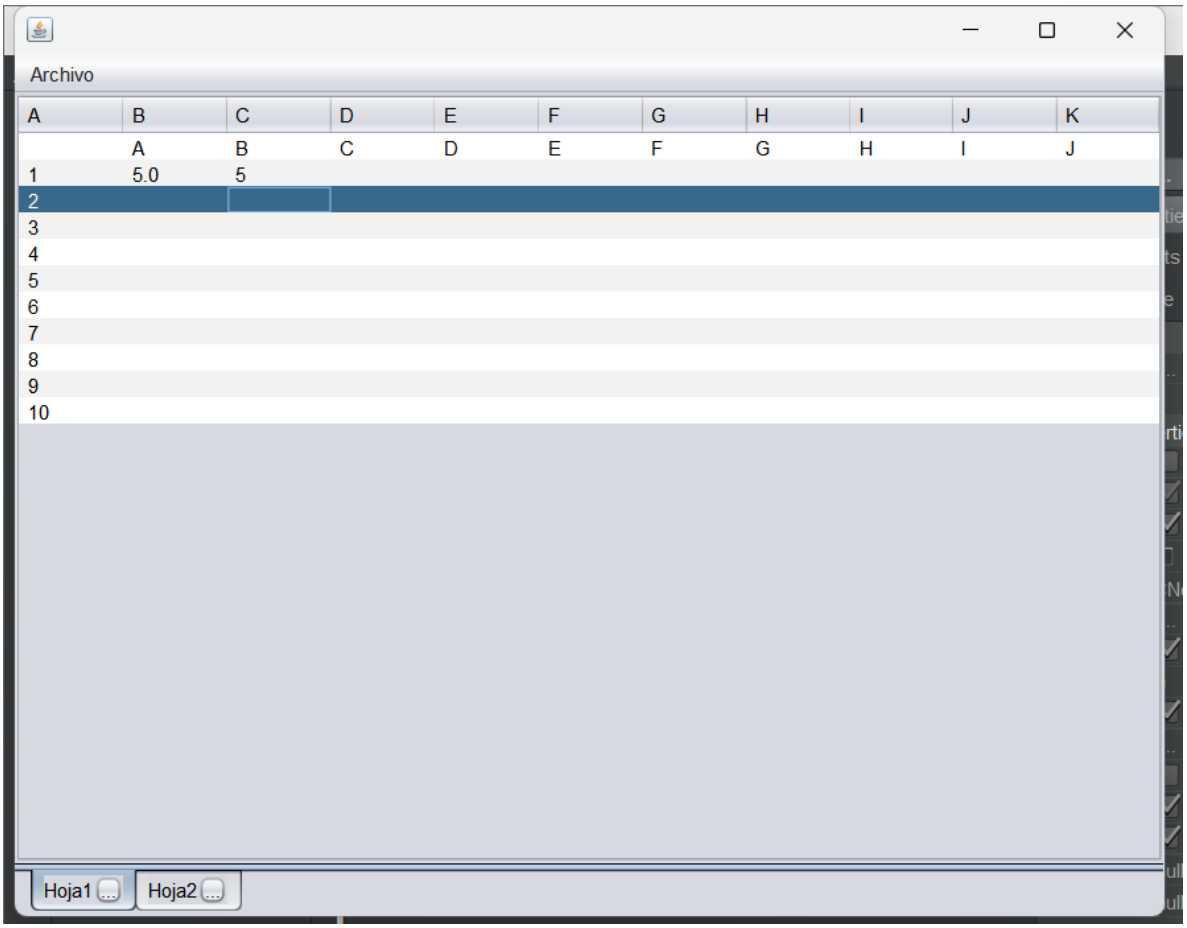
Lo principal al ejecutar el programa será lo siguiente:

