UNIVERSIDAD CENTROAMERICANA “JOSÉ SIMEÓN CAÑAS”

FACULTAD DE INGENIERÍA Y ARQUITECTURA

Tarea 1: MiniExcel en consola

Asignatura:

Técnicas de Simulación

Catedrático:

Ing. Enmanuel Araujo

Integrantes:

García Arévalo, José Enrique 0009361
Hernández García, Rodrigo Anibal 00050519
Eddy Rene Barahona Hernández 00074320
Katerin Alexandra Alfaro Angel 00240620

Fecha de entrega:

27/05/2023

Este documento es una guía que pretende mostrar al usuario la estructura y flujo funcional del programa excel en consola realizado para la tarea 1 de Técnicas de Simulación en Computadoras. El programa descrito a continuación es una aplicación en consola que genera una matriz sobre la cual el usuario puede realizar operaciones como copiar, pegar, insertar y modificar valores.

Estructura de carpetas

El programa consta del archivo main.cpp, que es el punto de entrada al programa, y las carpetas controllers, services y utils, que contienen las funciones necesarias para el uso de la matriz y las operaciones que el usuario puede realizar. Las carpetas están segmentadas de acuerdo a su contenido de la siguiente manera:

Controllers

Esta carpeta contiene los archivos:

- **menu_principal_controller.hpp:** este archivo implementa un menú iterativo que le permite al usuario generar una nueva matriz para realizar operaciones, ver sus archivos guardados o salir del programa. Para este proposito se utiliza una estructura de flujo do-while y un switch que recibe el input de usuario validado como un entero
- **new_sheet_controller.hpp:** en este archivo se implementa un menú que permite al usuario utilizar las funcionalidades de una matriz creada, a traves de una estructura de control de flujo de tipo while, se ofrecen opciones de operación al usuario y se utiliza un switch que llama a las funciones que realizan las operaciones basado en la opción seleccionada por el usuario.
- **open_sheet_controller.hpp:** permite al usuario acceder a una matriz previamente guardada en un archivo .JSON, esto se logra parseando el archivo .JSON y presentándose nuevamente al usuario en la consola.

Services

La carpeta services guarda los archivos:

- **matrix_service.hpp:** contiene todas las funciones necesarias para crear, imprimir, expandir filas y columnas, así como también escribir valores en celdas.

Utils

Esta carpeta contiene archivos que guardan funciones utilizadas repetidamente en el flujo del programa, de ahí su nombre Utils (utilidades). En esta carpeta se encuentran los archivos:

- **Utils.hpp:** define funciones de validación, impresión y conversión de char a int para permitir la navegación alfanumérica.

- **JSON.hpp:** es un archivo con código desarrollado por joaquinrmi (https://github.com/joaquinrmi/JSON), diseñado para manipular archivos y objetos en formato JSON utilizando c++. Este archivo permite guardar una hoja de cálculo de nuestro programa como un archivo .JSON.

Descripción de funciones

El programa utiliza un grupo de funciones para ofrecer operaciones en las hojas de cálculo a los usuarios, la lista siguiente ofrece una descripción breve de cada función, organizada por archivo:

menu_principal_controller.hpp

- **menu_principal:** muestra un menú en la consola y permite al usuario seleccionar opciones. Dependiendo de la opción seleccionada, se llamará a otras funciones para realizar diferentes tareas. La función se repetirá hasta que el usuario seleccione la opción de salir (3).

new_sheet_controller.hpp

- **menu_matriz:** muestra una hoja de cálculo en la consola y permite al usuario interactuar con ella seleccionando diferentes opciones. Dependiendo de la opción seleccionada, se llamará a diferentes funciones para realizar diversas tareas relacionadas con la hoja de cálculo, como insertar contenido, saltar a una celda, copiar, cortar, pegar, moverse en la hoja, guardar y salir. El bucle while se repetirá hasta que el usuario seleccione la opción de salir (opción 11).
- **insertar_contenido:** permite al usuario ingresar un valor en una celda de la hoja de cálculo. Se valida que el valor cumpla con ciertas restricciones de longitud y espacios. Se repite el proceso hasta que se ingrese un valor válido.
- **saltar_celda:** permite al usuario saltar a una celda específica de la hoja de cálculo. El usuario ingresa el número de fila y la letra de la columna a la que desea saltar. Se realizan validaciones de entrada, se ajusta la estructura de la hoja de cálculo si es necesario y se actualiza la posición actual.
- **guardar:** Guarda el contenido de una hoja de cálculo en un archivo JSON, solicitando al usuario un nombre de archivo, generando la cadena JSON correspondiente y guardándola en el archivo. Luego muestra un mensaje de éxito o fracaso y espera la interacción del usuario.

open_sheet_controller.hpp

- **menu_archivos:** La función menu_archivos lee un archivo JSON llamado "prueba.json" y extrae las filas del objeto JSON. Elimina los corchetes y espacios en blanco de cada fila y las imprime en la consola, mientras obtiene el número de columnas en cada fila.

