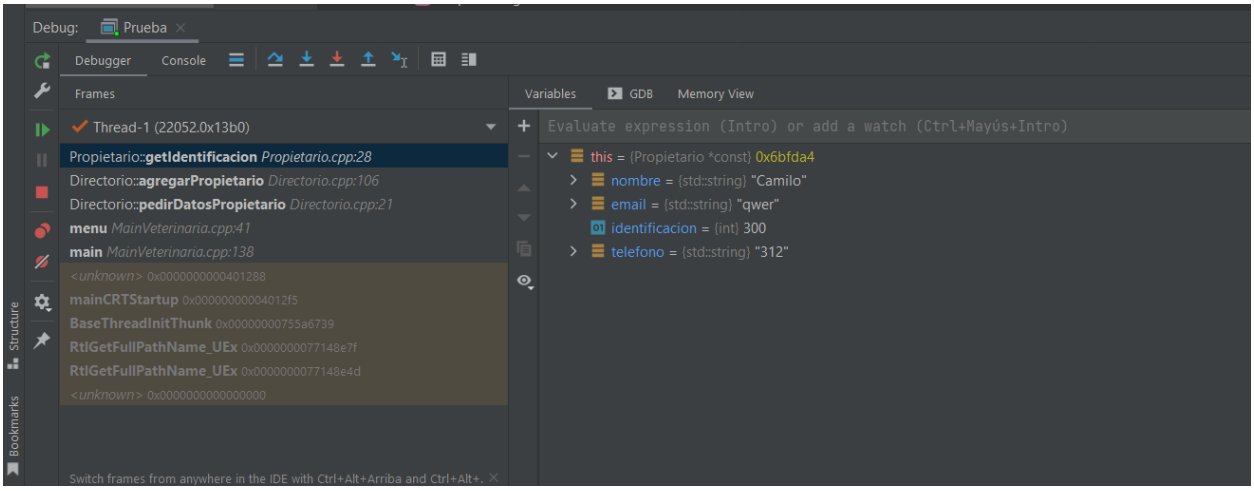
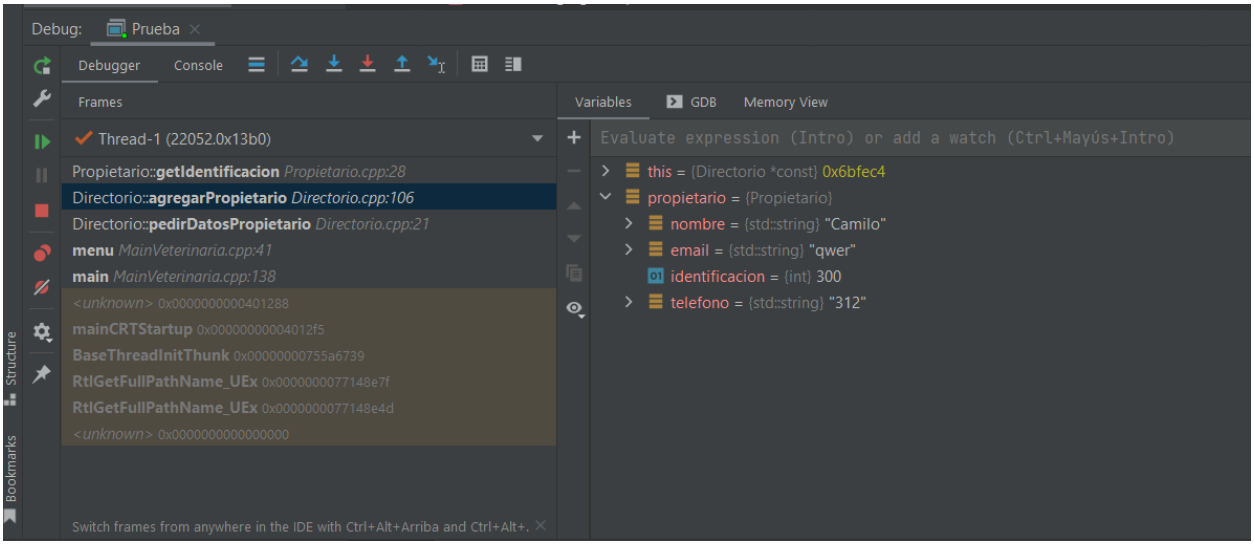
Punto 1

Para este punto se utiliza el método getIdentificación, debido a que el otro método no estaba implementado en el código.



