Universidad Distrital Francisco José de Caldas

Curso: Ciencias de la computación 1

Nombres y códigos :

Sebastián Salazar - 20172020006

Javier Andres Aponte Quevedo - 20172020036

Daniel Alejandro Rodriguez Suarez -20172020009

**Documentación del proyecto final:**

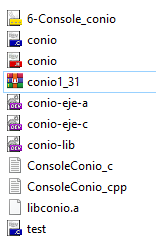
El propósito de nuestro proyecto final es el hacer un sistema gestor de información para una comunidad religiosa. Dadas las circunstancias de la sustentación de este trabajo nos hemos tomado la libertad de añadir una librería adicional para implementar una mejor interfaz visual.

Instalación

Se requiere descargar la libreria **conio.h** para lograr la ejecución, la podrás descargar en <https://bit.ly/2Ju62WF> , lo siguiente que debes hacer es descomprimir el archivo en una carpeta nueva.

Hacer una nueva carpeta y descomprimir el archivo en esta



Estos son los archivos que aparecerán

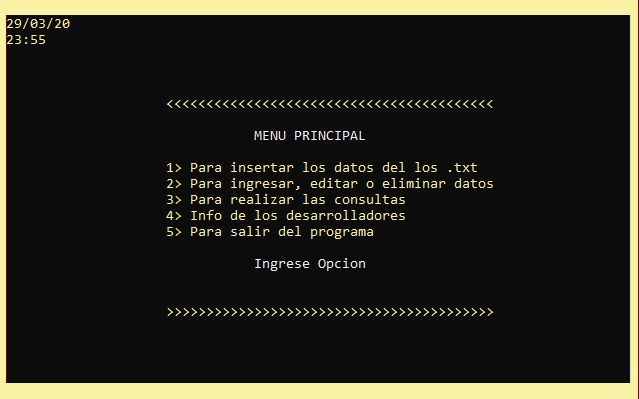
Ahora abriremos una nueva carpeta en la dirección donde hemos instalado Dev c++, esta suele ser **C:\Program Files(x86)\Dev-Cpp,** luego nos dirigimos a la carpeta **MinGW64** , una vez aquí nos vamos a dirigir a **include** y vamos a copiar los archivos **conio.h** y **conio.cpp** y los vamos a pegar (se necesitan permisos de administrador).

Luego volvemos a **MinGWG64** y nos dirigimos a **lib**, aquí vamos a pasar el archivo **libconio.a**. Para finalizar nos devolvemos a **Dev-Cpp** y entramos a **templates** y pegamos **6-Console\_conio.template** en esta carpeta.

Como paso opcional queda el eliminar el archivo comprimido y la carpeta que se hizo en los primeros pasos.

