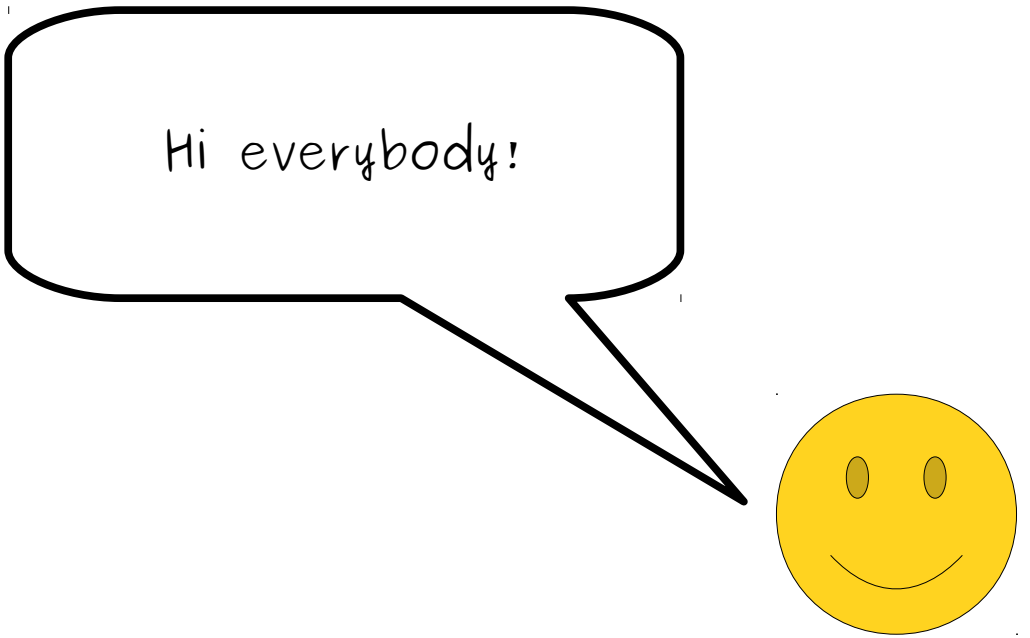
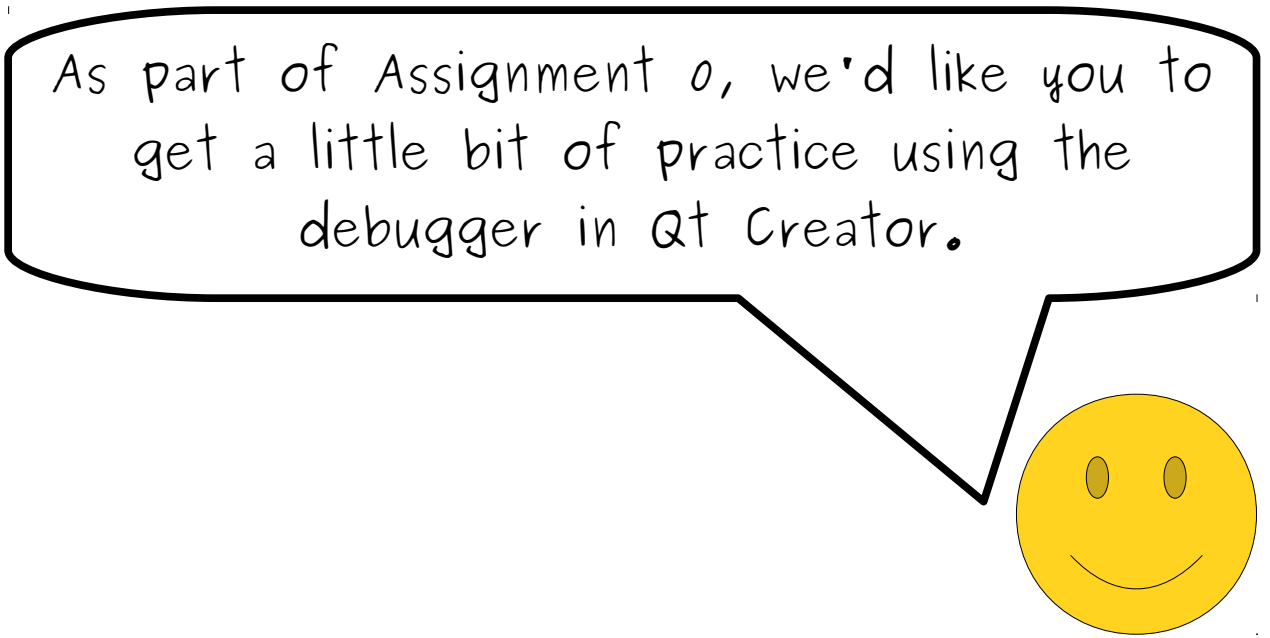
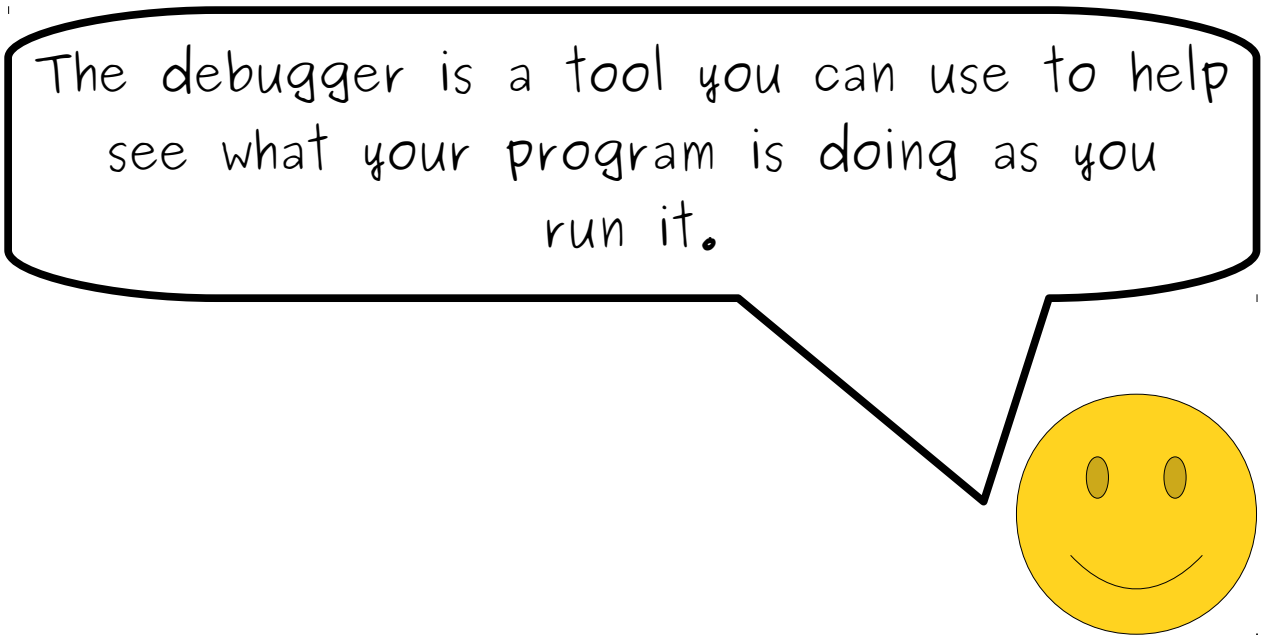
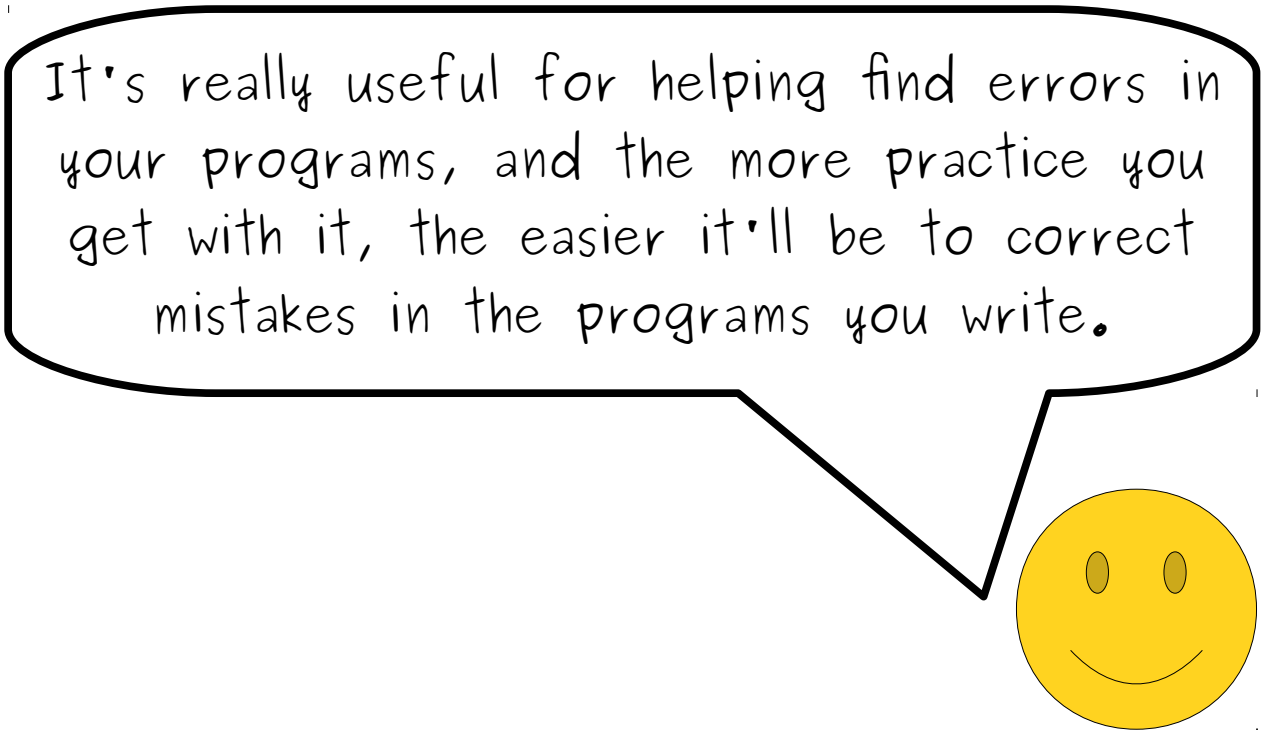
Assignment 0: Using the Debugger

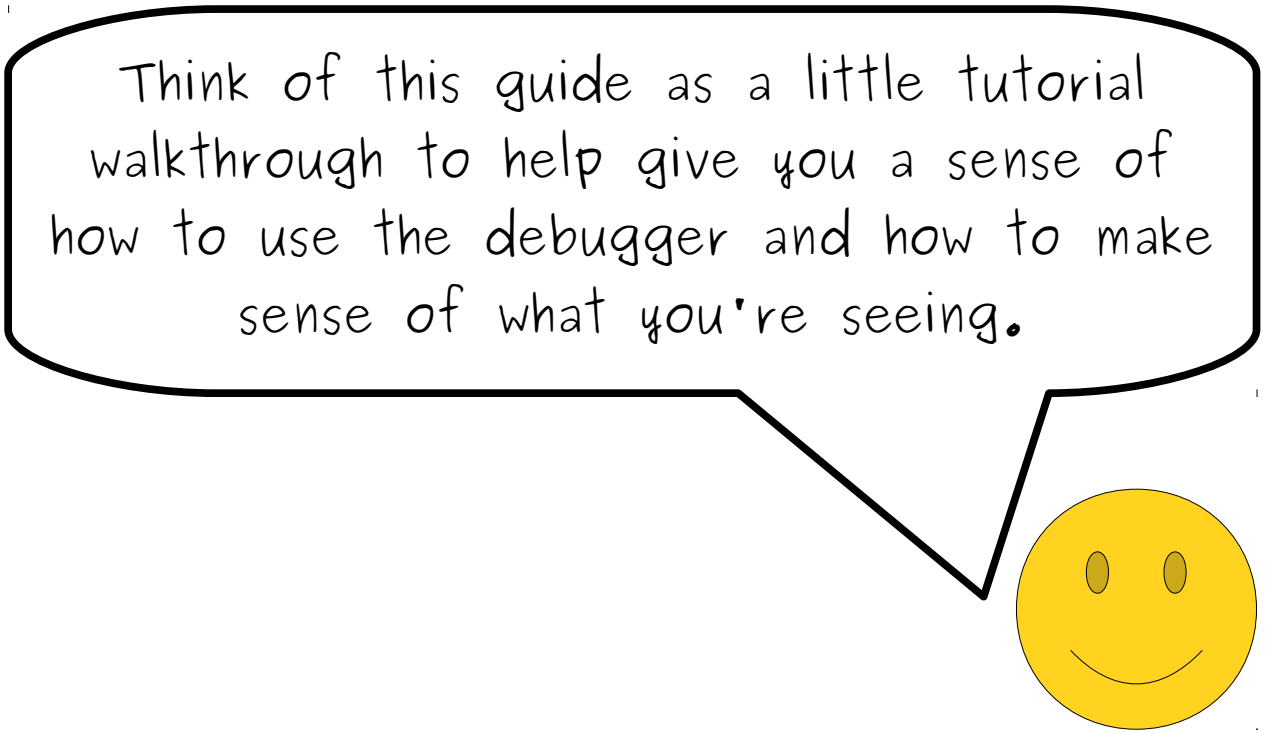
This assignment was written by Keith Schwartz. Thanks, Keith!


Hi everybody!


As part of Assignment 0, we'd like you to get a little bit of practice using the debugger in Qt Creator.

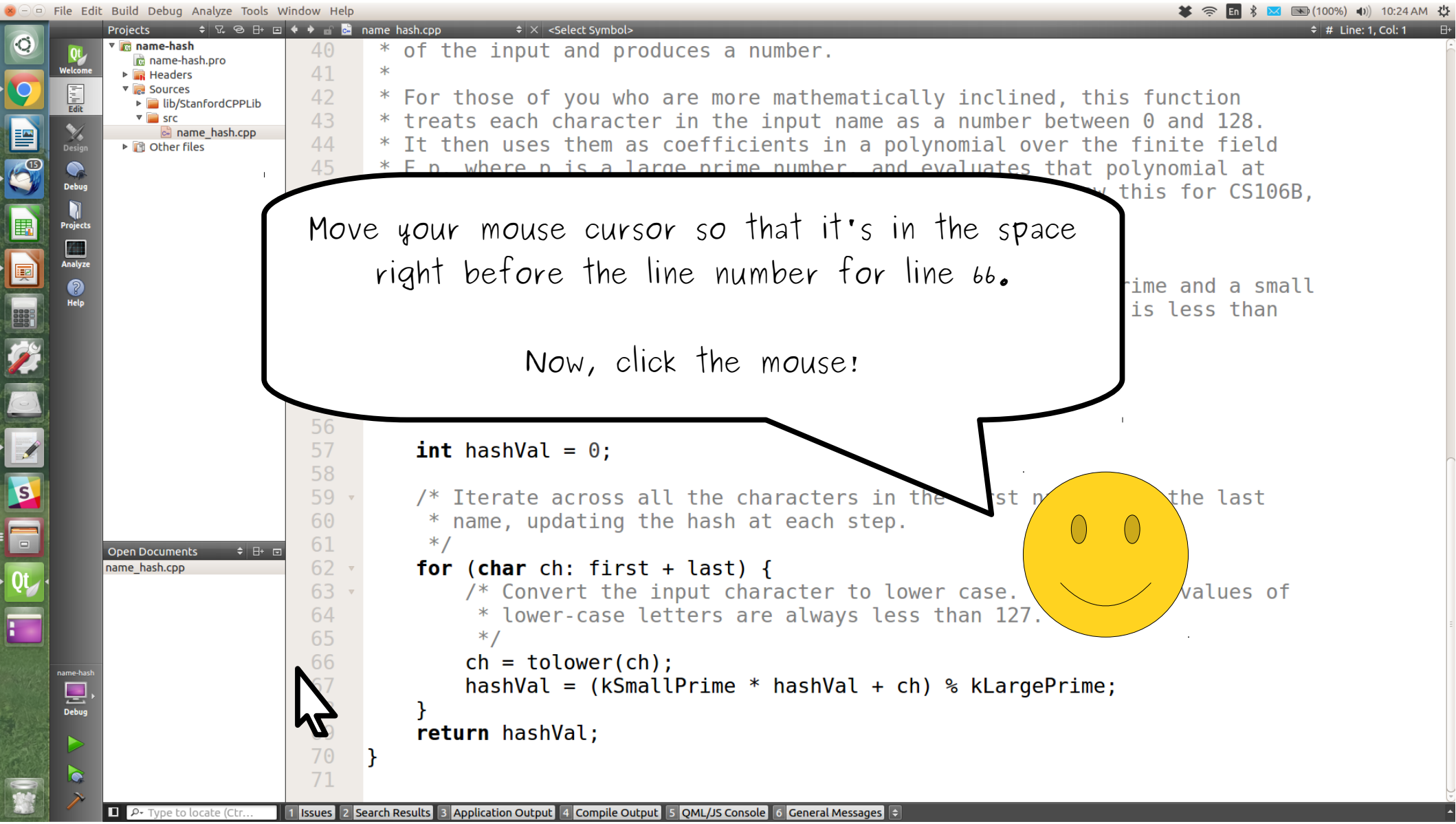

The debugger is a tool you can use to help see what your program is doing as you run it.

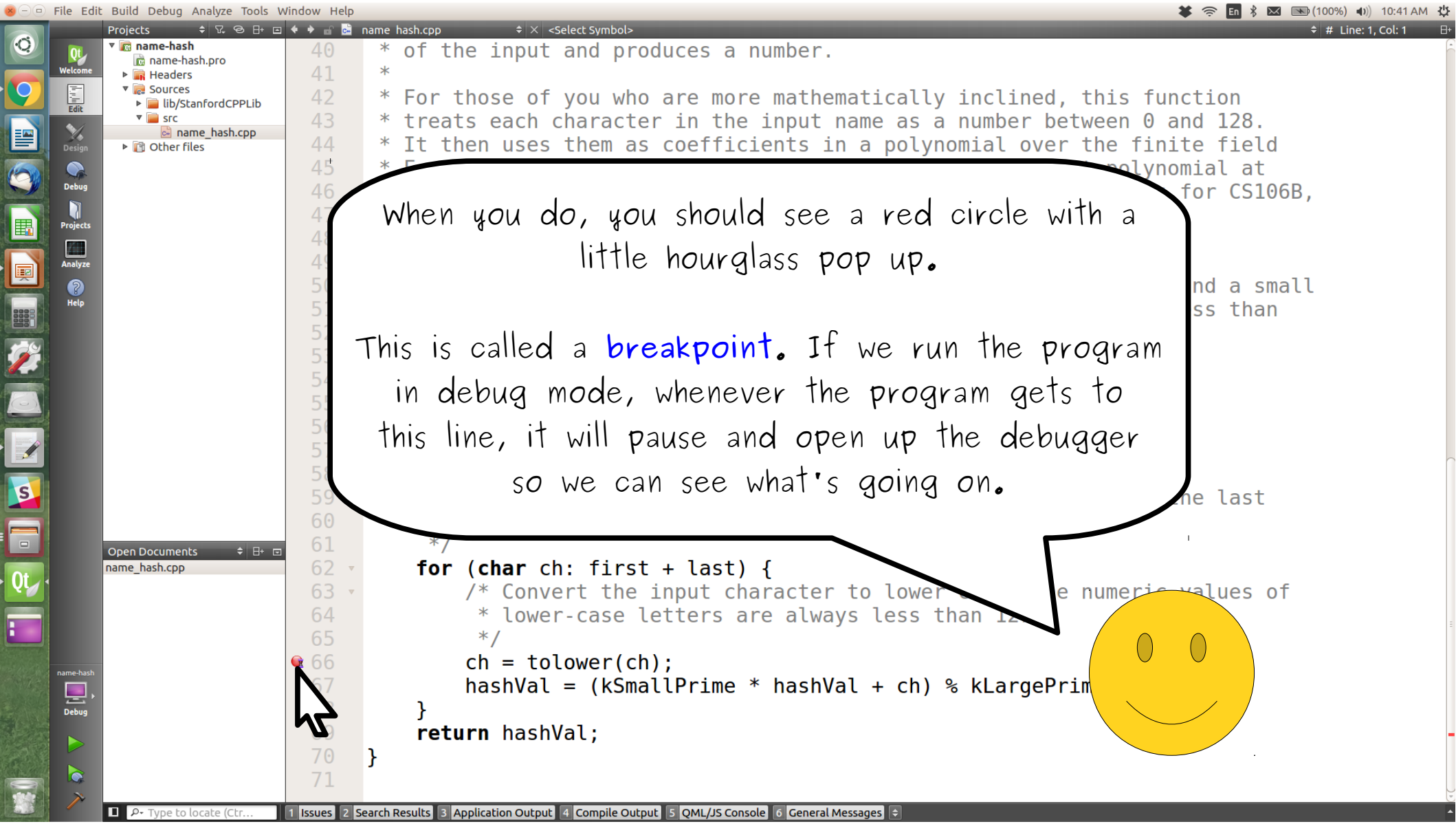

It's really useful for helping find errors in your programs, and the more practice you get with it, the easier it'll be to correct mistakes in the programs you write.

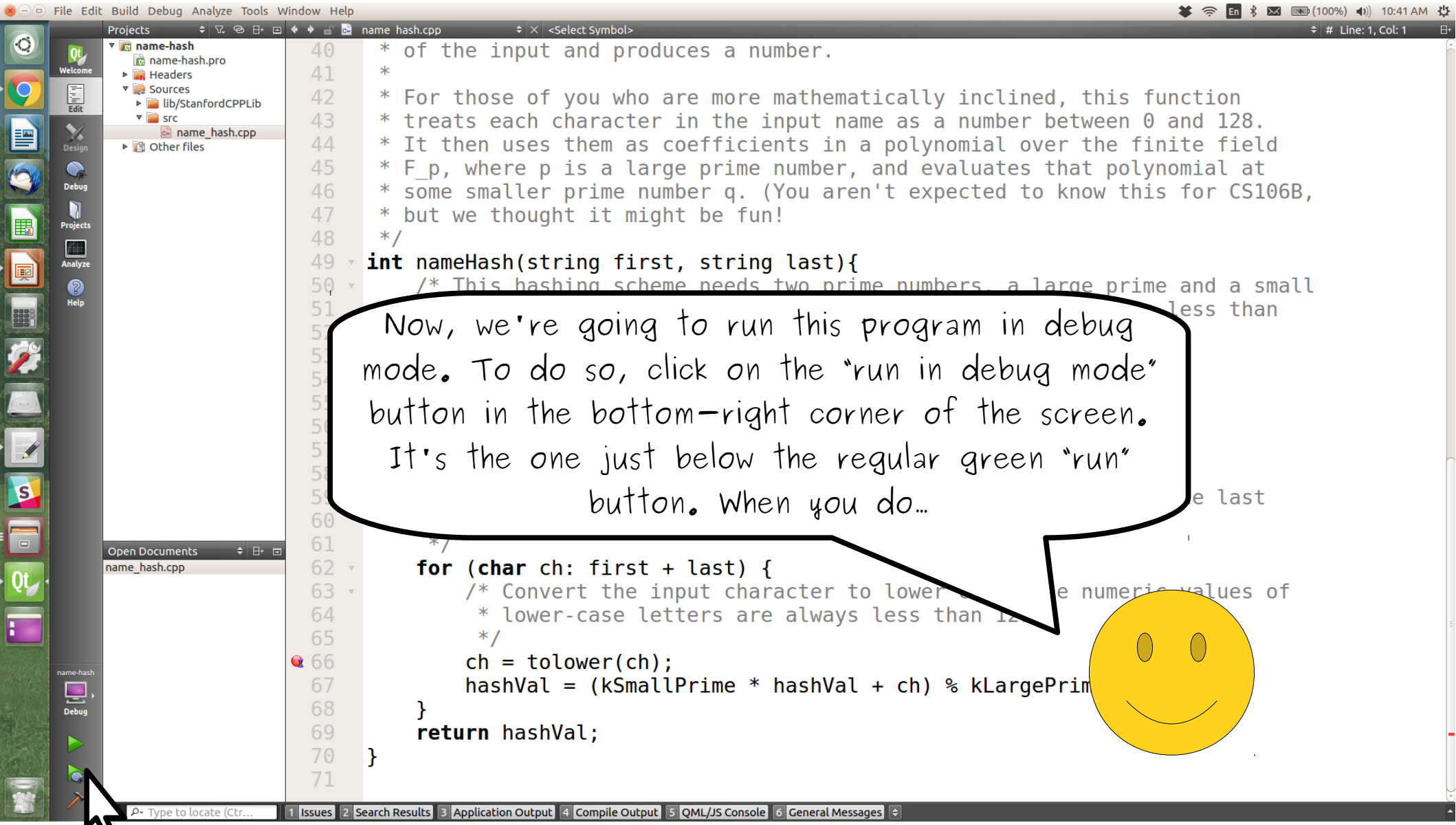

Think of this guide as a little tutorial walkthrough to help give you a sense of how to use the debugger and how to make sense of what you're seeing.

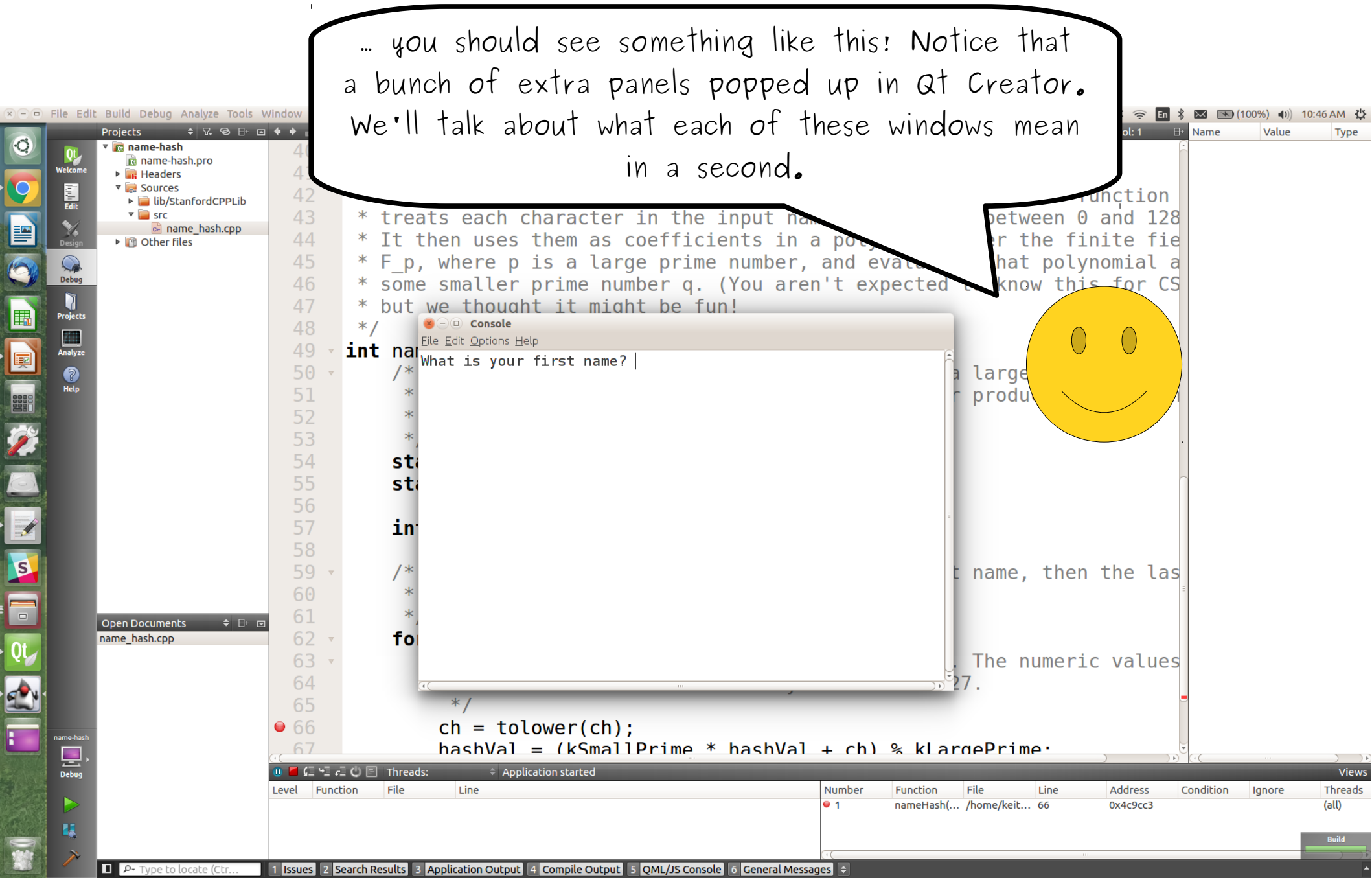
To start things off, open up the Name Hash program you ran in Part One of this assignment. Scroll down to the **nameHash** function so that you can see the entire function in your window.

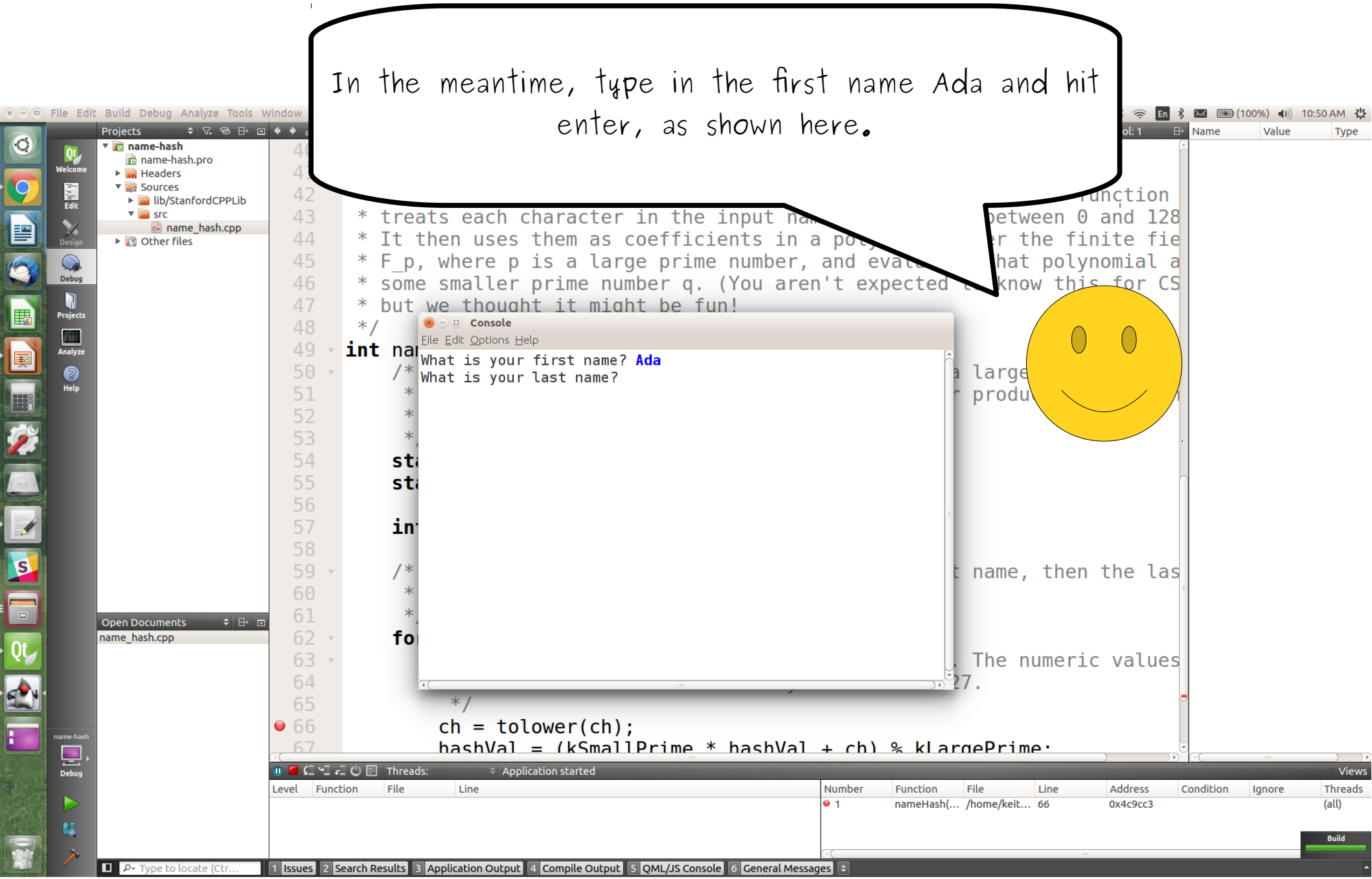
```
 * of the
 *
 * For tho
 * treats
 * It then
 * F_p, whe
 * some smaller
 * but we thought it might be fun!
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers, a large
     * prime. These numbers were chosen because their product
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the first name, then the last
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower case. The numeric values of
         * lower-case letters are always less than 127.
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
    return hashVal;
}
```


Move your mouse cursor so that it's in the space right before the line number for line 66.
Now, click the mouse!

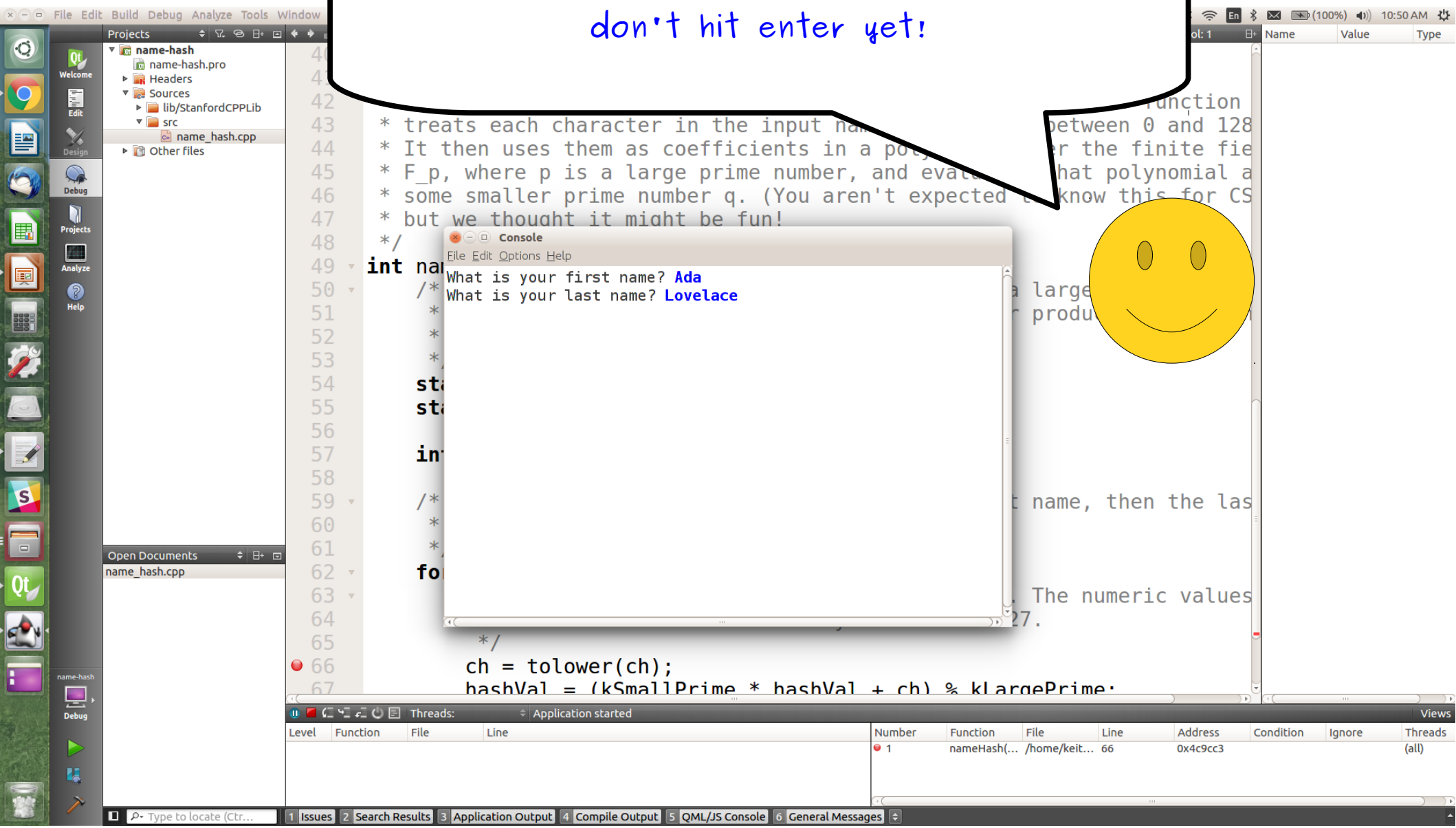

When you do, you should see a red circle with a little hourglass pop up.
This is called a breakpoint. If we run the program in debug mode, whenever the program gets to this line, it will pause and open up the debugger so we can see what's going on.


Now, we're going to run this program in debug mode. To do so, click on the "run in debug mode" button in the bottom-right corner of the screen. It's the one just below the regular green "run" button. When you do…

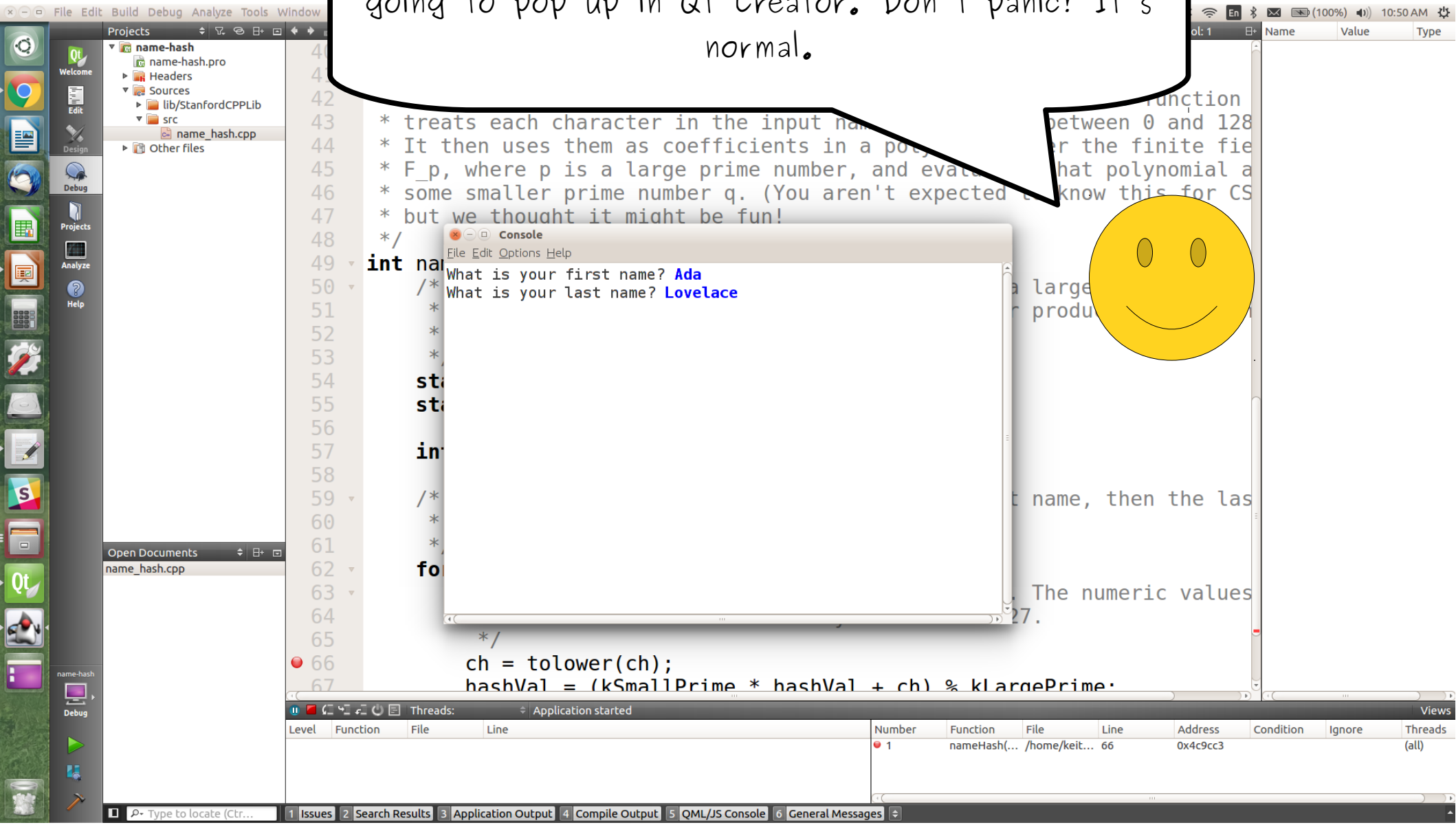

… you should see something like this! Notice that a bunch of extra panels popped up in Qt Creator. We'll talk about what each of these windows mean in a second.
Console
File Edit Options Help
What is your first name?
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
Threads:
Application started
Level
Function
File
Line
Number
Function
File
Line
Address
Condition
Ignore
Threads
1
nameHash(...
/home/keit...
66
0x4c9cc3
(all)
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages


In the meantime, type in the first name Ada and hit enter, as shown here.
Console
File Edit Options Help
What is your first name? Ada
What is your last name?