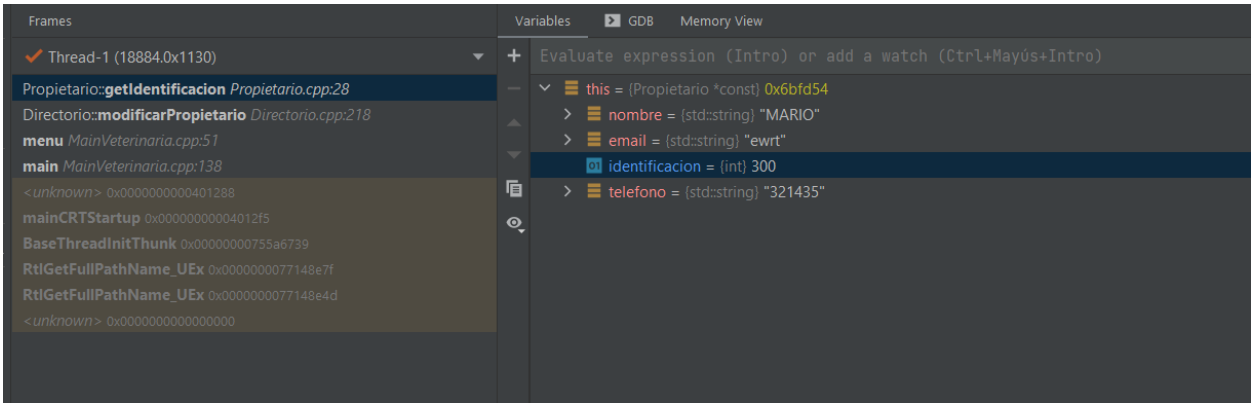


El método que inició toda la ejecución fue el método pedirDatosPropietario de la clase Directorio.

Punto 2

```
Para el usuario identificado con el numero: 800
Los datos de propietario son:
El nombre del propietario es : MARIO
La direccion de correo electronico del propietario es : ewrt
La identificacion del propietario es : 800
El numero de telefono del propietario es : 24324
```

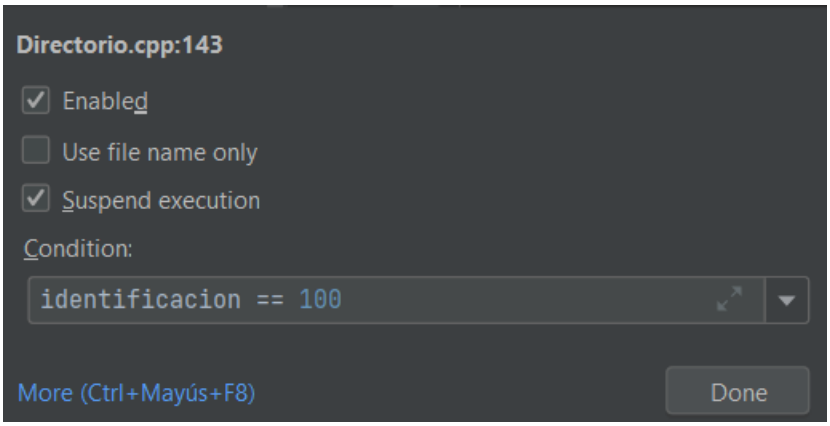
Se contaba inicialmente con este propietario



Se modificó dentro del debugger el número de identificación donde paso de ser 800 a 300.