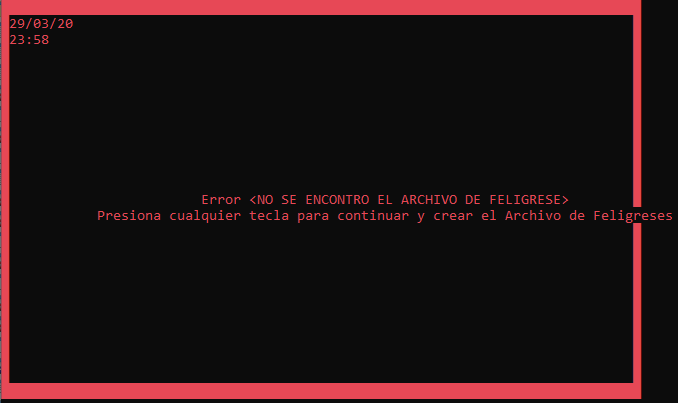
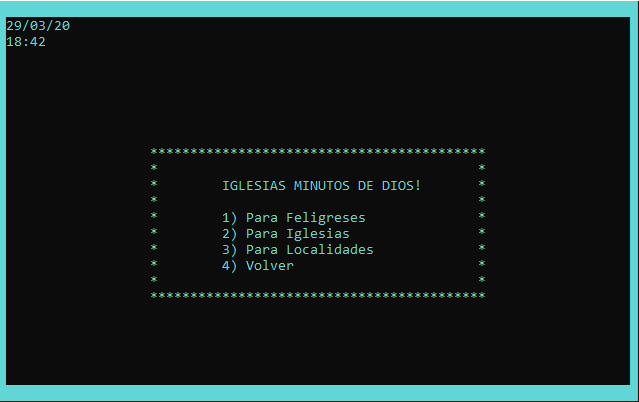
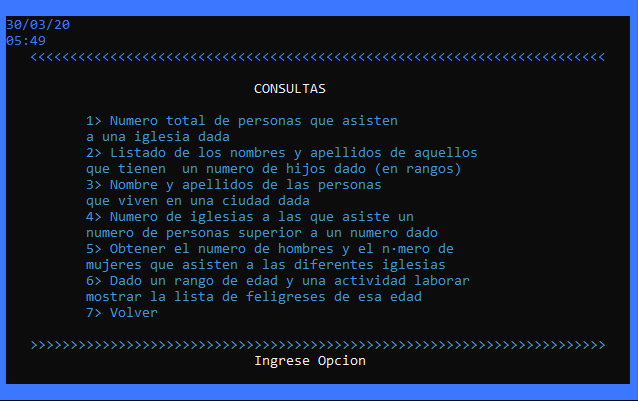
Si se tienen dudas con los anteriores pasos puede dirigirse a este vídeo: <https://youtu.be/S4az7cVyvT0>

Guía de usuario



Si se completaron los anteriores pasos podrá ejecutar el código y se encontrará con la pantalla principal.

Dependiendo de la opción que ingrese podrá acceder a los diferentes submenús que hay en la aplicación.

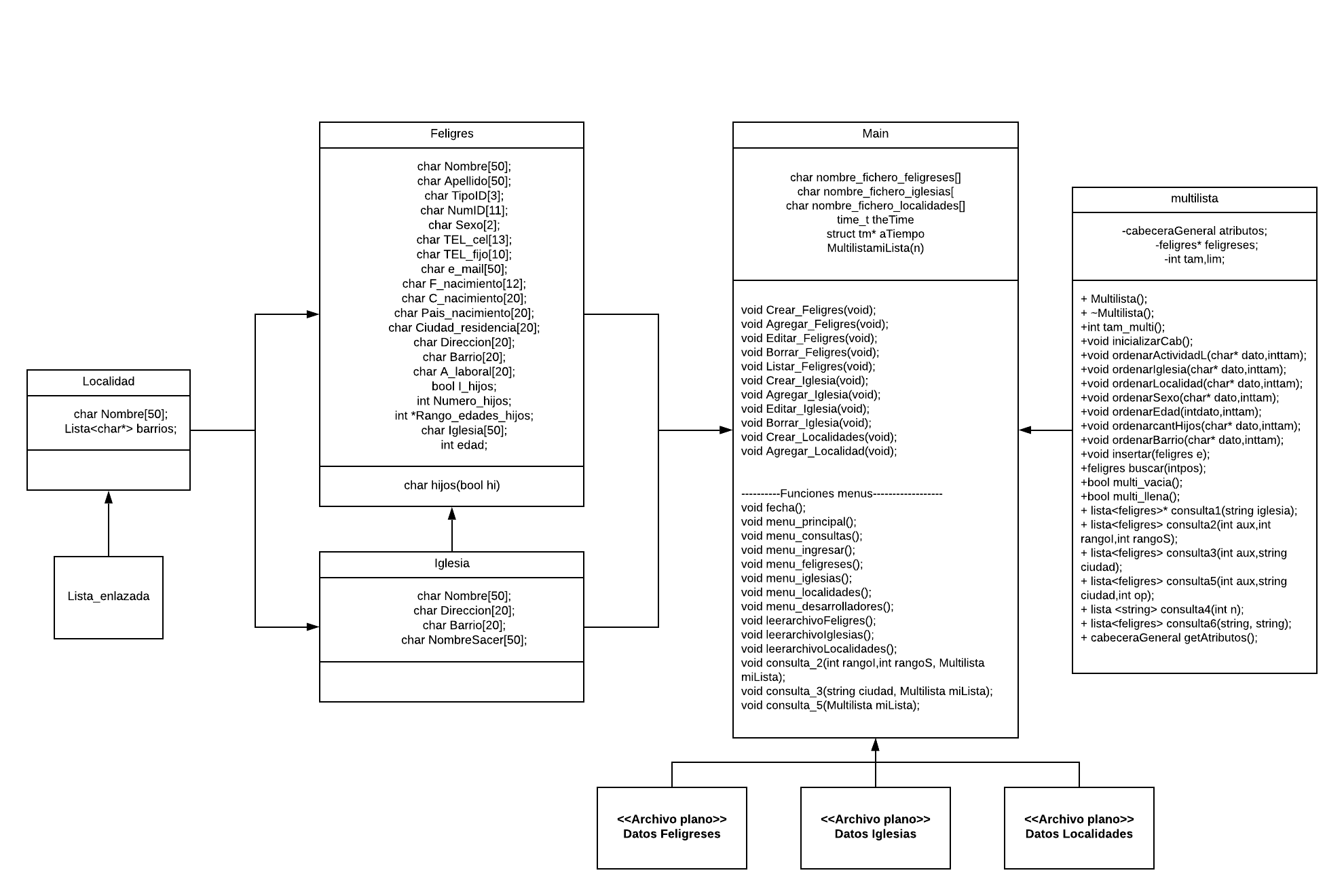
* **Submenú para registrar la información de los archivos planos  
  **Si no encuentra los archivos con la información binaria, usa los archivos planos txt para crear los registros y poder trabajar con ellos.
* **Submenú para ingresar, editar o eliminar datos  
  **
* **Submenú para realizar las consultas  
    