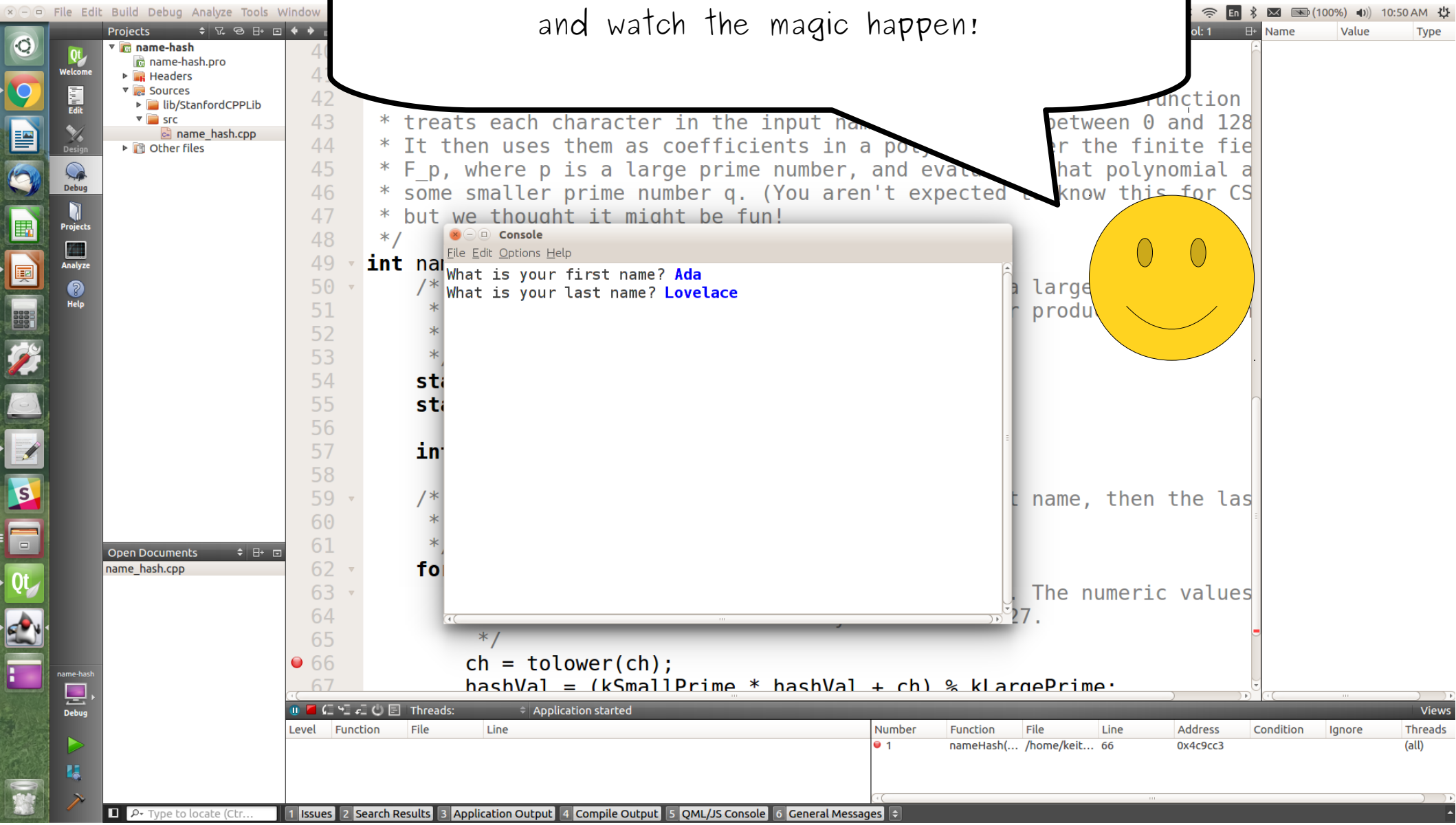
Now, type in "Lovelace" as a last name, but don't hit enter yet!



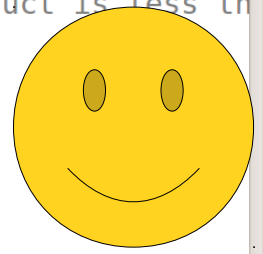
As soon as you hit enter, a bunch of things are going to pop up in Qt Creator. Don't panic! It's normal.



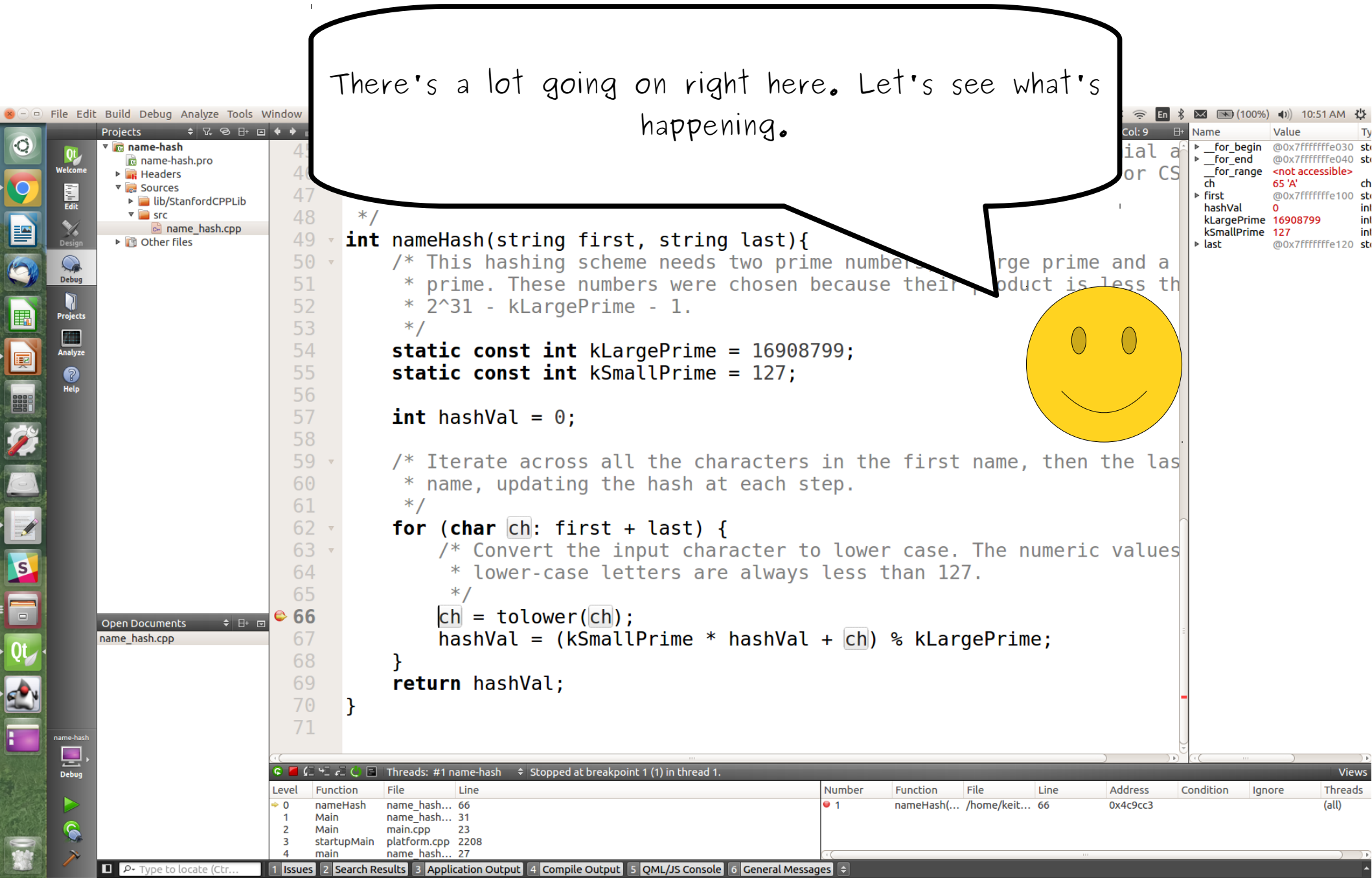
With that said, hit enter,
and watch the magic happen!

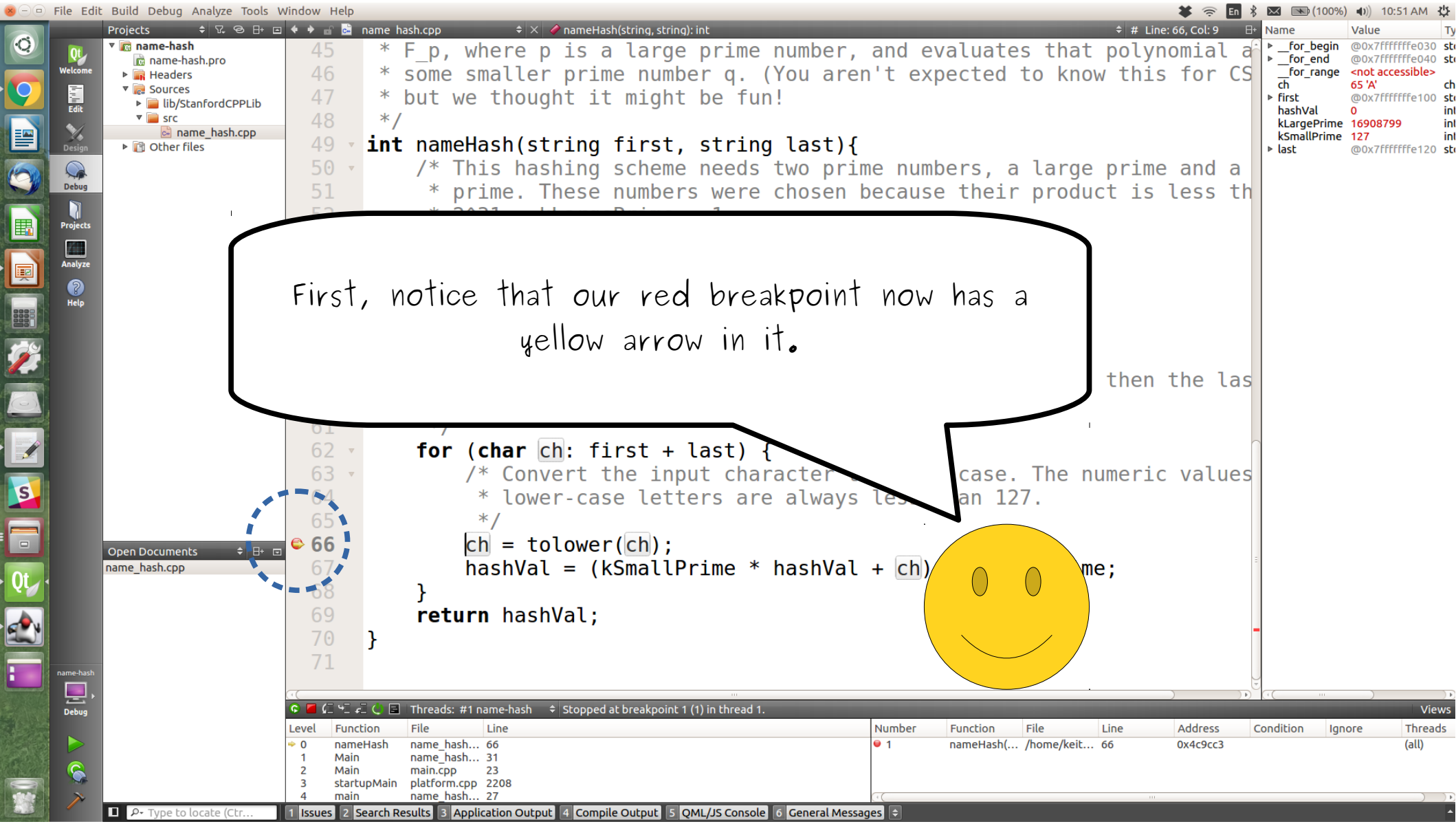


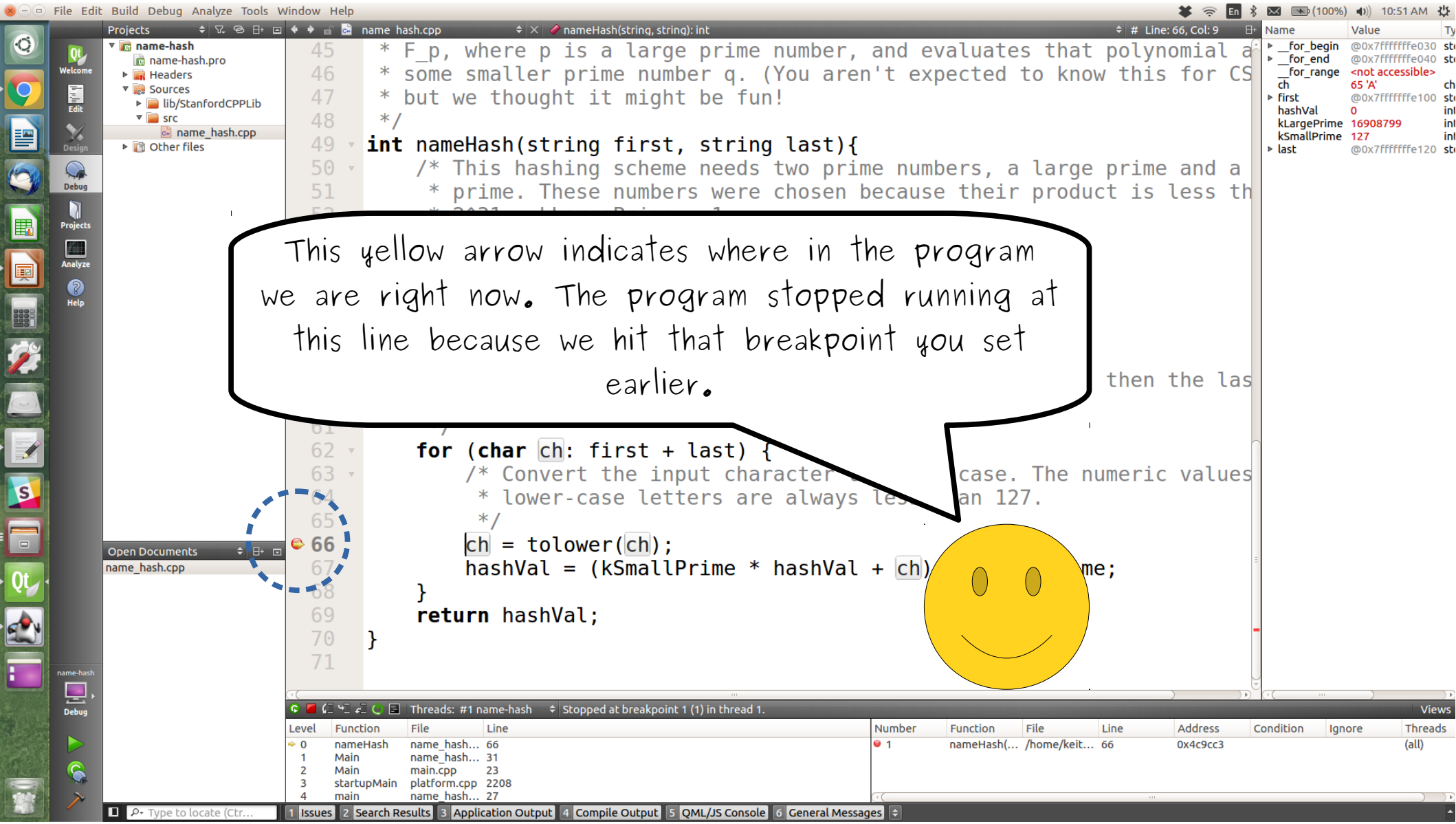
Shazam! We're back in Qt Creator, and there's tons of values showing up everywhere.

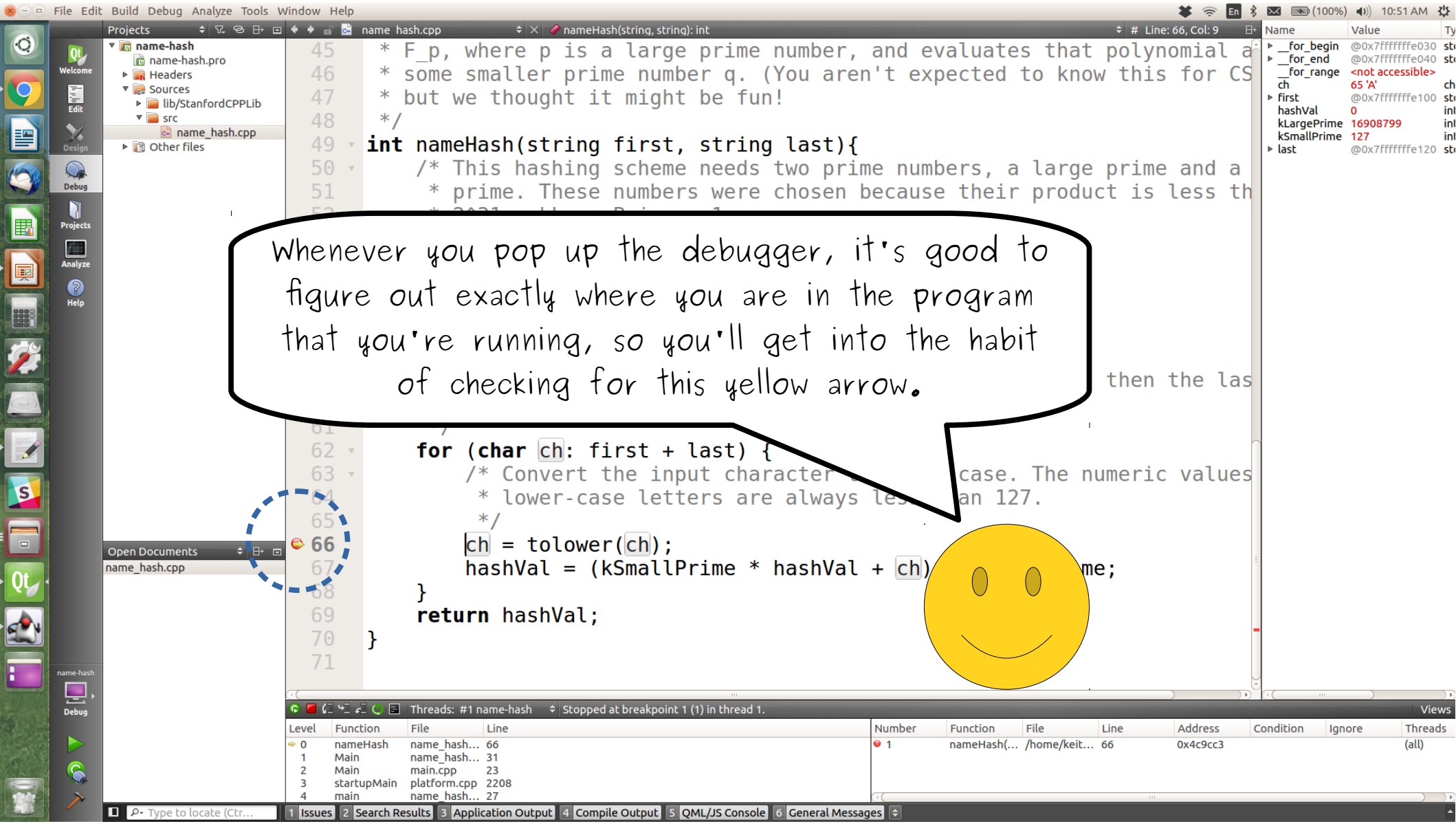


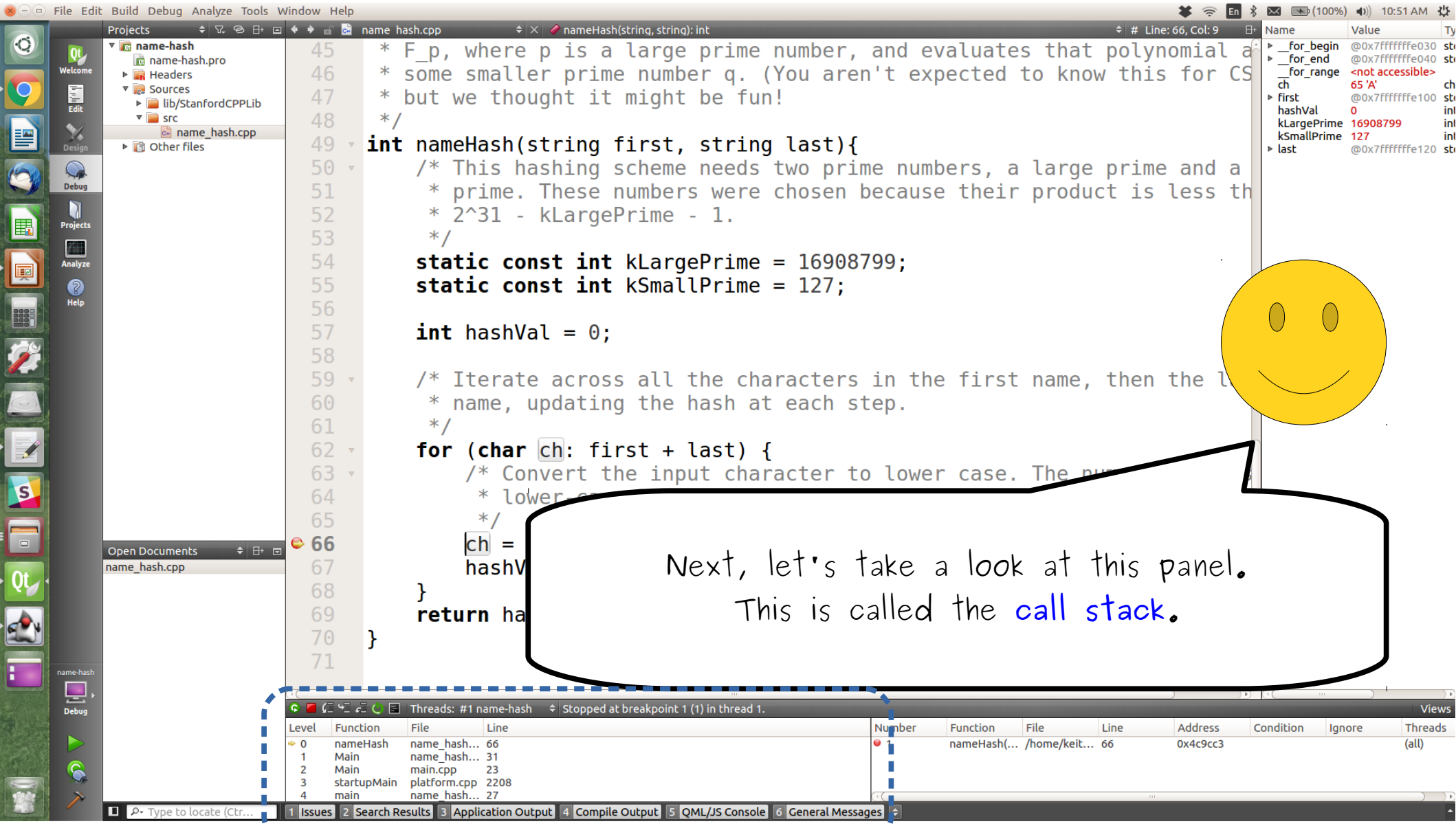
```
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers, a large prime and a
     * prime. These numbers were chosen because their product is less th
     * 2^31 - kLargePrime - 1.
     */
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the first name, then the las
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower case. The numeric values
         * lower-case letters are always less than 127.
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
    return hashVal;
}
```

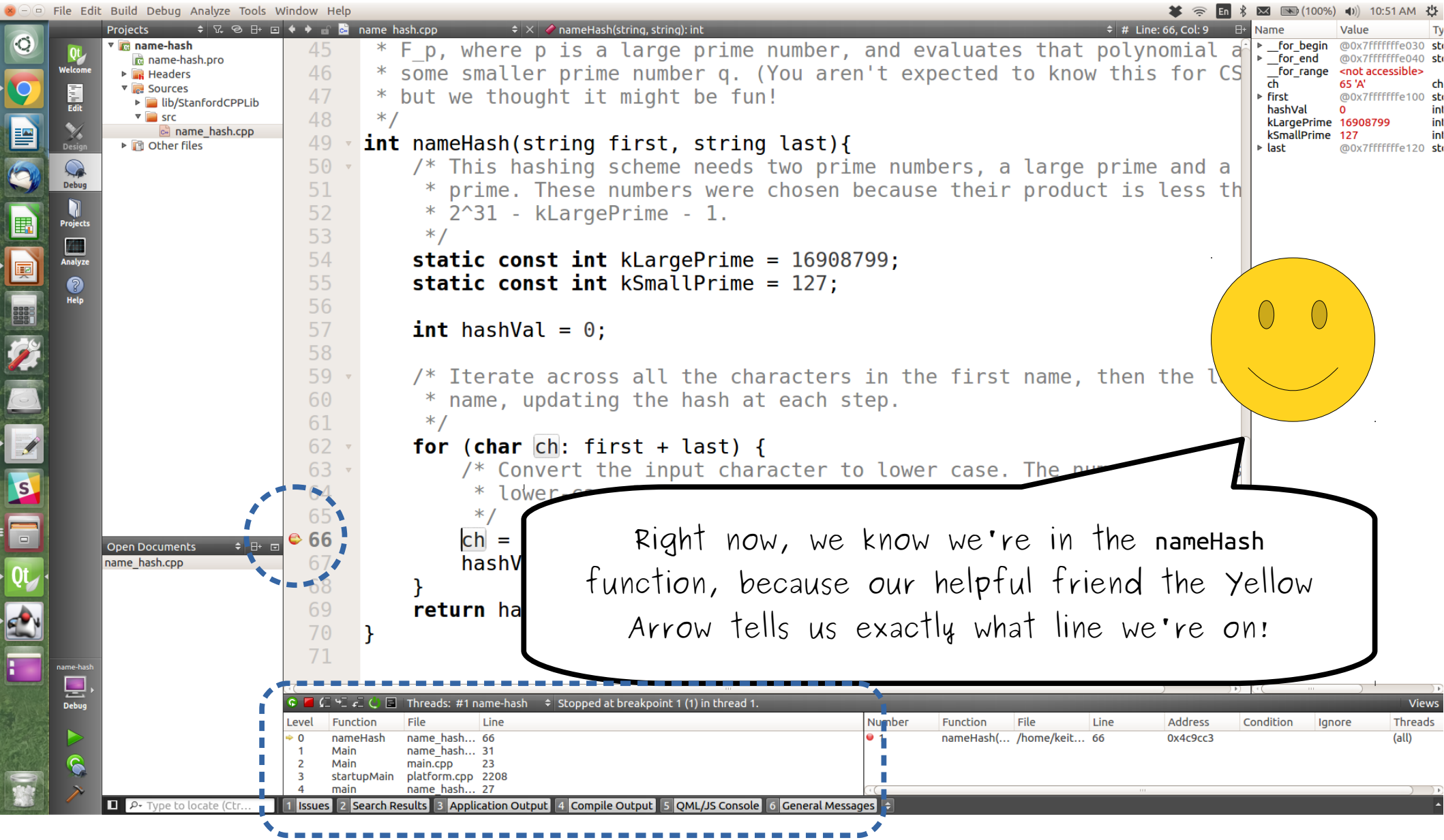

There's a lot going on right here. Let's see what's happening.
File Edit Build Debug Analyze Tools Window
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers
* prime. These numbers were chosen because their
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
Number
Address
Condition
Ignore
Threads
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages

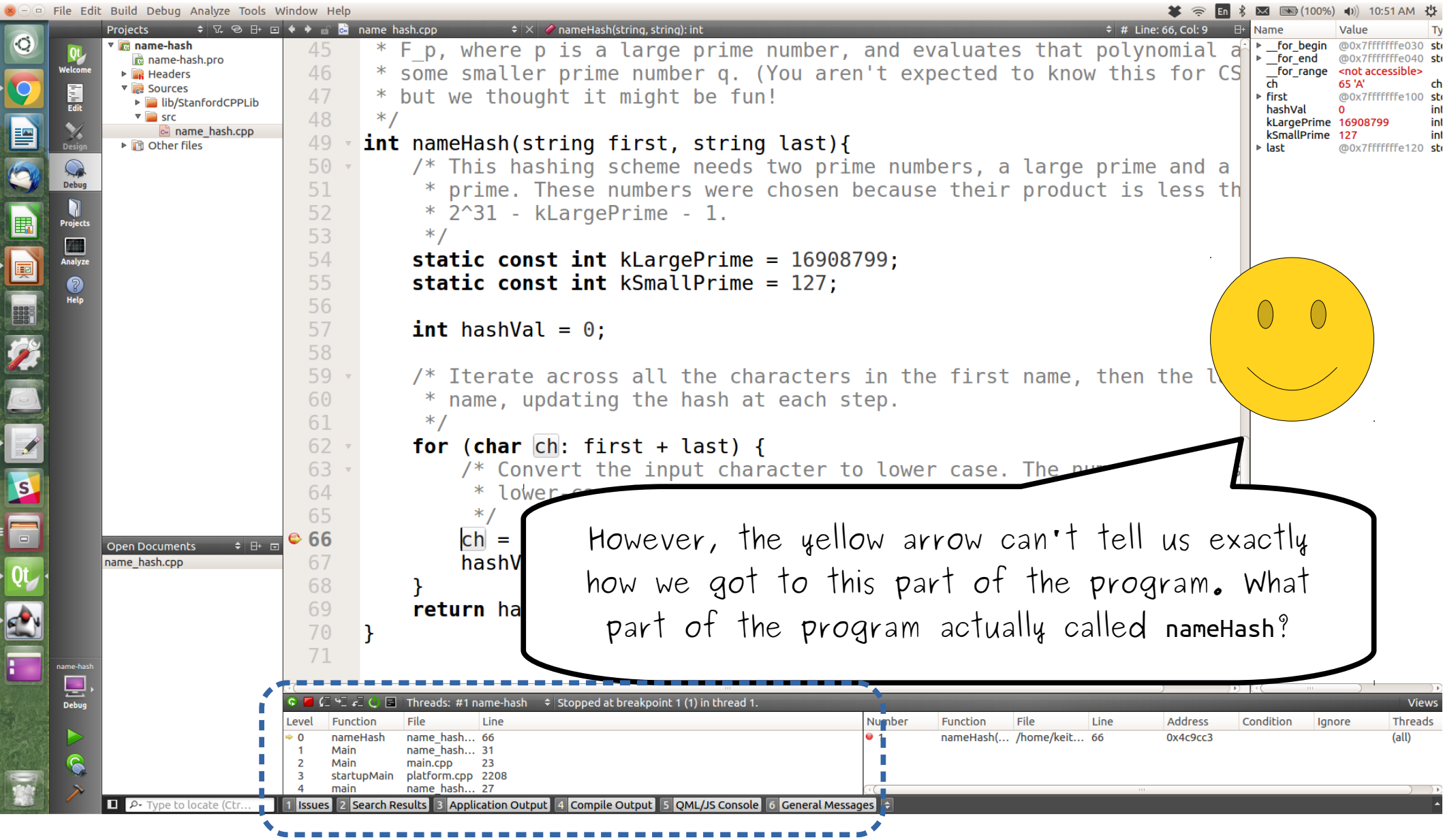

First, notice that our red breakpoint now has a yellow arrow in it.

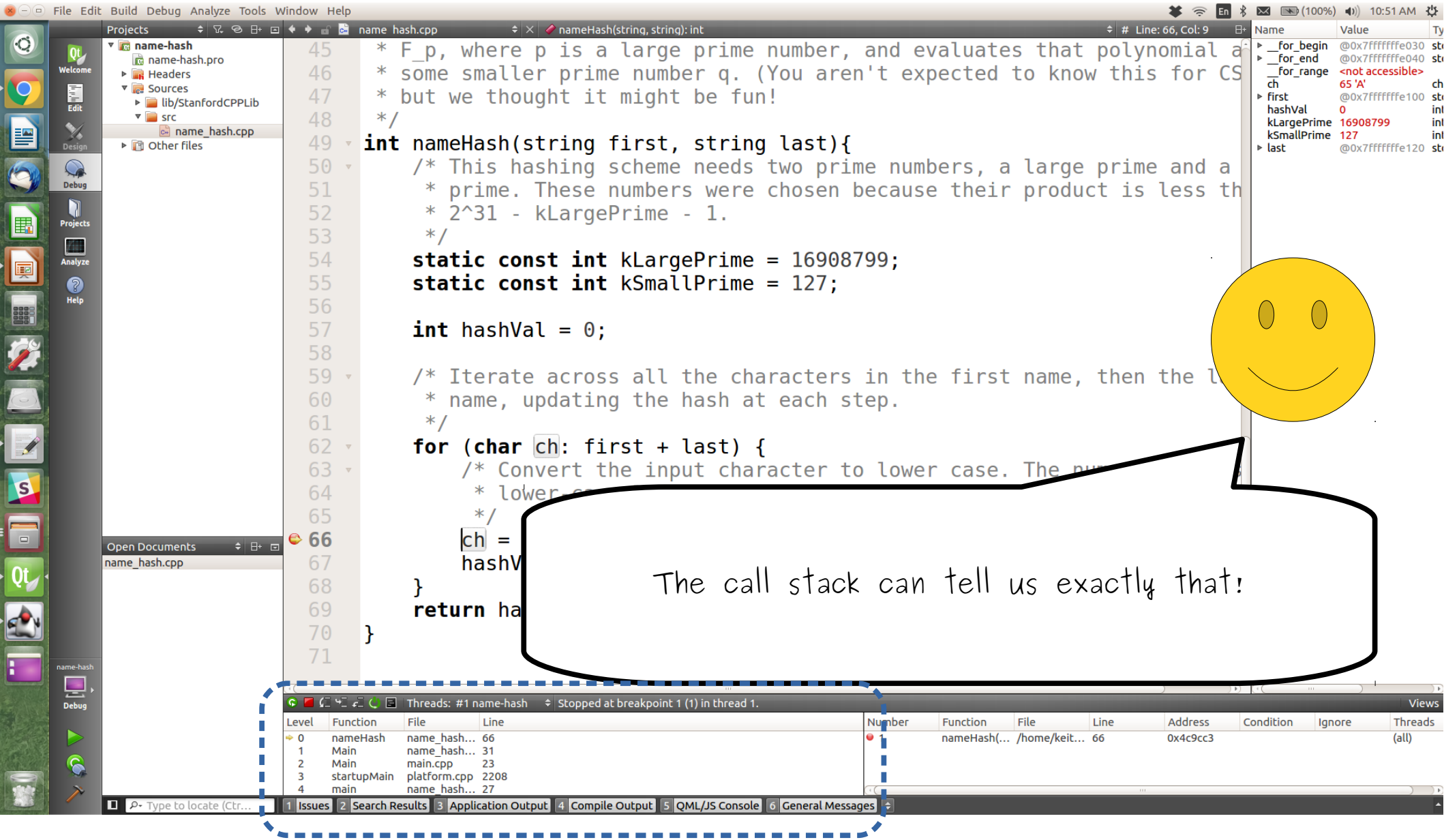

This yellow arrow indicates where in the program we are right now. The program stopped running at this line because we hit that breakpoint you set earlier.

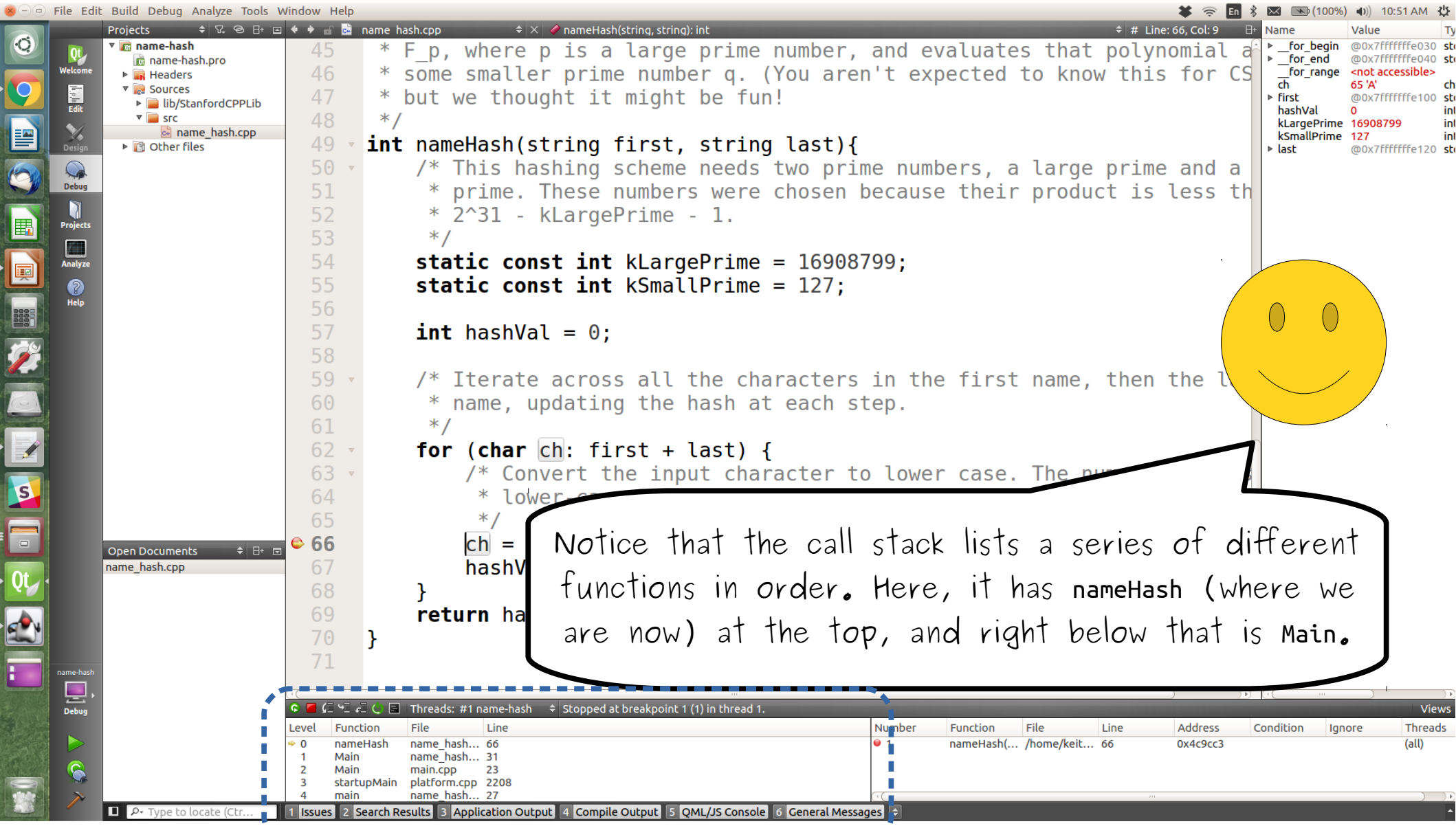

Whenever you pop up the debugger, it's good to figure out exactly where you are in the program that you're running, so you'll get into the habit of checking for this yellow arrow.

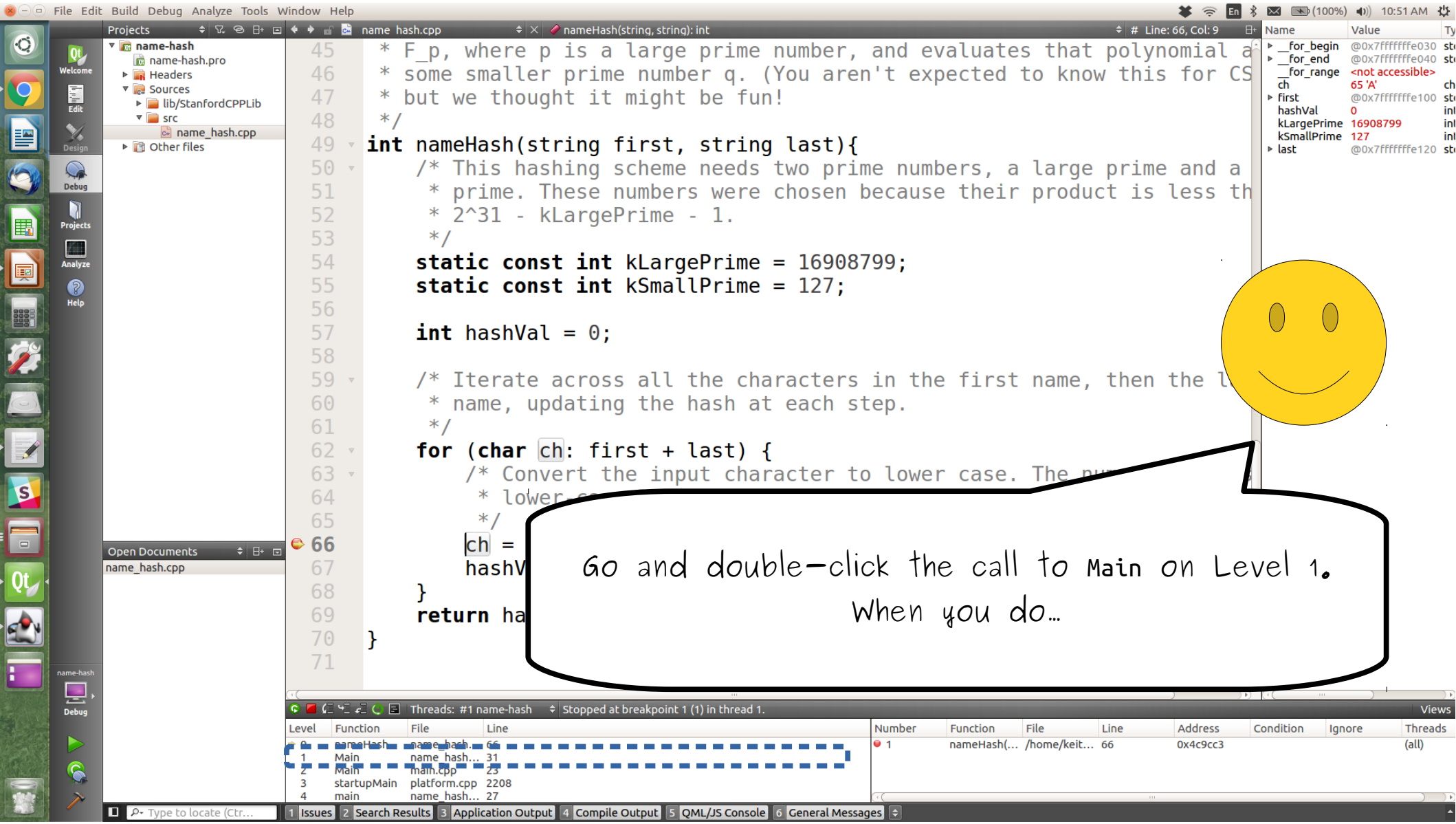

Next, let's take a look at this panel.
This is called the call stack.

