Universidad de San Carlos de Guatemala
Centro Universitario de Occidente División de Ciencias de la Ingeniería
Estructura de Datos
Aux. Yefer Rodrigo Alvarado Tzul
Sección: A

Manual Técnico

Carlos Raúl Alberto López Peláez 202031871

Quetzaltenango,2 de Abril del 2024

Main del programa de Sistema de gestión de Contactos

1. Inclusión de bibliotecas y archivos de encabezado:

    - #include <iostream>: Incluye la biblioteca estándar de entrada y salida de C++.
    - #include <vector>: Incluye la biblioteca de vectores estándar de C++.
    - #include <string>: Incluye la biblioteca de cadenas estándar de C++.
    - #include <sstream>: Incluye la biblioteca de flujos de cadena estándar de C++.
    - "Parametro/Parametro.h", "Hash/Hash.h", "Comandos/Comandos.h": Incluye los archivos de encabezado de las clases definidas en estos archivos.

2. Espacio de nombres:

- using namespace std;: Hace que todas las clases y funciones del espacio de nombres std estén disponibles sin tener que calificarlos con std::.

3. Función main():

- Se define la función principal del programa.
- Se crean instancias de las clases Hash, Parametro y Comandos.
- Se comentan algunas líneas de código que parecen ser pruebas o ejemplos de uso de las clases.
- Se declara una cadena comando para almacenar la entrada del usuario.
- Se inicia un bucle while que solicita al usuario que ingrese un comando hasta que escriban "fin".
- Dentro del bucle, se llama al método ejecutar de la instancia comandos para procesar el comando ingresado por el usuario.
- Finalmente, se liberan los recursos asignados dinámicamente al eliminar las instancias creadas con new.

Parametro

Directivas de preprocesador:

- #ifndef POYECTO_2_PARAMETRO_H: Comienza un bloque condicional que verifica si el identificador POYECTO_2_PARAMETRO_H no está definido. Si no está definido, se incluirá el código hasta #endif.
- #define POYECTO_2_PARAMETRO_H: Define el identificador POYECTO_2_PARAMETRO_H, evitando que el código dentro del bloque se incluya más de una vez en el mismo archivo de código fuente.

2. **Inclusión de archivos de encabezado y espacio de nombres**:
   - #include <string>: Incluye la biblioteca estándar de cadenas de C++.
   - "../Nodos/NodoGrupo.h": Incluye el archivo de encabezado que define la clase NodoGrupo.
3. **Declaración de la clase Parametro**:
   - class Parametro { ... };: Define la clase Parametro, que contiene varios miembros de datos y funciones miembro.
   - public:: Especifica que los miembros de la clase que siguen a esta etiqueta son accesibles desde fuera de la clase.
   - string valor;: Una cadena que representa el valor del parámetro.
   - int valorAscii;: Un entero que almacena el valor ASCII del parámetro (aunque no parece ser utilizado en el código que has proporcionado).
   - NodoGrupo *grupo;: Un puntero a un objeto de la clase NodoGrupo, que representa el grupo al que pertenece el parámetro.
   - string tipo;: Una cadena que representa el tipo de parámetro (por ejemplo, "STRING", aunque podría ser cualquier tipo).
   - Parametro *parametroAnt; Parametro *parametroSig;: Punteros a objetos de la clase Parametro que representan el parámetro anterior y siguiente en una lista enlazada (aunque no se utiliza explícitamente en el código proporcionado).
   - Parametro();: Declaración del constructor por defecto de la clase Parametro.
   - ~Parametro();: Declaración del destructor de la clase Parametro.
4. **Directiva de preprocesador de cierre**:
   - #endif: Termina el bloque condicional iniciado por #ifndef, evitando que el código sea incluido más de una vez.

Hash

Directivas de preprocesador:

- #ifndef POYECTO_2_HASH_H: Comienza un bloque condicional que verifica si el identificador POYECTO_2_HASH_H no está definido. Si no está definido, se incluirá el código hasta #endif.
- #define POYECTO_2_HASH_H: Define el identificador POYECTO_2_HASH_H, evitando que el código dentro del bloque se incluya más de una vez en el mismo archivo de código fuente.