Lo que en el caso propuesto por la profe sería cambiar el nombre a “PEDRO PEREZ”

Punto 3



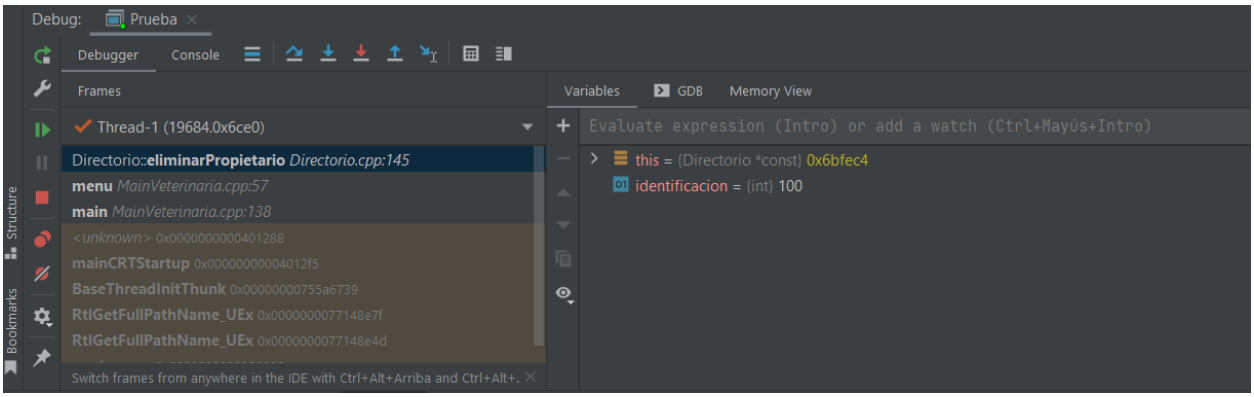
Condición establecida

```cpp
void Directorio::eliminarPropietario(int identificacion){
    //Este if permite verificar si el propietarios con la identificacion seleccionada, está registrado en el mapa o no
    if(mapaPropietario.find( x: identificacion)!= mapaPropietario.end()){ // Buscamos la llave en el mapa (identificacion)
        mapaPropietario.erase( x: identificacion);
        cout << "El propietario con identificacion " << identificacion << " ha sido eliminado\n";
    }else{
        cout << "El propietario con identificacion " << identificacion << " no existe en el sistema\n";
    }
```

Breakout condicional en el método eliminarPropietario(int identificación)

```
9.  Consultar las mascotas de un propietario
10. Asociar una mascota con un propietario
11. Asociar un propietario con una mascota
12. Cambiar el estado de una mascota
13. Eliminar un propietario de una mascota
14. Eliminar una mascota de un propietario
15. Mostrar todas las mascotas que están vivas
4
Escriba la identificacion del propietario que desea eliminar
100
```

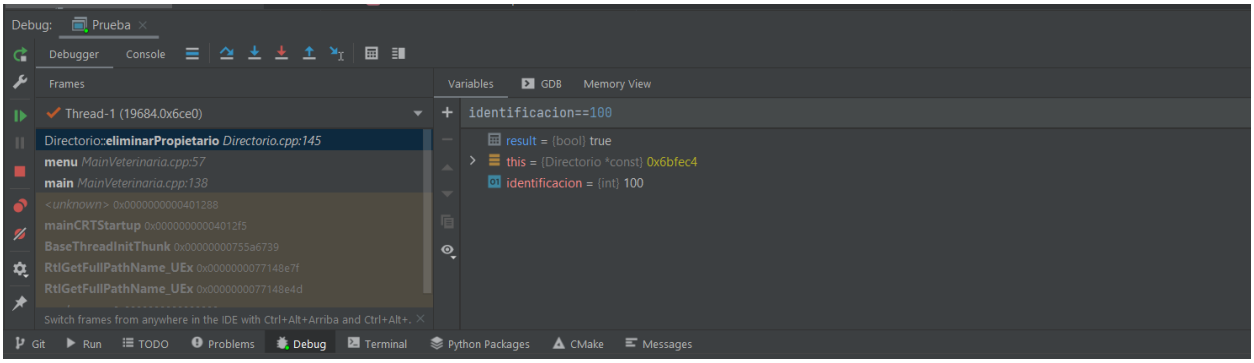
Desde el directorio, se ingresa desde el terminal el valor de Identificación; para este caso, va a 100