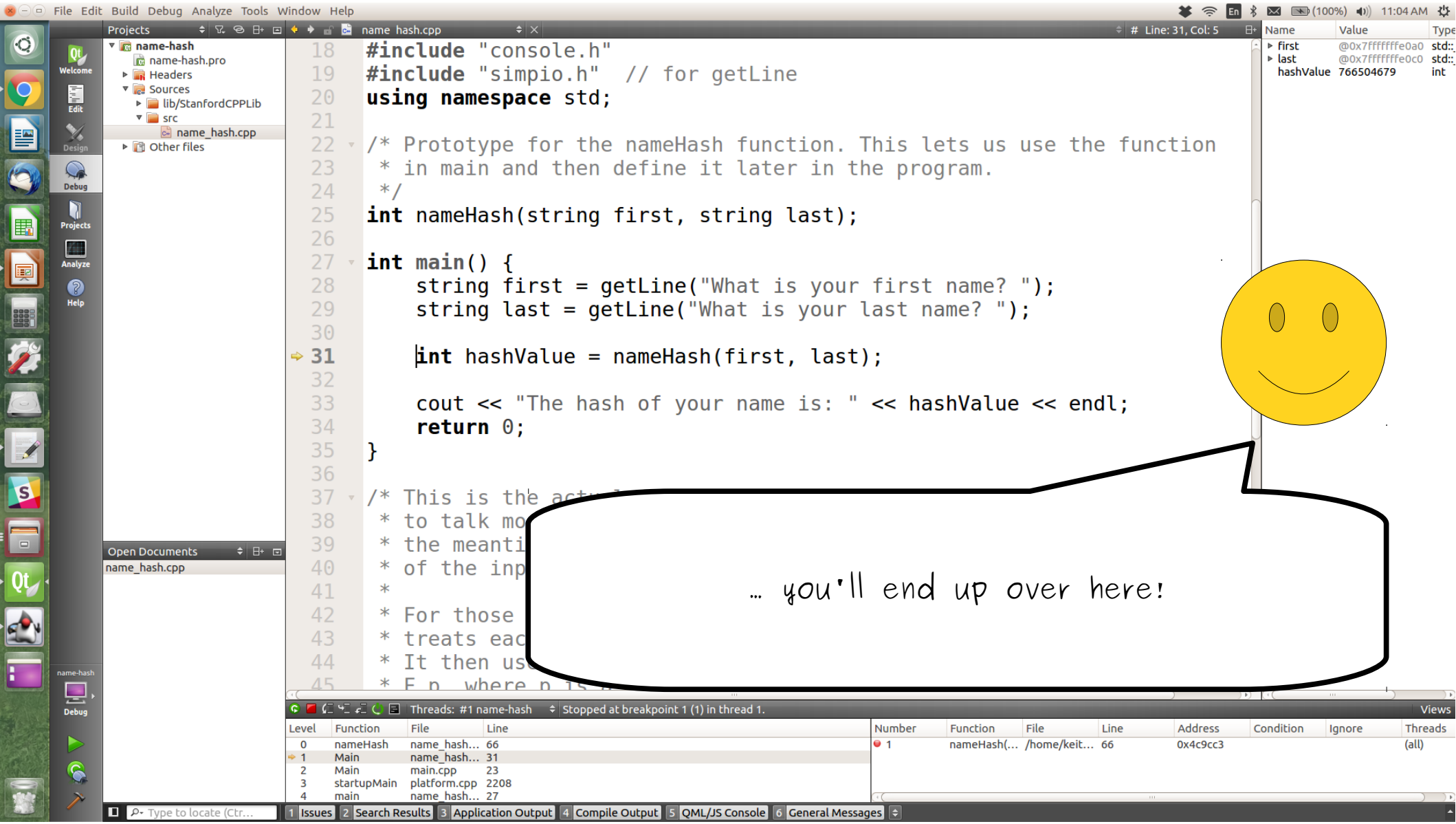

Right now, we know we're in the nameHash function, because our helpful friend the Yellow Arrow tells us exactly what line we're on!

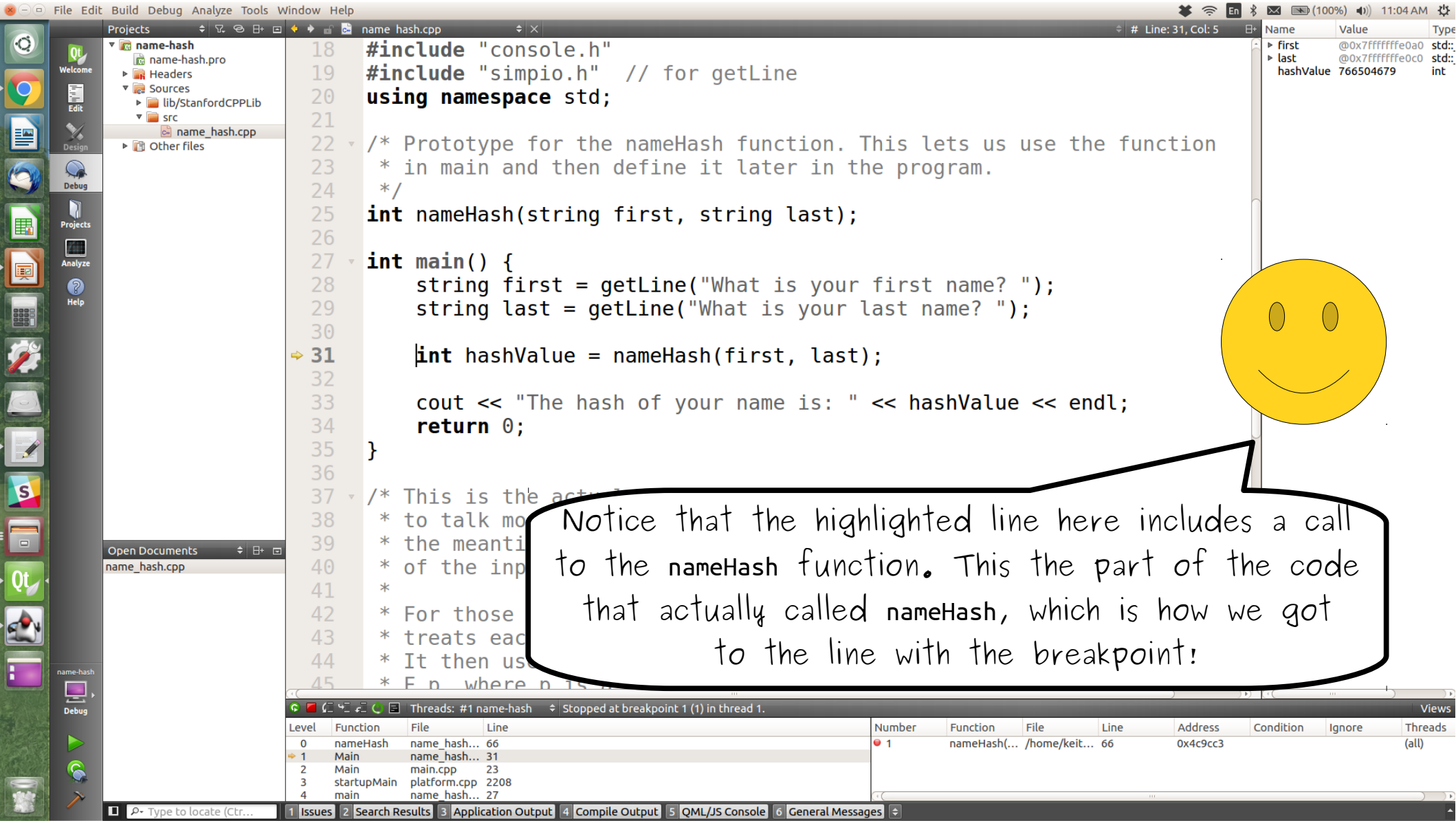

However, the yellow arrow can't tell us exactly how we got to this part of the program. What part of the program actually called nameHash?

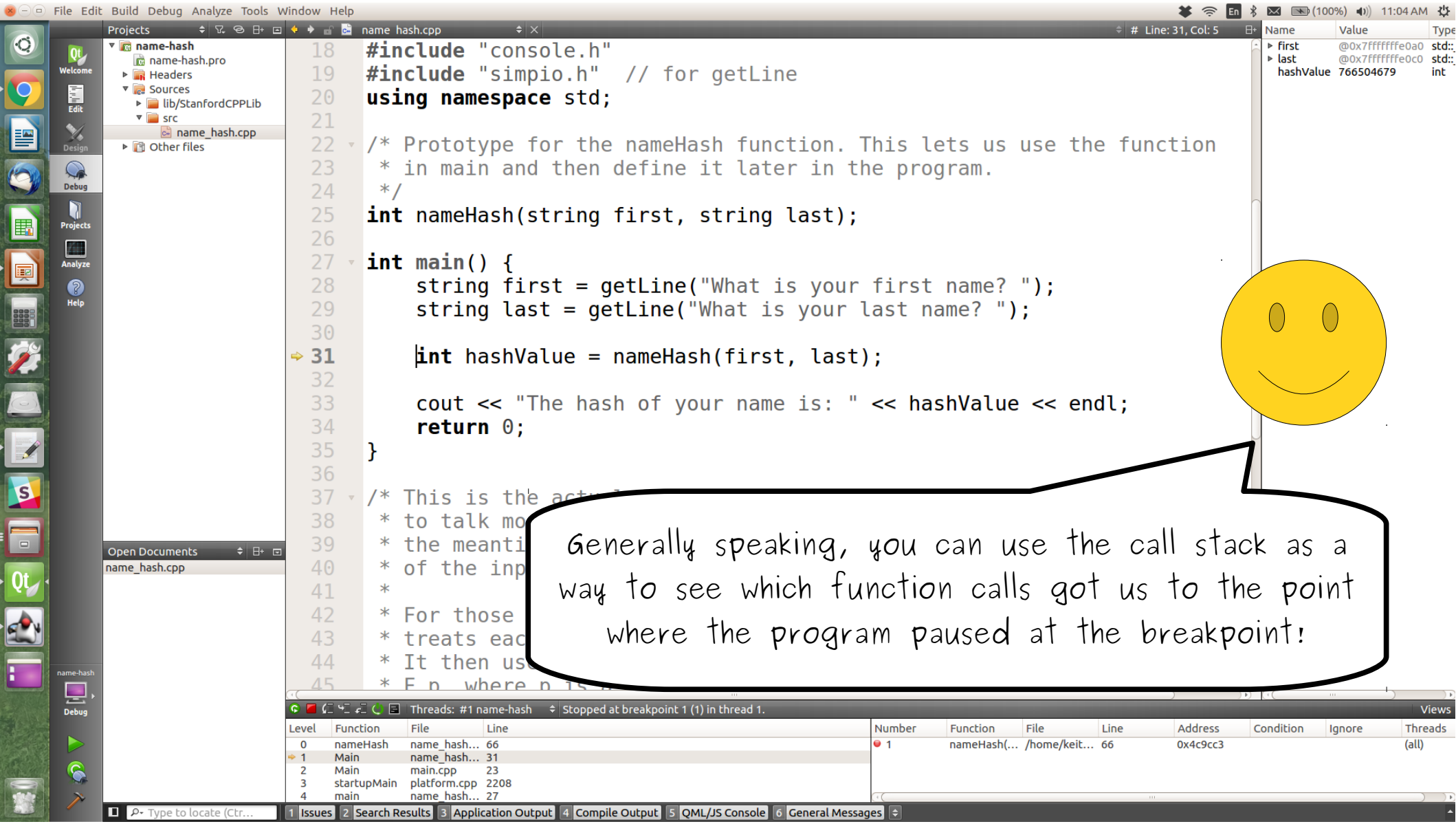

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the l
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The nu
*
*/
ch =
hashV
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
The call stack can tell us exactly that!
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

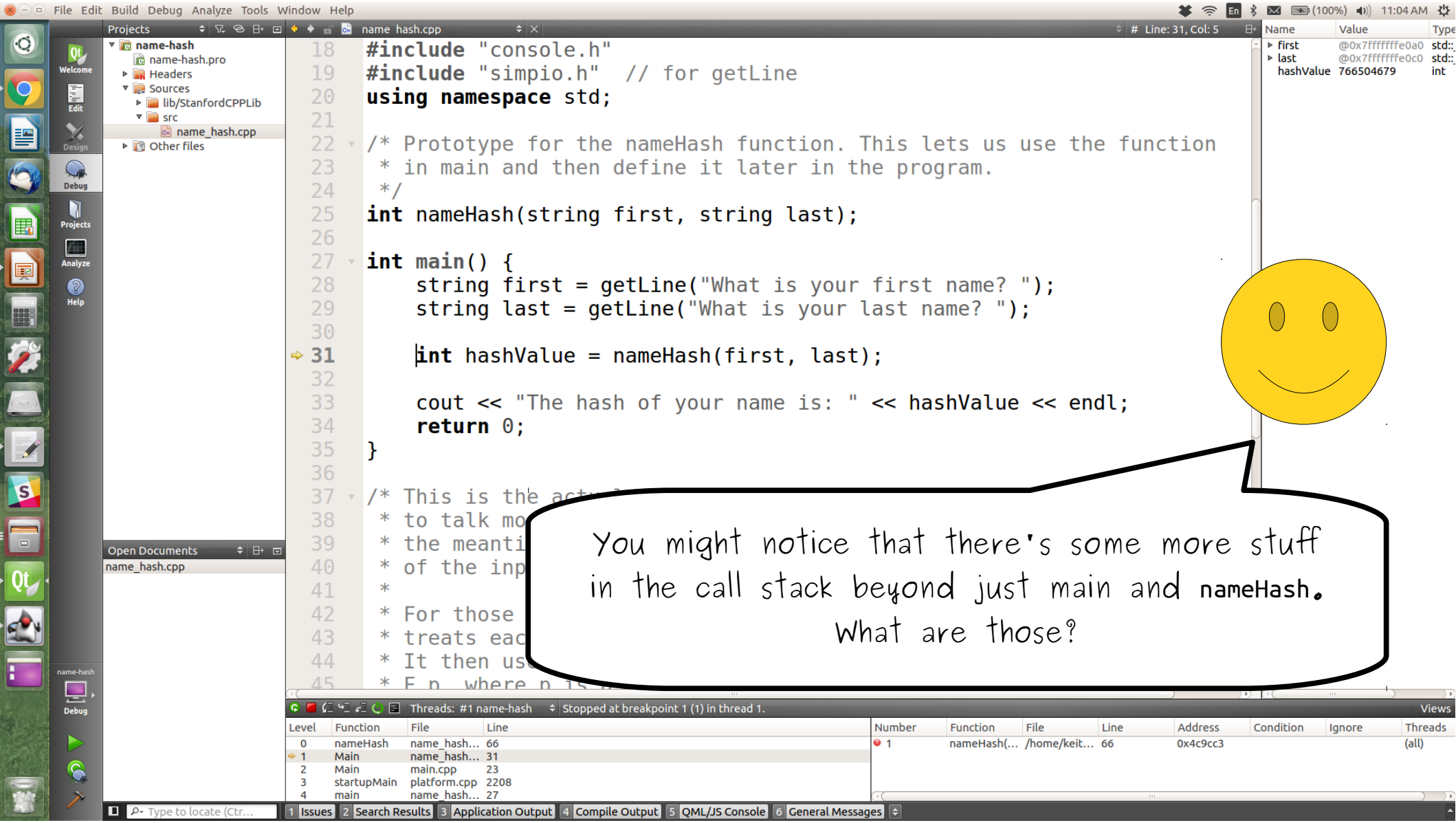

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the l
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The nu
* lower
*/
ch =
hashV
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Notice that the call stack lists a series of different functions in order. Here, it has nameHash (where we are now) at the top, and right below that is Main.
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

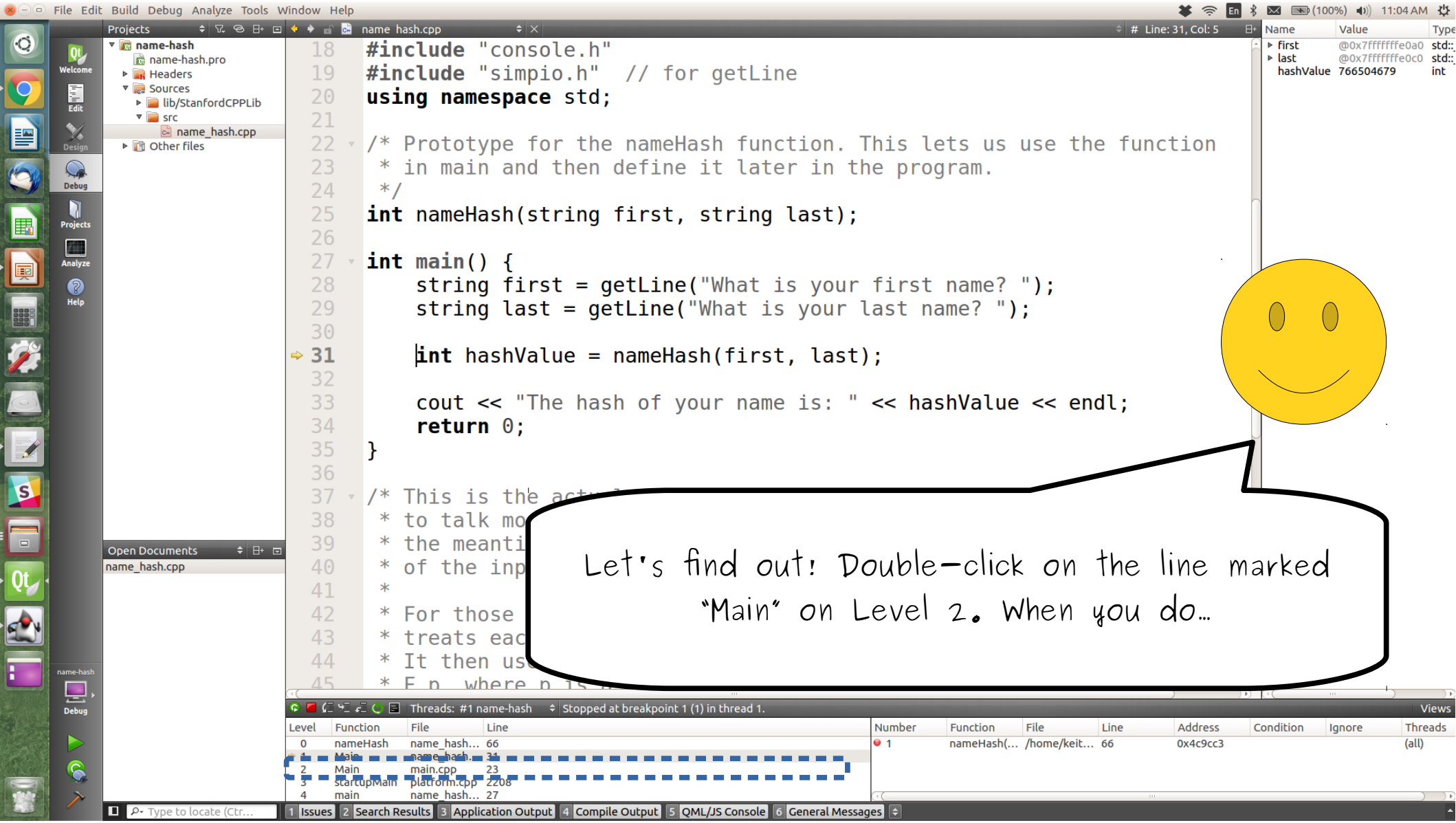

Go and double-click the call to Main on Level 1.
When you do…

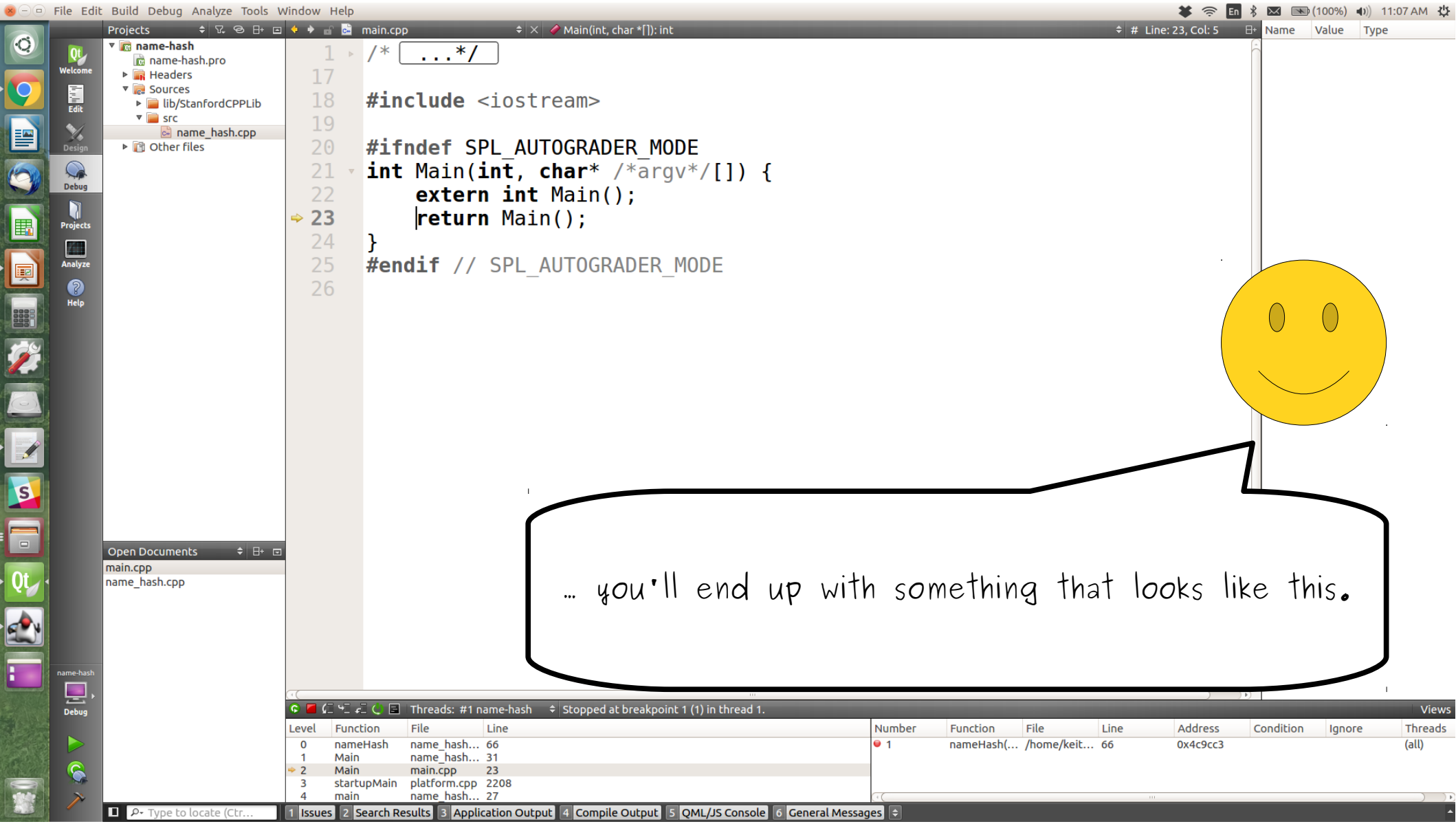

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name_hash.cpp
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Name
Value
Type
first
last
hashValue
766504679
int
… you'll end up over here!
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

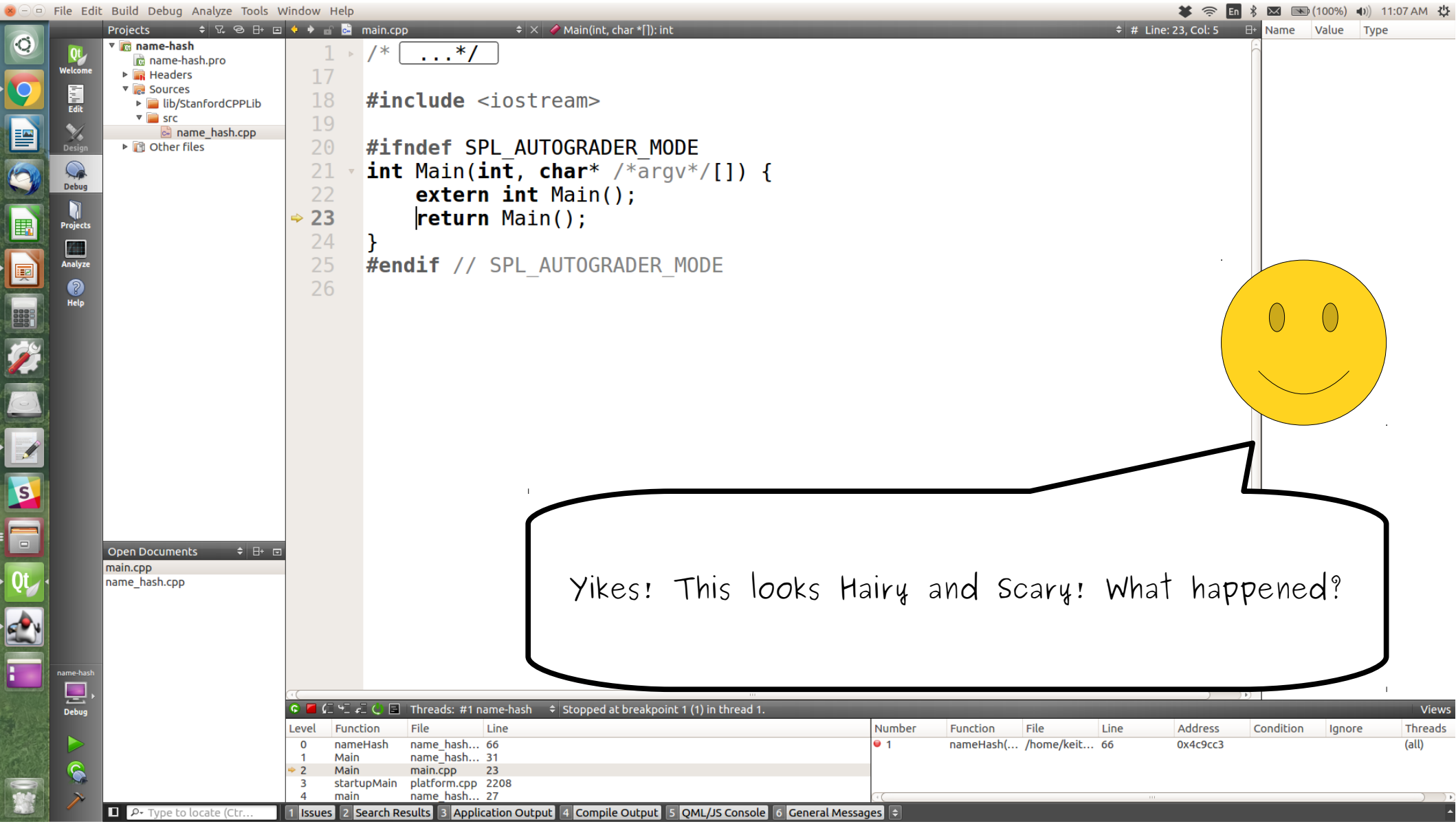

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name hash.cpp
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Name
Value
Type
first
last
hashValue
766504679
int
Notice that the highlighted line here includes a call to the nameHash function. This the part of the code that actually called nameHash, which is how we got to the line with the breakpoint!
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages

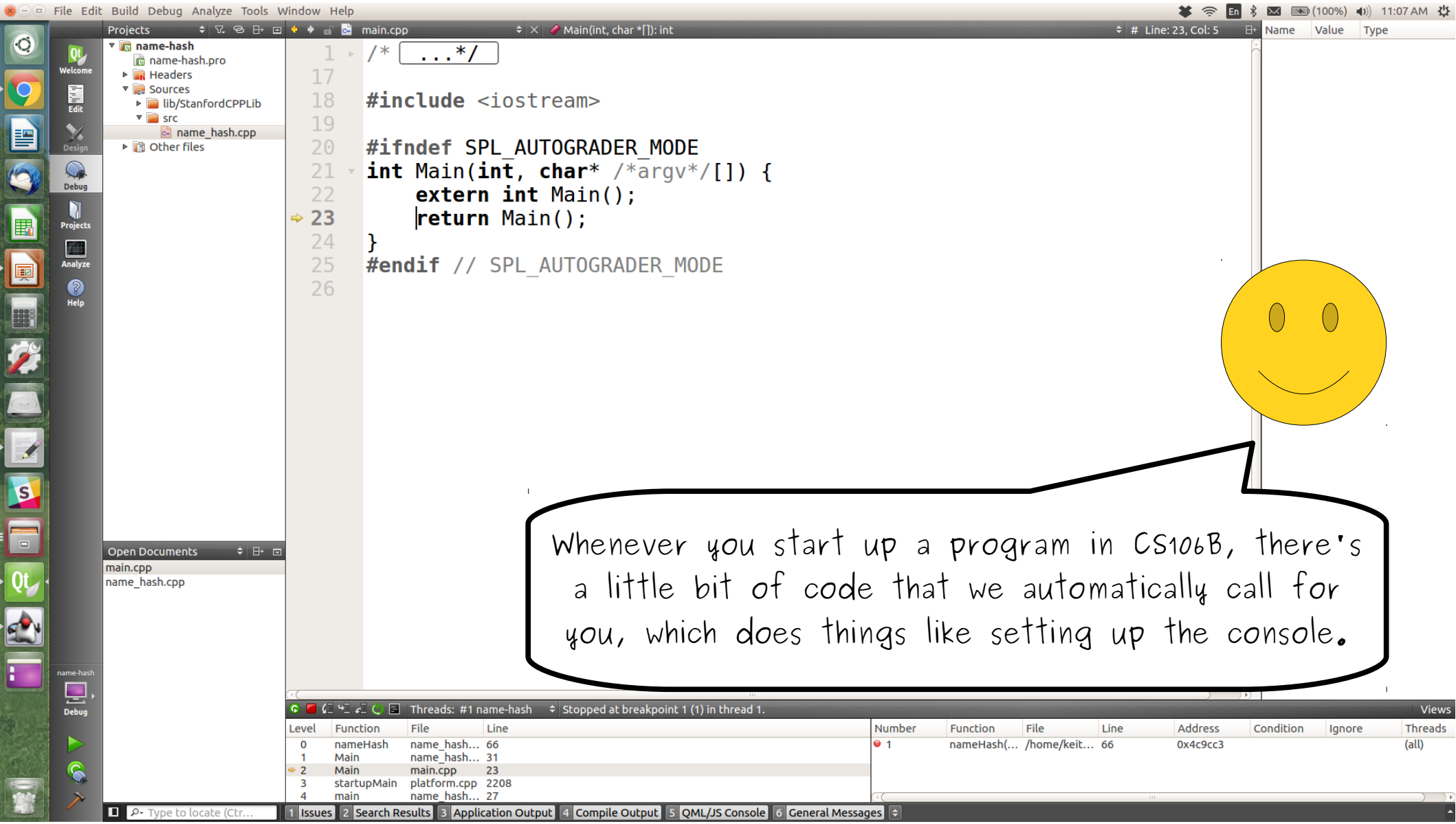

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name_hash.cpp
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Name
Value
Type
first
last
hashValue
766504679
int
Generally speaking, you can use the call stack as a way to see which function calls got us to the point where the program paused at the breakpoint!
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

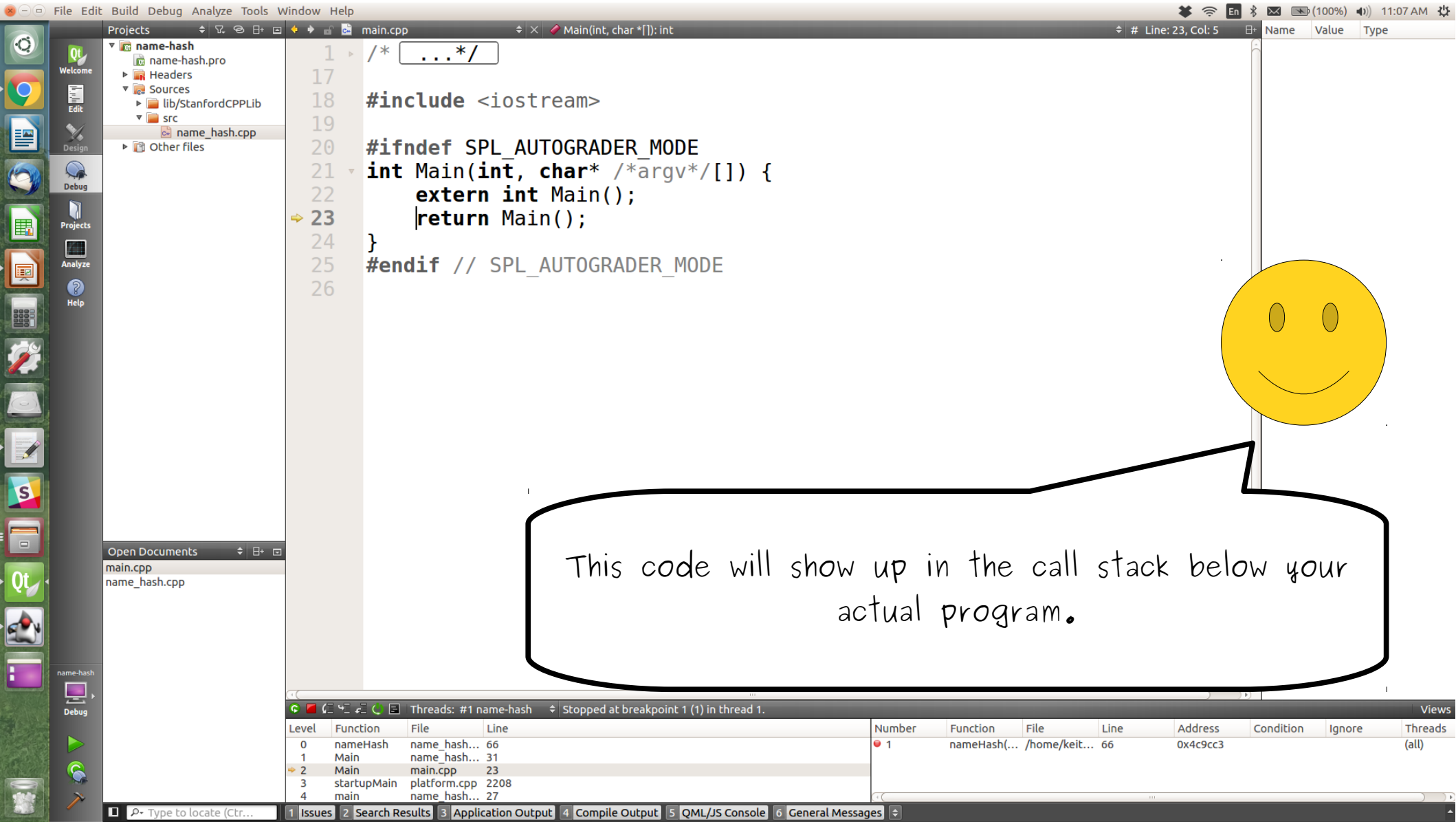

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name_hash.cpp
Line: 31, Col: 5
Name Value Type
first @0x7fffffffe0a0 std::
last @0x7fffffffe0c0 std::
hashValue 766504679 int
18 #include "console.h"
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the function
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28 string first = getLine("What is your first name? ");
29 string last = getLine("What is your last name? ");
30
31 int hashValue = nameHash(first, last);
32
33 cout << "The hash of your name is: " << hashValue << endl;
34 return 0;
35 }
36
37 /* This is the
38 * to talk mo
39 * the meanti
40 * of the inp
41 *
42 * For those
43 * treats eac
44 * It then us
45 * E.g. where n is
You might notice that there's some more stuff in the call stack beyond just main and nameHash. What are those?
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

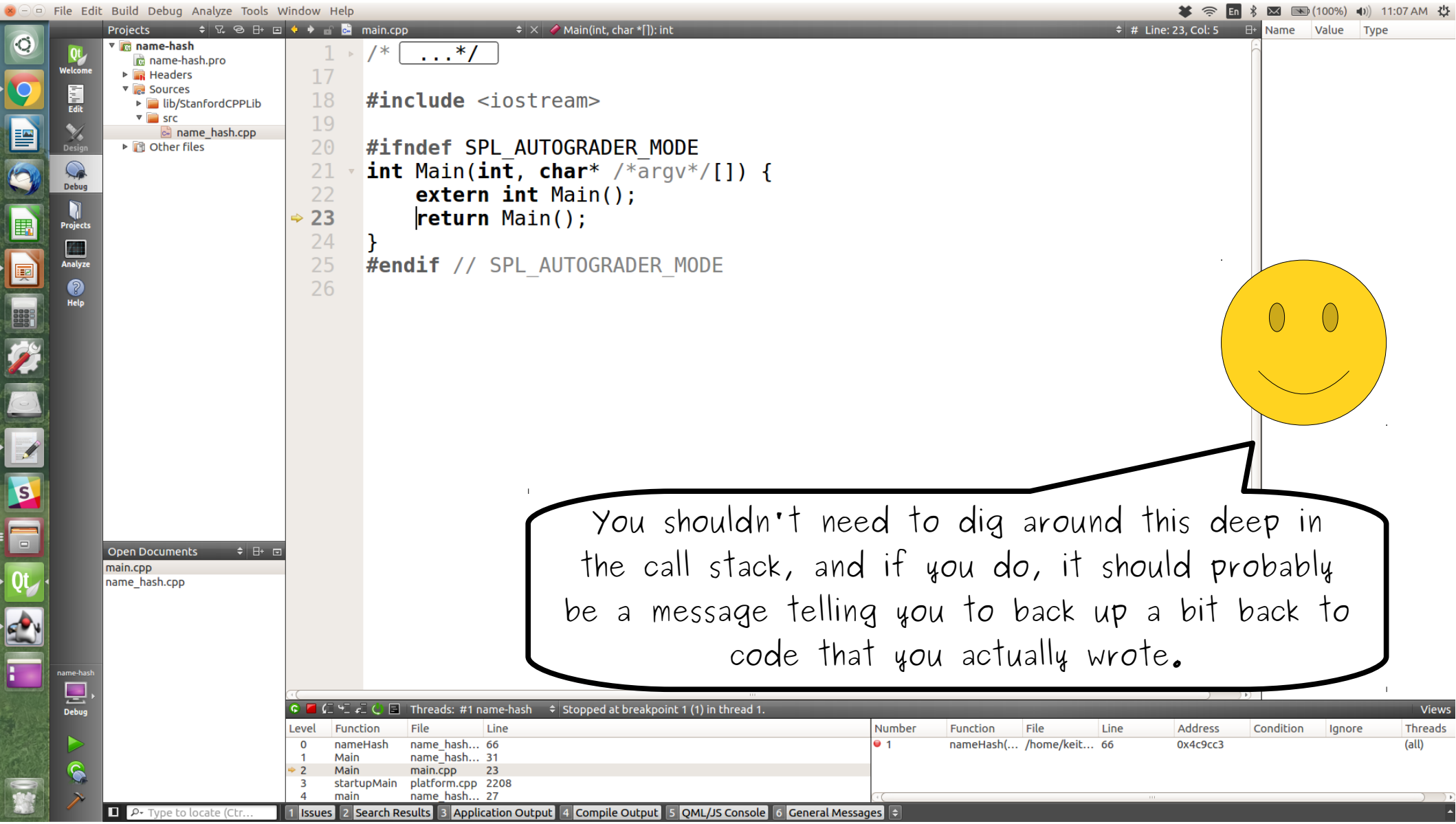

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
name_hash.cpp
name_hash.cpp
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Name
Value
Type
first
last
hashValue
766504679
int
Let's find out! Double-click on the line marked "Main" on Level 2. When you do…
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

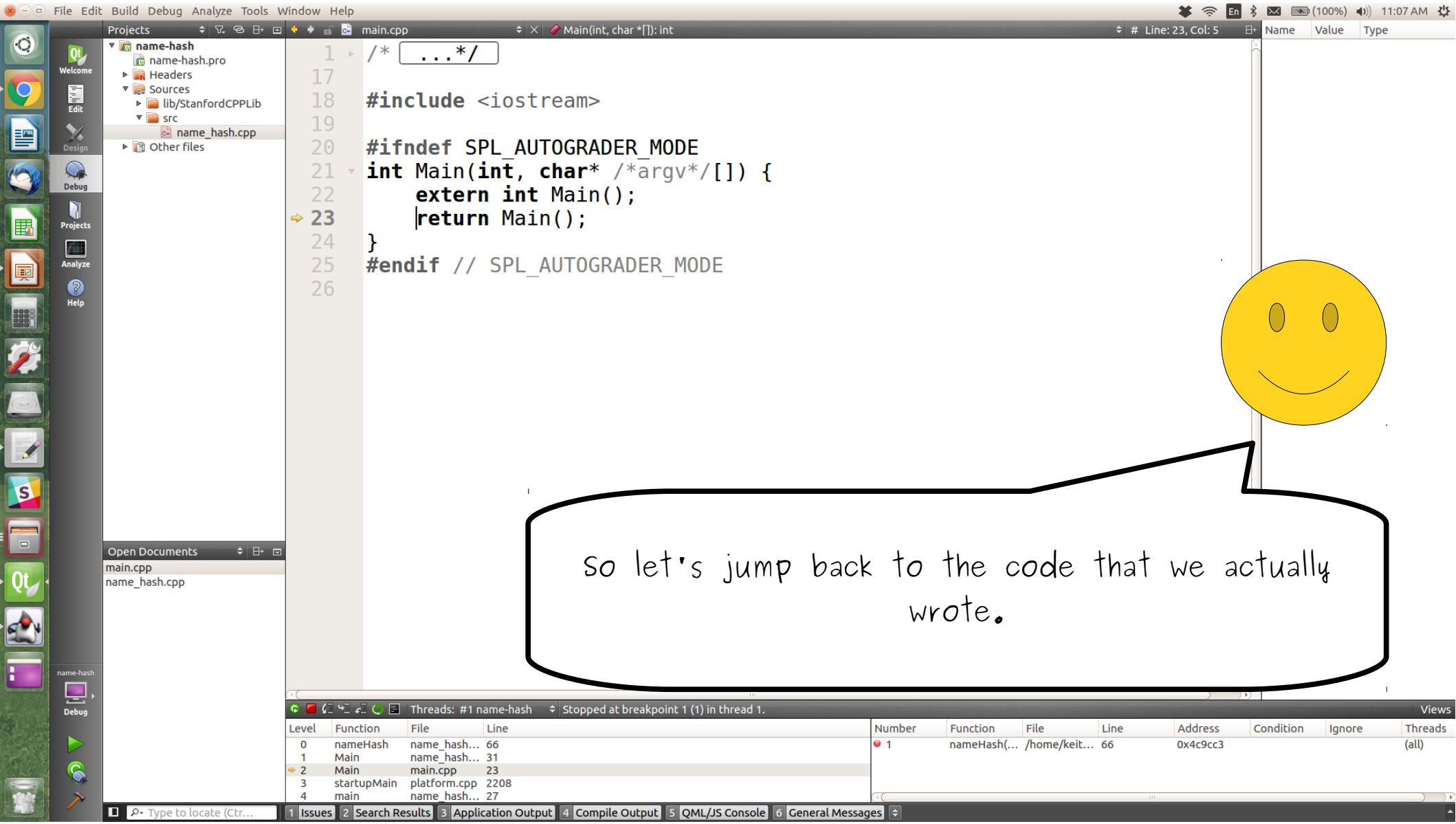

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
/* ...*/
#include <iostream>
#ifndef SPL_AUTOGRADER_MODE
int Main(int, char* /*argv*/[]) {
extern int Main();
return Main();
}
#endif // SPL_AUTOGRADER_MODE
Name
Value
Type
… you'll end up with something that looks like this.
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