matrix_service.hpp

- **agregar_fila:** Crea un nuevo nodo “MainListNode” y un nuevo nodo Cell. Luego, se agregan columnas a la nueva fila mediante la función “agregar_columna”. El nuevo nodo “MainListNode” se enlaza correctamente en la lista enlazada “list” al final de la misma, manteniendo la estructura adecuada de punteros previo y siguiente.
- **agregar_columna:** La función "agregar_columna" crea un nuevo nodo "Cell" y asigna un valor a su propiedad "value". Luego, el nuevo nodo se enlaza correctamente en la lista enlazada "list" al final de la misma, manteniendo la estructura adecuada de punteros previo y siguiente.
- **crear_hoja_calculo:** Crea una hoja de cálculo con el número especificado de filas y columnas. Utiliza la función "agregar_fila" para agregar filas a la hoja de cálculo, incrementando el índice de fila en cada iteración.
- **imprimir_hoja_calculo:** Muestra en la consola el contenido de una hoja de cálculo representada por una lista enlazada bidimensional. Recorre la lista de filas y las correspondientes celdas, imprimiendo los valores de cada celda en un formato de tabla. La celda actual (especificada por "actual_row" y "actual_column") se resalta en la salida con un guion bajo (_) en lugar del valor.
- **escribir_contenido:** Actualiza el valor de una celda en una hoja de cálculo representada por una lista enlazada bidimensional. Recorre la lista de filas y las celdas correspondientes hasta llegar a la celda especificada por "actual_row" y "actual_col". Luego asigna el valor "data" a esa celda.
- **contar_filas:** Cuenta el número total de filas en una hoja de cálculo representada por una lista enlazada bidimensional. Comienza desde el nodo inicial de la lista y recorre los nodos siguientes mientras incrementa un contador en cada iteración. Finalmente, devuelve el valor del contador que representa la cantidad de filas en la hoja de cálculo.
- **contar_columnas:** Cuenta el número total de columnas en una fila representada por una lista doblemente enlazada. Comienza desde el nodo inicial de la lista y recorre los nodos siguientes mientras incrementa un contador en cada iteración. Finalmente, devuelve el valor del contador que representa la cantidad de columnas en la fila.
- **vaciar:** Elimina la hoja de cálculo y libera la memoria ocupada por las celdas. Recorre la lista enlazada de nodos principales y, para cada nodo, libera la memoria y avanza al siguiente nodo. Al finalizar, establece el inicio de la lista en NULL para indicar que está vacía.
- **save_file:** Guarda los datos en un archivo especificado por "filename". Si el archivo se abre exitosamente, los caracteres de la cadena de datos se escriben en el archivo y se retorna true. En caso contrario, retorna false.

utils.hpp

El archivo de utilidades contiene las siguientes funciones:

- **clean_cin:** Limpia el estado de error y vacía el búfer de entrada "cin" restableciendo y descartando cualquier entrada no procesada.

- **validate_cin:** Limpia el estado de error y vacía el búfer de entrada "cin", muestra un mensaje de error en la consola y espera a que el usuario presione una tecla para continuar.
- **letter_to_number:** Convierte una letra en minúscula en su equivalente numérico. En resumen, la función realiza las siguientes acciones: convierte la letra en mayúscula, obtiene el código ASCII de la letra restando 'A' y luego devuelve el resultado del código ASCII módulo 26, obteniendo así un número entre 0 y 25.
- **relleno:** Establece un carácter de relleno y un ancho de salida para la impresión en la consola, lo que permite ajustar el formato de los valores mostrados.
- **imprimir_mensaje_confirmacion:** Muestra un mensaje en la consola solicitando al usuario que presione una tecla para continuar.
- **has_space:** La función verifica si una cadena de texto contiene espacios en blanco. Recorre cada carácter de la cadena y si encuentra un espacio en blanco (código ASCII 32), devuelve verdadero (true). Si no encuentra ningún espacio en blanco, devuelve falso (false).
- **contar_archivos:** Cuenta la cantidad de archivos en un directorio específico. Recorre los elementos dentro del directorio y aumenta el contador si el nombre del elemento no tiene una longitud de 1 o 2 caracteres. Al finalizar, devuelve la cantidad de archivos encontrados en el directorio.
- **insertar_elemento:** Crea un nuevo nodo de tipo "File" con un valor y un índice especificados. Luego, verifica si la lista está vacía y, en ese caso, asigna el nuevo nodo como el primer elemento de la lista. Si la lista no está vacía, recorre la lista hasta llegar al último nodo y agrega el nuevo nodo al final de la lista.
- **recuperar_archivos:** Recorre el directorio "files" y recupera el nombre de cada archivo en él. Luego, utiliza la función "insertar_elemento" para agregar cada archivo a la lista "list", asignándole un índice incremental. Finalmente, cierra el directorio.
- **extraer_nombre_archivos:** Busca en la lista "list" el archivo con el índice "file_index" y devuelve su nombre. Recorre la lista hasta encontrar el archivo con el índice correspondiente. Si lo encuentra, devuelve su nombre; de lo contrario, devuelve una cadena vacía.