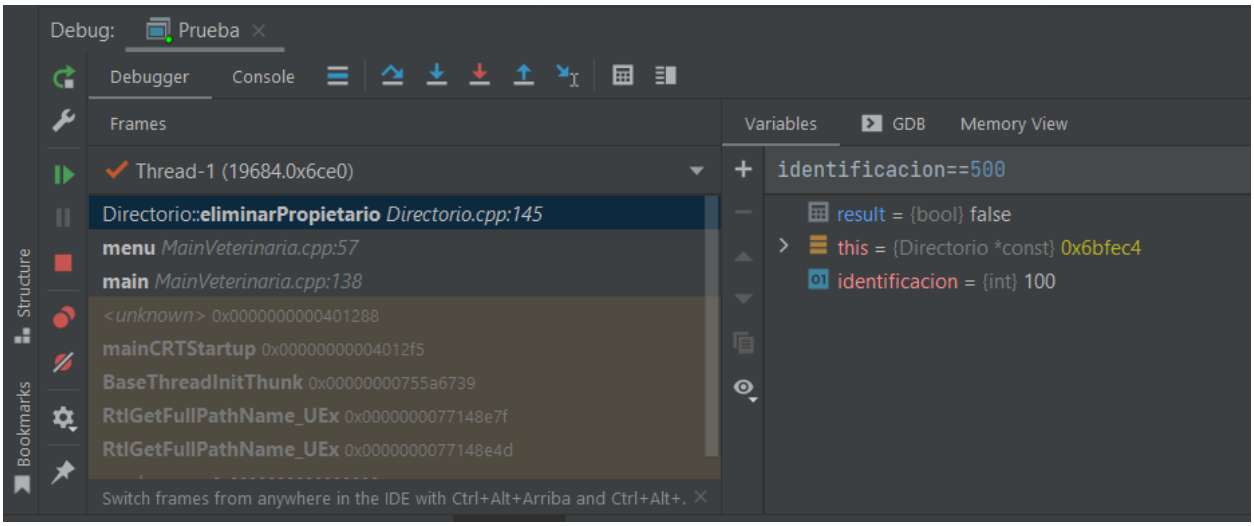
Como resultado, se entra al breakout condicional

Punto 4

Se trabaja con el mismo método del punto anterior. En este caso, se define una expresión como identificacion ==100 y como el propietario con esta identificación había sido previamente agregado, la función da como resultado true para eliminarlo.