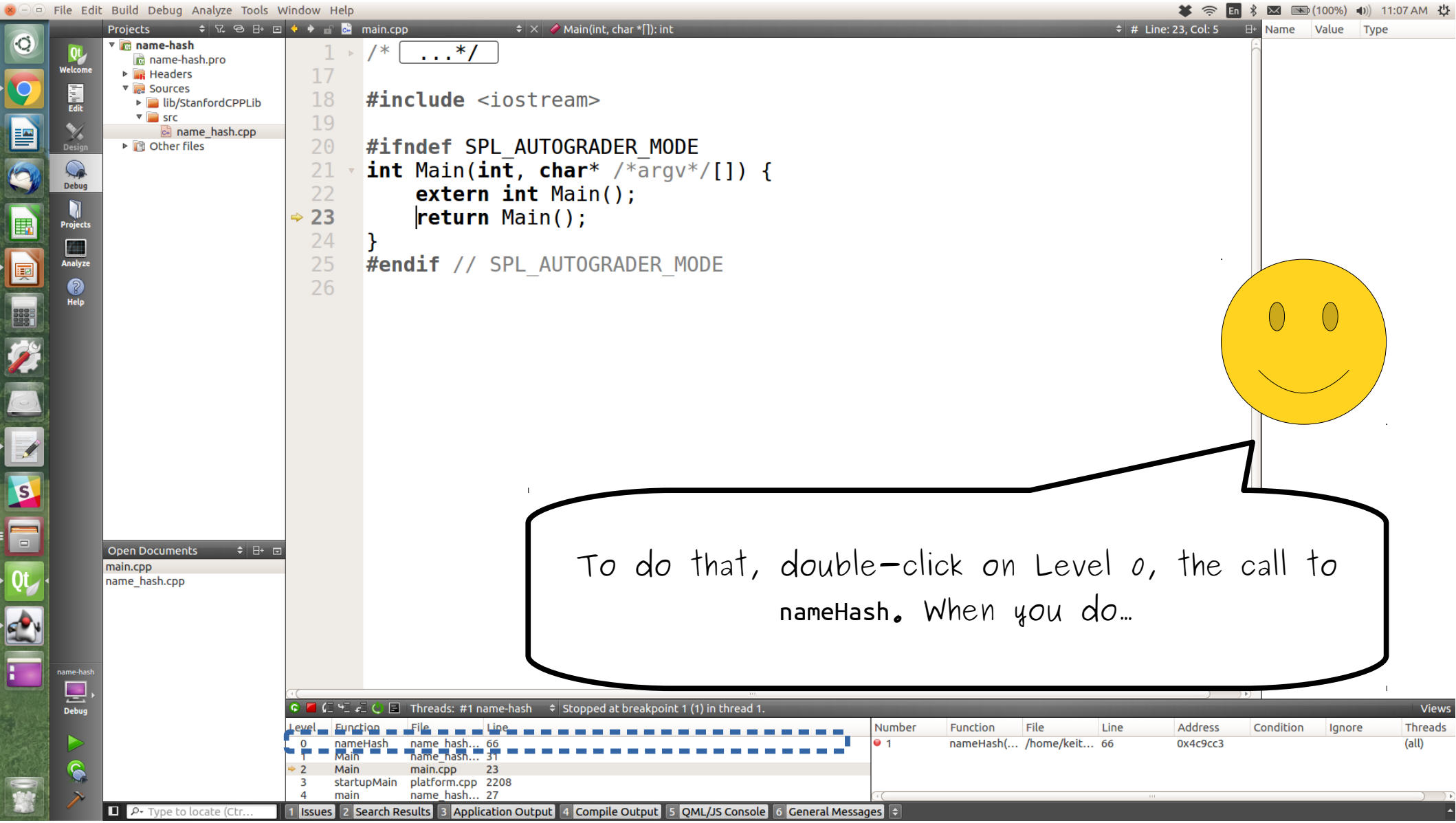

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
/* ...*/
#include <iostream>
#ifndef SPL_AUTOGRADER_MODE
int Main(int, char* /*argv*/[]) {
extern int Main();
return Main();
}
#endif // SPL_AUTOGRADER_MODE
Yikes! This looks Hairy and Scary! What happened?
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages

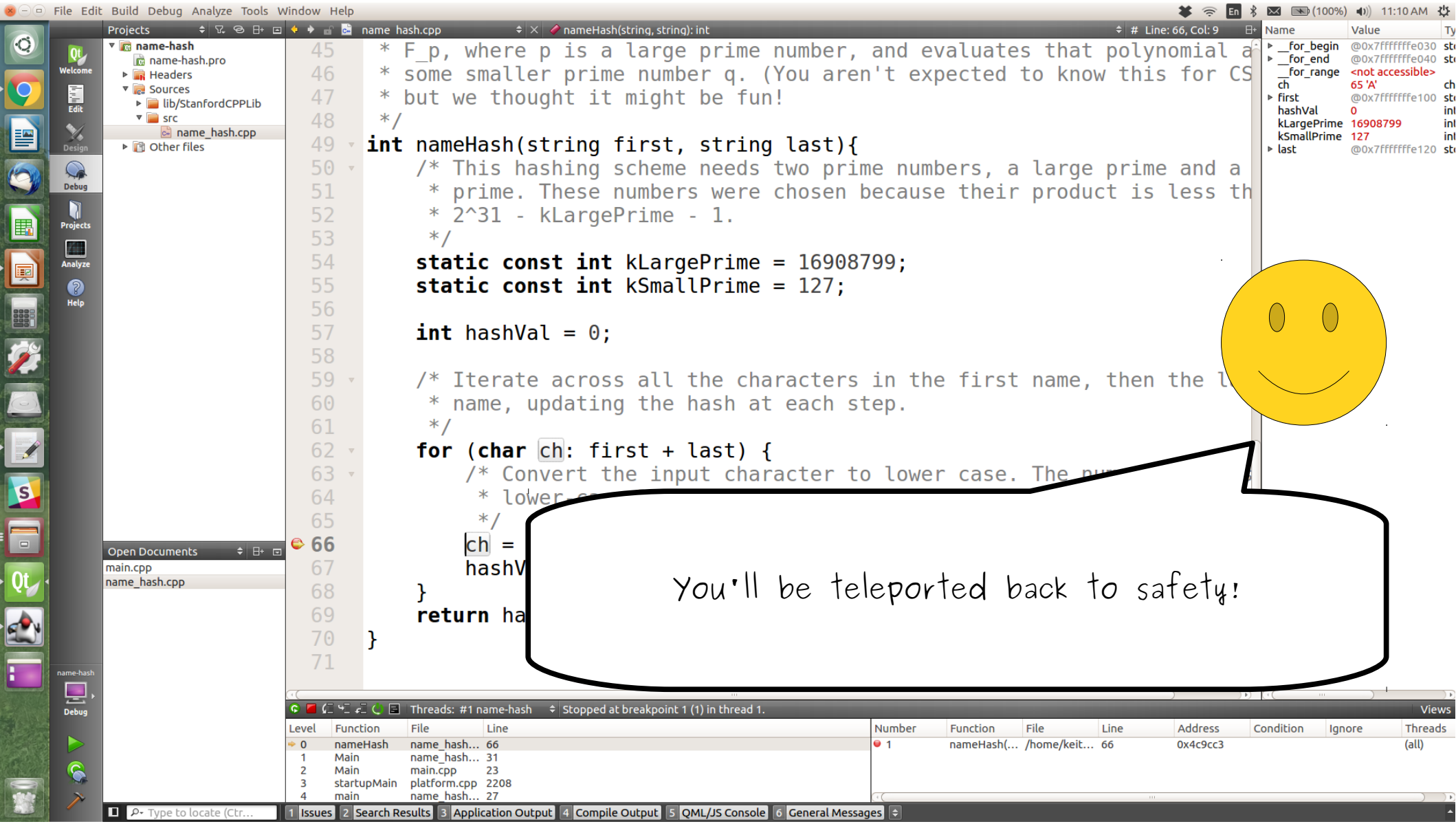

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
Name Value Type
/* ...*/
#include <iostream>
#ifndef SPL_AUTOGRADER_MODE
int Main(int, char* /*argv*/[]) {
extern int Main();
return Main();
}
#endif // SPL_AUTOGRADER_MODE
Whenever you start up a program in CS106B, there's a little bit of code that we automatically call for you, which does things like setting up the console.
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

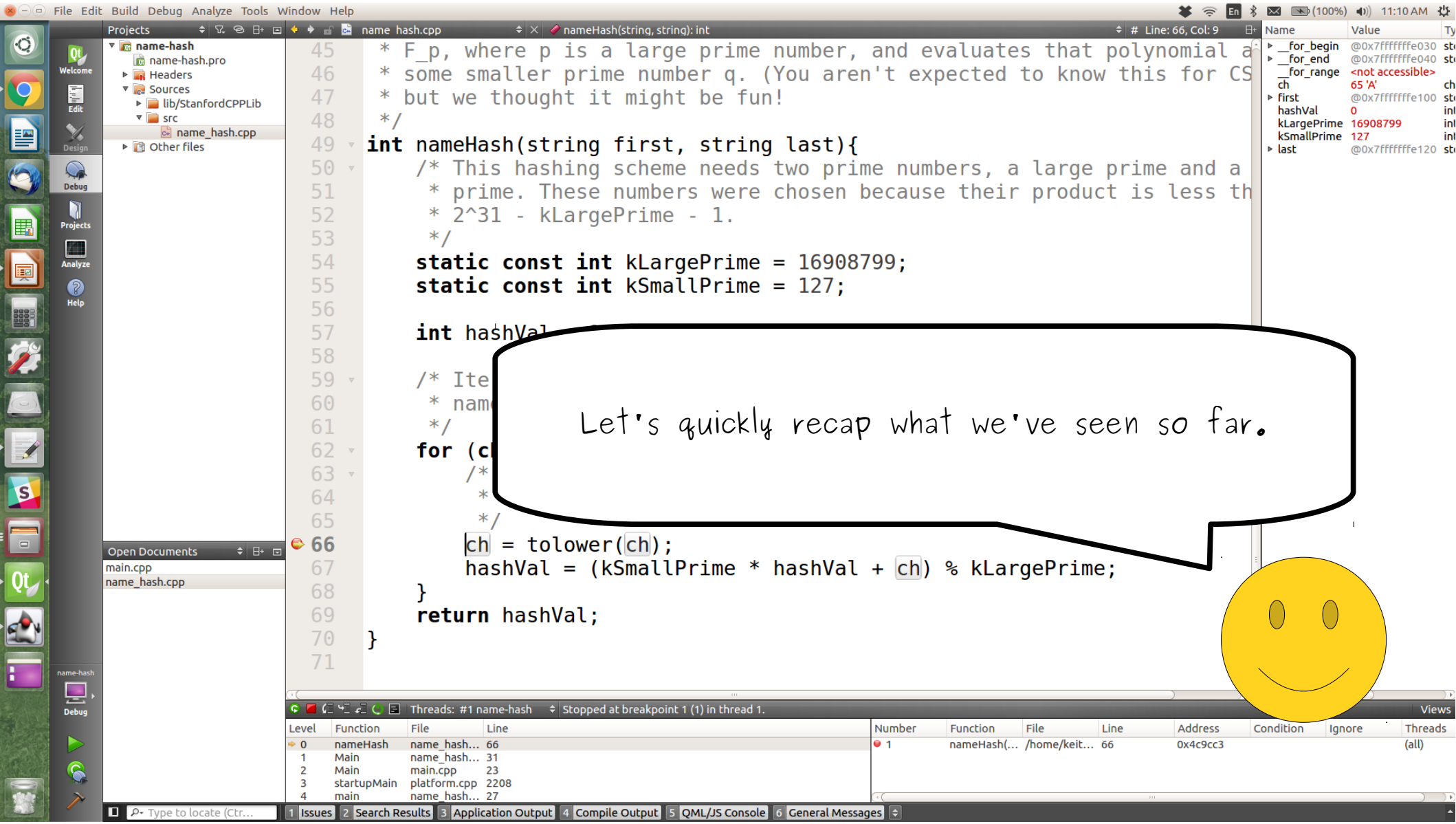

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
1 /* ...*/
17
18 #include <iostream>
19
20 #ifndef SPL_AUTOGRADER_MODE
21 int Main(int, char* /*argv*/[]) {
22 extern int Main();
23 return Main();
24 }
25 #endif // SPL_AUTOGRADER_MODE
26
This code will show up in the call stack below your actual program.
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

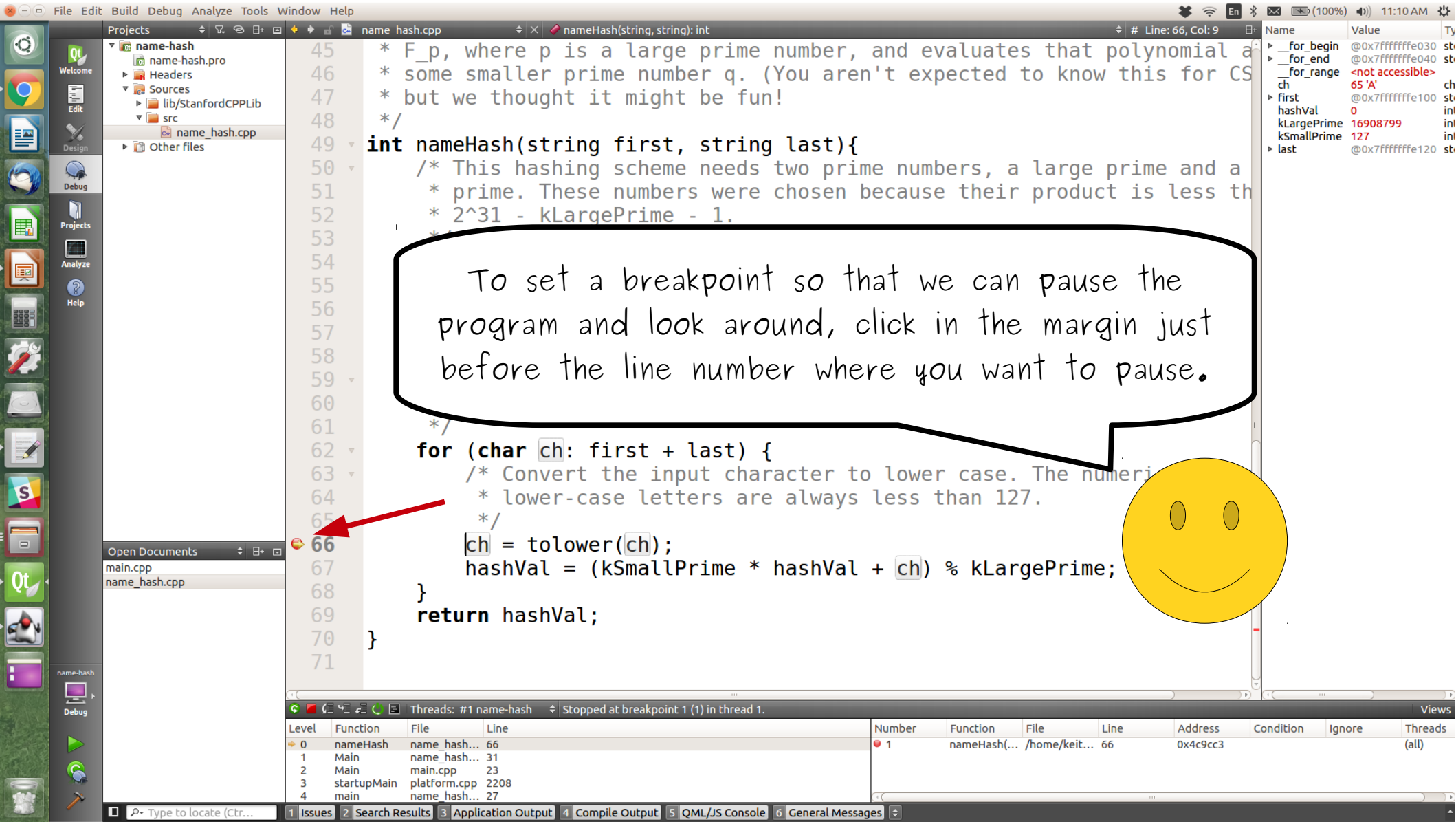

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
Name
Value
Type
/* ...*/
#include <iostream>
#ifndef SPL_AUTOGRADER_MODE
int Main(int, char* /*argv*/[]) {
extern int Main();
return Main();
}
#endif // SPL_AUTOGRADER_MODE
You shouldn't need to dig around this deep in the call stack, and if you do, it should probably be a message telling you to back up a bit back to code that you actually wrote.
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

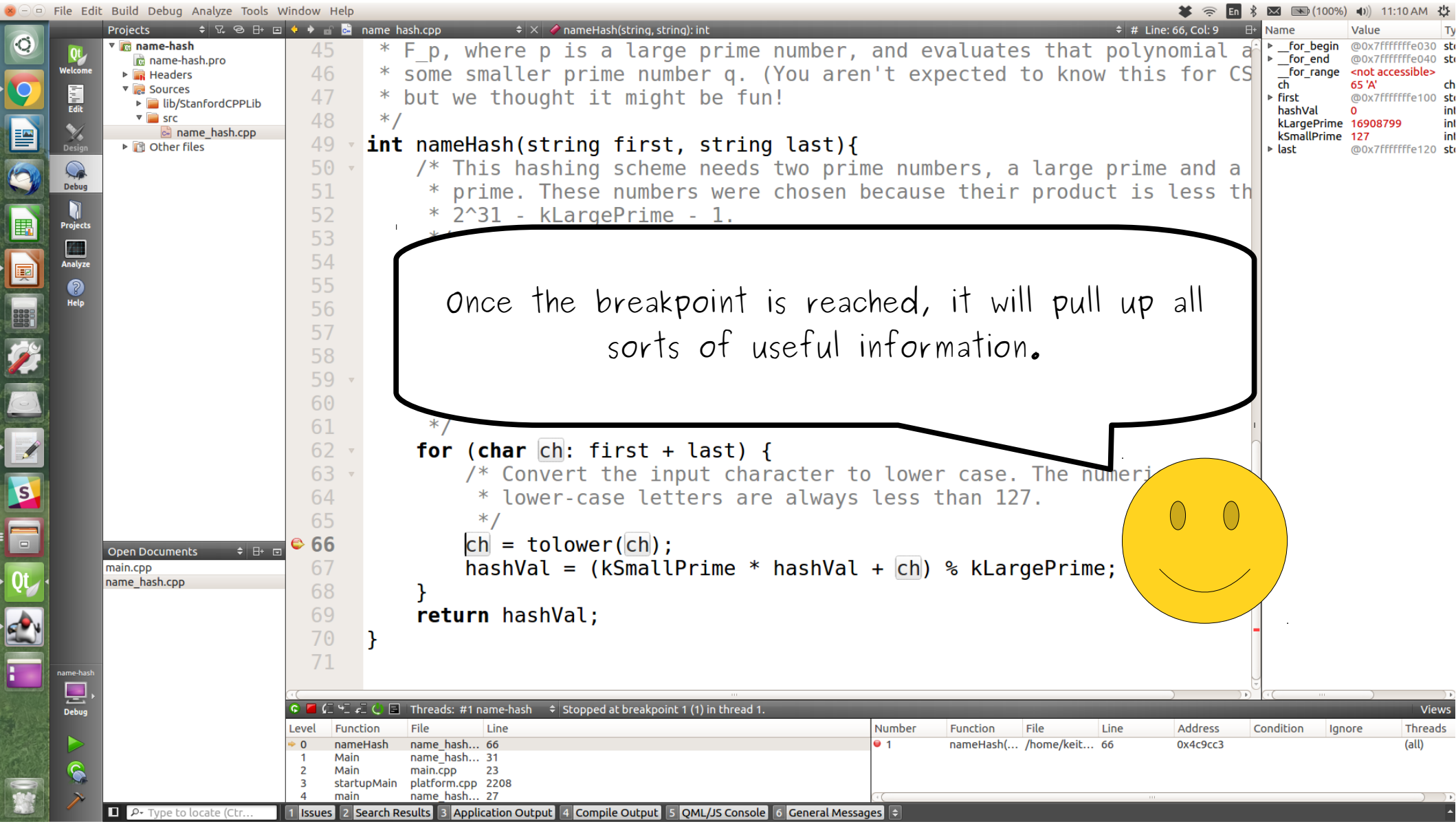

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
Name
Value
Type
/* ...*/
#include <iostream>
#ifndef SPL_AUTOGRADER_MODE
int Main(int, char* /*argv*/[]) {
extern int Main();
return Main();
}
#endif // SPL_AUTOGRADER_MODE
so let's jump back to the code that we actually wrote.
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

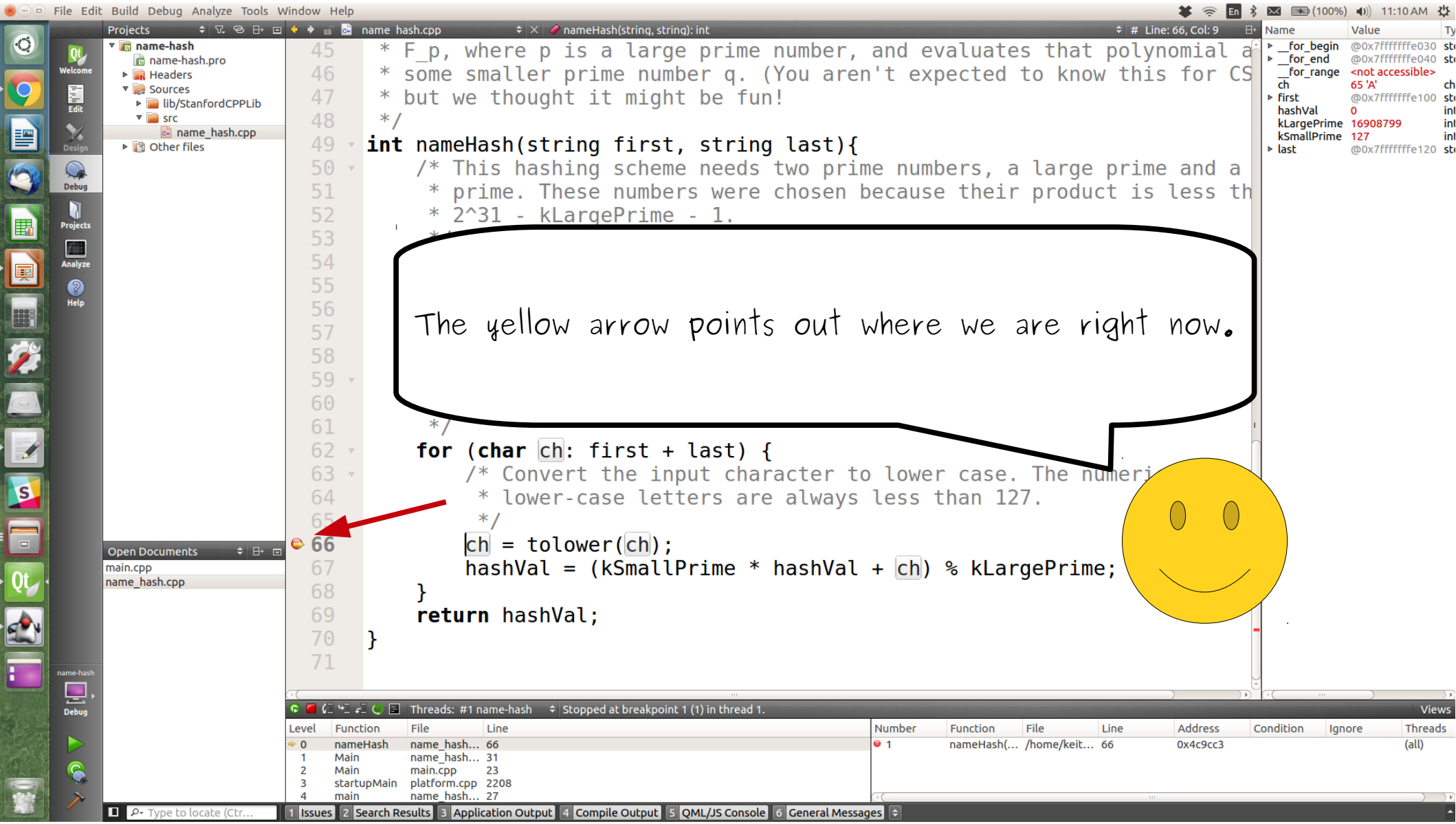

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
main.cpp
Main(int, char *[]): int
Line: 23, Col: 5
/* ...*/
#include <iostream>
#ifndef SPL_AUTOGRADER_MODE
int Main(int, char* /*argv*/[]) {
extern int Main();
return Main();
}
#endif // SPL_AUTOGRADER_MODE
To do that, double-click on Level 0, the call to nameHash. When you do…
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages

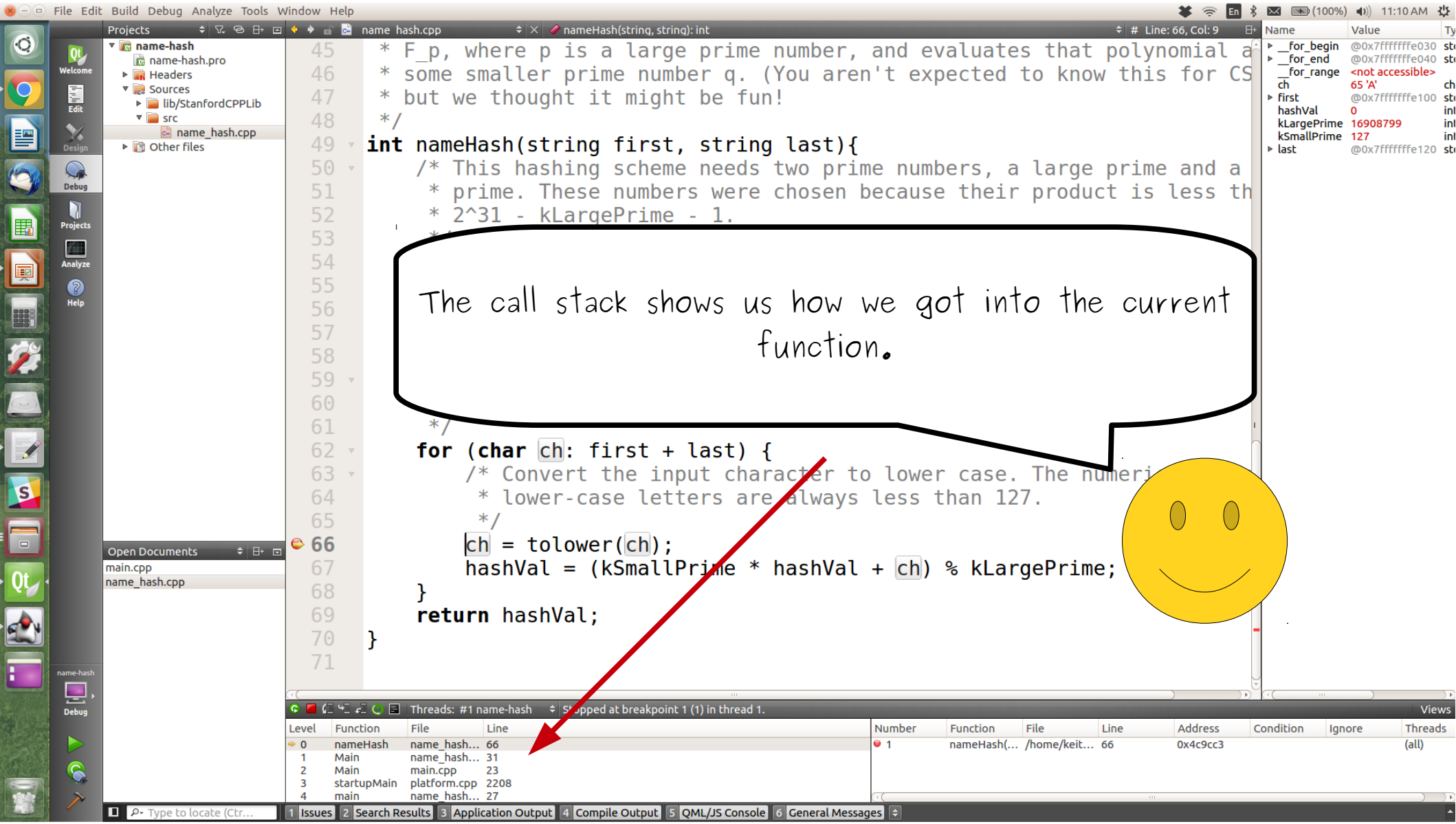

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the l
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The nu
* lower
*/
ch =
hashV
}
return ha
}
You'll be teleported back to safety!
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

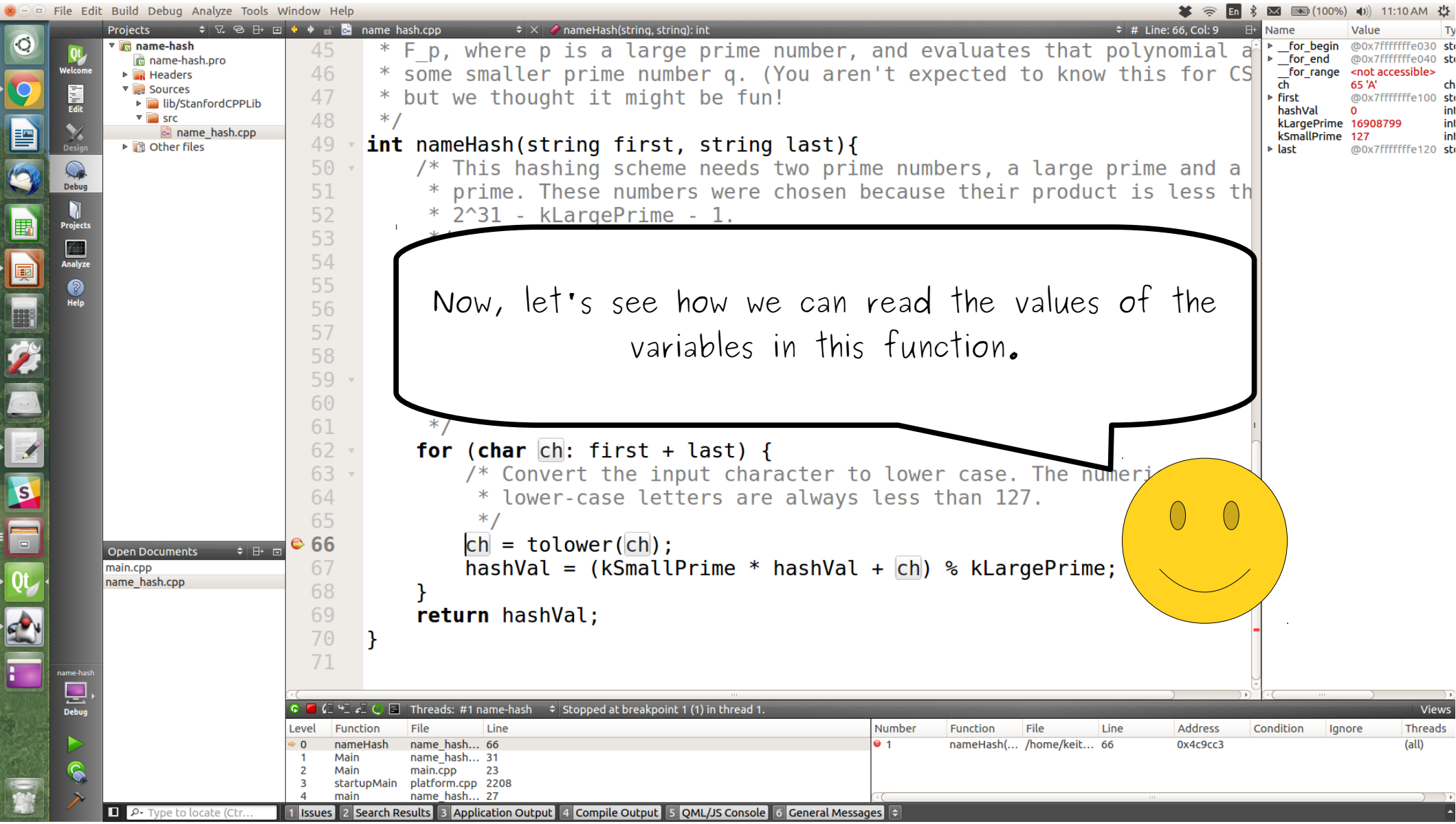

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal
/* Ite
* nam
*/
for (c
/*
*
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Let's quickly recap what we've seen so far.
Name
Value
__for_begin @0x7fffffffe030
__for_end @0x7fffffffe040
__for_range <not accessible>
ch 65 'A'
first @0x7fffffffe100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

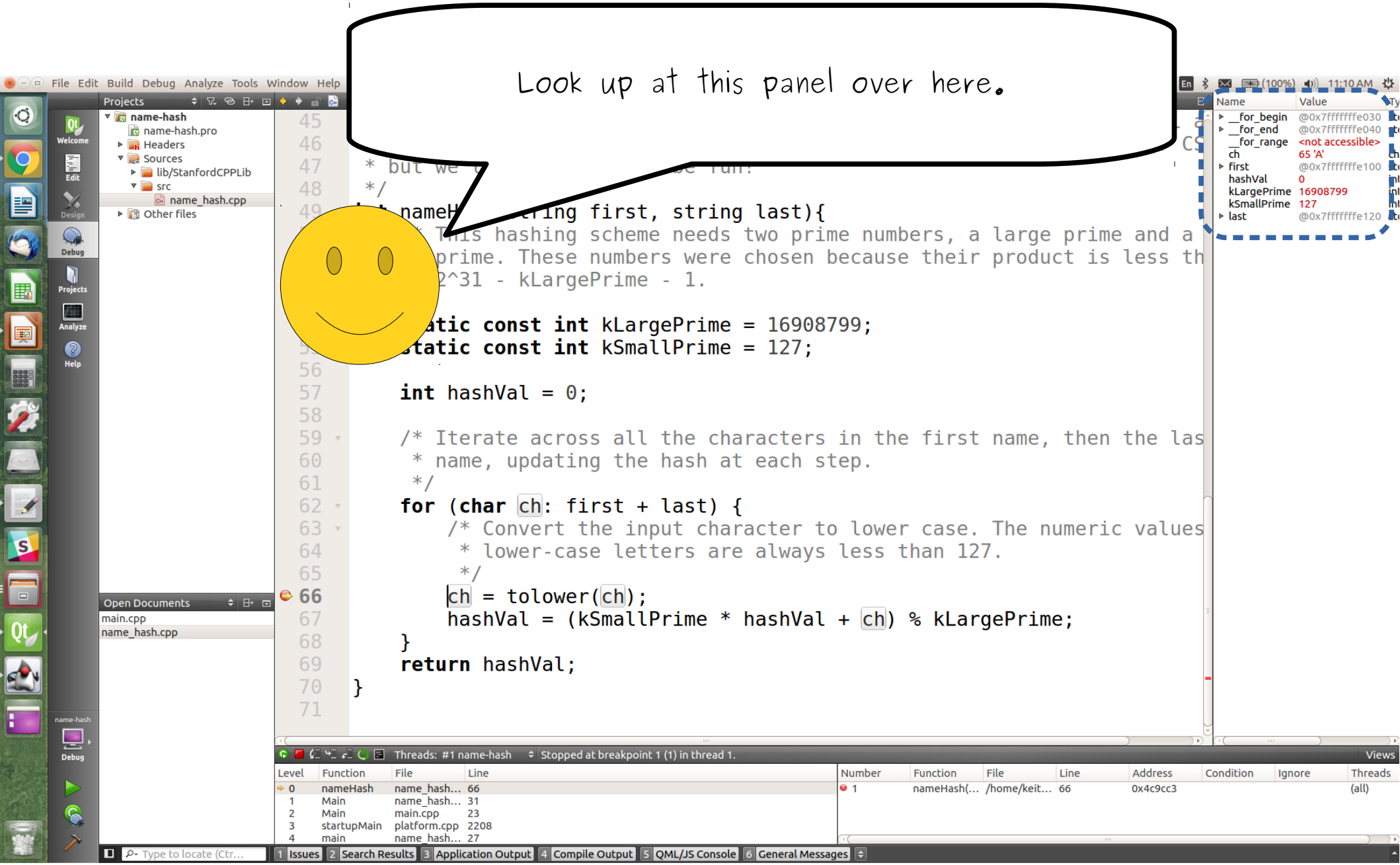

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
To set a breakpoint so that we can pause the program and look around, click in the margin just before the line number where you want to pause.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeri
* lower-case letters are always less than 127.
*/
66
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

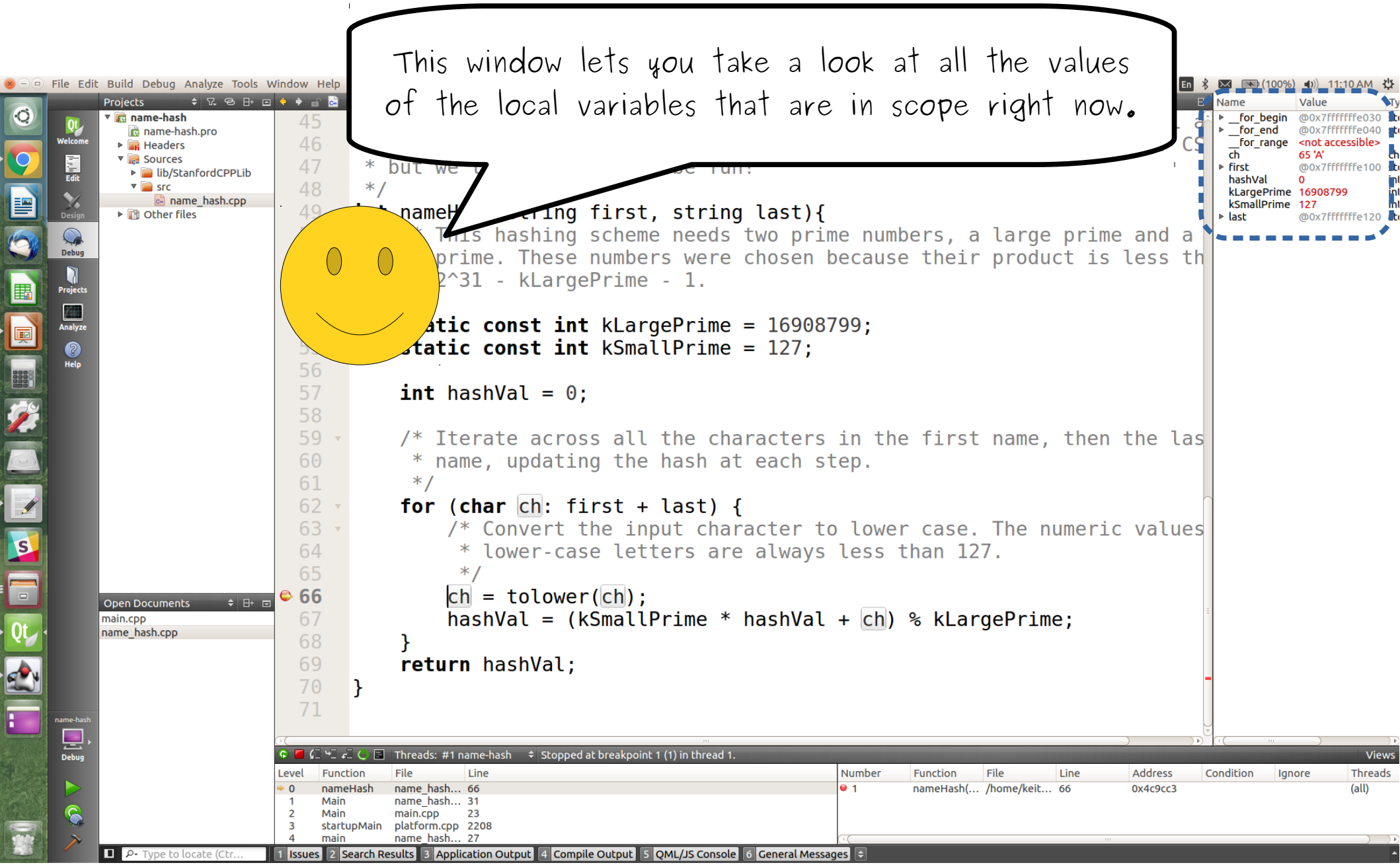

Once the breakpoint is reached, it will pull up all sorts of useful information.

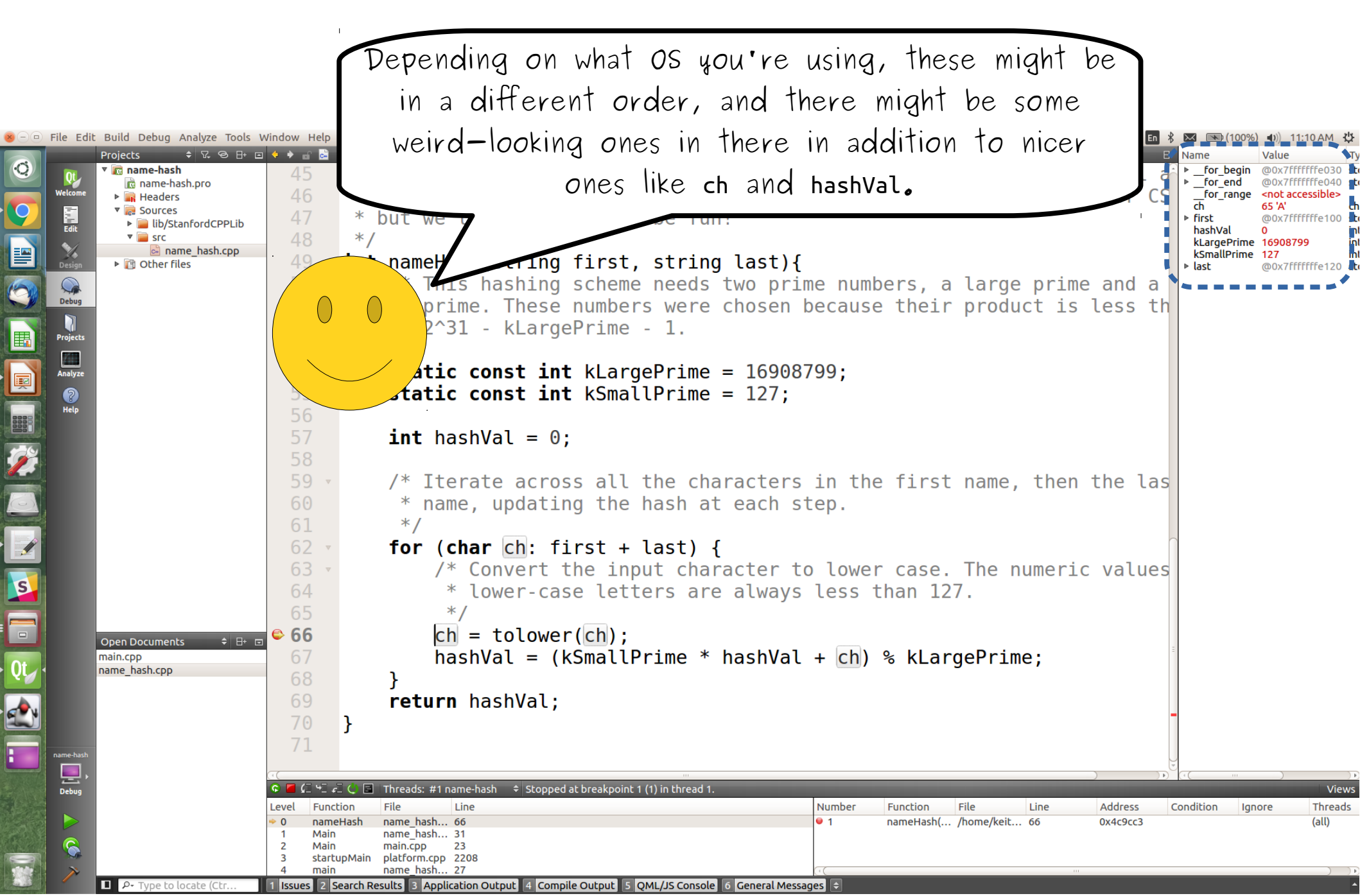

The yellow arrow points out where we are right now.