  **

Invitamos al lector a revisar los submenús restantes.

Diseño de la solución

Para cumplir con los requerimientos del proyecto se realizaron distintos diagramas que en primera instancia dieron una idea de cómo debíamos proceder, y que después se fueron refinando mientras se hacía el desarrollo.

**Diseño general**



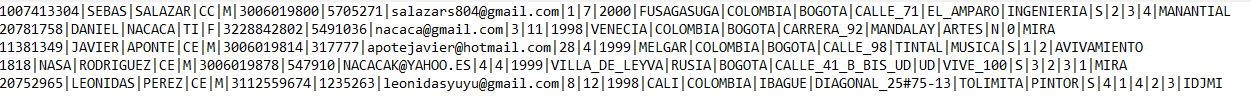
Estructura de los archivos planos

Los archivos planos se asemejan a la estructura de un archivo csv, con la particularidad de que el elemento que hace la separación es el carácter “|” y no la coma.

**Ejemplo archivo feligreses**

En la siguiente imagen se separa la información de la siguiente forma:

ID|nombre|apellido|tipodocumento|sexo|celular|telefono|e-mail|dia\_n|mes\_n|año\_n|c\_nacimiento| p\_nacimiento|c\_residencia|direccion|barrio|tiene\_hijos|(edades de los hijos)|iglesia|edad



**Ejemplo archivo iglesias**

En la siguiente imagen se separa la información de la siguiente forma:

