Universidad De San Carlos de Guatemala
Centro Universitario de Occidente CUNOC

MANUAL TÉCNICO

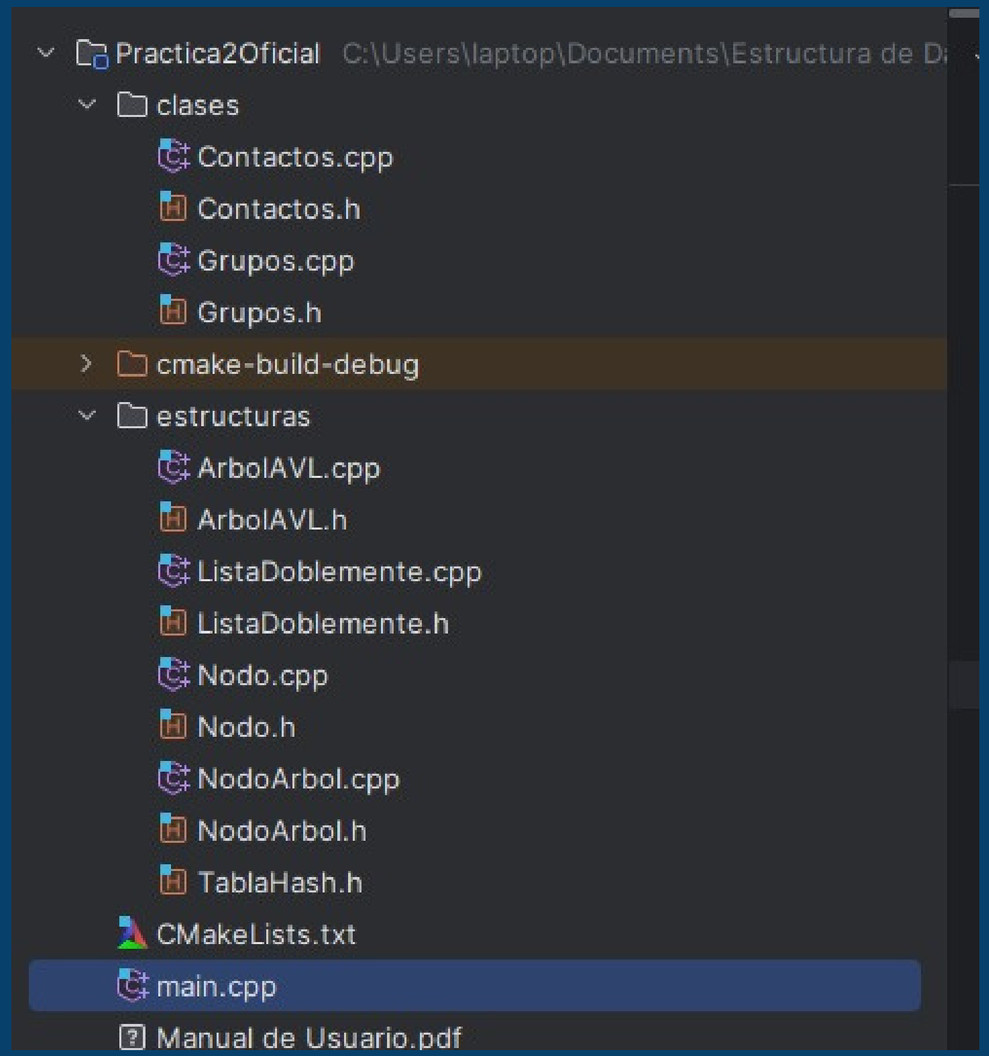
202031773 Enmer Oswaldo Sandoval Mazariegos

Sistema de Gestor de Contactos: Resumen del Manual de Usuario

Un sistema de gestor de contactos es una herramienta diseñada para organizar y gestionar información de contactos individuales y grupos. Permite:

- Registrar detalles como nombres, números de teléfono, correos electrónicos y direcciones.
- Crear y administrar grupos de contactos, como familiares, amigos o compañeros de trabajo.
- Editar, eliminar y buscar contactos de manera eficiente.
- Exportar listas de contactos para transferencia de datos.

Distribución del proyecto.



Clase main

```cpp
using namespace std;
void menuPrincipal() {
    cout << "****** CODE & BUGS ******" << endl;
    cout << "1. Ingrese el comando" << endl;
    cout << "2. Graficas" << endl;
    cout << "3. Exportacion de contactos" << endl;
    cout << "4. Reportes" << endl;
    cout << "5. Salir" << endl;
    cout << "Ingrese la opcion deseada: ";
}

void submenuGraficas() {
    cout << "****** GRAFICAS ******" << endl;
    cout << "1. Estado actual de la estructura completa." << endl;
    cout << "2. Estado actual de cada una de las estructuras por separado mostrara solo 1 grupo
    cout << "3. Estado de visualizacion de todos los datos contenidos." << endl;
    cout << "5. Regresar" << endl;
    cout << "Ingrese la opcion deseada: ";
}
```

```cpp
template <typename T>
void verificacionCreacion(std::string &comando, TablaHash<T> &tablaHash){
    std::istringstream iss( str: comando);
    limpiarPantalla();
    std::string add, newgroup, nameGroup, fields, attributes;
    iss >> add >> newgroup >> nameGroup >> fields;
    if (add == "ADD" && newgroup == "NEW-GROUP" && fields == "FIELDS" ){
        Grupos* grupo = new Grupos();
        grupo->setNombreGrupo(nameGroup);
        cout << "Grupo creado con exito " << attributes << endl;
        cout << "El nombre del grupo es: " <<  grupo->getNombreGrupo() << endl;
        std::getline( &: iss,  &: attributes,  delim: ';');
        attributes.erase( pos: 0,  n: 2);
        attributes.erase( pos: attributes.size() - 1);
        std::istringstream iss2( str: attributes);
        std::string attribute;
        while (std::getline( &: iss2,  &: attribute,  delim: ',')) {
            std::istringstream iss3( str: attribute);
            std::string key, value;
            iss3 >> key >> value;
            grupo->group[key] = value;
        }
        for (const auto& pair : pair<...> const & : grupo->group) {
            cout << pair.first << ": " << pair.second << endl;
        }
rificacionCreacion
```

Descripción

1. **menuPrincipal():**
   - **Descripción: Muestra el menú principal con opciones para agregar comandos, visualizar estadísticas, exportar datos, generar reportes y salir del programa.**
   - **Uso: Al iniciar el programa, seleccione una opción del menú principal para acceder a las funcionalidades deseadas.**

limpiarPantalla():
Descripción: Limpia la pantalla mostrando varios saltos de línea para mejorar la visualización.
Uso: Utilizado antes de mostrar nuevos datos o menús para mantener una interfaz limpia y organizada.

submenuGraficas():
Descripción: Presenta el submenú de gráficas con opciones para visualizar el estado de grupos y contactos.
Uso: Acceda al submenú de gráficas desde el menú principal para obtener información detallada sobre el estado de las estructuras de datos.

verificacionCreacion(std::string &comando, TablaHash<T> &tablaHash)
Descripción: Verifica y crea un nuevo grupo con los campos y atributos especificados en el comando.
Uso: Llamado al ingresar un comando "ADD NEW-GROUP FIELDS" para crear un nuevo grupo con campos personalizados.

Estructuras Utilizadas

1. ListasDoblementeEnlazadas
2. ÁrbolAVL
3. NodoÁrbol
4. TablaHash
5. Grupos
6. Contactos

Estructuras Utilizadas