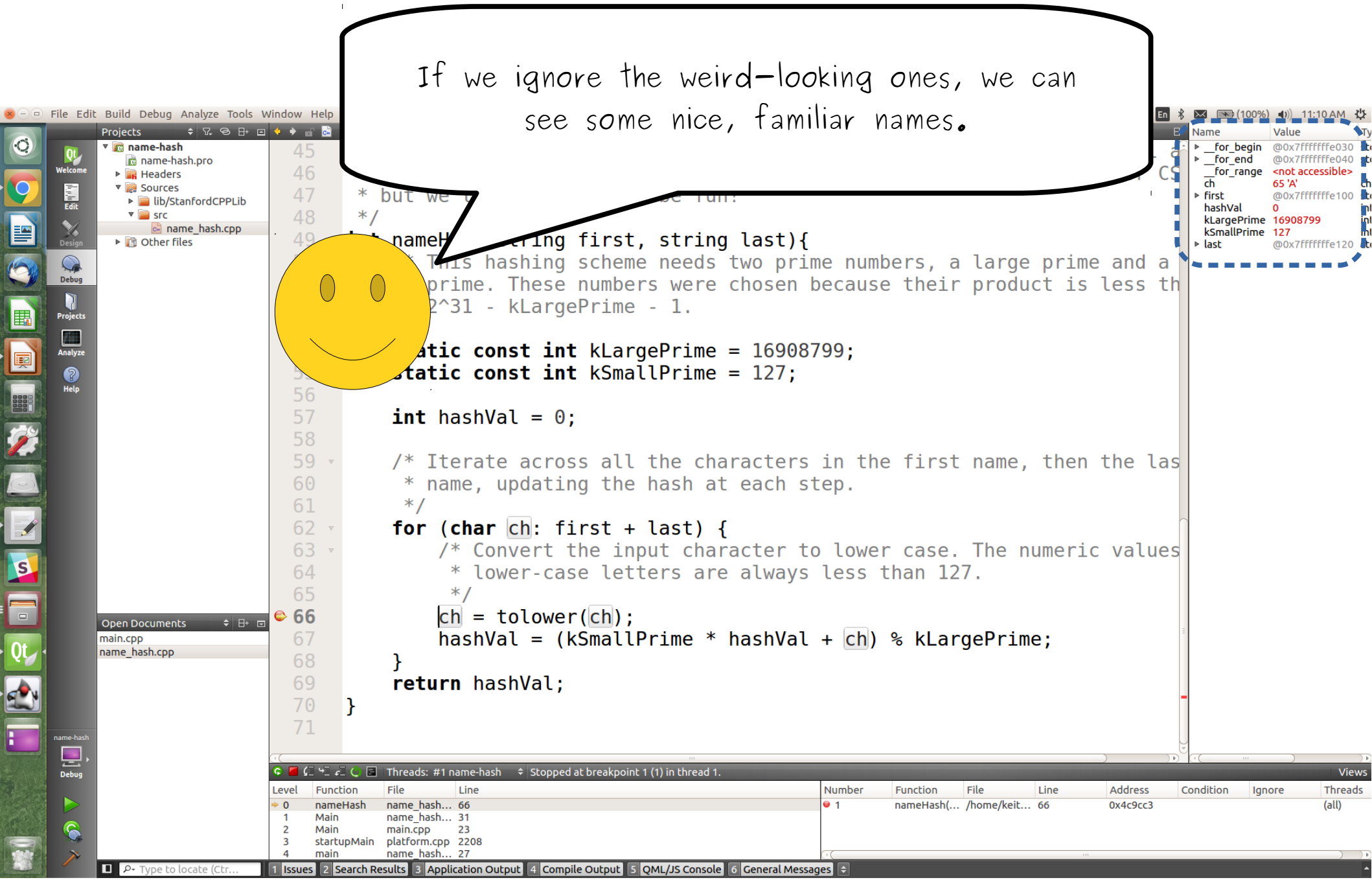

The call stack shows us how we got into the current function.

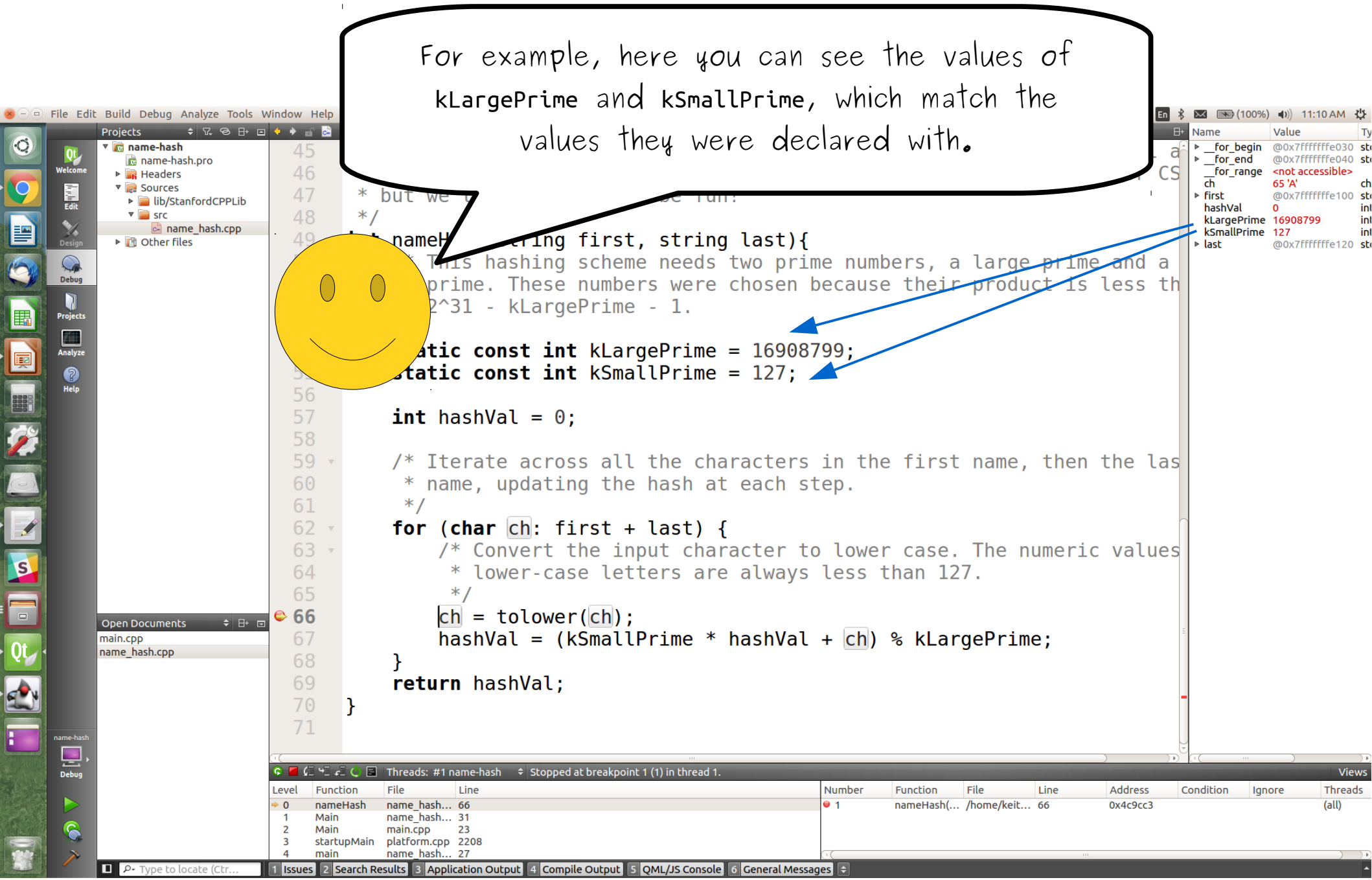

Now, let's see how we can read the values of the variables in this function.

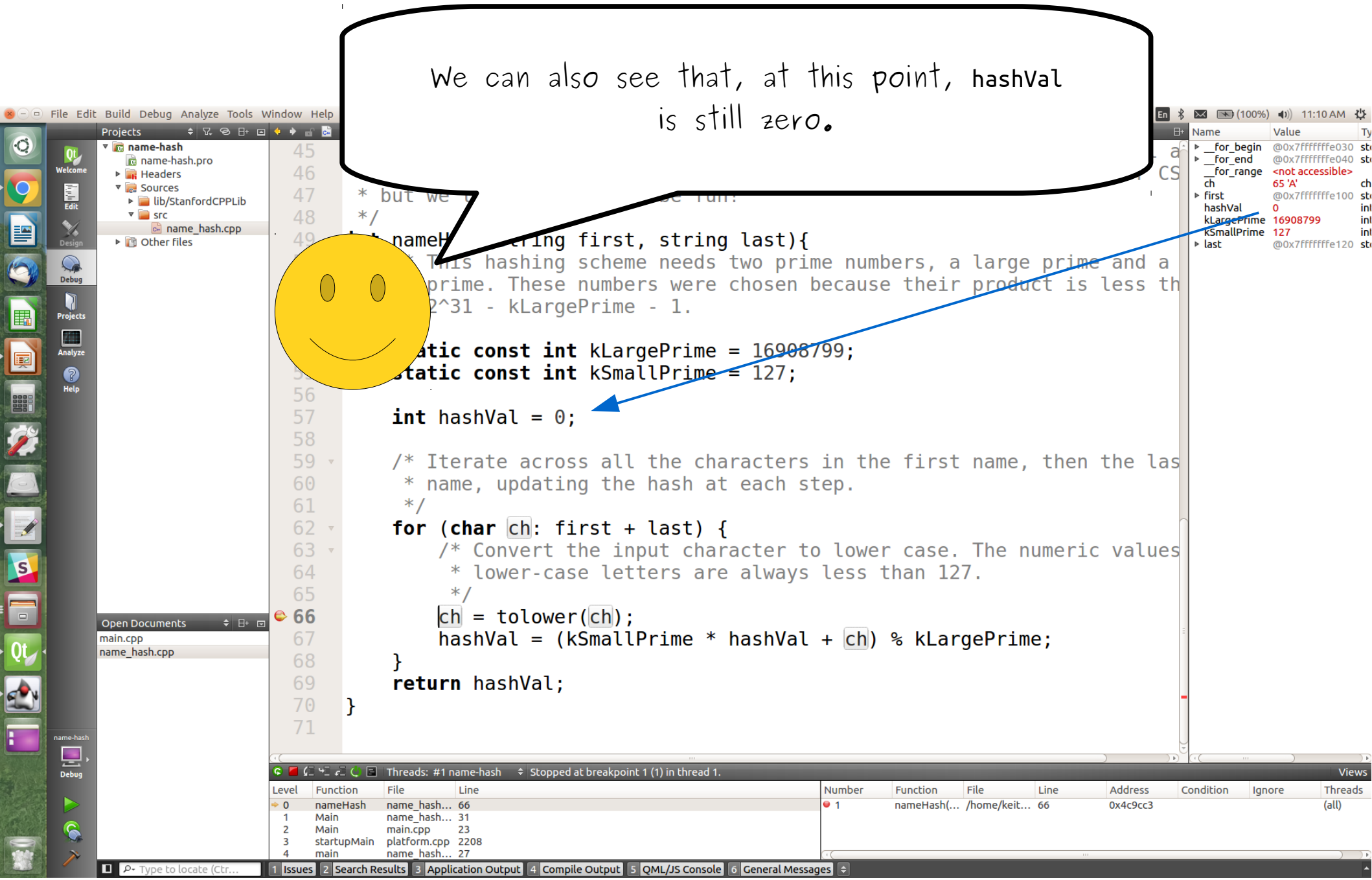

Look up at this panel over here.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
* but we
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
@0x7fffffffe030
__for_end
@0x7fffffffe040
__for_range
<not accessible>
ch
65 'A'
first
@0x7fffffffe100
hashVal
0
kLargePrime
16908799
kSmallPrime
127
last
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0
nameHash
name_hash...
66
1
Main
name_hash...
31
2
Main
main.cpp
23
3
startupMain
platform.cpp
2208
4
main
name_hash...
27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1
nameHash(...
/home/keit...
66
0x4c9cc3
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

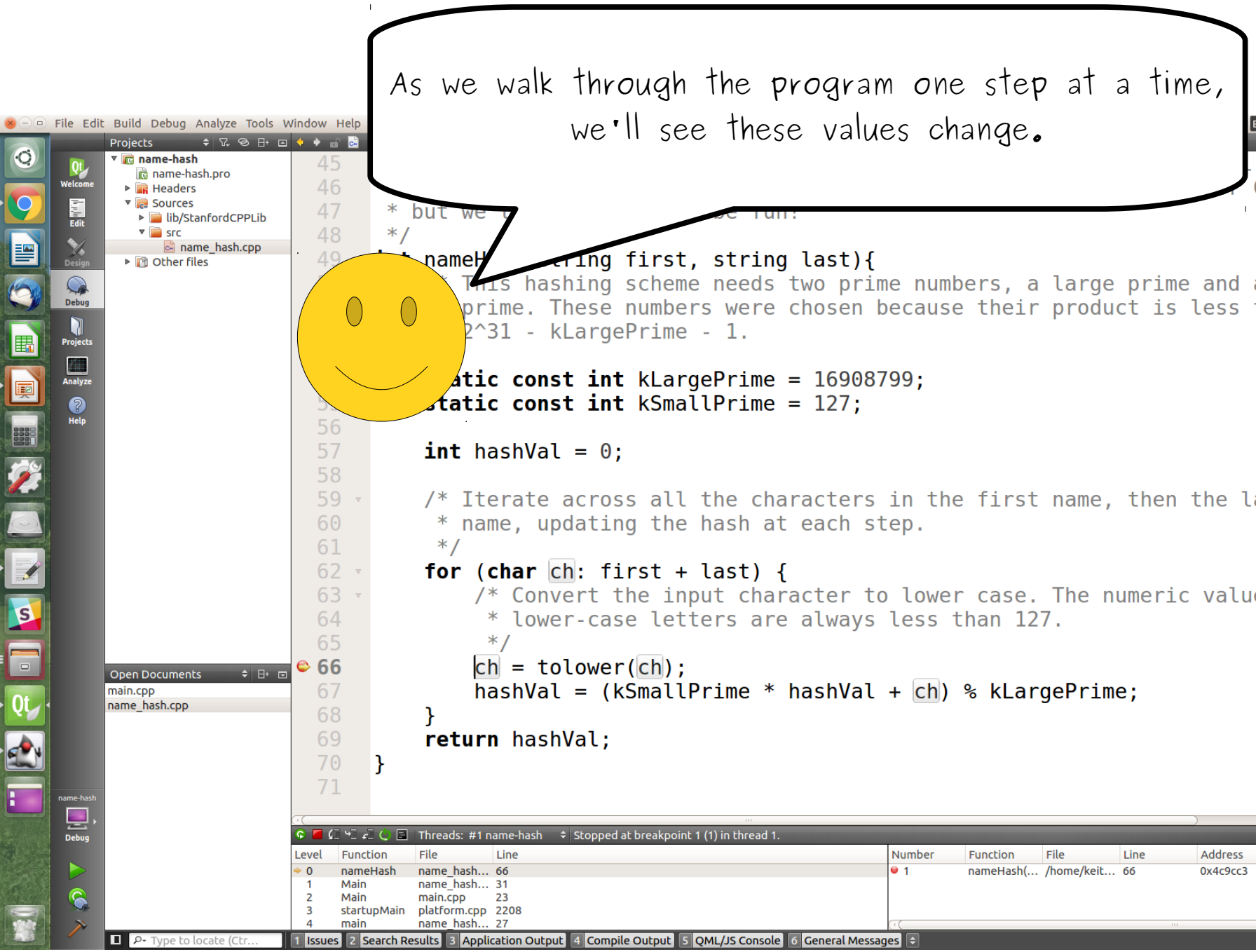

This window lets you take a look at all the values of the local variables that are in scope right now.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
* but we
*/
nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
@0x7fffffffe030
__for_end
@0x7fffffffe040
__for_range
<not accessible>
ch
65 'A'
first
@0x7fffffffe100
hashVal
0
kLargePrime
16908799
kSmallPrime
127
last
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0
nameHash
name_hash...
66
1
Main
name_hash...
31
2
Main
main.cpp
23
3
startupMain
platform.cpp
2208
4
main
name_hash...
27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1
nameHash(...
/home/keit...
66
0x4c9cc3
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

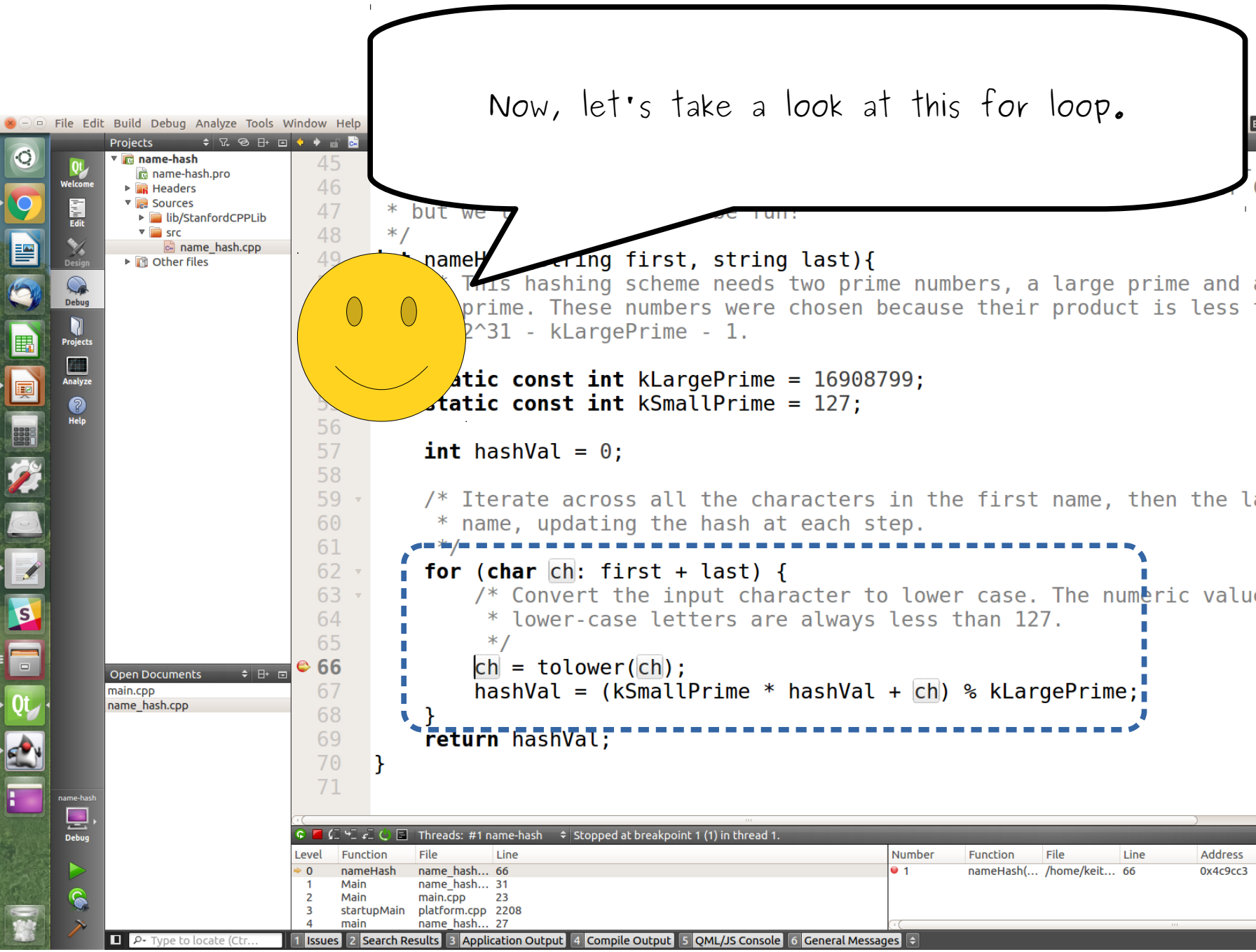

Depending on what OS you're using, these might be in a different order, and there might be some weird-looking ones in there in addition to nicer ones like ch and hashVal.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
* but we
*/
nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
@0x7fffffffe030
__for_end
@0x7fffffffe040
__for_range
<not accessible>
ch
65 'A'
first
@0x7fffffffe100
hashVal
0
kLargePrime
16908799
kSmallPrime
127
last
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0
nameHash
name_hash...
66
1
Main
name_hash...
31
2
Main
main.cpp
23
3
startupMain
platform.cpp
2208
4
main
name_hash...
27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1
nameHash(...
/home/keit...
66
0x4c9cc3
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

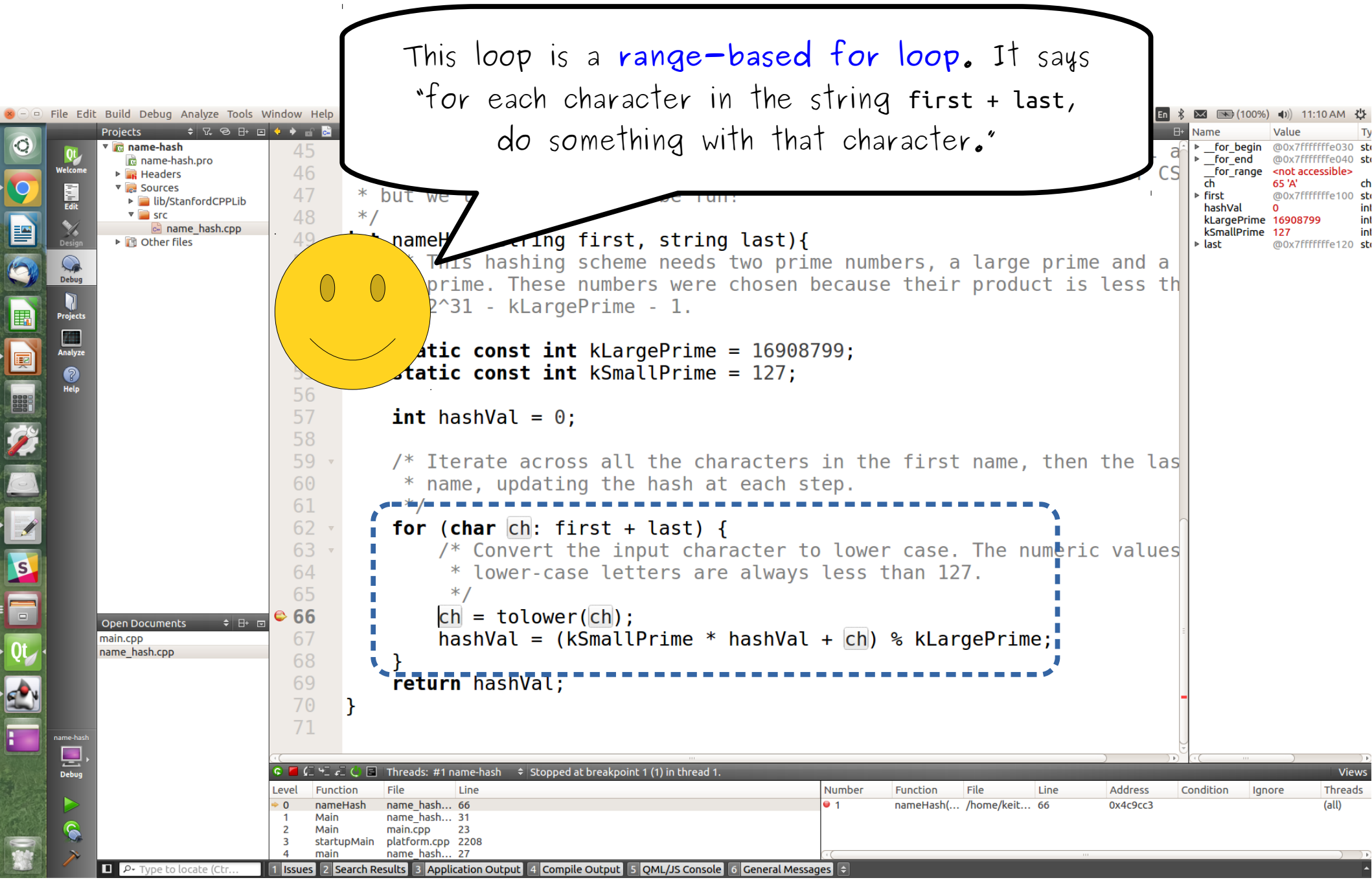

If we ignore the weird-looking ones, we can see some nice, familiar names.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
* but we
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

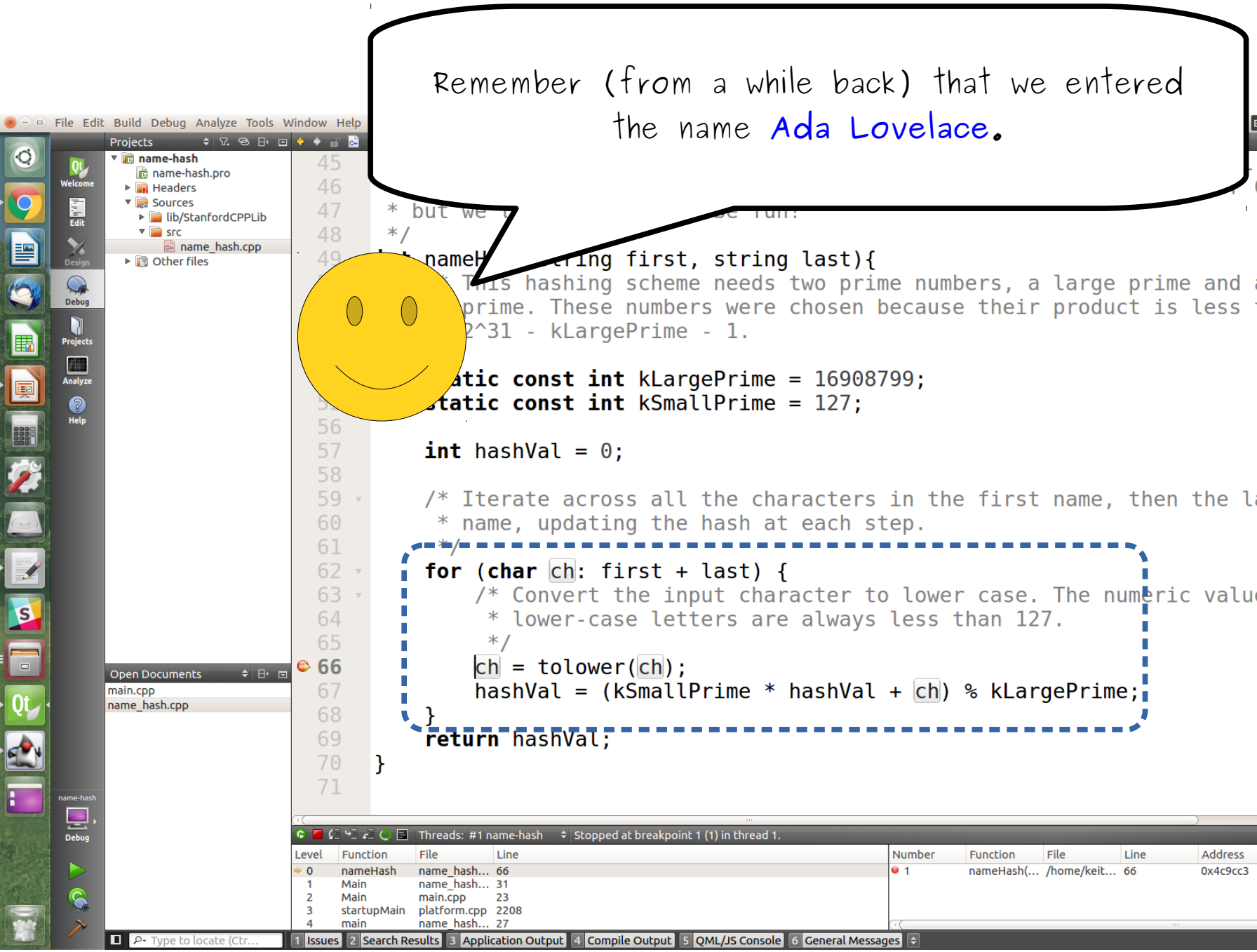

For example, here you can see the values of kLargePrime and kSmallPrime, which match the values they were declared with.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45 * but we
46 */
47 int nameHash(string first, string last){
48 /* This hashing scheme needs two prime numbers, a large prime and a
49 * prime. These numbers were chosen because their product is less th
50 * 2^31 - kLargePrime - 1.
51 */
52 static const int kLargePrime = 16908799;
53 static const int kSmallPrime = 127;
54
55 int hashVal = 0;
56
57 /* Iterate across all the characters in the first name, then the las
58 * name, updating the hash at each step.
59 */
60 for (char ch: first + last) {
61 /* Convert the input character to lower case. The numeric values
62 * lower-case letters are always less than 127.
63 */
64 ch = tolower(ch);
65 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
66 }
67 return hashVal;
68 }
69
70
71
Name
Value
__for_begin @0x7fffffffe030
__for_end @0x7fffffffe040
__for_range <not accessible>
ch 65 'A'
first @0x7fffffffe100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

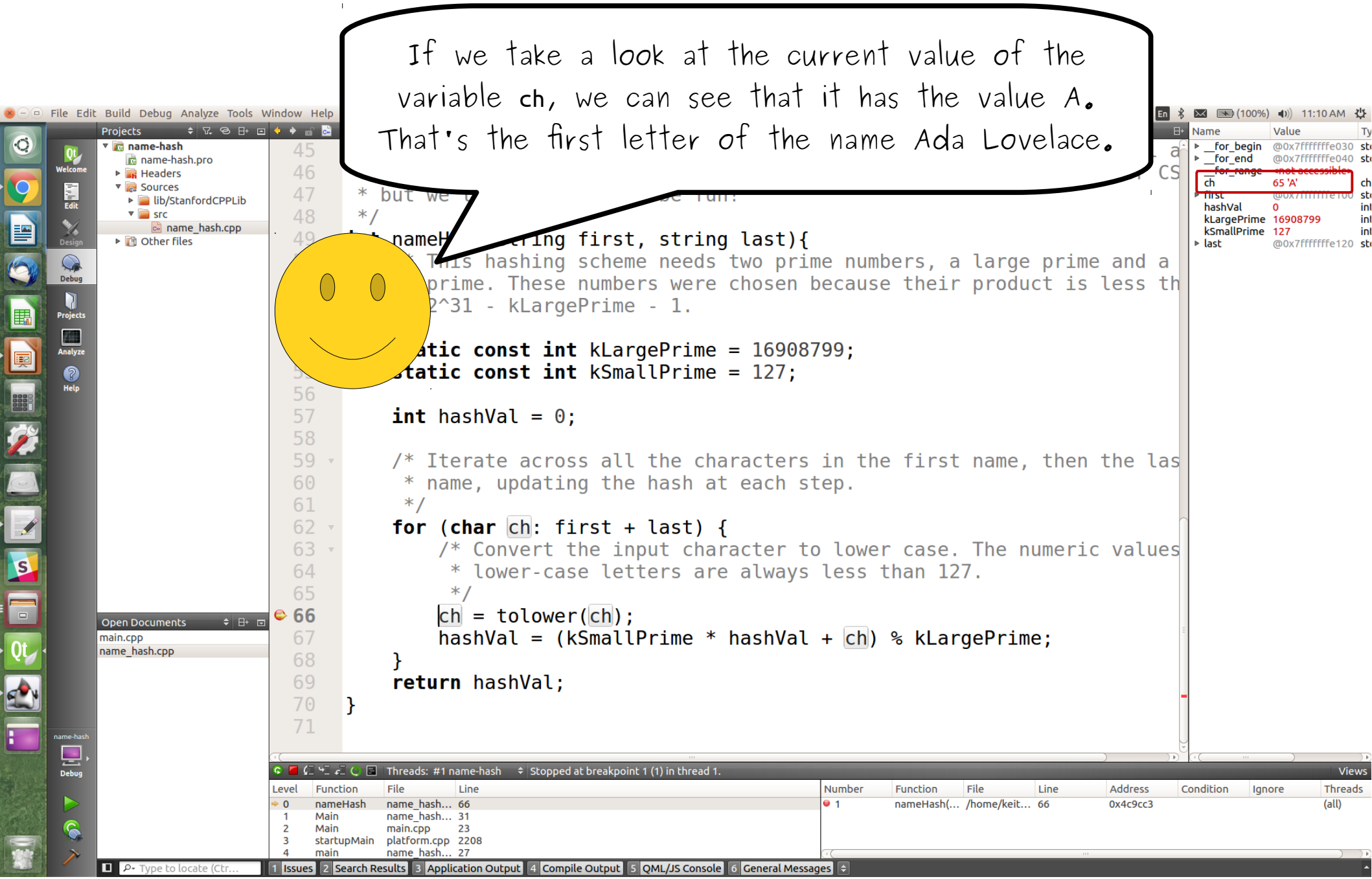

We can also see that, at this point, hashVal is still zero.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
* This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

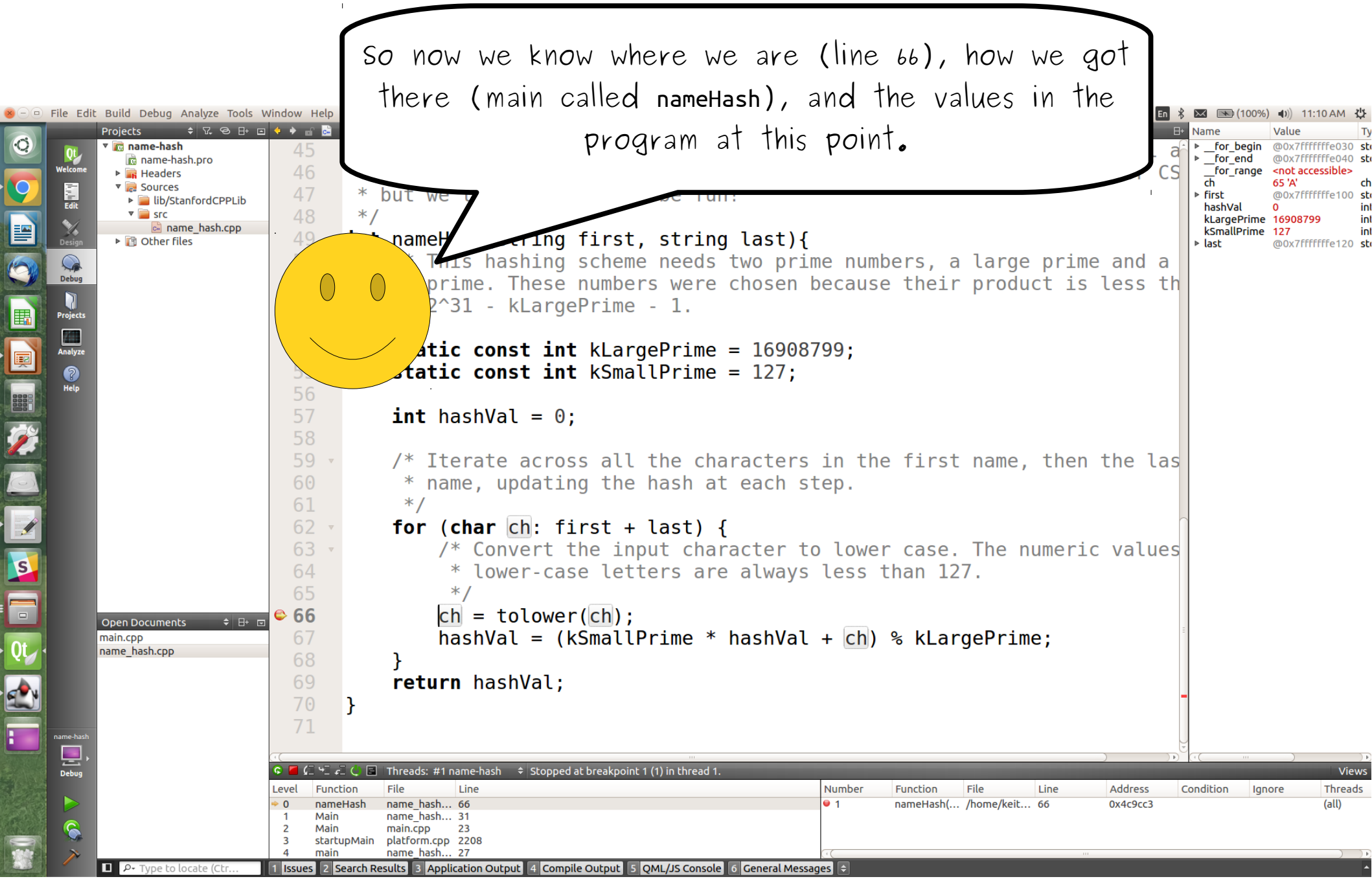

As we walk through the program one step at a time, we'll see these values change.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
* but we
*/
nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

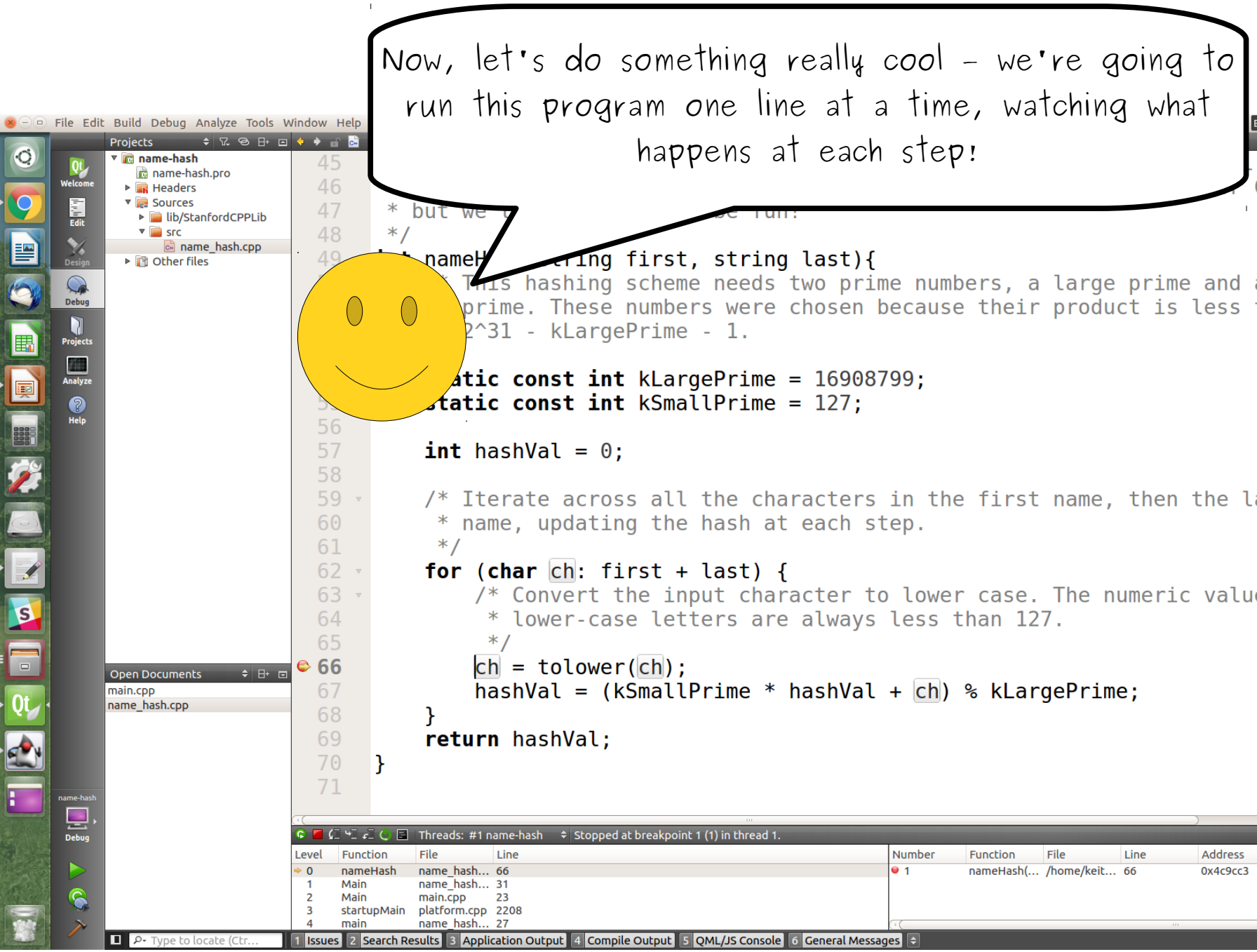

Now, let's take a look at this for loop.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
* but we
*/
nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

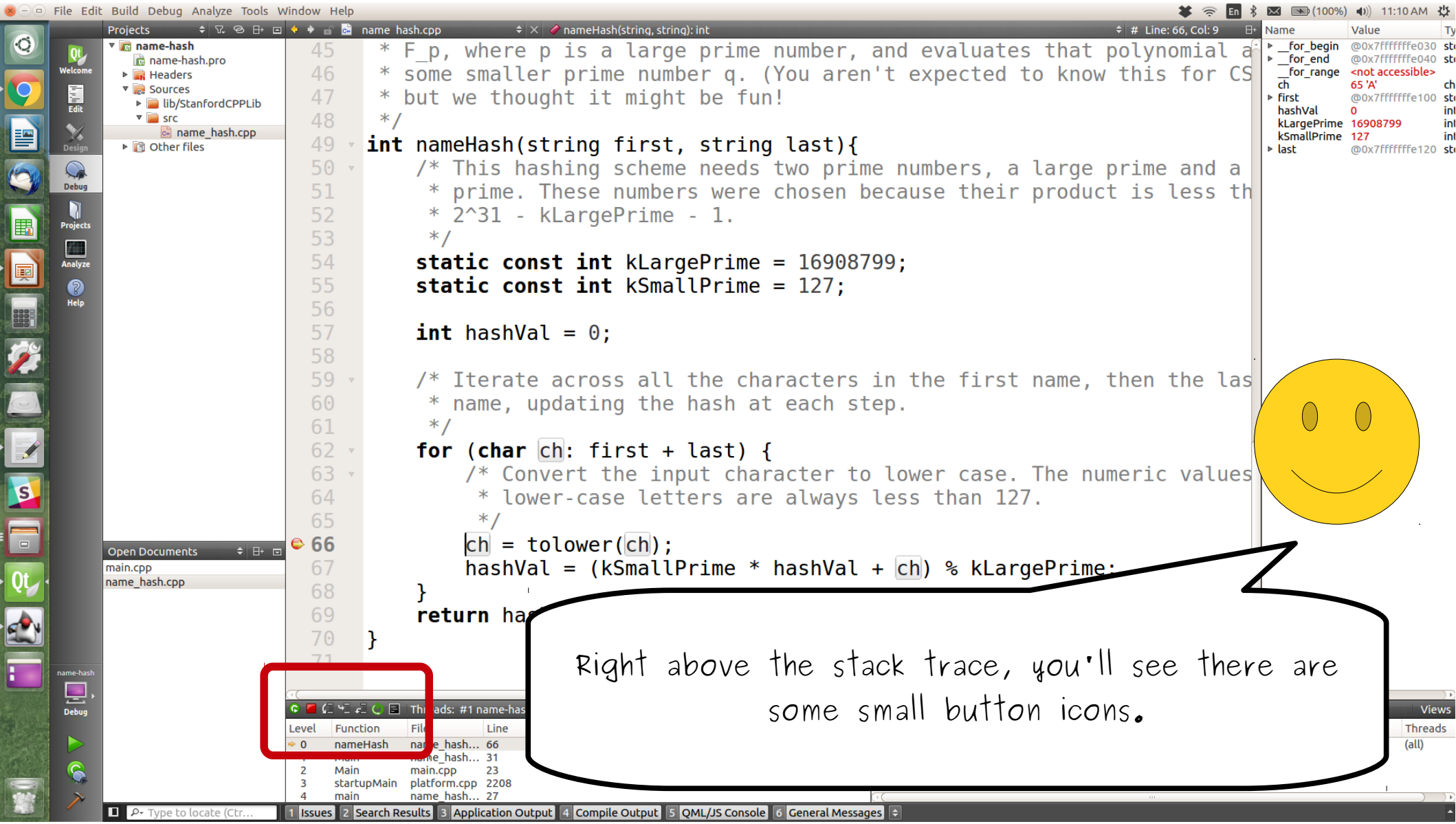

This loop is a range-based for loop. It says "for each character in the string first + last, do something with that character."
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
Number
Address
Condition
Ignore
Threads
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages

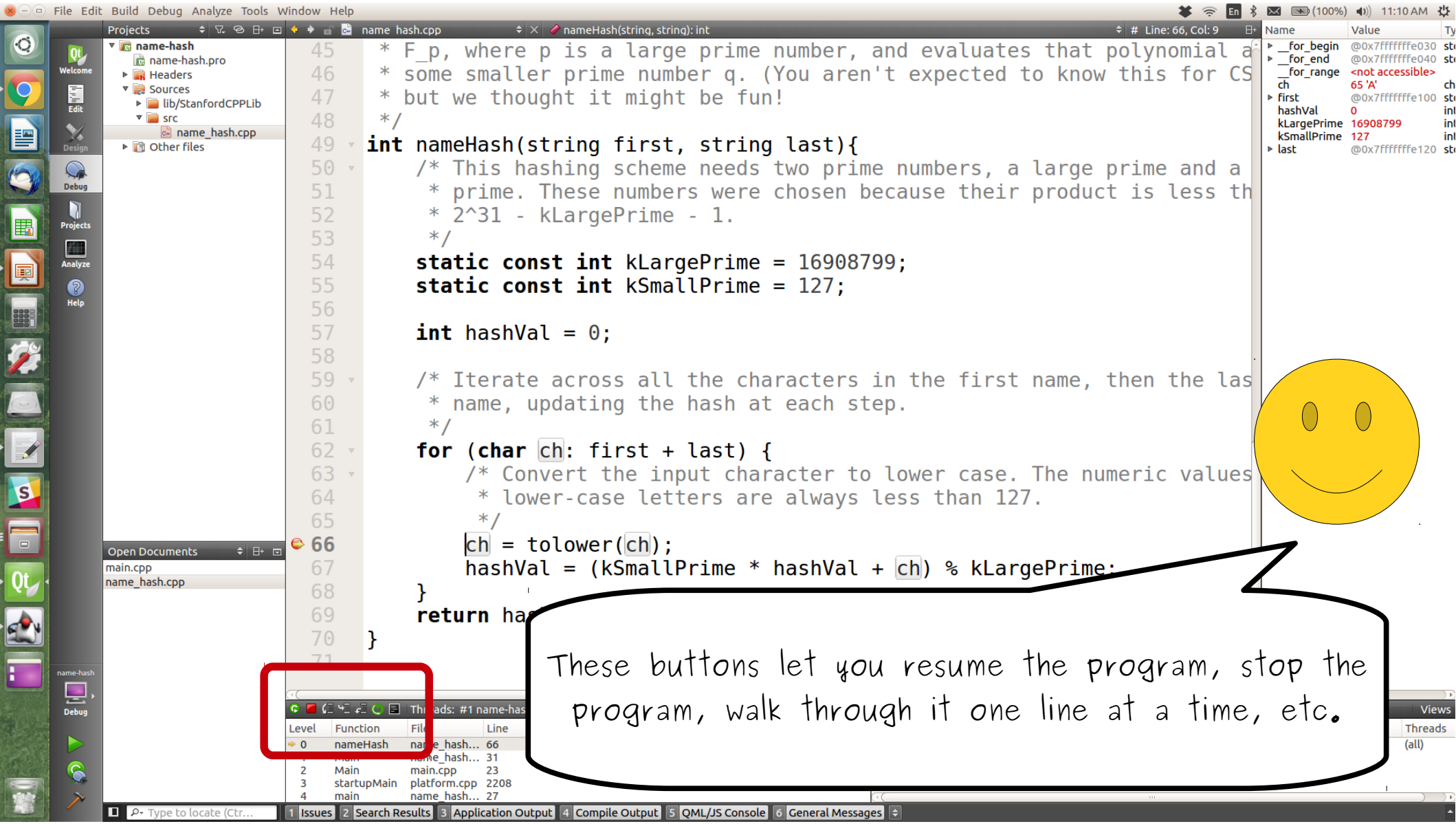

Remember (from a while back) that we entered the name Ada Lovelace.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
* but we
*/
nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

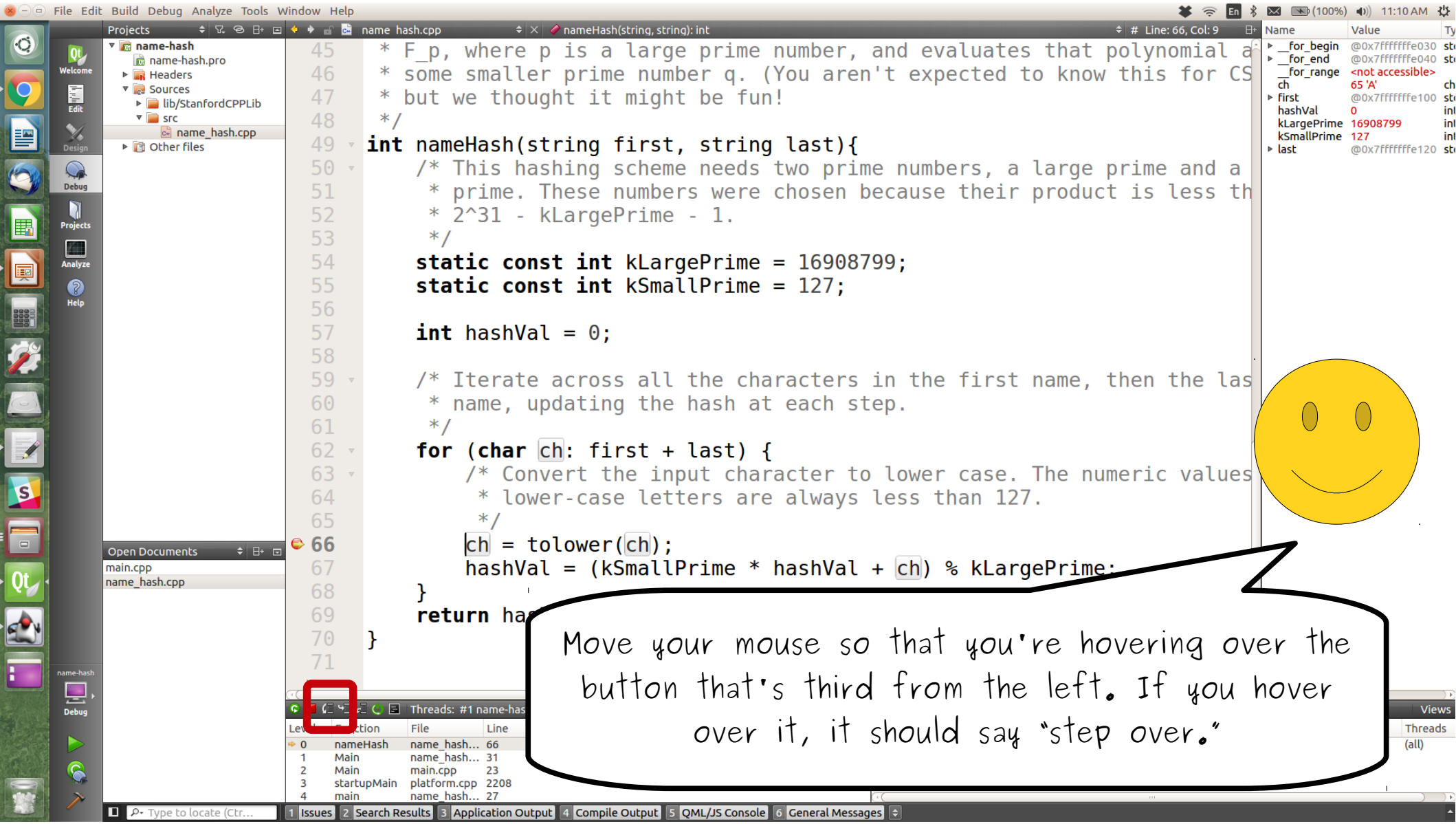

If we take a look at the current value of the variable ch, we can see that it has the value A. That's the first letter of the name Ada Lovelace.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45
46
47
48
49
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
* but we
*/
nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
65 'A'
first
hashVal
0
kLargePrime
16908799
kSmallPrime
127
last
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

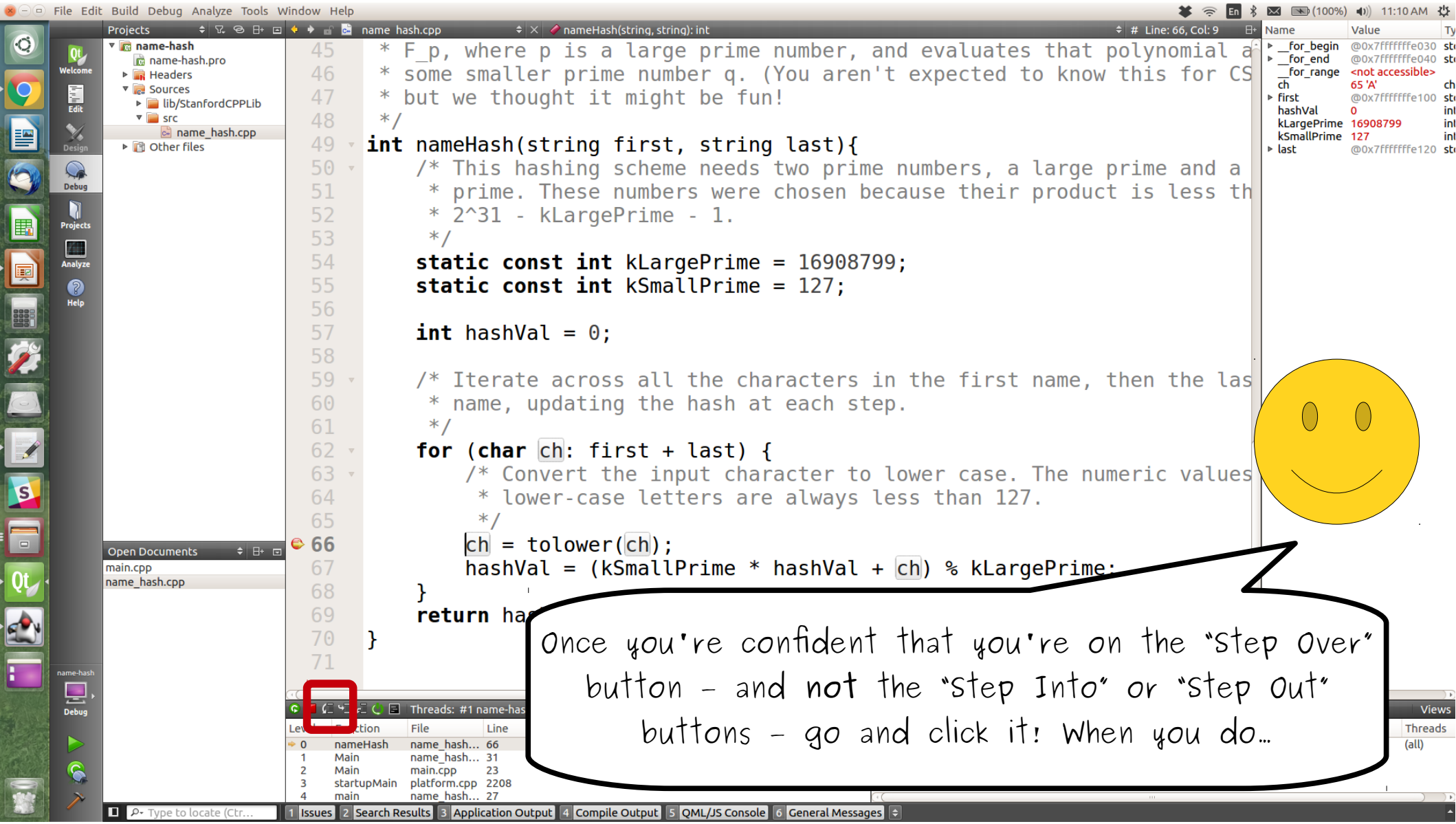

So now we know where we are (line 66), how we got there (main called nameHash), and the values in the program at this point.
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
* This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level
Function
File
Line
Number
Address
Condition
Ignore
Threads
Issues
Search Results
Application Output
Compile Output
QML/JS Console
General Messages

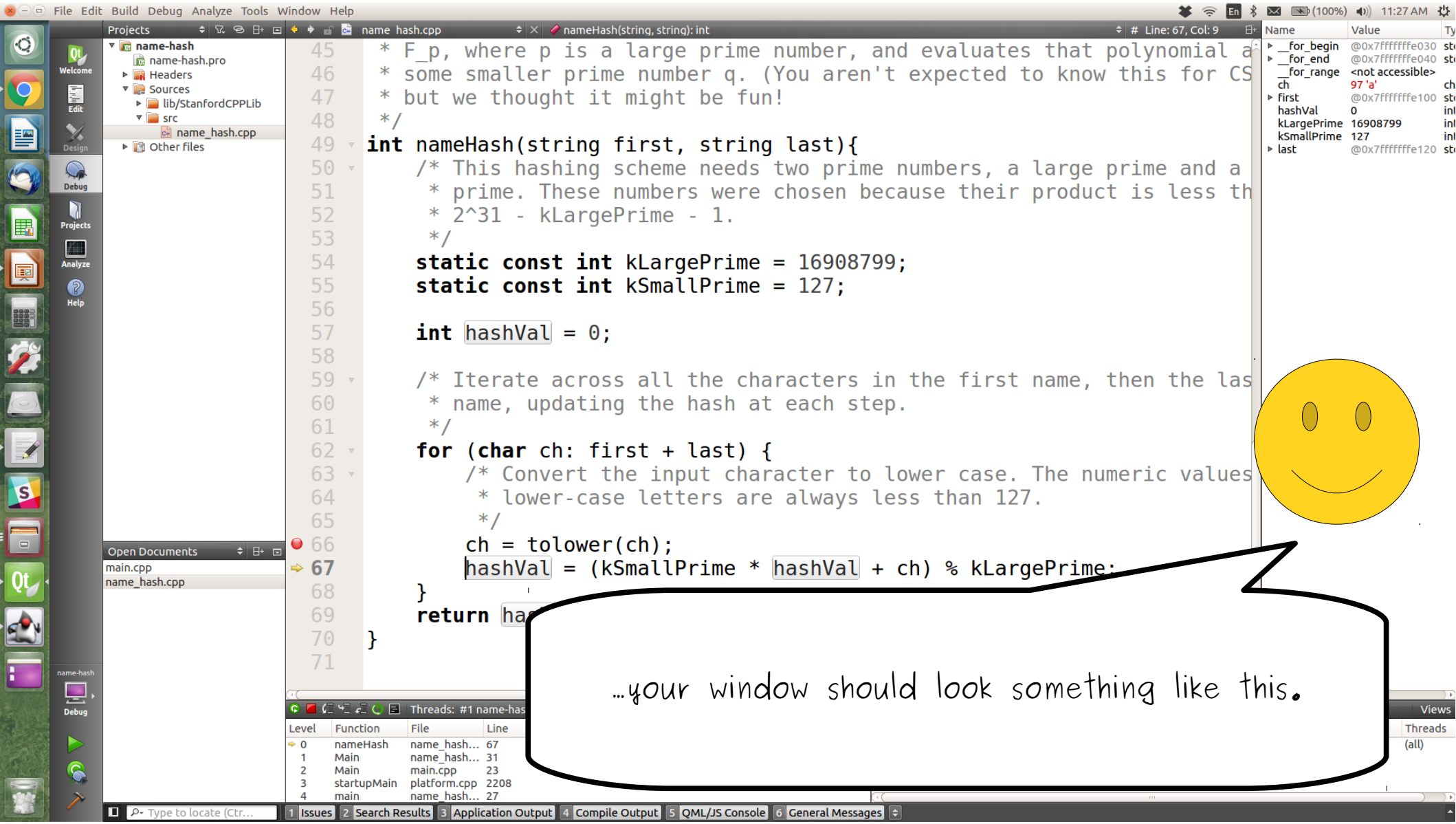

Now, let's do something really cool - we're going to run this program one line at a time, watching what happens at each step!
File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
45
46
47 * but we
48 */
49 int nameH ring first, string last){
This hashing scheme needs two prime numbers, a large prime and a
prime. These numbers were chosen because their product is less th
2^31 - kLargePrime - 1.
atic const int kLargePrime = 16908799;
tatic const int kSmallPrime = 127;
56
57 int hashVal = 0;
58
59 /* Iterate across all the characters in the first name, then the las
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values
64 * lower-case letters are always less than 127.
65 */
66 ch = tolower(ch);
67 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68 }
69 return hashVal;
70 }
71
Name Value
__for_begin @0x7fffffffe030
__for_end @0x7fffffffe040
__for_range <not accessible>
ch 65 'A'
first @0x7fffffffe100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffffe120
Threads: #1 name-hash
Stopped at breakpoint 1 (1) in thread 1.
Views
Level Function File Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number Function File Line Address Condition Ignore Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

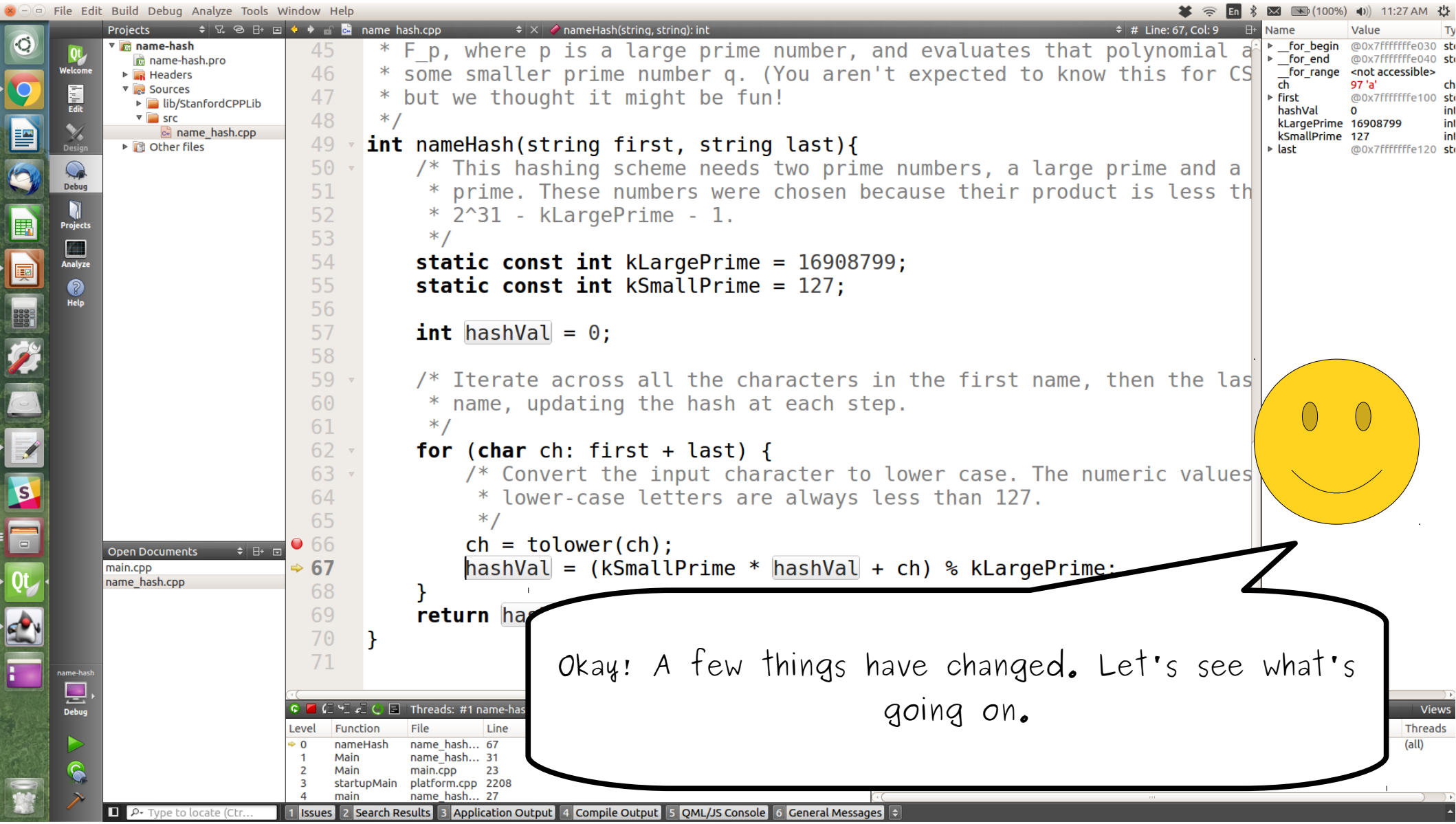

Right above the stack trace, you'll see there are some small button icons.

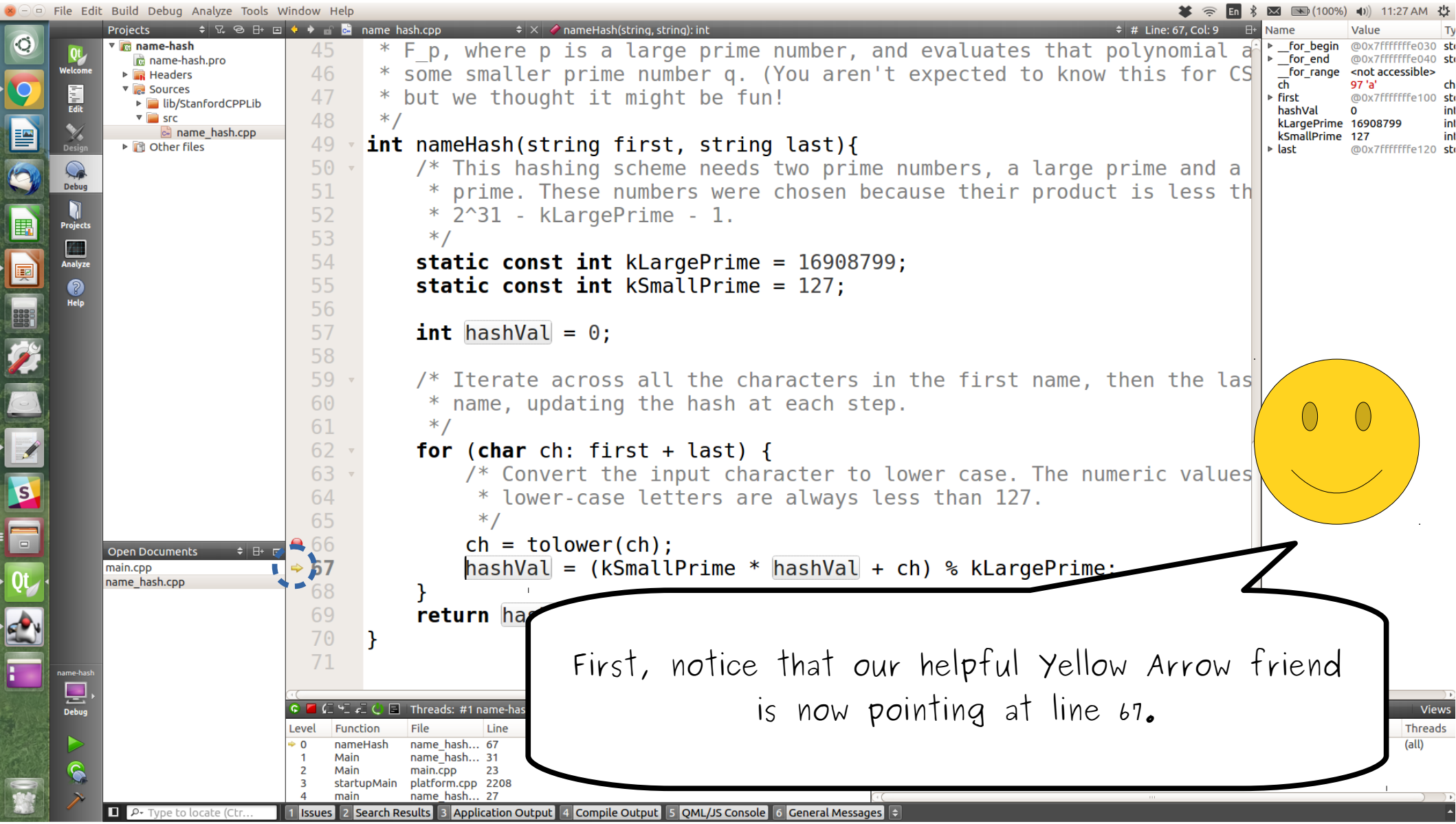

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Level
Function
Line
0 nameHash 66
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main 27
Views
Threads
(all)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages
These buttons let you resume the program, stop the program, walk through it one line at a time, etc.


Move your mouse so that you're hovering over the button that's third from the left. If you hover over it, it should say "step over."

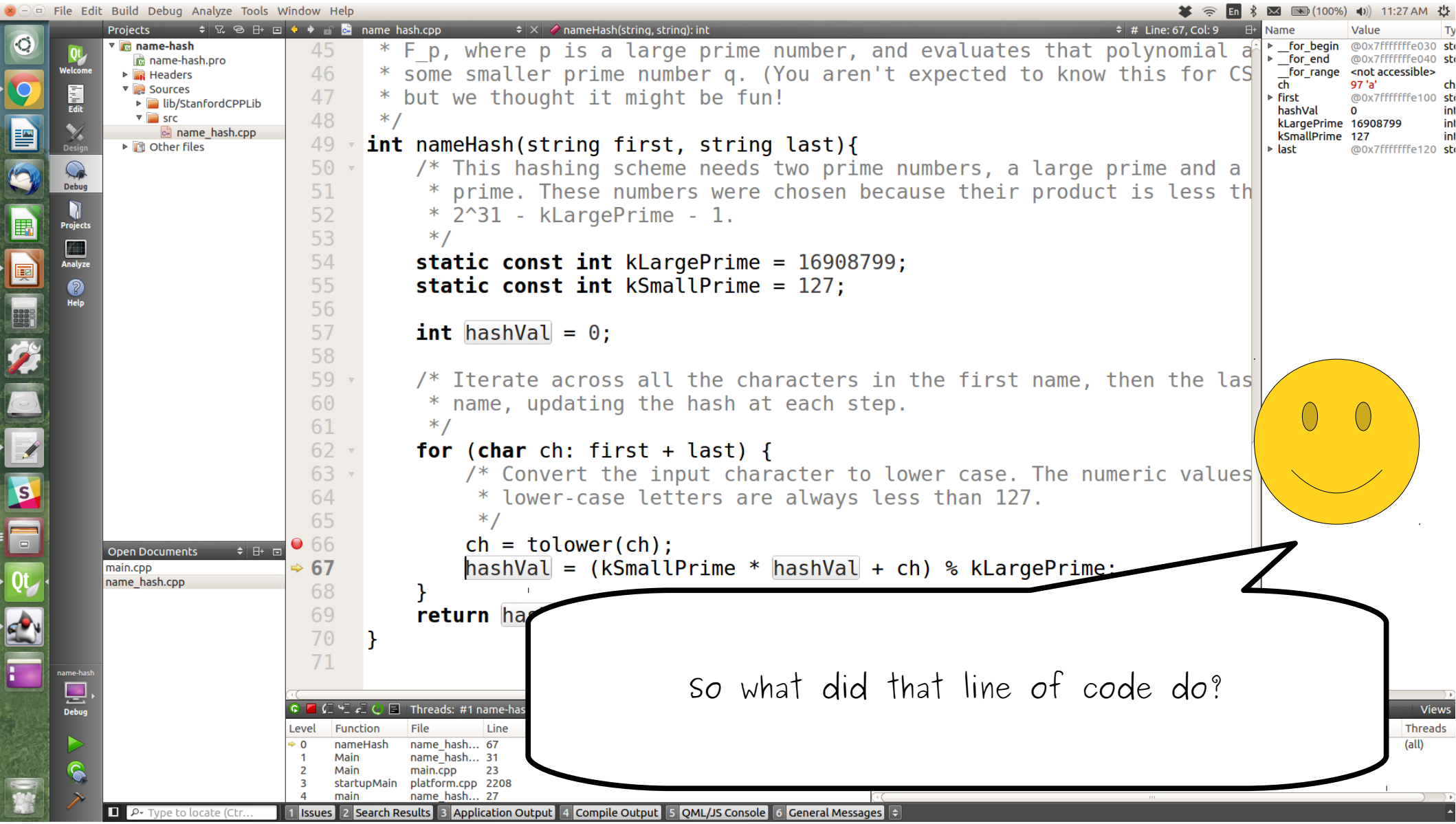

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 66, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
65 'A'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-has
Views
Threads
(all)
Level
Function
File
Line
0 nameHash name_hash... 66
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages
Once you're confident that you're on the "Step Over" button - and not the "Step Into" or "Step Out" buttons - go and click it! When you do...

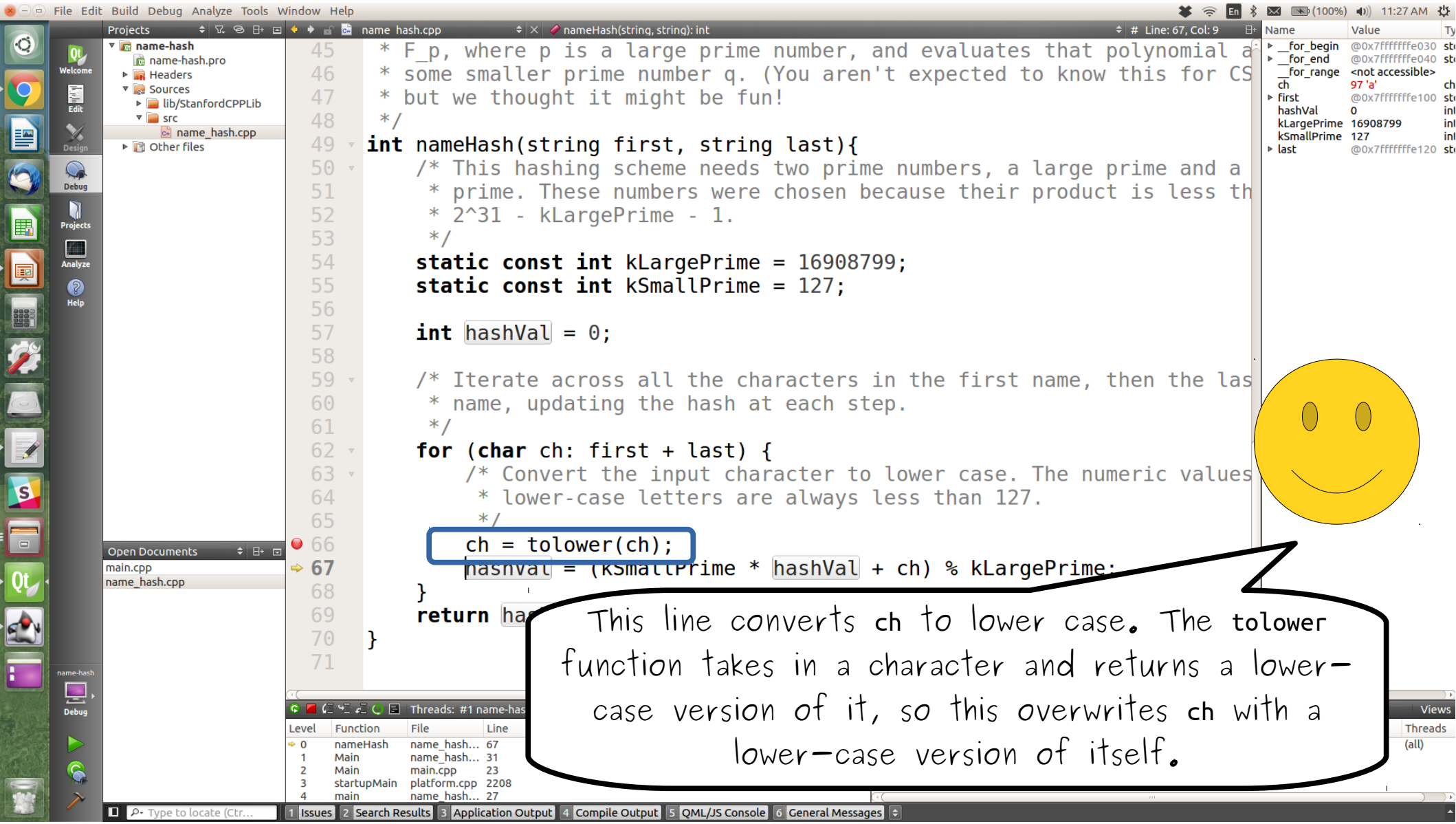

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
97 'a'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-has
Level
Function
File
Line
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Threads
(all)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages
…your window should look something like this.

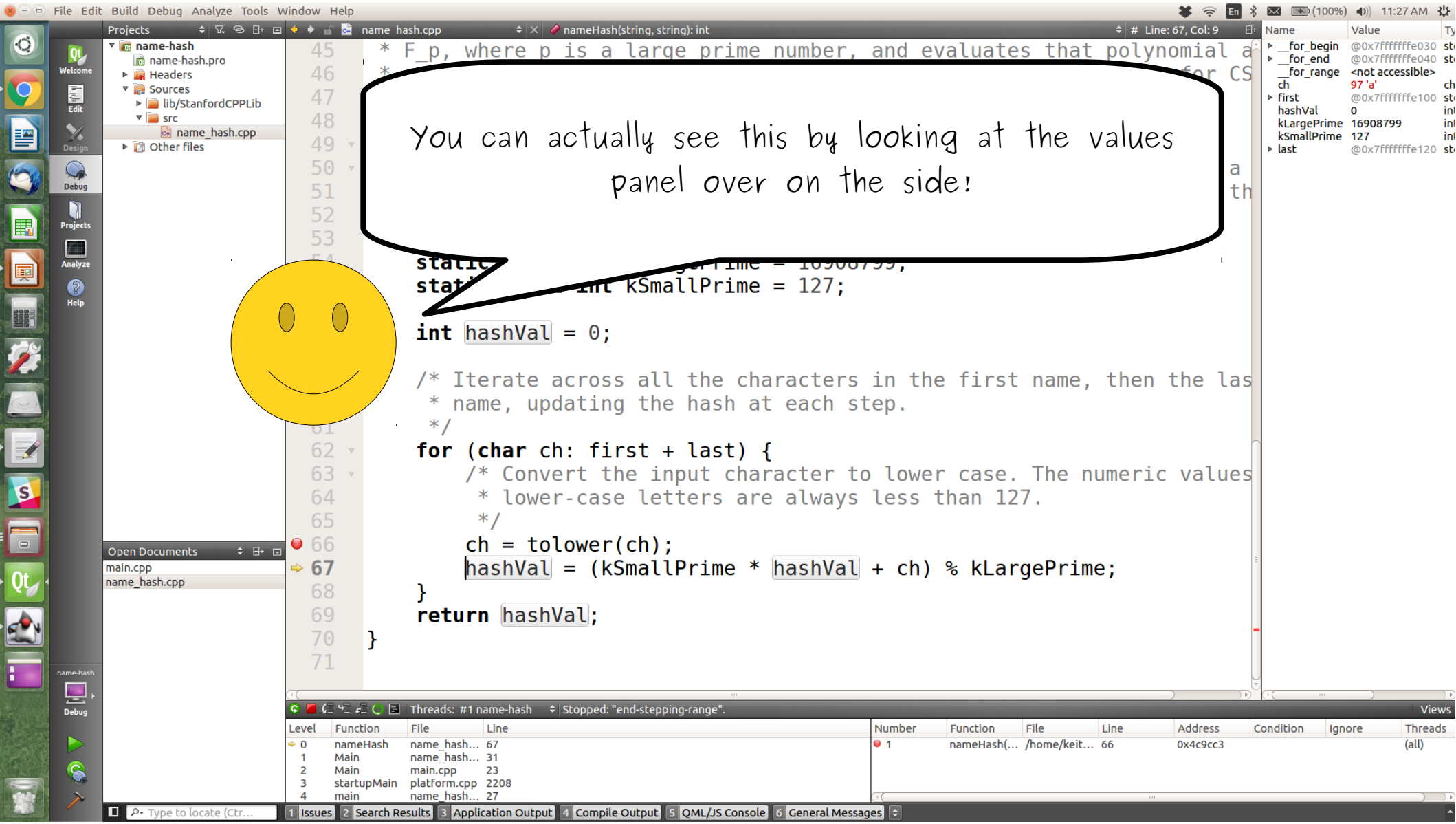

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
97 'a'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Okay! A few things have changed. Let's see what's going on.
Threads: #1 name-has
Level
Function
File
Line
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Threads
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

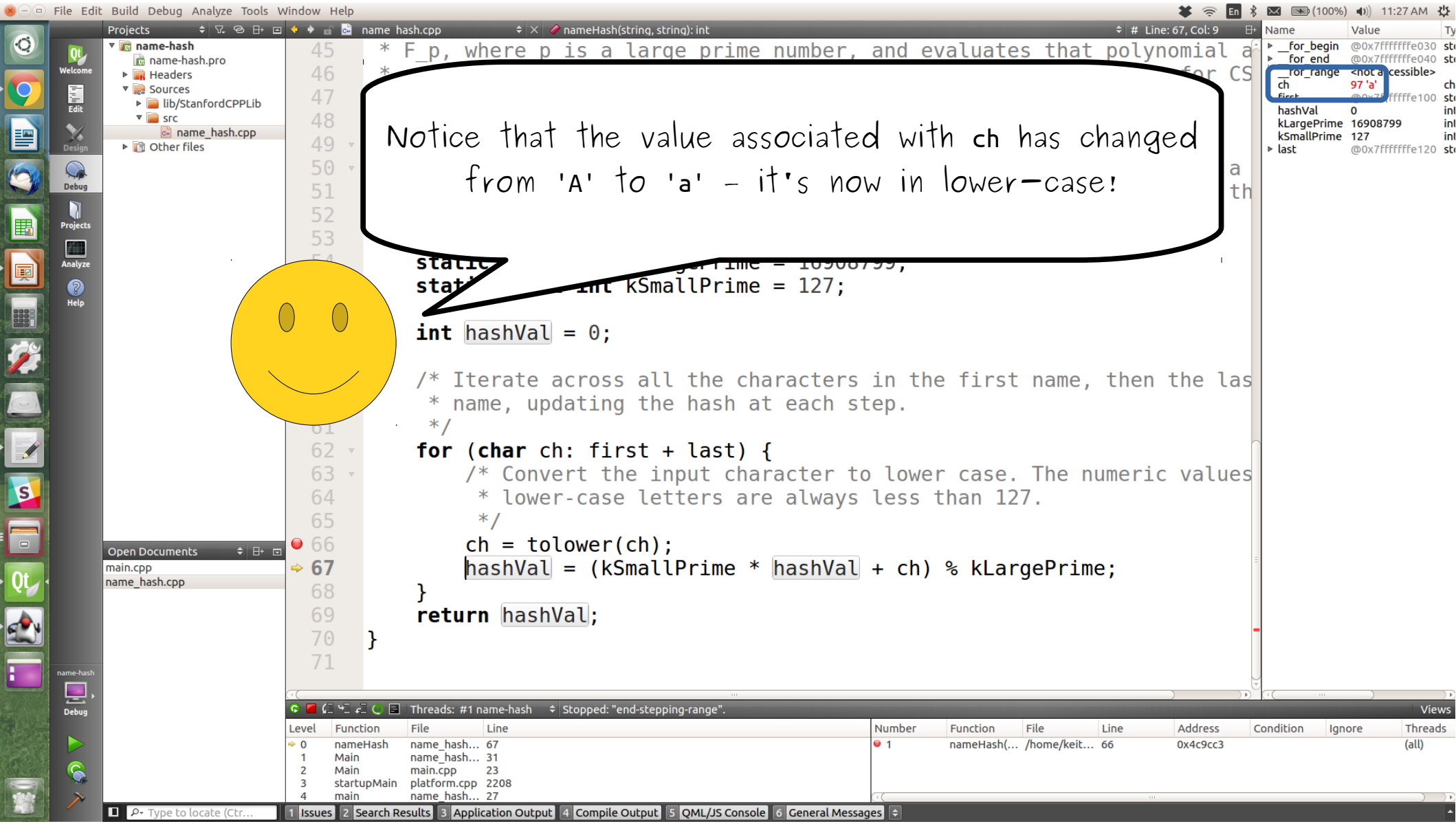

First, notice that our helpful Yellow Arrow friend is now pointing at line 67.