|nombre|direccion|barrio|nombre\_sacerdote|



**Ejemplo archivo localidades**

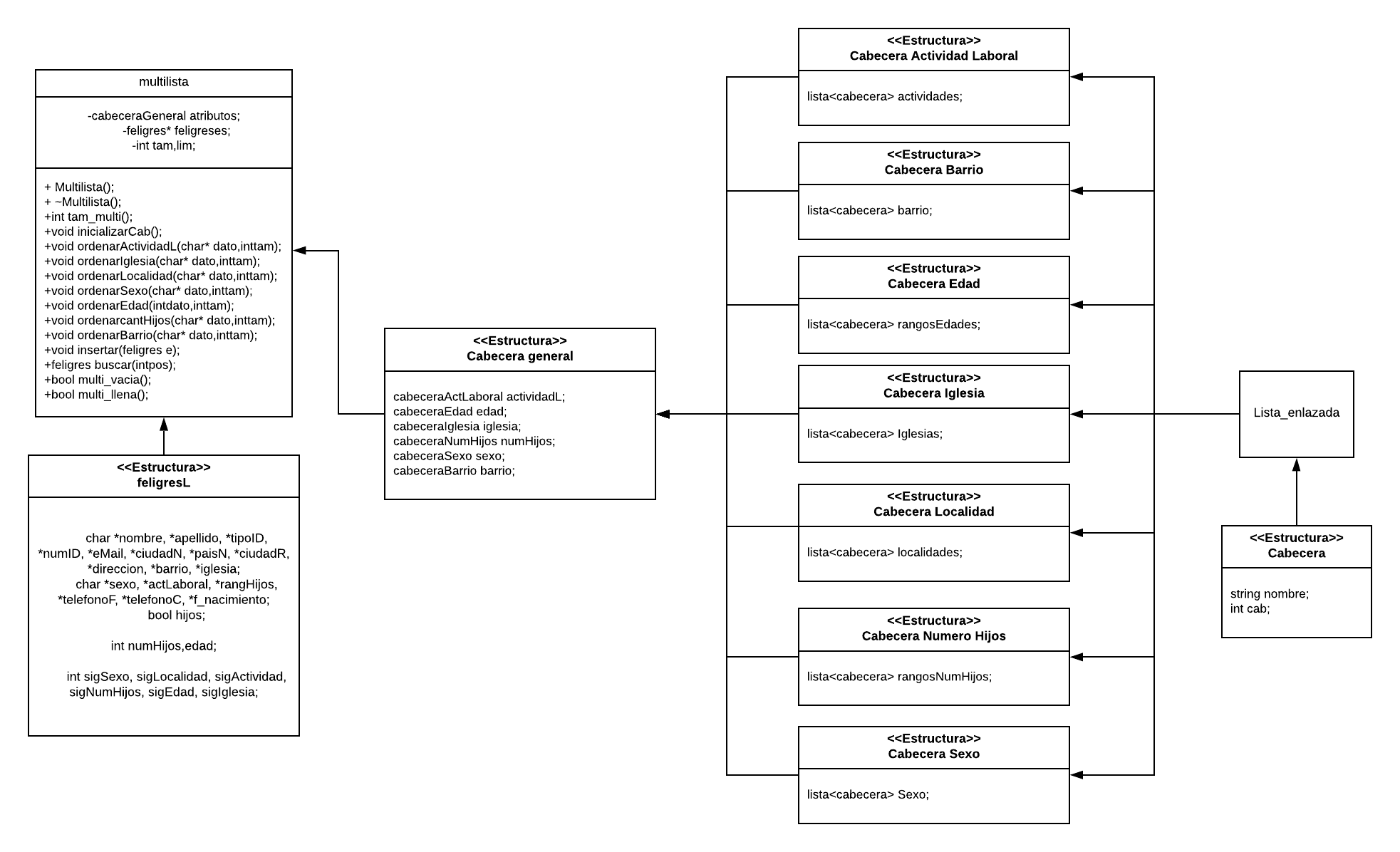
En la siguiente imagen se separa la información de la siguiente forma:

|Nombre|(lista de barrios)|



Estructura de datos en memoria principal

Se utilizó una multilista dinámica para clasificar a los feligreses dependiendo de distintos parámetros.



Las consultas usan una combinación de la información guardada en la memoria principal y la estática. Un ejemplo de esto es la consulta 4 en la cual con la información de la multilista se realizan los filtros que permiten saber los nombres de las iglesias a las que asisten más de n personas, y con estos nombres se realiza una búsqueda en los archivos para poder desplegar la información de las iglesias.