2. **Inclusión de archivos de encabezado**:
    - #include <string>: Incluye la biblioteca estándar de cadenas de C++.
    - "../Nodos/NodoParametro.h" y "../Nodos/NodoGrupo.h": Incluye los archivos de encabezado que definen las clases NodoParametro y NodoGrupo, respectivamente.
3. **Declaración de la clase Hash**:
    - class Hash { ... };: Define la clase Hash, que contiene miembros de datos y funciones miembro para implementar una tabla hash.
    - private:: Especifica que los miembros de la clase que siguen a esta etiqueta solo son accesibles dentro de la clase.
    - NodoGrupo *grupo[10]; NodoParametro *parametro[10];: Arreglos de punteros a objetos de las clases NodoGrupo y NodoParametro, respectivamente, que representan las listas de grupos y parámetros almacenados en la tabla hash.
    - int espacioGrupo; int espacioParametro;: Variables enteras que representan el espacio disponible en los arreglos grupo y parametro.
    - int convertirAscii(string valor);: Declaración de una función miembro que convierte una cadena en su valor ASCII (aunque no se utiliza explícitamente en el código proporcionado).
    - public:: Especifica que los miembros de la clase que siguen a esta etiqueta son accesibles desde fuera de la clase.
    - NodoParametro claveParametro; NodoGrupo claveGrupo;: Objetos de las clases NodoParametro y NodoGrupo que se utilizan como claves para la búsqueda en la tabla hash.
    - int indice;: Una variable entera que representa el índice de un elemento en la tabla hash.

- void ingresarGrupo(NodoGrupo *nodoGrupo); void ingresarParametro(NodoParametro *parametroNuevo);: Métodos para ingresar un grupo y un parámetro en la tabla hash.
- void imprimirGrupo(); void imprimirParametro();: Métodos para imprimir los grupos y parámetros almacenados en la tabla hash.
- NodoParametro* buscarParametro(string parametron); NodoGrupo* buscarGrupo(string grupon);: Métodos para buscar un parámetro o grupo en la tabla hash y devolver un puntero a su nodo correspondiente.

4. **Directiva de preprocesador de cierre**:
    - #endif: Termina el bloque condicional iniciado por #ifndef, evitando que el código sea incluido más de una vez.

Árbol

Directivas de preprocesador:

- #ifndef POYECTO_2_ARBOL_H: Comienza un bloque condicional que verifica si el identificador POYECTO_2_ARBOL_H no está definido. Si no está definido, se incluirá el código hasta #endif.
- #define POYECTO_2_ARBOL_H: Define el identificador POYECTO_2_ARBOL_H, evitando que el código dentro del bloque se incluya más de una vez en el mismo archivo de código fuente.

2. **Inclusión de archivos de encabezado**:
   - #include <string>: Incluye la biblioteca estándar de cadenas de C++.
   - "../Nodos/NodoArbol.h" y "../ListaDoblementeEnlazada/ListaDoblementeEnlazada.h": Incluye los archivos de encabezado que definen las clases NodoArbol y ListaDoblementeEnlazada, respectivamente.
3. **Declaración de la clase Arbol**:
   - class Arbol { ... };: Define la clase Arbol, que contiene miembros de datos y funciones miembro para implementar un árbol binario de búsqueda.
   - private:: Especifica que los miembros de la clase que siguen a esta etiqueta solo son accesibles dentro de la clase.
   - Varias variables privadas, incluyendo punteros a nodos y enteros para manipular el árbol y realizar operaciones como rotaciones y equilibrado.
   - int convertirAscii(string valor);: Declaración de una función miembro que convierte una cadena en su valor ASCII (aunque no se utiliza explícitamente en el código proporcionado).
   - Parametro* buscarDato(NodoArbol *recorrer, string buscado, int nuevP);: Método privado para buscar un dato en el árbol.
   - void insertarNuevo(NodoArbol *recorrer, NodoArbol *nuevo, NodoArbol *PadreAB);: Método privado para insertar un nuevo nodo en el árbol.
   - void verArbol(NodoArbol *recorrer, int n);: Método privado para visualizar el árbol.
   - void rotarID(); void rotarDI(); void rotarDD(); void rotarII();: Métodos privados para realizar rotaciones en el árbol.
   - void necesidadEquilibrar(NodoArbol *recorrer);: Método privado para verificar si se necesita equilibrar el árbol.
4. **Métodos públicos de la clase**:

- public:: Especifica que los miembros de la clase que siguen a esta etiqueta son accesibles desde fuera de la clase.
- Arbol();: Declaración del constructor por defecto de la clase Arbol.
- void ingresar(string nuevoDato, NodoGrupo *grupo, string tipo);: Método público para insertar un nuevo dato en el árbol.
- Parametro* buscar(string valor, int nuevP);: Método público para buscar un dato en el árbol.
- void imprimir(); void graficar(); void graficarArbol(NodoArbol *recorrer, int n, ofstream &file);: Métodos públicos para imprimir y graficar el árbol.

5. **Directiva de preprocesador de cierre**:
    - #endif: Termina el bloque condicional iniciado por #ifndef, evitando que el código sea incluido más de una vez.