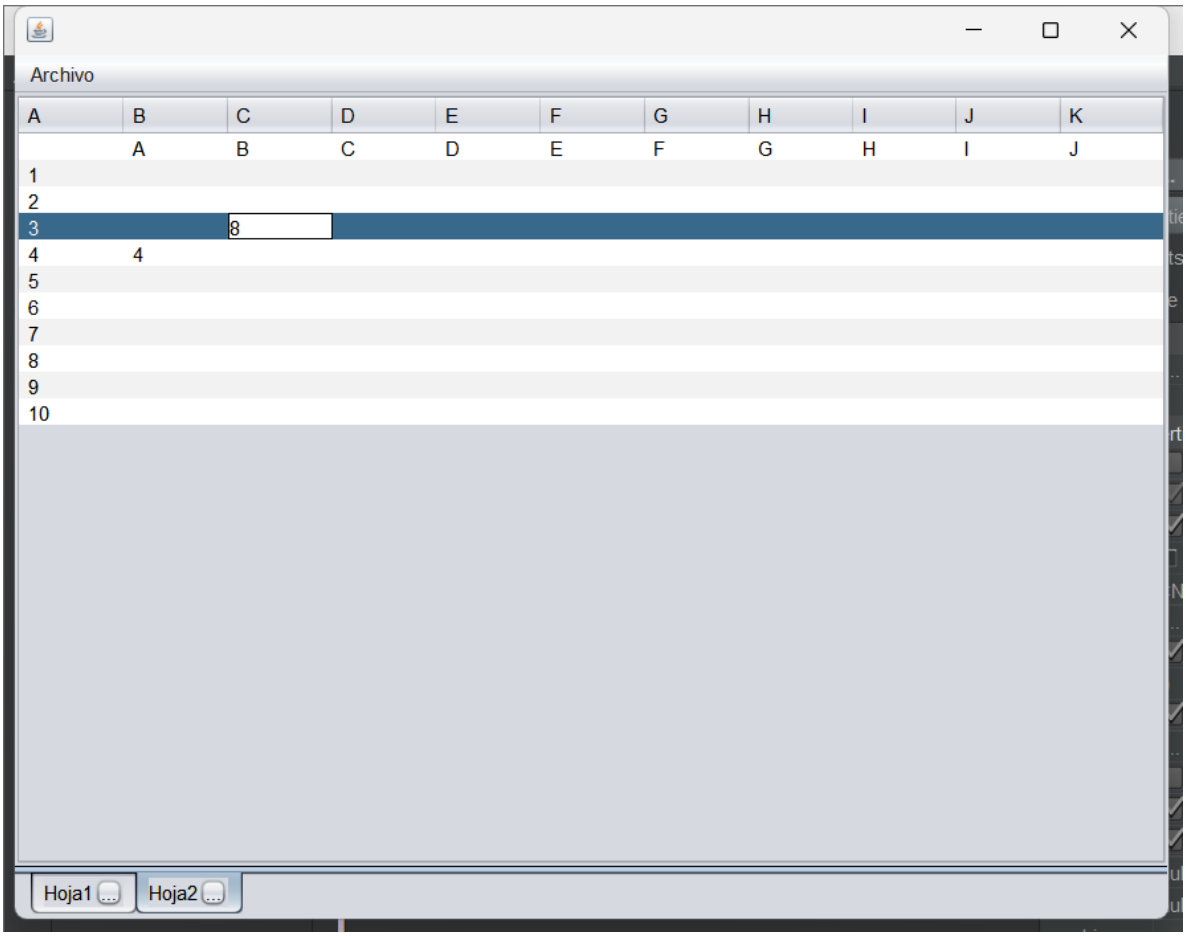


En donde podremos interactuar y seleccionar lo deseado en este caso al seleccionar Nueva Hoja podremos observar nuestro excel:



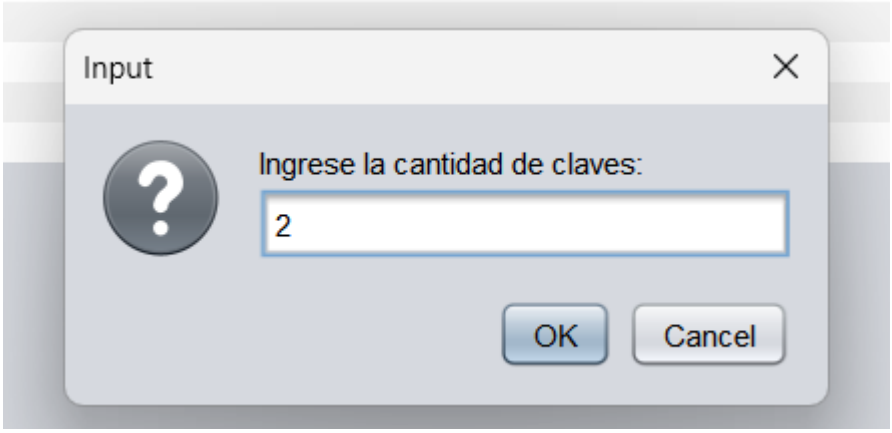
El cual es capaz de realizar varias operaciones en una celda y poder almacenar datos al crear otra hoja los datos se aguardaran de la hoja anterior, y con el funcionamiento anterior podremos observar que la nueva hoja es creada desde 0 sin números.


Archivo
A B C D E F G H I J K
A B C D E F G H I J
1 5.0 5
2
3
4
5
6
7
8
9
10
Hoja1
Hoja2