Flujo de ejecución

1. La ejecución inicia en main.cpp en la función main, la cual es la función principal que ejecuta el programa de c + +, esta hace el llamado de la función menú principal la cual se encuentra en la clase **Menú principal controller** y le pasa el valor de una variable int que será la opción del menú.
2. El menú principal ingresa en un bucle que solo se rompe cuando el usuario ingresa la opción 3. Dentro del bucle se le ofrece al usuario la opción de hacer una nueva hoja de cálculo, abrir una ya existente en formato JSON, o salir del programa, se valida mediante el uso de cin.fail() para detectar si el usuario ingresa un valor que no sea un dato numérico.
    a. En caso de que la entrada sea inválida se llama a la función validate_cin() que se encuentren utils.hpp, la cual limpia el estado de error y vacía el buffer.
    b. Si la entrada es válida, un switch verifica cuál opción fue seleccionada, se divide el flujo en 2 ramas principales, **nueva hoja de cálculo** y **abrir hoja de cálculo**.

i. Nueva hoja de cálculo llama a la función menu_matriz() la cual recibe una variable que maneja las opciones, la cual se encuentra en new_sheet_controller.hpp.
ii. Abrir hoja de cálculo llama a la función menu_archivos(), la cual se encuentra en open_sheet_controller.hpp
iii. Salir cierra el programa
iv. Default indica que se ingresó una opción inválida y llama a la función imprmir_mensaje_confirmación(), la cual imprime: “Presione una tecla para continuar”, luego lee carácter a carácter y muestra por pantalla el carácter leído, la función se encuentra en utils.hpp

Flujo de nueva hoja de cálculo.

Este flujo comienza cuando el usuario escoge crear una nueva hoja de cálculo en el menú de opciones principal, esto hace llamado a la función menu_matriz(), la cual consume a lo largo de toda su ejecución funciones tanto de utils.hpp como funciones de matrix_service.hpp.

1. Como primer paso se crea la hoja de cálculo mediante la llamada de la función crear_hoja_calculo(), la cual requiere una lista, columnas, filas y la constante ASCII_CODE_LETTER_A que sirve para almacenar el código ascii de la primera letra de los encabezados.
2. Se obtiene el número de filas y columnas de la lista principal.
3. Se imprime el menú que únicamente rompe bucle en la opción 11, este muestra una serie de opciones siendo estas, ingresar contenido, saltar a celda, copiar, pegar, Mover a la izquierda, derecha, arriba, abajo, guardar y salir.
4. Valida que la entrada sea un valor numérico válido.
5. Se ingresa a un switch el cual llama las funciones respectivas a la función seleccionada.
    a. La primera opción llama la función insertar_celda() la cual ingresa un valor dentro del nodo en el que se encuentra actualmente.
    b. La segunda opción llama la función saltar_celda() la cual moviliza al usuario a cualquier celda de la tabla dentro del rango máximo permitido siendo este 25xZ.
    c. La tercera opción llama a la función copy() guarda el valor del nodo actual en una variable auxiliar llamada copycell.
    d. La cuarta opción llama a la función cortar() que guarda el valor del nodo actual en una variable auxiliar llamada copycell y además limpia el contenido del nodo.
    e. La quinta opción llama a la función pegar() que inserta el contenido de la variable copycell en el nodo actual.
    f. La sexta opción llama a la función mover_izquierda() que reduce la columna actual en uno.
    g. La séptima opción llama a la función mover_derecha() que incrementa la columna actual en uno.

h. La octava opción llama a la función mover_arriba() que incrementa la fila actual en uno.
i. La novena opción llama a la función mover_abajo() que reduce la fila actual en uno.
j. La décima opción llama a la función guardar() que se encarga de guardar los datos de la hoja mediante la clase JSON.
k. La opción 11 cierra la hoja de cálculo y regresa al menú principal.
l. Default indica que se ingresó una opción inválida y llama a la función imprmir_mensaje_confirmación(), la cual imprime: “Presione una tecla para continuar”, luego lee carácter a carácter y muestra por pantalla el carácter leído, la función se encuentra en utils.hpp

Flujo de abrir hoja de cálculo.

Cuando el usuario seleccione la opción 2. Abrir hoja de cálculo, se llama la función que se menu_archivos() que se encuentra en la clase open_sheet_controller.hpp, la cual hace de la clase JSON.hpp

1. Primero crea un objeto tipo json, el cual usa la función parseFromFile(“nombre_de_archivo.json”), variable de fila, fila formateada y un iterador para esta, se crea un elemento inicial de tipo string que almacenará el contenido parseado a json en forma de cadena de texto y se inicializa una variable para la cantidad de elementos.
2. Mediante un bucle, cuenta las comas y cada una la cuenta como un elemento y posteriormente se adjunta dicha cantidad a las columnas
3. Luego mediante un bucle ingresa los datos del objeto en una nueva lista.

Diagrama del flujo de ejecució