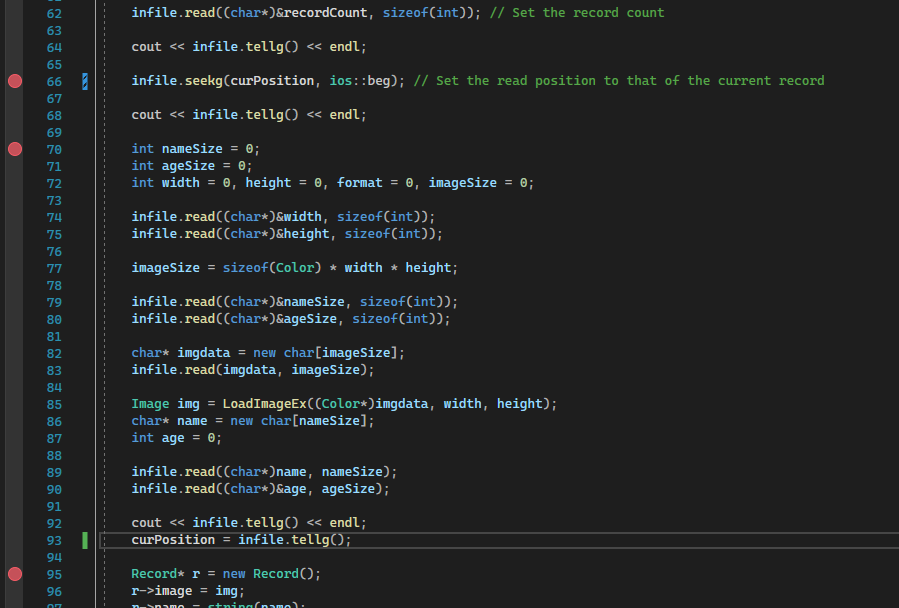
**NPC Data-Checker Program Fix**

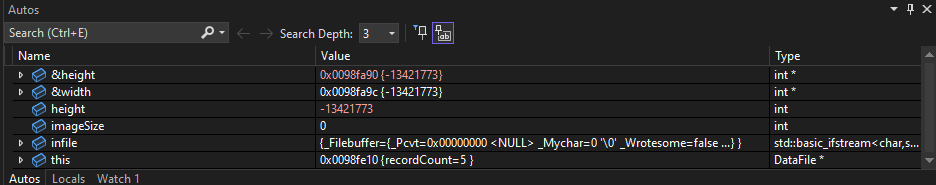
The following tests were conducted after I implemented my first attempt at correcting the program.

**Test 1 – Testing the updated ‘Load’ function**

* Input
  + Pass the database name into the load function
* Expected Output
  + First record in the database is loaded and appears onscreen
* Actual Output
  + First record correctly appears onscreen

The Autos menu, which I used to check if data was correctly being pulled from the database during Test 1.

Screen capture showing some breakpoints I used while conducting Test 1.