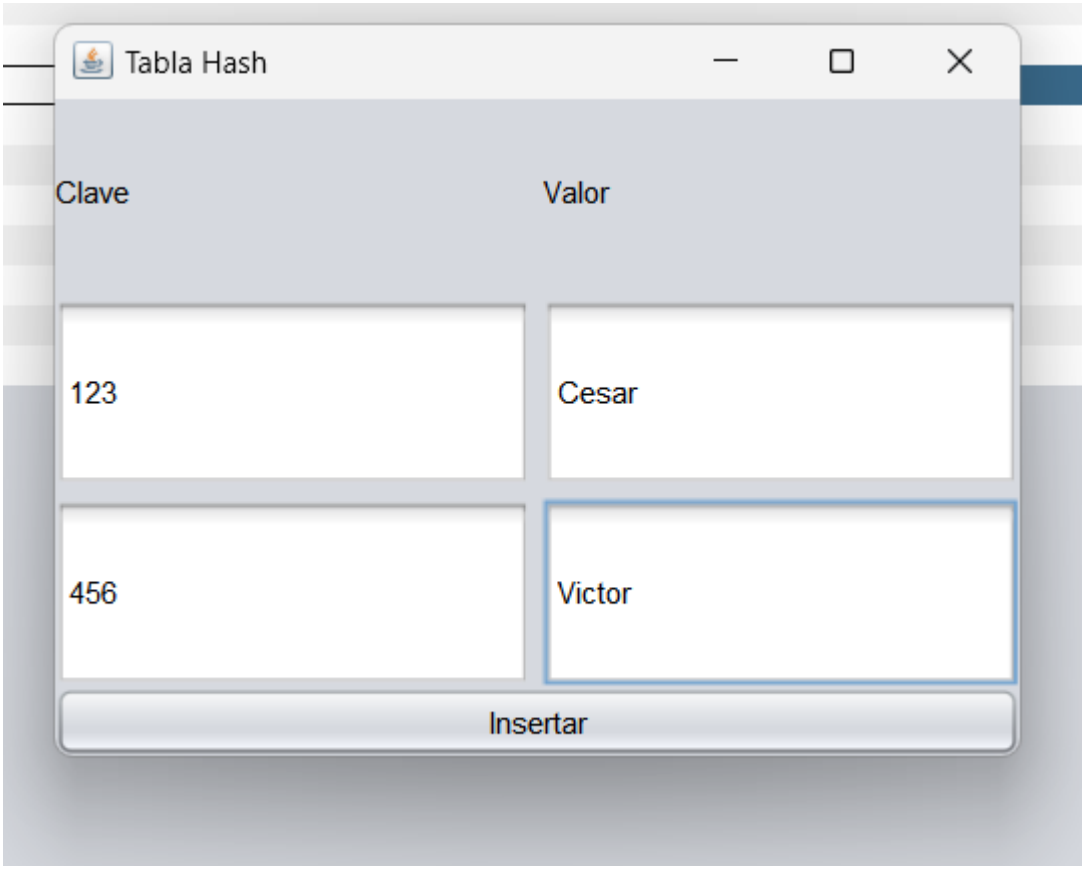


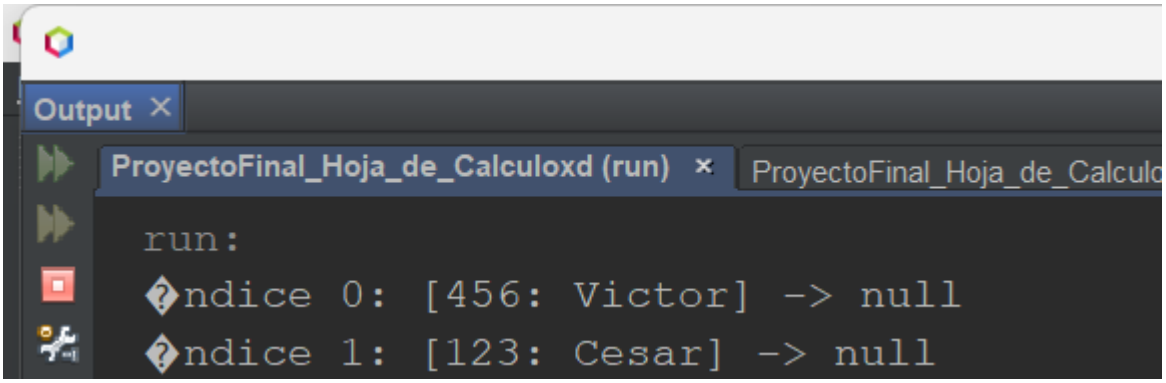
Y ese seria el funcionamiento de nuestra interfaz grafica cada button action estaría enlazado con una acción por lo que con esto podemos controlar las diferentes acciones de nuestro programa de forma grafica.

En cuanto la tabla hash lo primero nos solicitara ingresar el tamaño (N) y nuestro números de claves en este caso ambas son 2



Nos creara la tabla para realizar su funcionamiento:





Y en la consola nos vera representada nuestra tabla hash.

Especificaciones de las versiones para el Desarrollo del software:

Versión de java: jdk20 (ApacheNeatBeans V18,20,21)

Memoria utilizada: 400/420 MBS

Espacio recomendado: 700 MBS

Sistema Operativo donde se desarrolló: Windows 11 y Sonoma 14.4.1

Librerías requeridas:

```
import Modelo.TablaHash;

import Vista.HojaCalculo;

import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import Modelo.Hoja;

import Modelo.Libro;

import Vista.HojaCalculo;

import javax.swing.*;

import javax.swing.event.TableModelEvent;

import javax.swing.event.TableModelListener;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Stack;

import java.util.regex.Matcher;

import java.util.regex.Pattern;
```

Model Graphical: Java swing y MVC

Directrices de codificación: Utilizamos un correcto nombramiento de las variables explicitas para mejorar la comprensión al desarrollador y complementarnos como equipo.