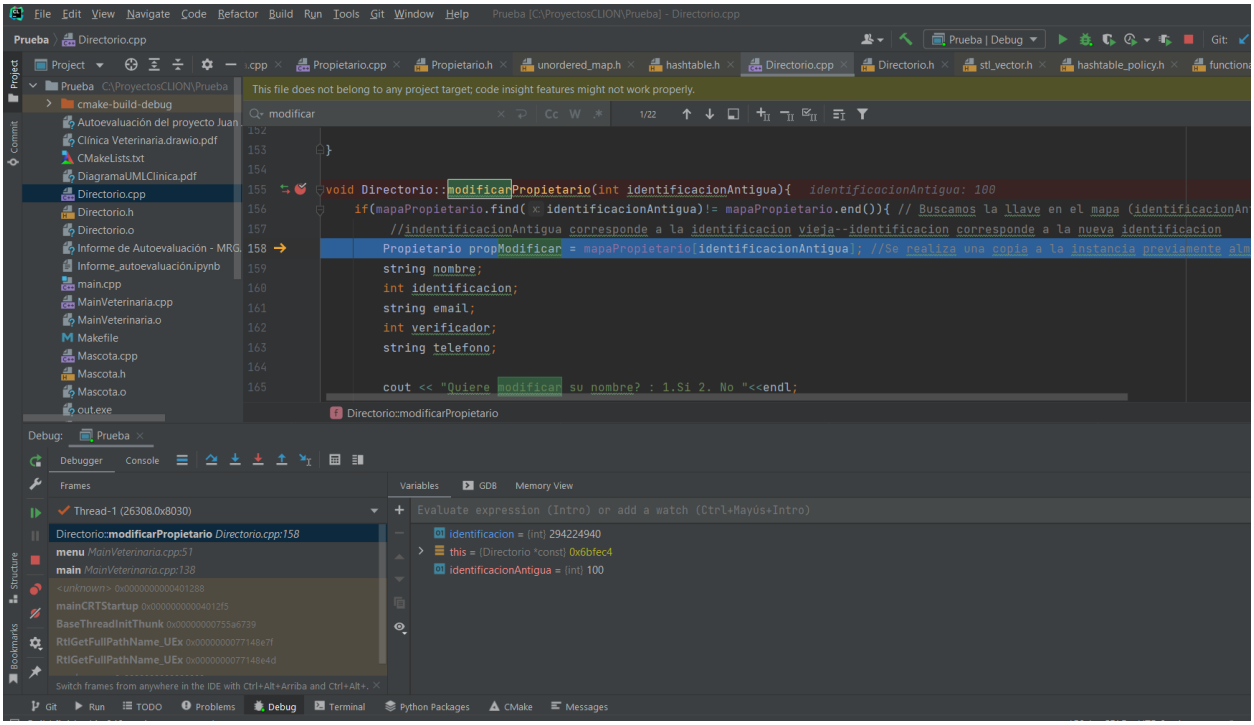


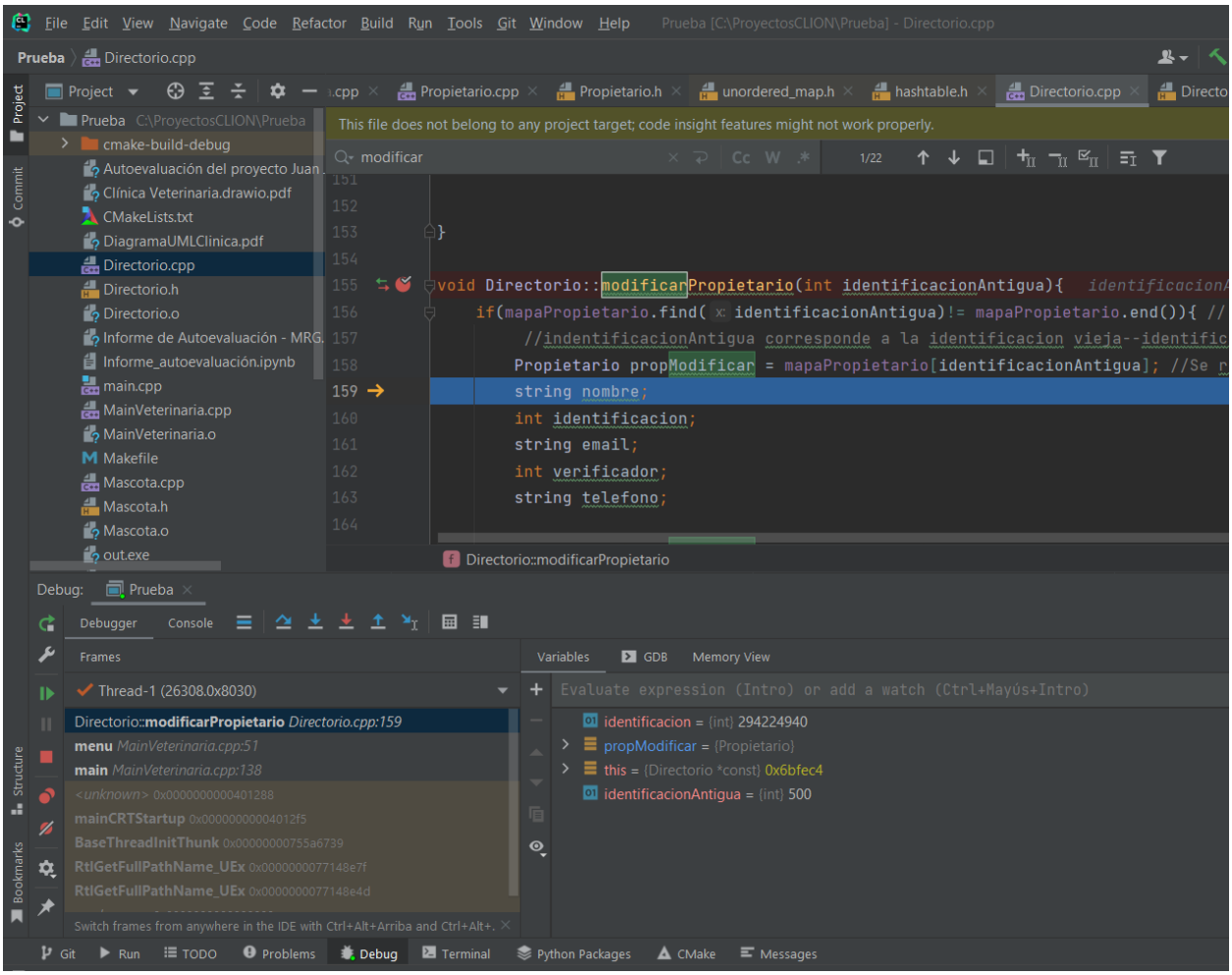
En el otro caso, se define una expresión como identificacion ==500 y como el propietario con esta identificación no existe en el sistema, no es posible eliminarlo y la función da como resultado false.