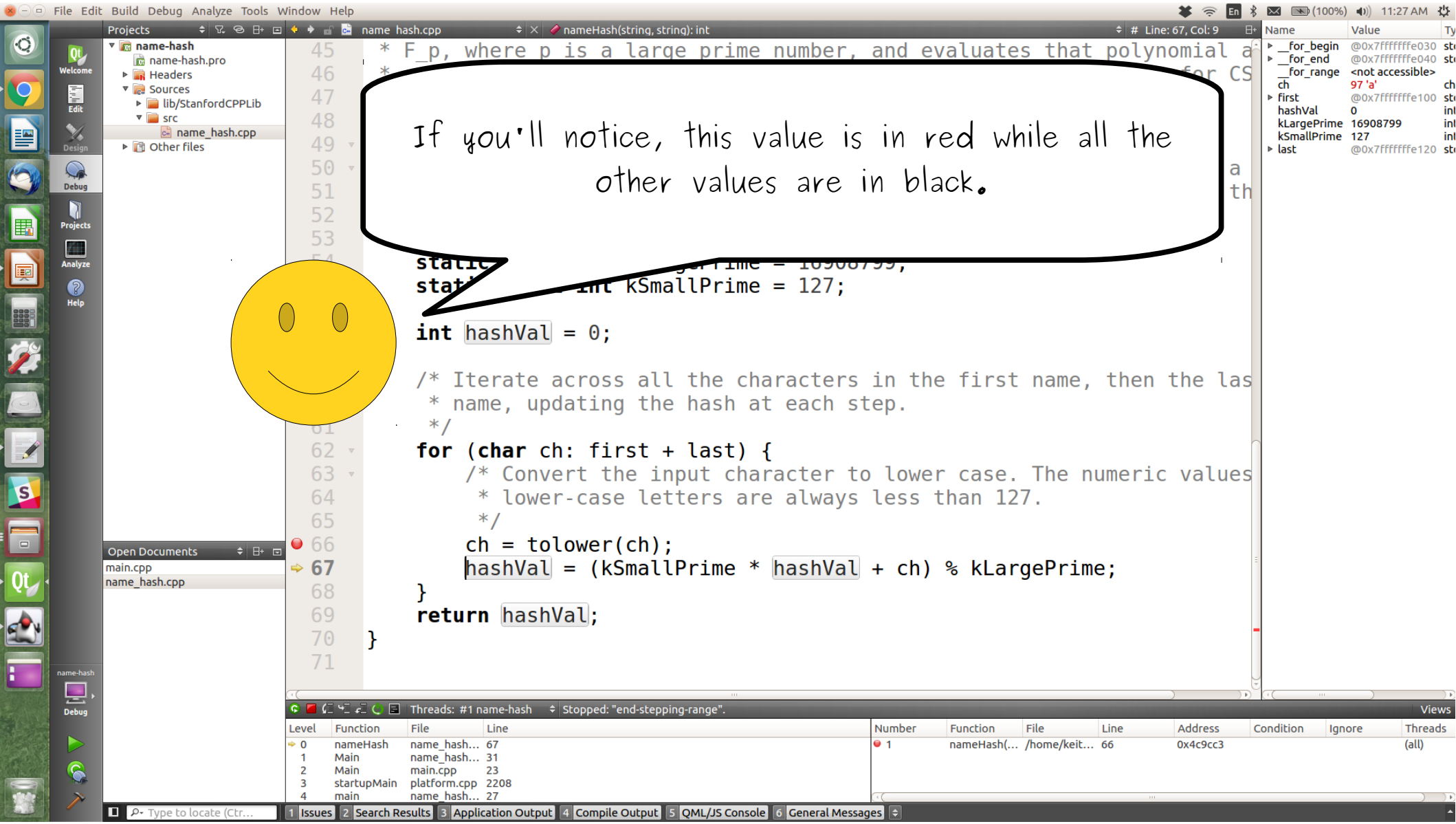

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
97 'a'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-has
Level
Function
File
Line
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Threads
(all)
We're now at the line right after the one where we stopped. You just ran a single line of the program! Pretty cool!
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

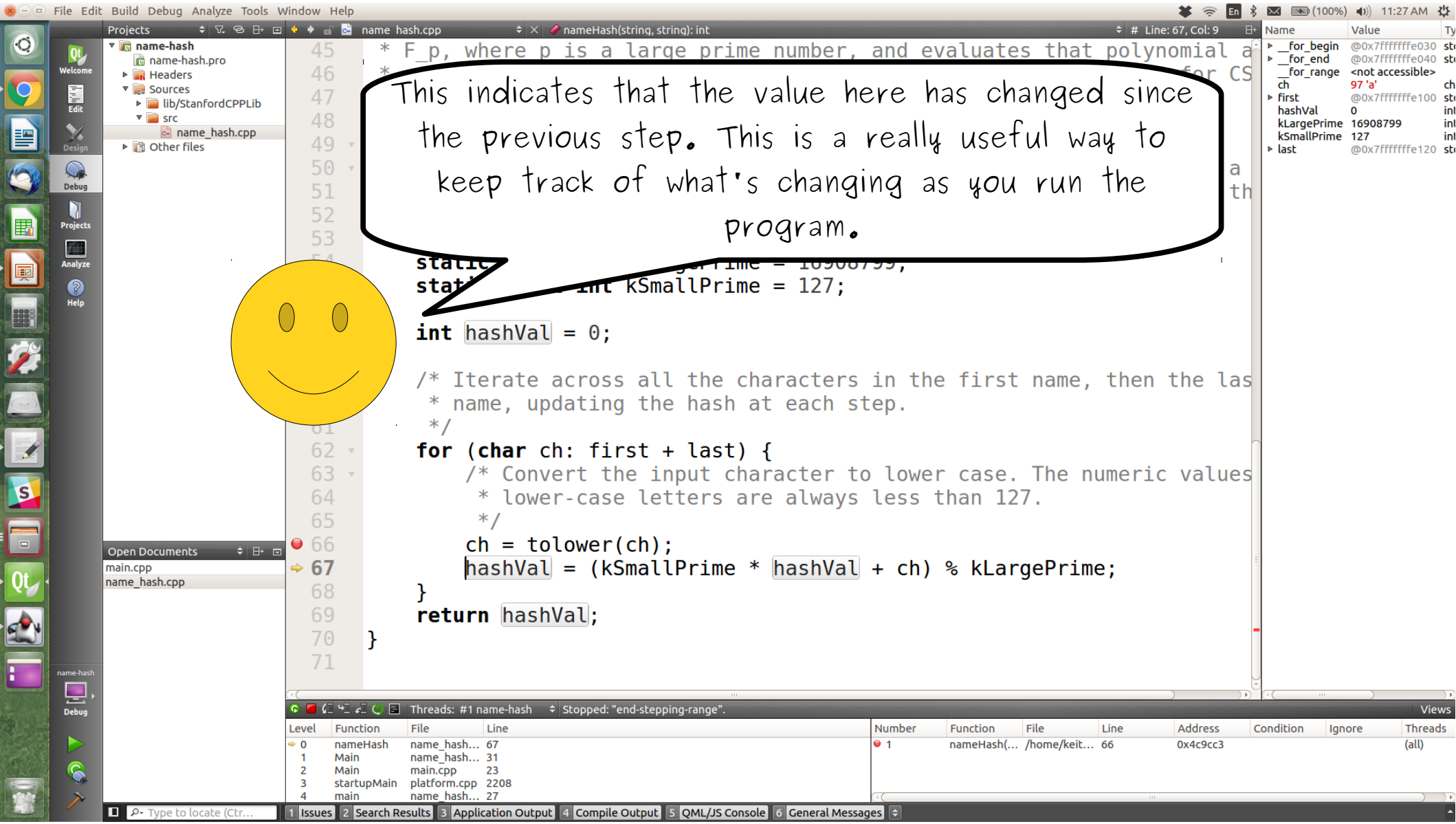

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
Threads: #1 name-has
Level
Function
File
Line
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Threads
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages
so what did that line of code do?

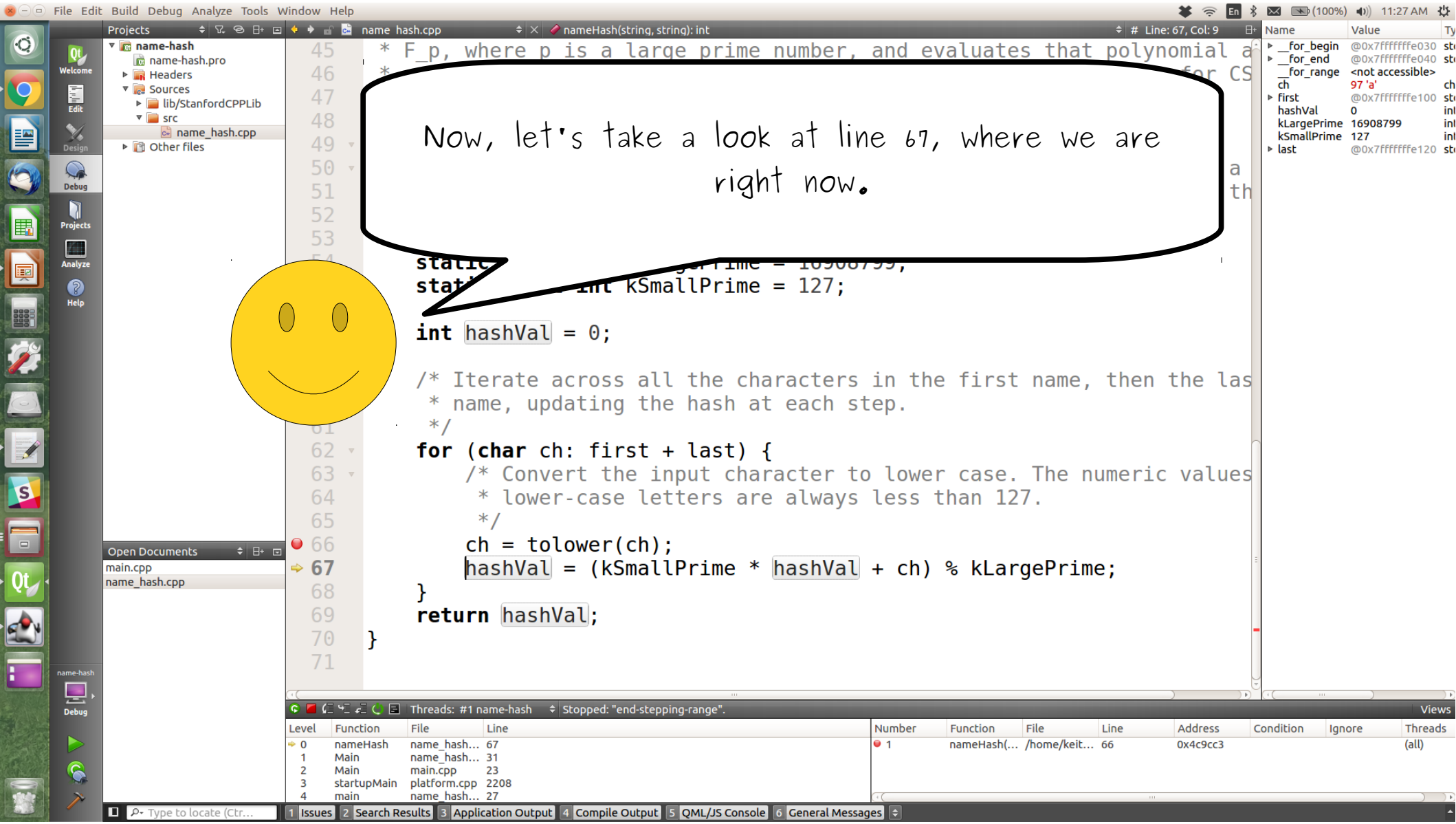

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return ha
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
97 'a'
@0x7fffffffe100
0
16908799
127
@0x7fffffffe120
Threads: #1 name-has
Level
Function
File
Line
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Threads
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages
This line converts ch to lower case. The tolower function takes in a character and returns a lower-case version of it, so this overwrites ch with a lower-case version of itself.

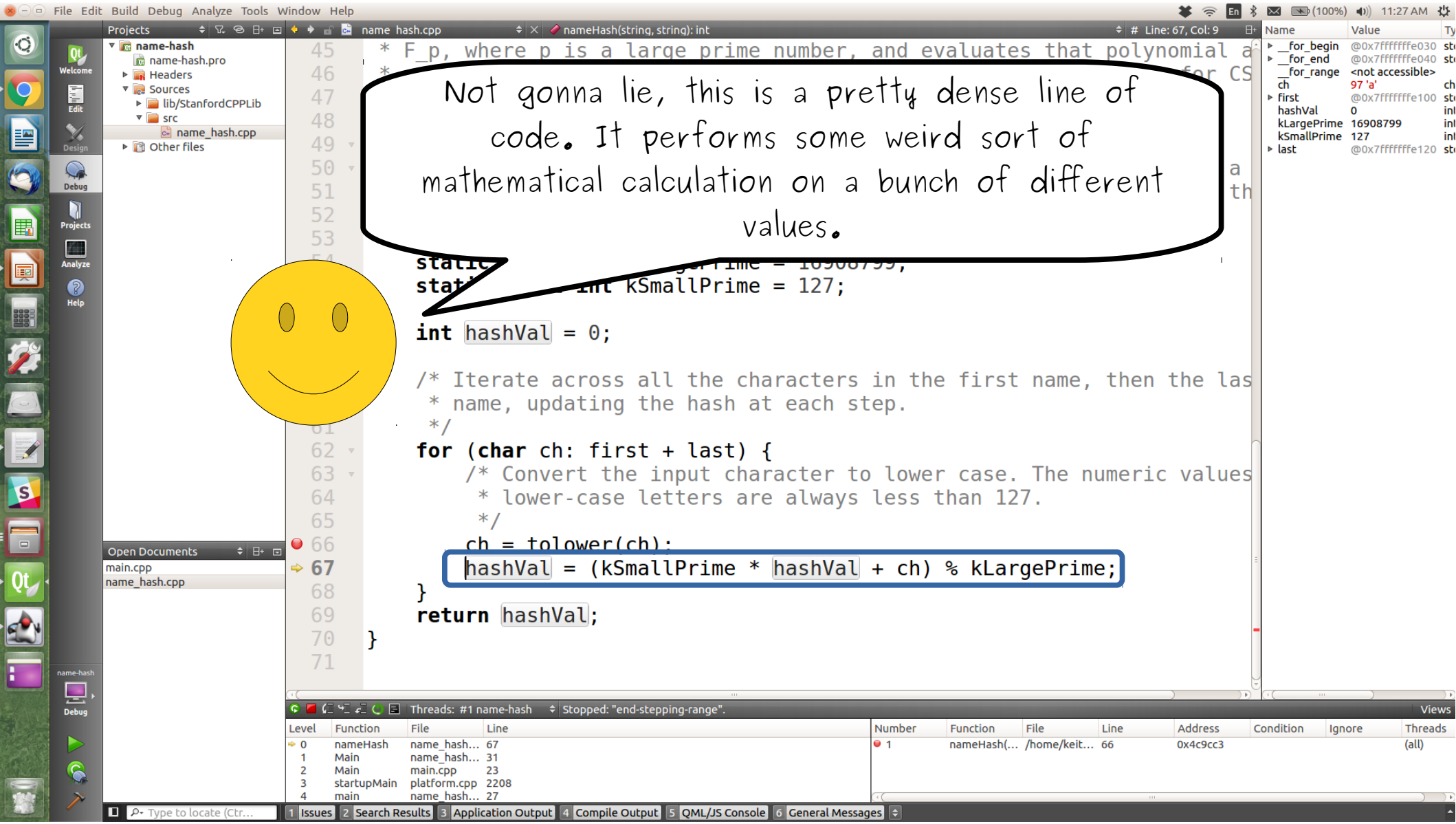

You can actually see this by looking at the values panel over on the side!
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}

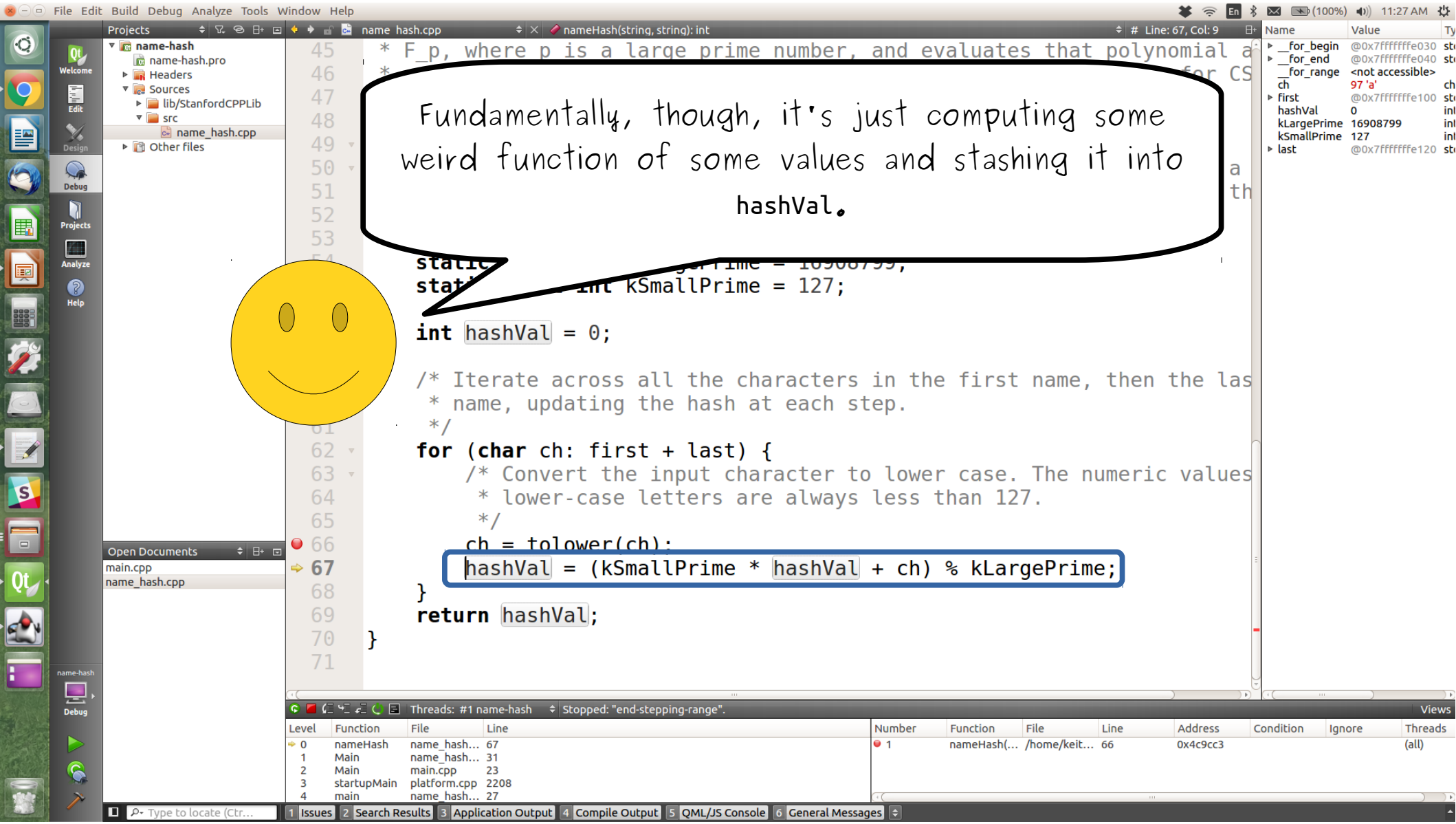

Notice that the value associated with ch has changed from 'A' to 'a' - it's now in lower-case!
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
int kSmallPrime = 127;
name_hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
Name
Value
__for_begin
__for_end
__for_range
ch
97 'a'
hashVal
0
kLargePrime
16908799
kSmallPrime
127
last
Threads: #1 name-hash
Stopped: "end-stepping-range".
Level
Function
File
Line
Number
Address
Condition
Ignore
Threads

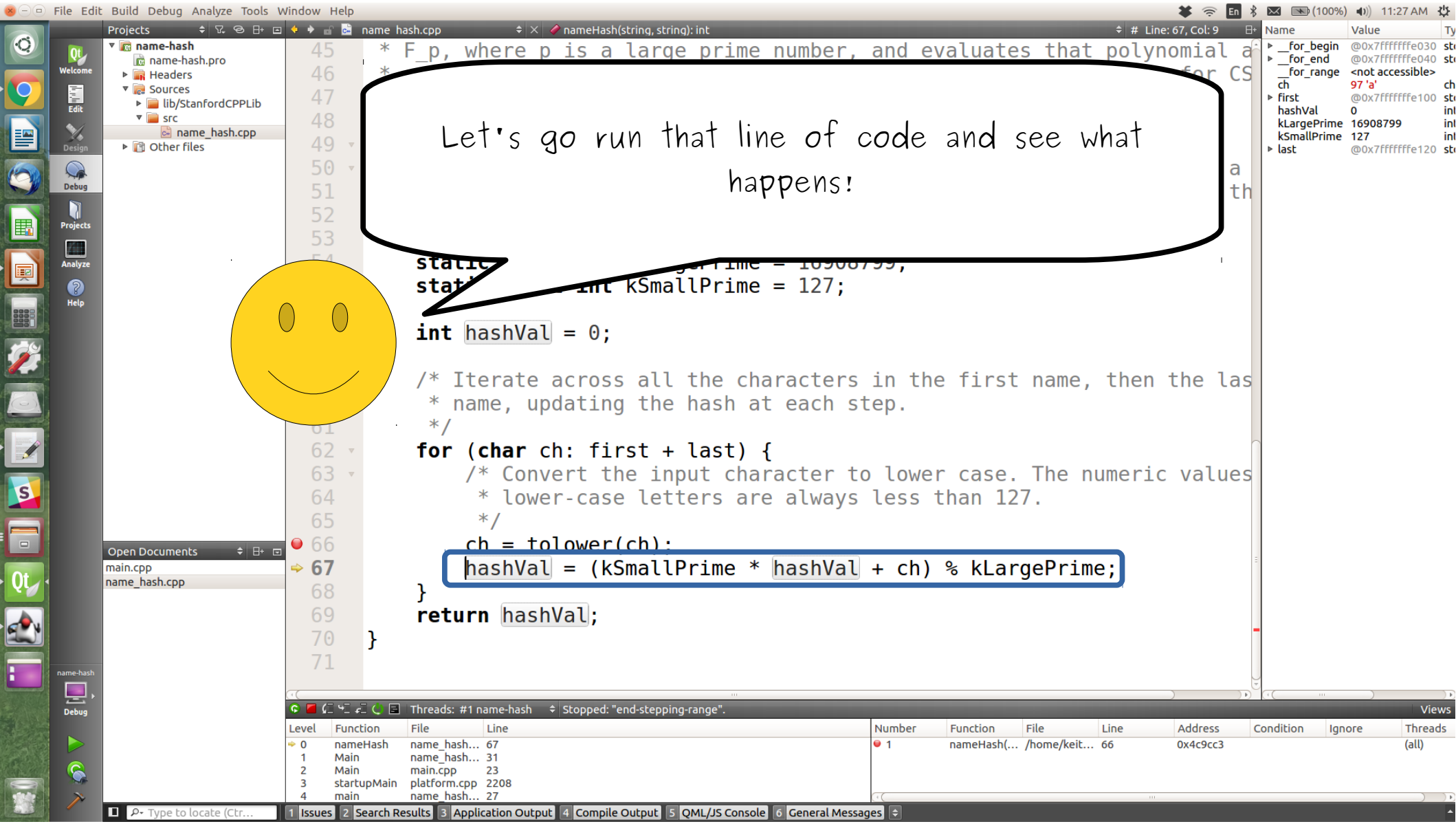

If you'll notice, this value is in red while all the other values are in black.
int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}

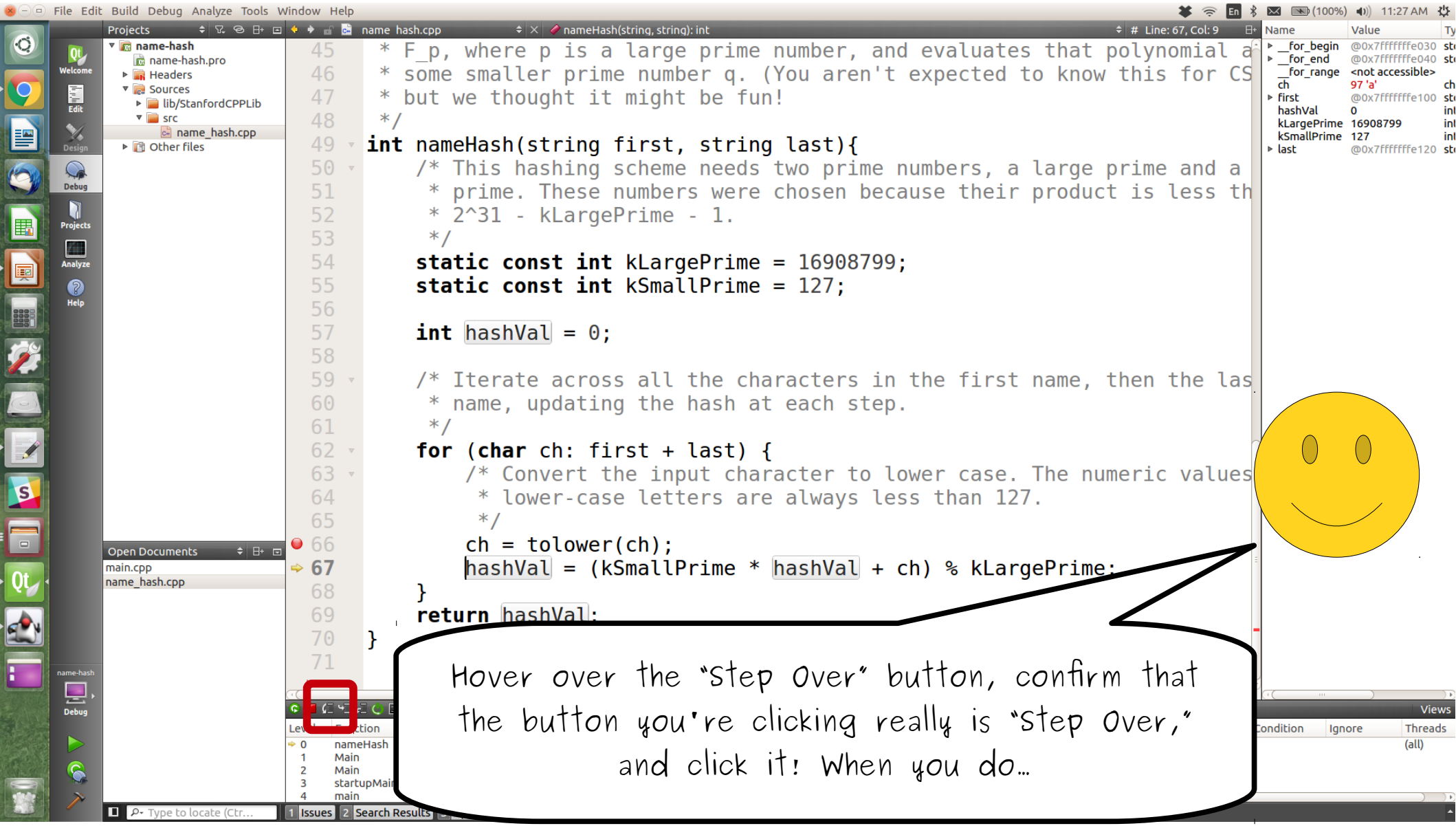

This indicates that the value here has changed since the previous step. This is a really useful way to keep track of what's changing as you run the program.
int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}

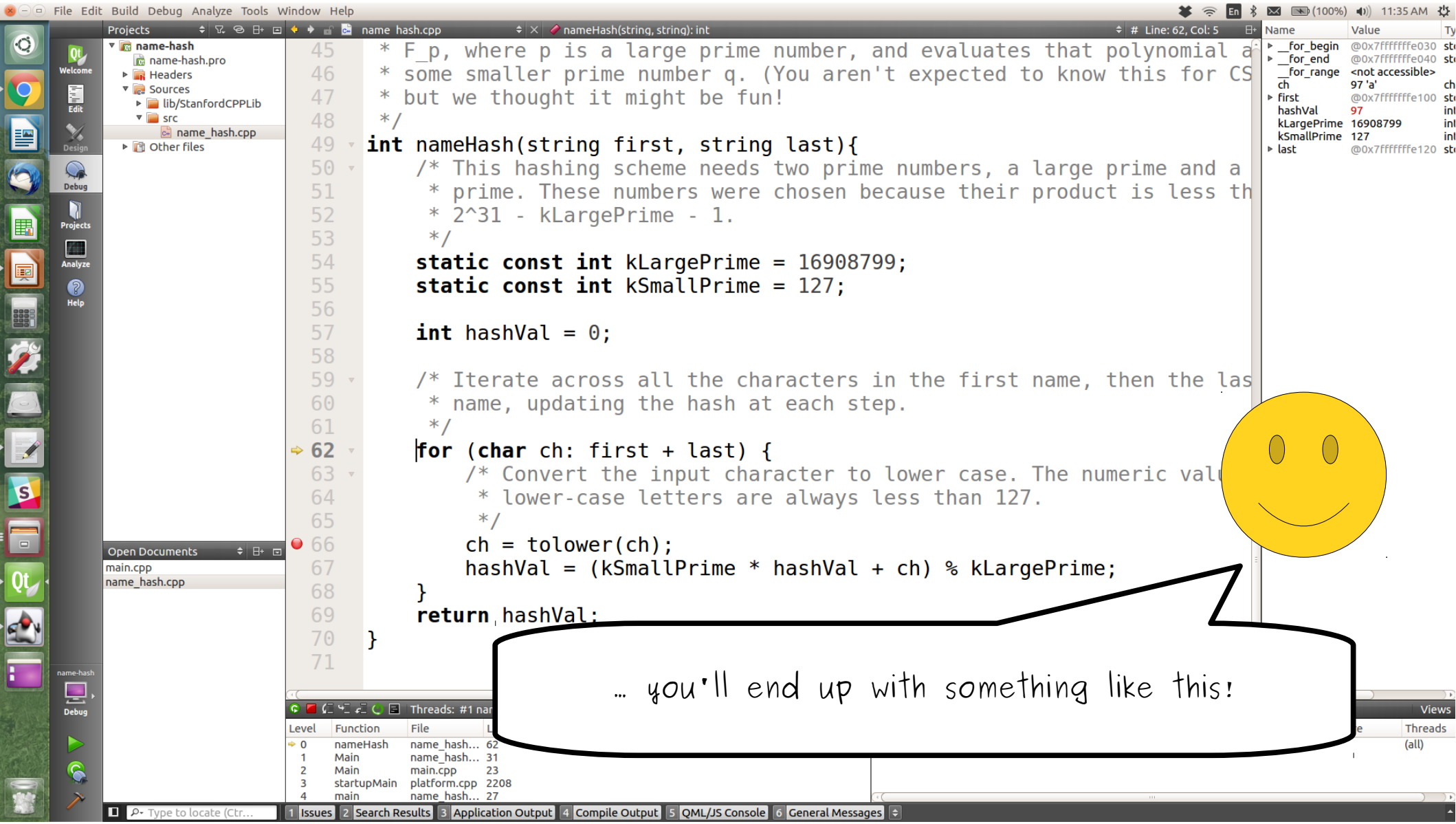

Now, let's take a look at line 67, where we are right now.
int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}

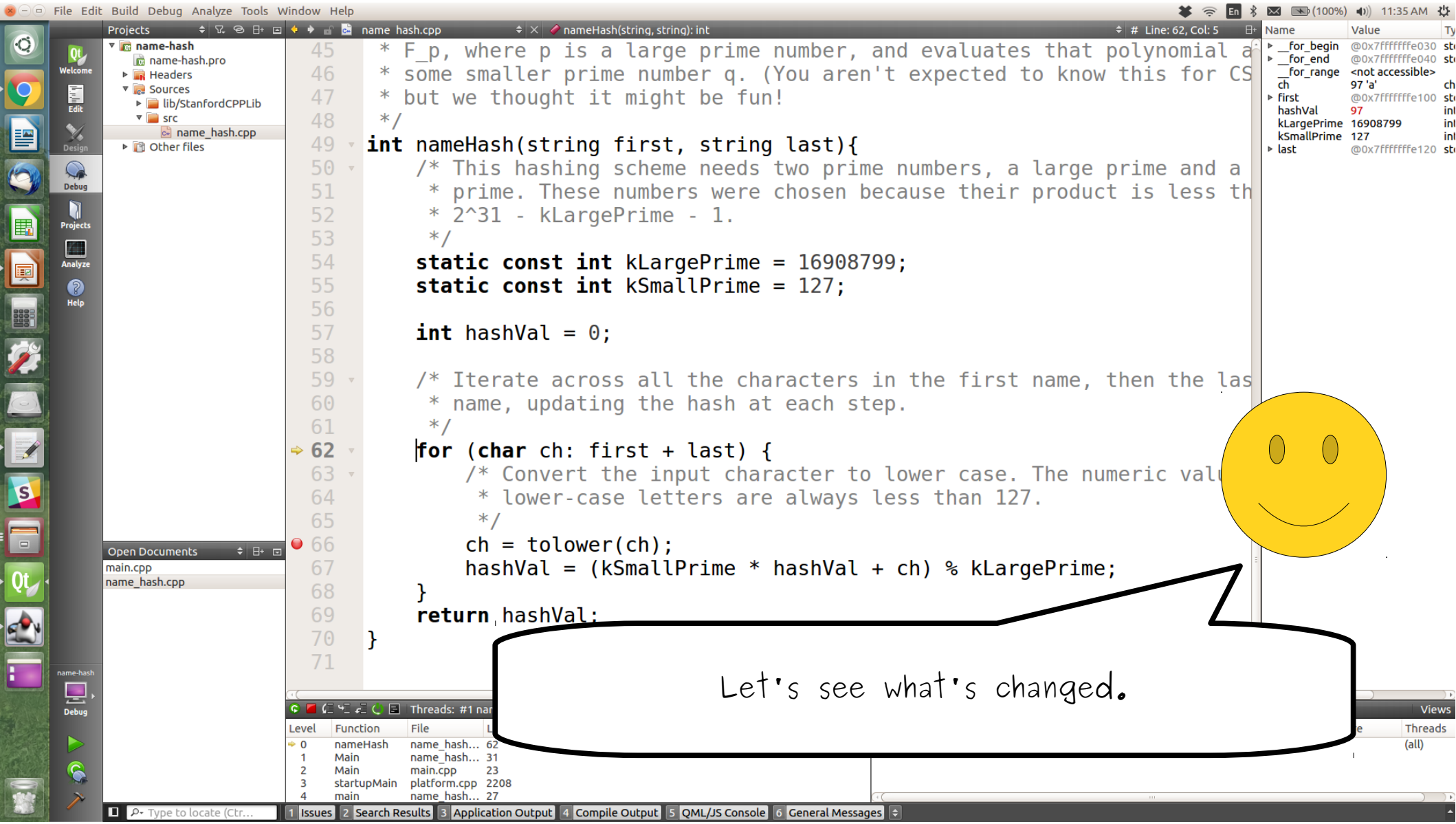

Not gonna lie, this is a pretty dense line of code. It performs some weird sort of mathematical calculation on a bunch of different values.
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}


Fundamentally, though, it's just computing some weird function of some values and stashing it into hashVal.
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}

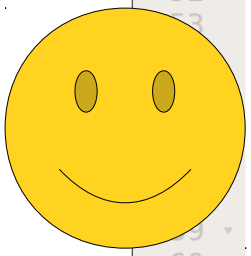

Let's go run that line of code and see what happens!
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric values
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}


Hover over the "Step Over" button, confirm that the button you're clicking really is "Step Over," and click it! When you do…

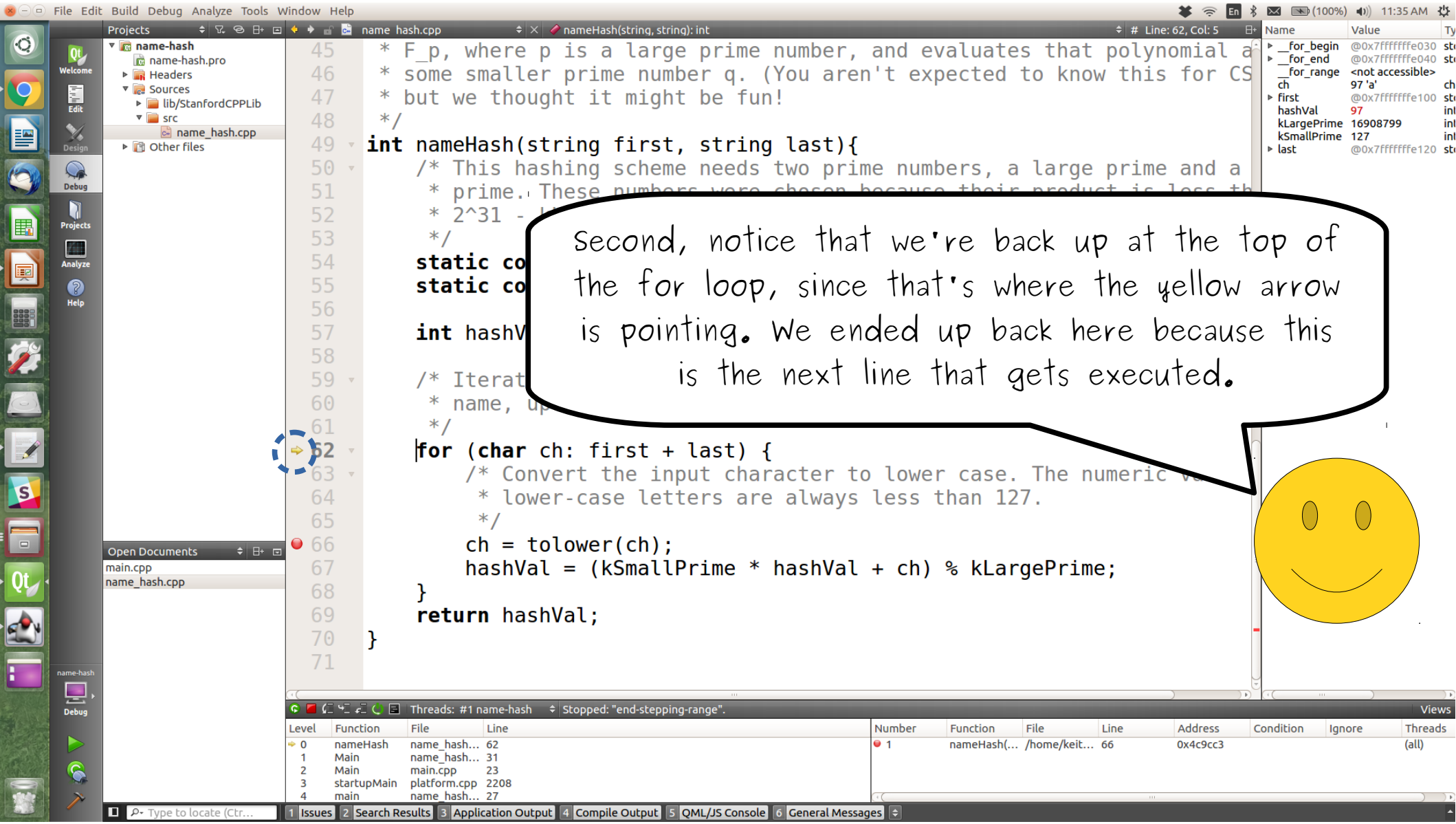

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 62, Col: 5
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric valu
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
97 'a'
@0x7fffffffe100
97
16908799
127
@0x7fffffffe120
Threads: #1
Level
Function
File
0 nameHash name_hash... 62
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Threads
(all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages
… you'll end up with something like this!

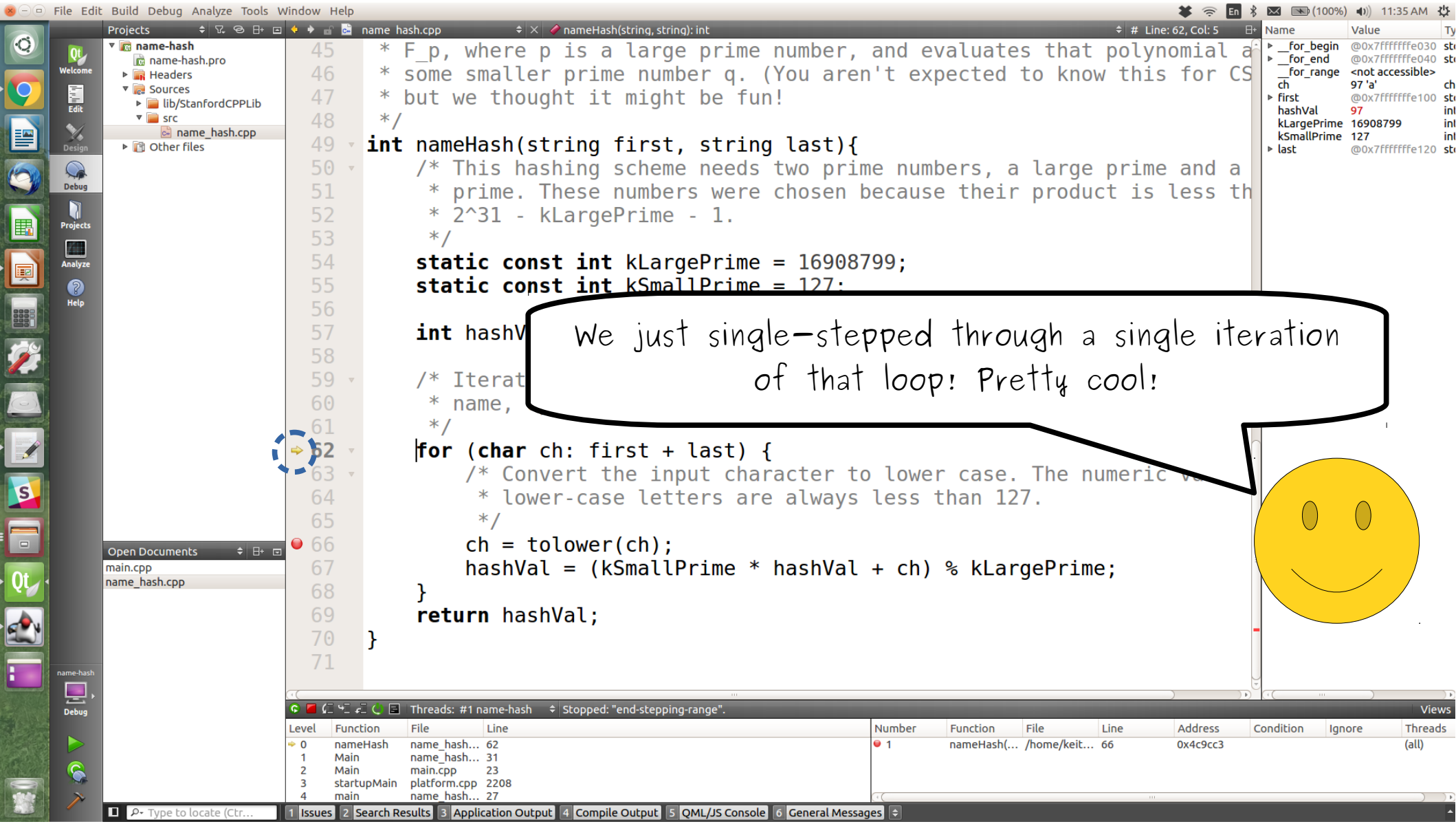

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 62, Col: 5
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric val
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
97 'a'
@0x7fffffffe100
97
16908799
127
@0x7fffffffe120
Let's see what's changed.
Level
Function
File
0 nameHash name_hash...
1 Main name_hash...
2 Main main.cpp
3 startupMain platform.cpp
4 main name_hash...
Threads
(all)
Views
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

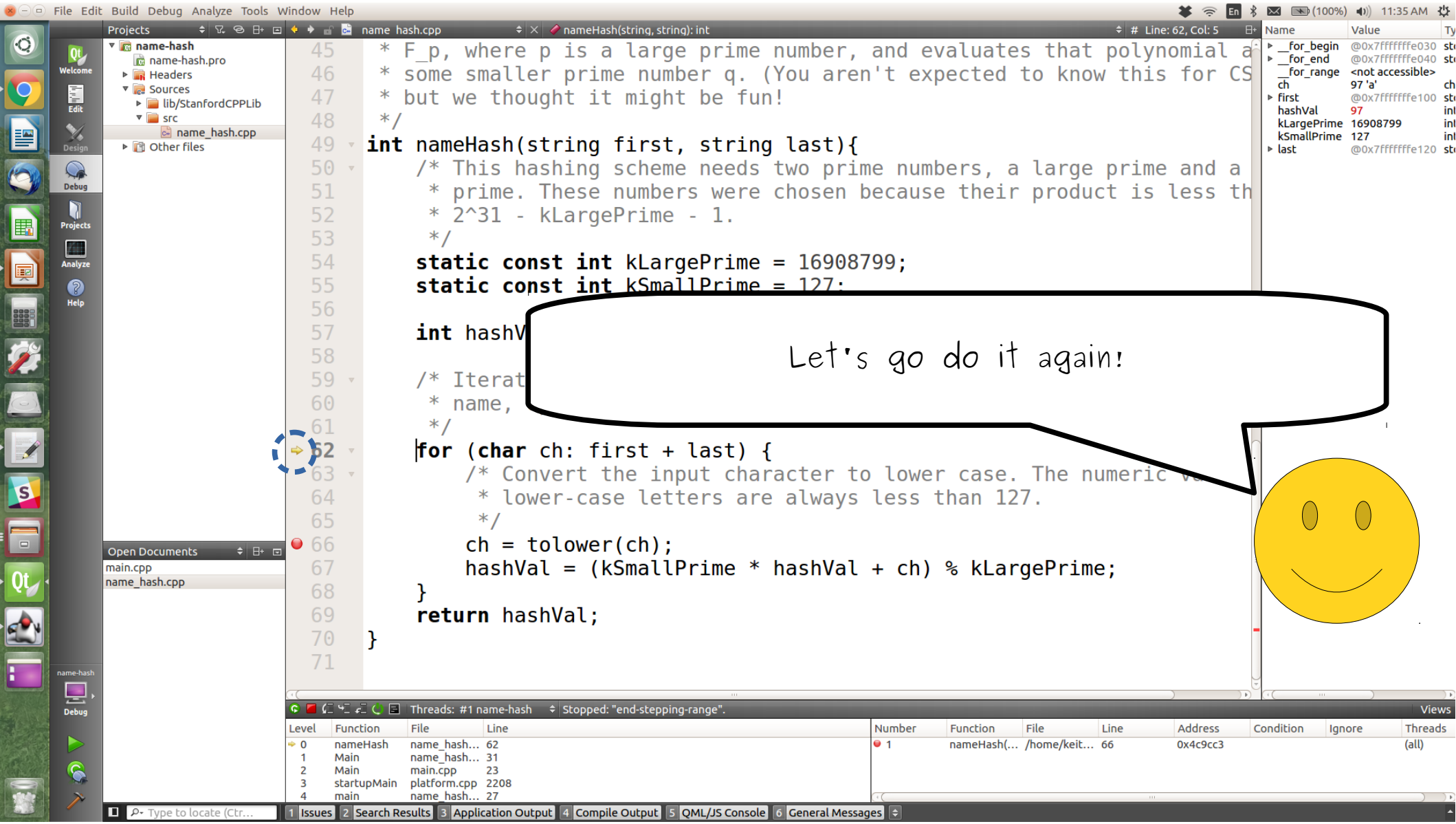
First, notice that the value stored in hashVal changed to 97. We know that it changed because the value is in red, and we know that nothing else changed because nothing else is in red!