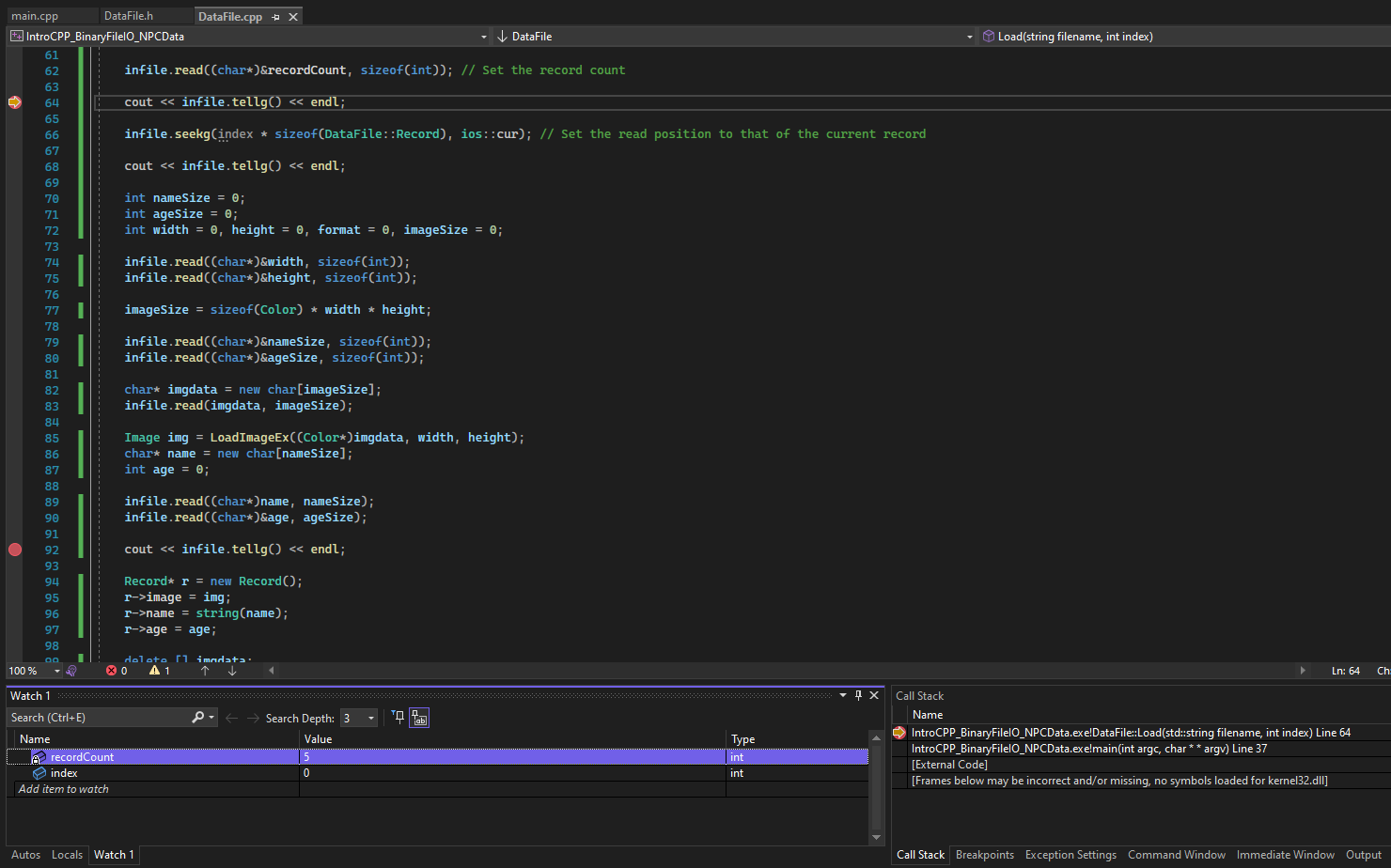


**Test 2 – Attempting name display fix**

* Input
  + Name is pulled from the record to be displayed onscreen
* Expected Output
  + Name appears in the correct spot onscreen when a record is loaded
* Actual Output
  + Name appears onscreen in the correct spot but has “junk” characters at its end
* Changes Made
  + A new char array is made using the name string from the record by only taking chars with letter values. This char array is passed into the name display code instead of the record name
* Updated Output
  + Name appears onscreen in the correct spot without any extra characters attached

**Test 3 – Iterating through entries in the database**

* Input
  + Press left or right arrow
* Expected Output
  + Iterates through all records, stopping once it reaches the first and last entries
* Actual Output
  + Iterates correctly through records, but there is an extra empty profile at the end
* Changes Made
  + The program was checking the index against the total record count. Realised that the index starts at 0, so the issue was solved by reducing the maximum value of the index by one
* Updated Output
  + Iterates through all records, stopping once it reaches both the first and last entries with no incorrect or empty profiles

Here’s some proof that I understand how to effectively use Visual Studio.

The screen capture above shows:

* Breakpoints I was using to figure out the size of each record in the database, as the output would be lost by errors caused by code I had not fixed yet.
* The ‘recordCount’ and ‘index’ variables in the Watch, allowing me to keep track of them during my first implementation of the seekg() function.
* The callstack, which showed me which functions the program was calling on startup.

Some keyboard shortcuts you can use while debugging with breakpoints are:

* Step into (F11), which will run the next line in the program,
* Step over (F10), which will run the next line in the program but will ignore any operation calls such as functions,
* and Step out (Shift + F11), which will run the program until the current operation is completed.

Hope this fix was what you wanted!

Thanks,

Halen