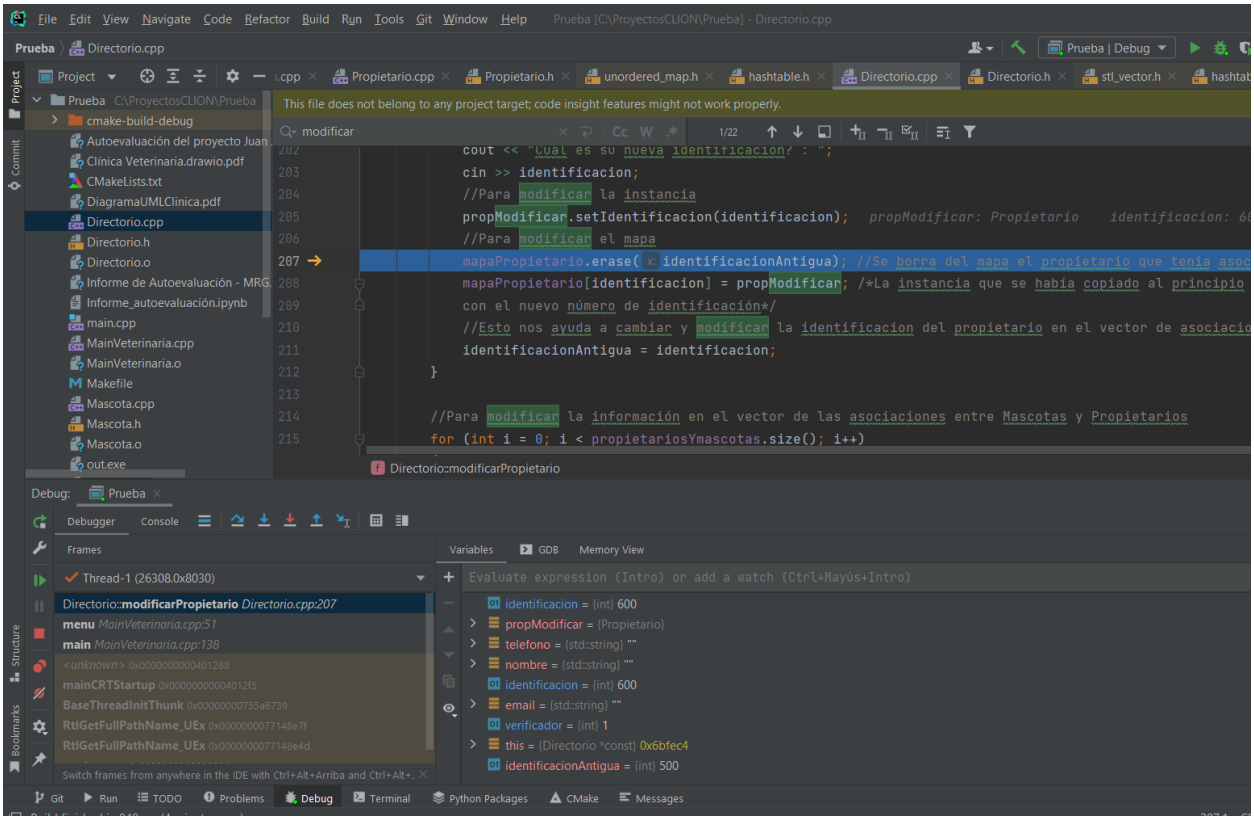


Punto 5

Para este punto se define un watch que tenga en la “mira” a la variable identificación, para ver cómo va ir cambiando a lo largo que se corre el programa.




File Edit View Navigate Code Refactor Build Run Tools Git Window Help
Prueba [C:\ProyectosCLION\Prueba] - Directorio.cpp
Prueba
Directorio.cpp
Project
Propietario.cpp
Propietario.h
unordered_map.h
hashtable.h
Directorio.cpp
This file does not belong to any project target; code insight features might not work properly.
modificar
1/22
Prueba C:\ProyectosCLION\Prueba
cmake-build-debug
Autoevaluación del proyecto Juan
Clínica Veterinaria.drawio.pdf
CMakeLists.txt
DiagramaUMLClinica.pdf
Directorio.cpp
Directorio.h
Directorio.o
Informe de Autoevaluación - MRG
Informe_autoevaluación.ipynb
main.cpp
MainVeterinaria.cpp
MainVeterinaria.o
Makefile
Mascota.cpp
Mascota.h
Mascota.o
out.exe
151
152
153 }
154
155 void Directorio::modificarPropietario(int identificacionAntigua){
156 if(mapaPropietario.find(x identificacionAntigua)!= mapaPropietario.end()){ //
157 //indentificacionAntigua corresponde a la identificacion vieja--identific
158 Propietario propModificar = mapaPropietario[identificacionAntigua]; //Se
159 string nombre;
160 int identificacion;
161 string email;
162 int verificador;
163 string telefono;
164
Directorio::modificarPropietario
Debug: Prueba
Debugger Console
Frames
Thread-1 (26308.0x8030)
Directorio::modificarPropietario Directorio.cpp:159
menu MainVeterinaria.cpp:51
main MainVeterinaria.cpp:138
<unknown> 0x0000000000401288
mainCRTStartup 0x00000000004012f5
BaseThreadInitThunk 0x00000000755a6739
RtlGetFullPathName_UEx 0x0000000077148e7f
RtlGetFullPathName_UEx 0x0000000077148e4d
Switch frames from anywhere in the IDE with Ctrl+Alt+Arriba and Ctrl+Alt+.
Variables GDB Memory View
Evaluate expression (Intro) or add a watch (Ctrl+Mayús+Intro)
identificacion = {int} 294224940
propModificar = {Propietario}
this = {Directorio *const} 0x6bfec4
identificacionAntigua = {int} 500
Git Run TODO Problems Debug Terminal Python Packages CMake Messages
Structure
Bookmarks
Commit



Se puede ver como fue cambiando la variable identificación durante el flujo del código. Para, finalmente, tener un valor de 500 debido a la modificación que se realizó, desde el terminal, a la identificación del propietario.

Punto 6

A lo largo del trabajo con del debugger, pude notar la diferencia que tiene el step over(F8) y el step into(F7). El primero ayuda a recorrer paso a paso las líneas de código, pero sin la necesidad de entrar tanto a detalle de lo que hay detrás de cierta función o implementación. Por otro lado, el step into brinda muchos detalles de lo que conforma a cierta función o variable, de tal manera que había momentos en los que incluso se baja tanto en el nivel de abstracción, que se llegaba a ver cómo cierta parte del código está implementada en lenguaje ensamblador.

Por ejemplo, cuando se pone un breakout desde el archivo MainVeterinaria y se comienzan a utilizar step into, el debugger te lleva a recorrer las líneas de código donde cierto método fue previamente definido o incluso la clase a donde pertenece un método en particular.

Otro ejemplo, es que, en el código de clínica veterinaria, se planteó un método que pide los datos del propietario y con ellos crea una instancia de tipo propietario a partir del constructor con parámetros; por esto, cuando se utiliza el step into, el debugger lo lleva a uno hasta el archivo de la clase donde está definido ese constructor en particular. No obstante, lo mismo no sucedía con el step over, donde simplemente no se mostraba ese nivel de detalle.