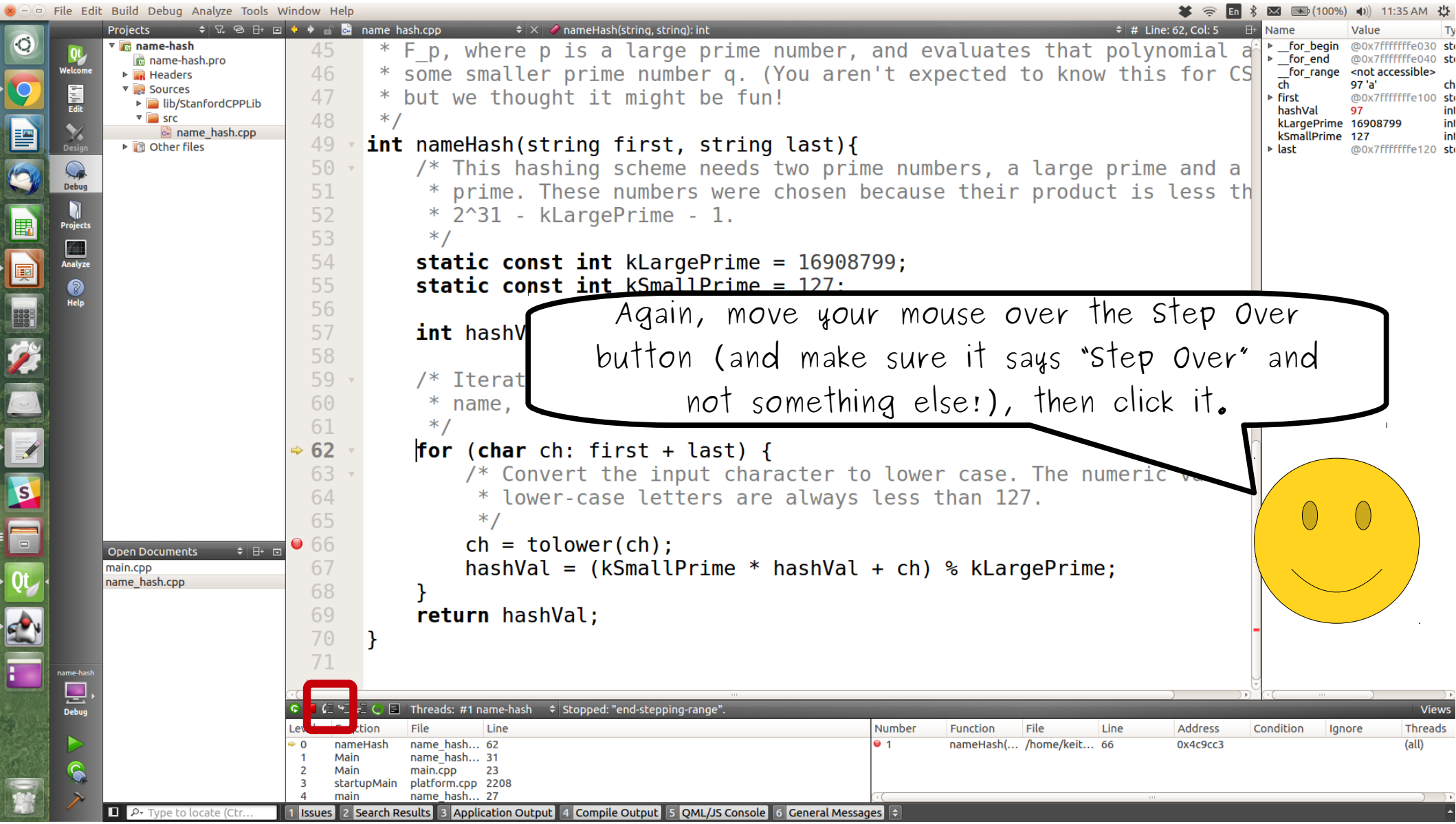


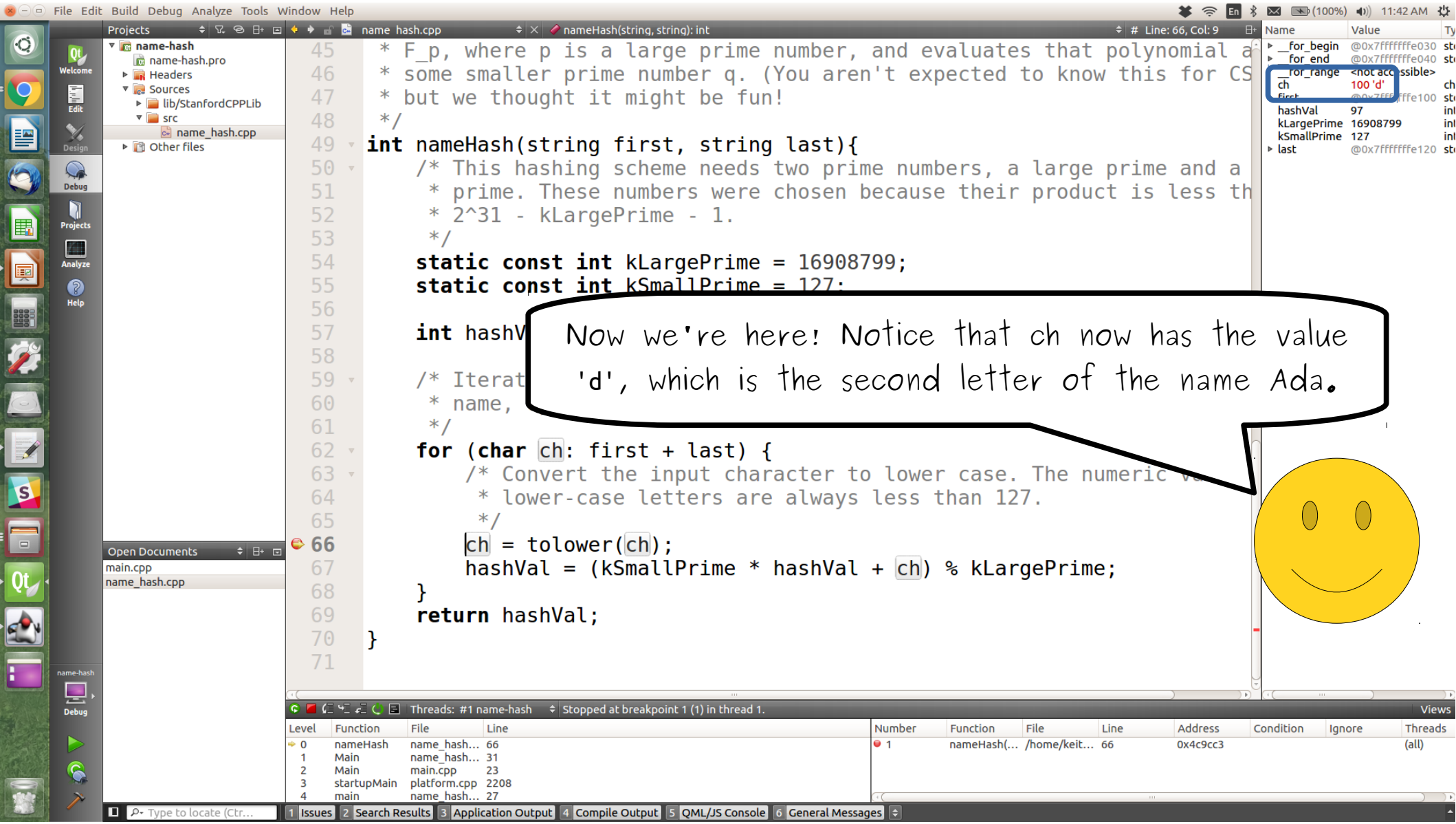
```
    static const int kLargePrime = 16908799;
    static const int kSmallPrime = 127;

    int hashVal = 0;

    /* Iterate across all the characters in the first name, then the las
     * name, updating the hash at each step.
     */
    for (char ch: first + last) {
        /* Convert the input character to lower case. The numeric values
         * lower-case letters are always less than 127.
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
    return hashVal;
}
```

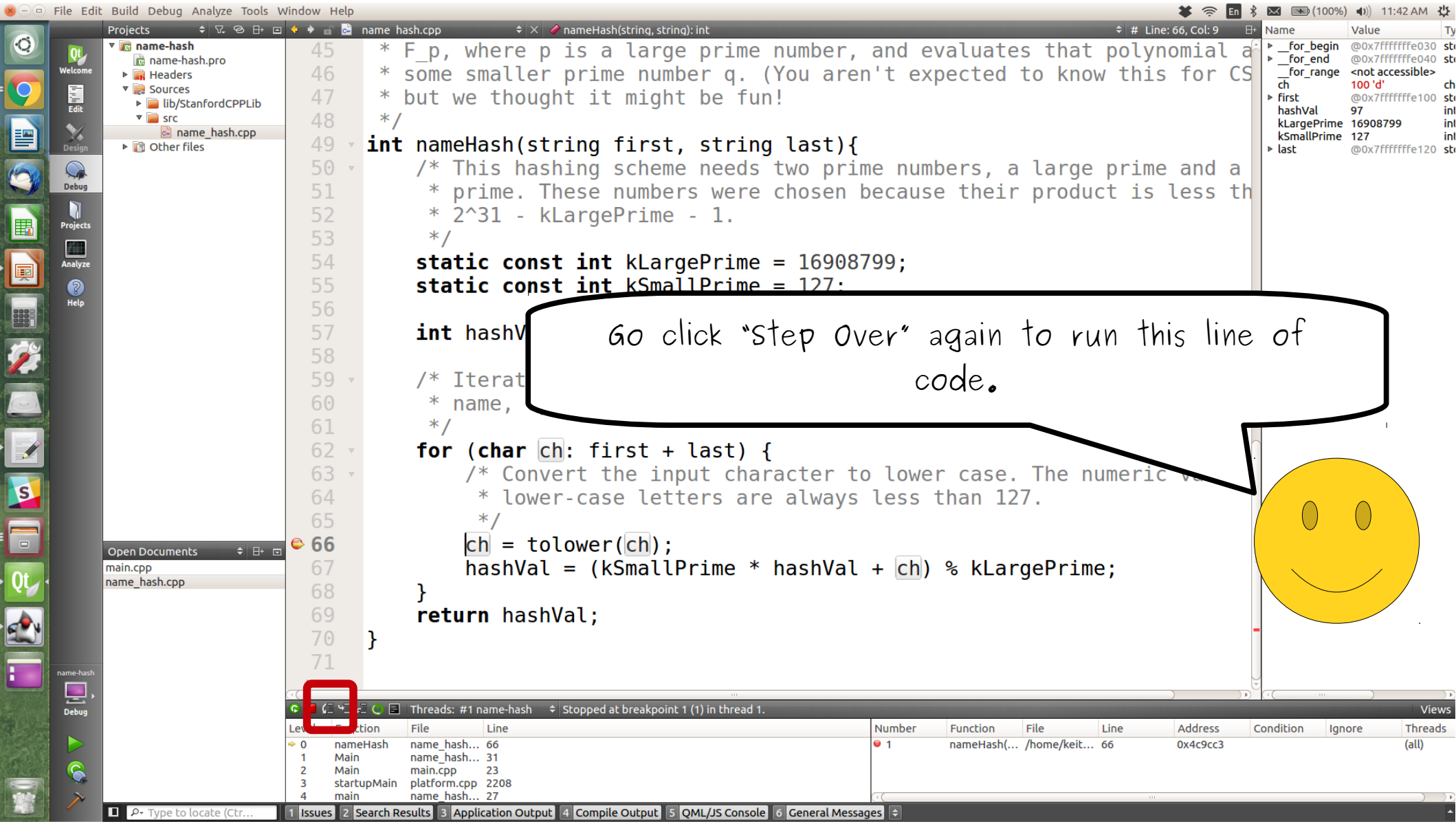

Second, notice that we're back up at the top of the for loop, since that's where the yellow arrow is pointing. We ended up back here because this is the next line that gets executed.
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These
* 2^31 -
*/
static co
static co
int hashV
/* Iterat
* name, u
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}

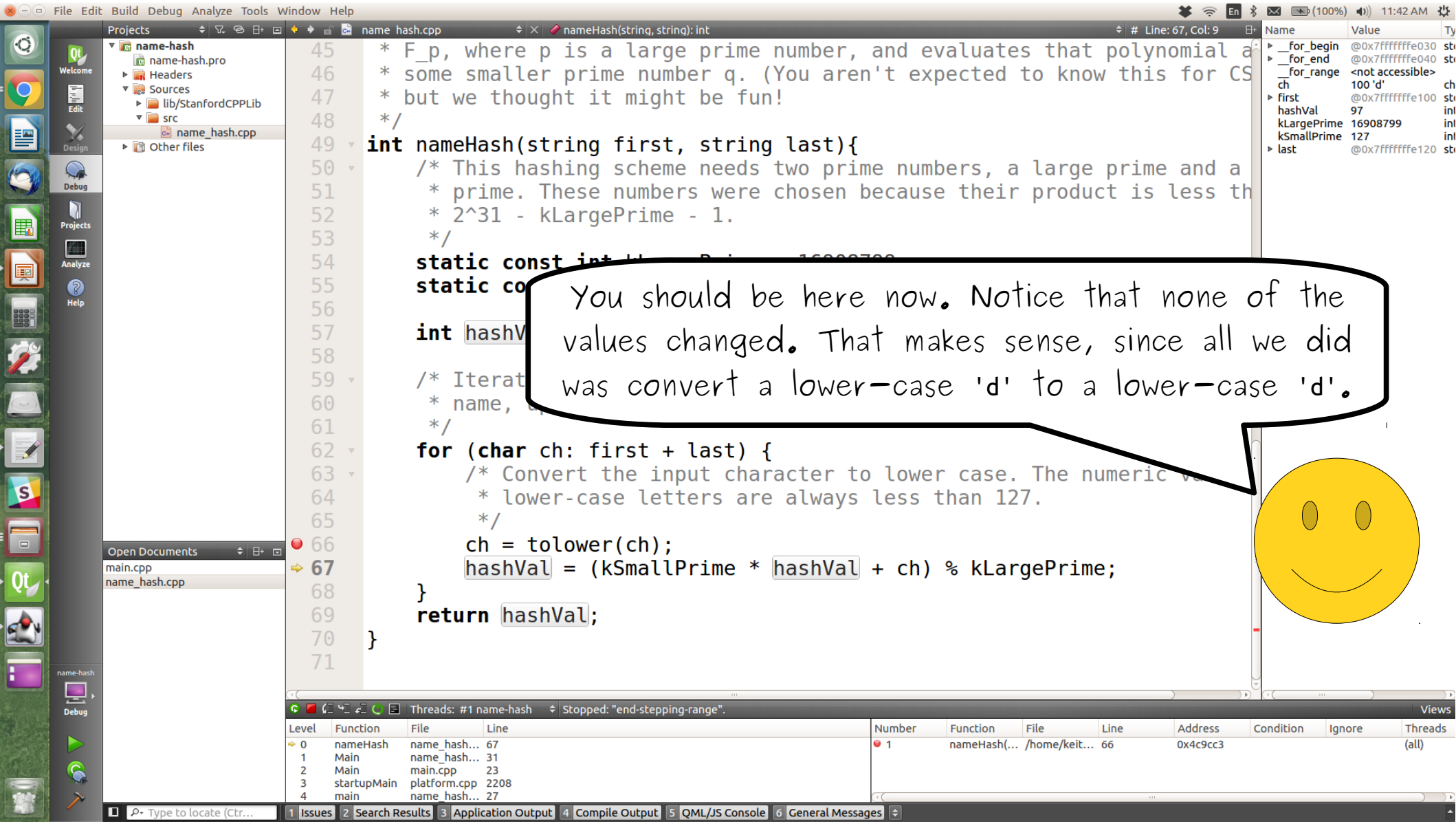

We just single-stepped through a single iteration of that loop! Pretty cool!

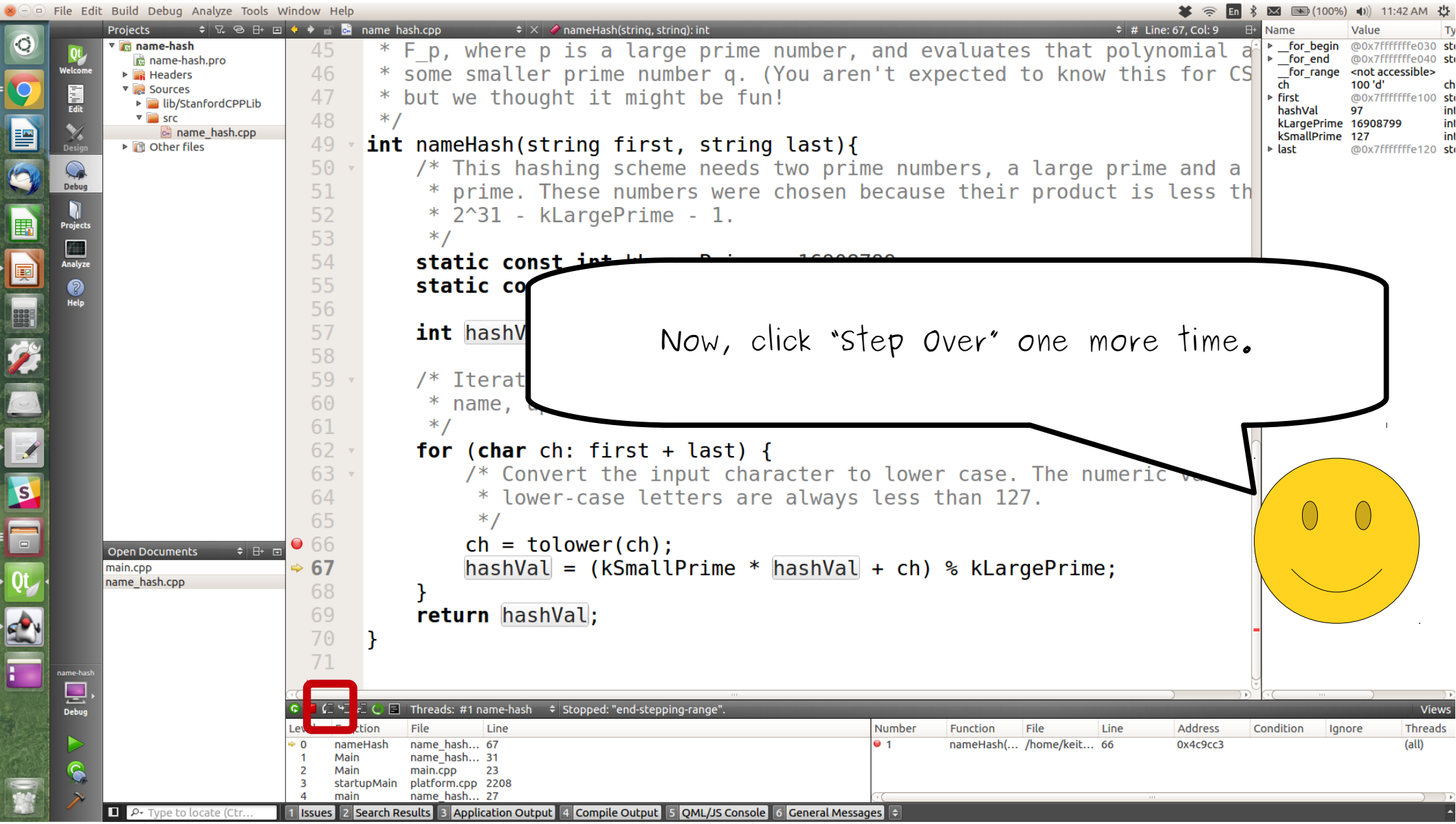

Let's go do it again!

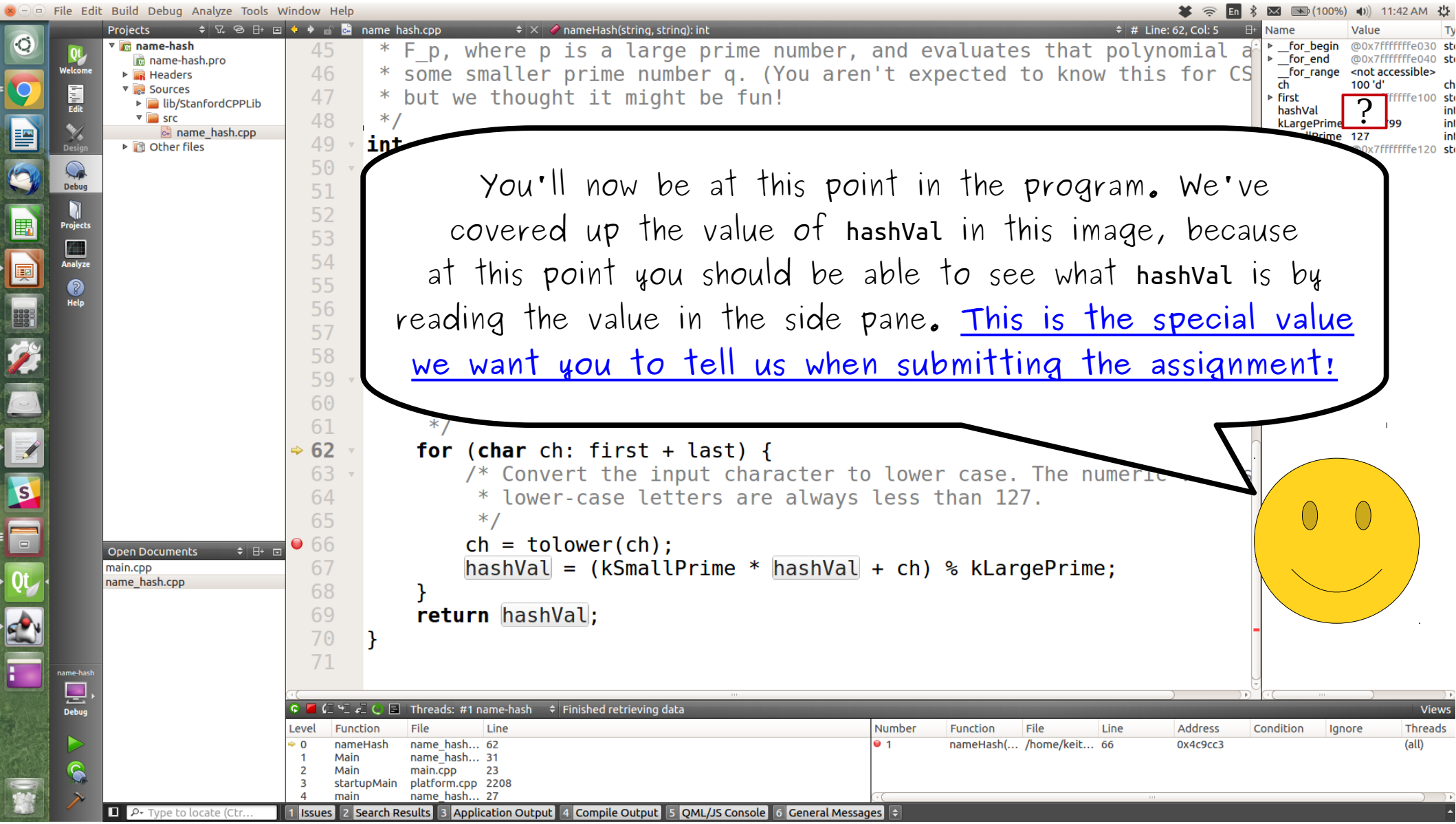

Again, move your mouse over the Step Over button (and make sure it says "Step Over" and not something else!), then click it.

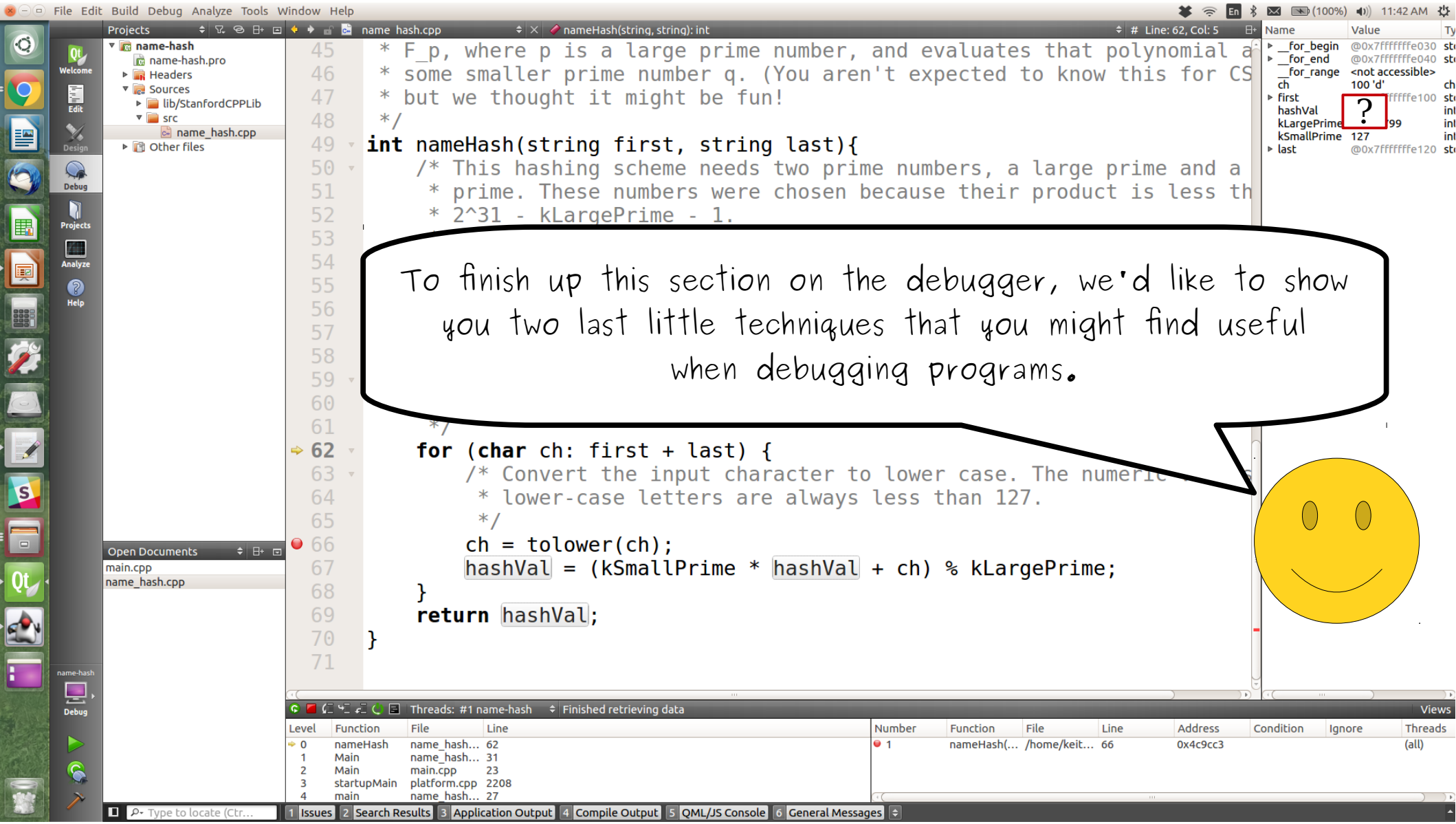

Now we're here! Notice that ch now has the value 'd', which is the second letter of the name Ada.

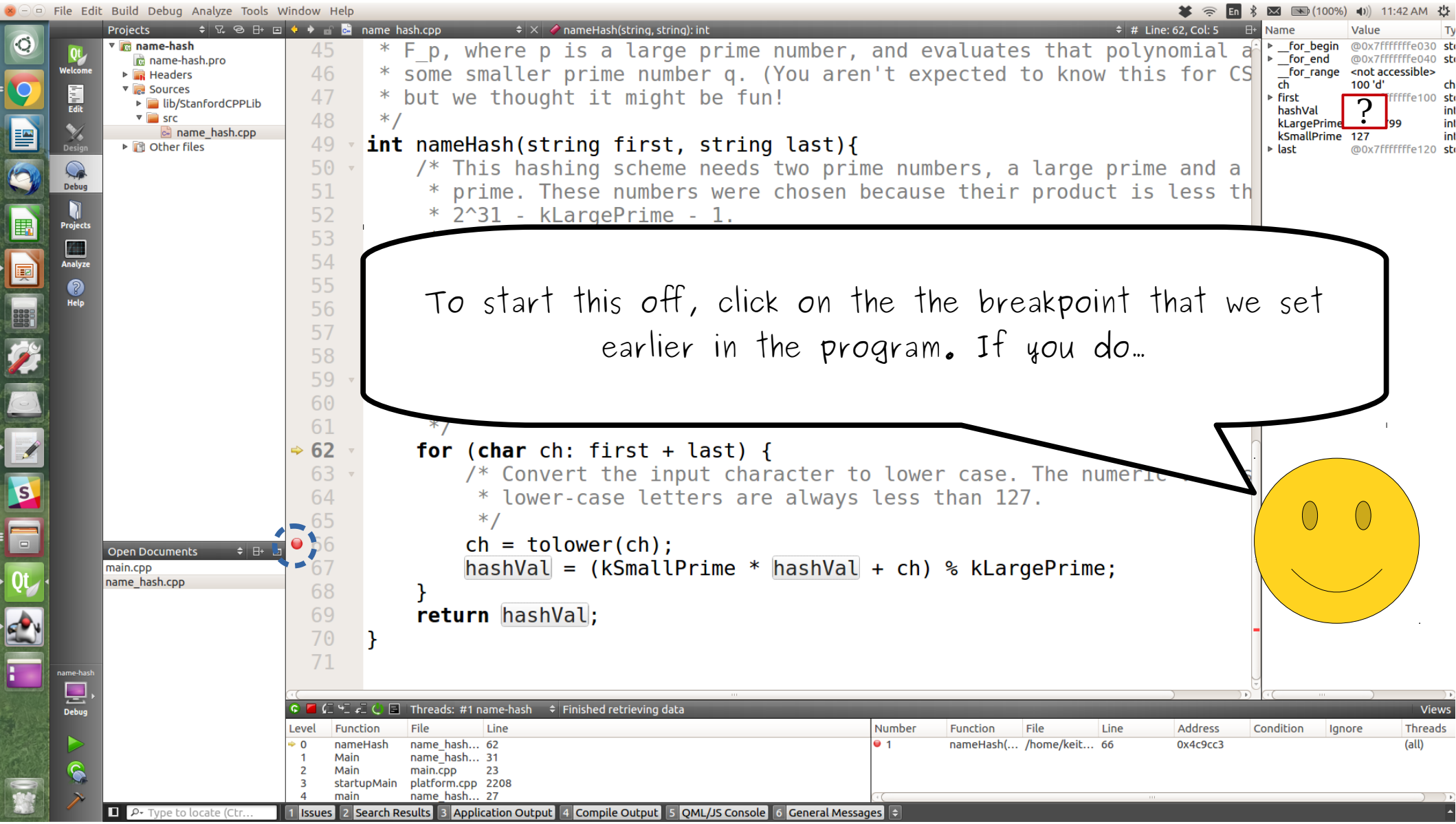

Go click "Step Over" again to run this line of code.

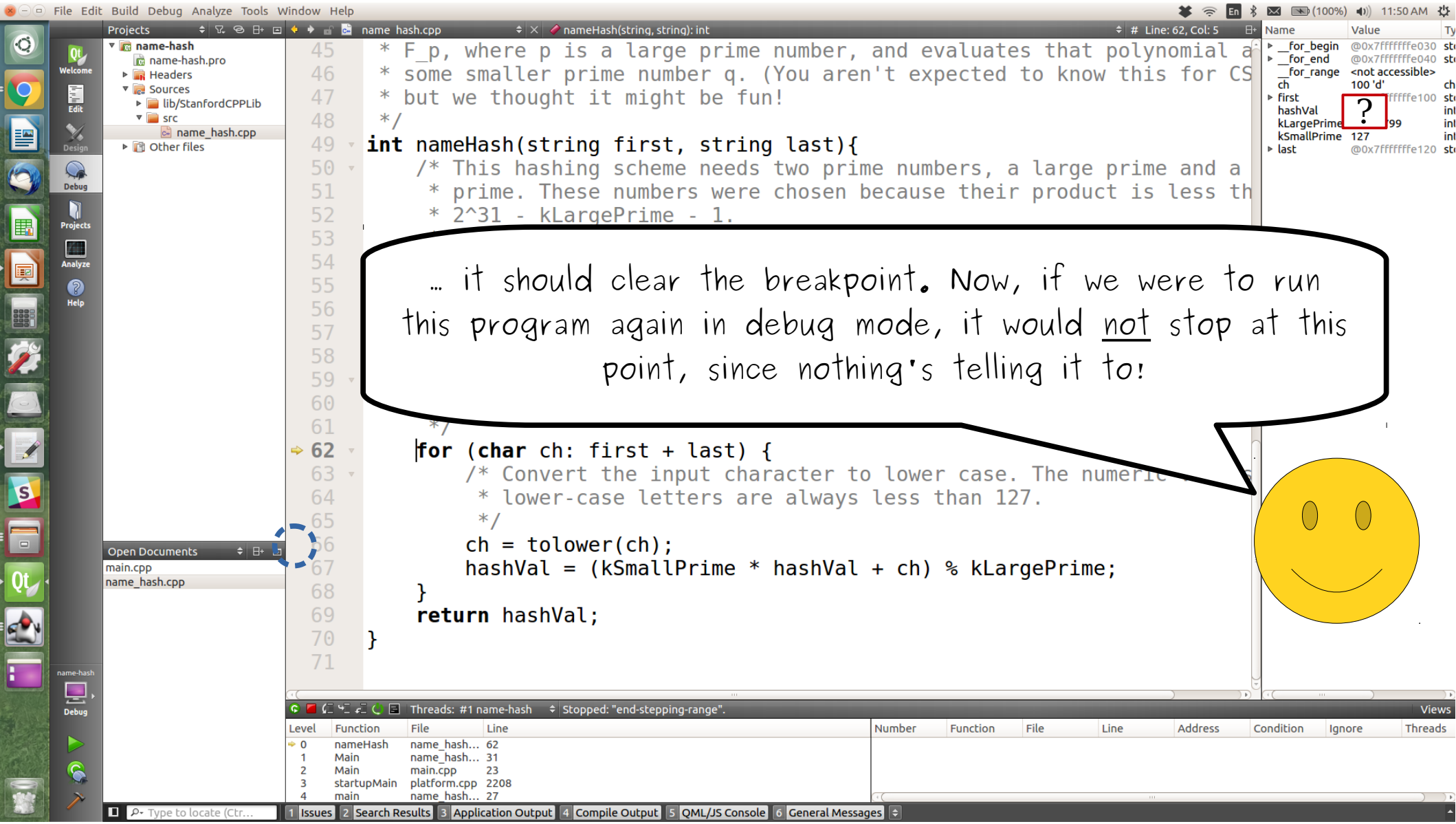

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 67, Col: 9
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
int hashV
/* Iterat
* name,
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
@0x7fffffffe030
@0x7fffffffe040
<not accessible>
100 'd'
@0x7fffffffe100
97
16908799
127
@0x7fffffffe120
You should be here now. Notice that none of the values changed. That makes sense, since all we did was convert a lower-case 'd' to a lower-case 'd'.
Threads: #1 name-hash
Stopped: "end-stepping-range".
Views
Level
Function
File
Line
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
1 nameHash(... /home/keit... 66 0x4c9cc3 (all)
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

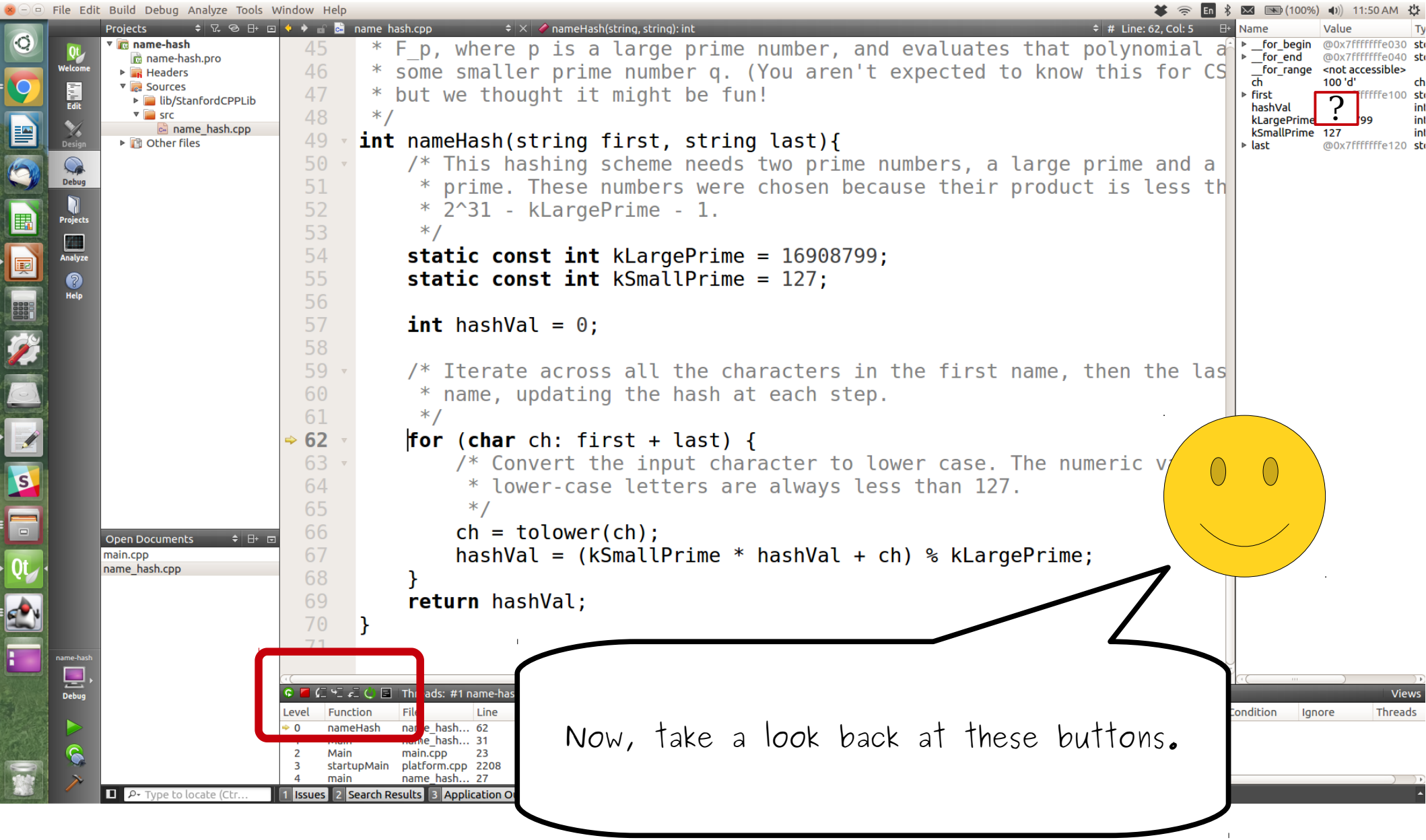

Now, click "Step Over" one more time.

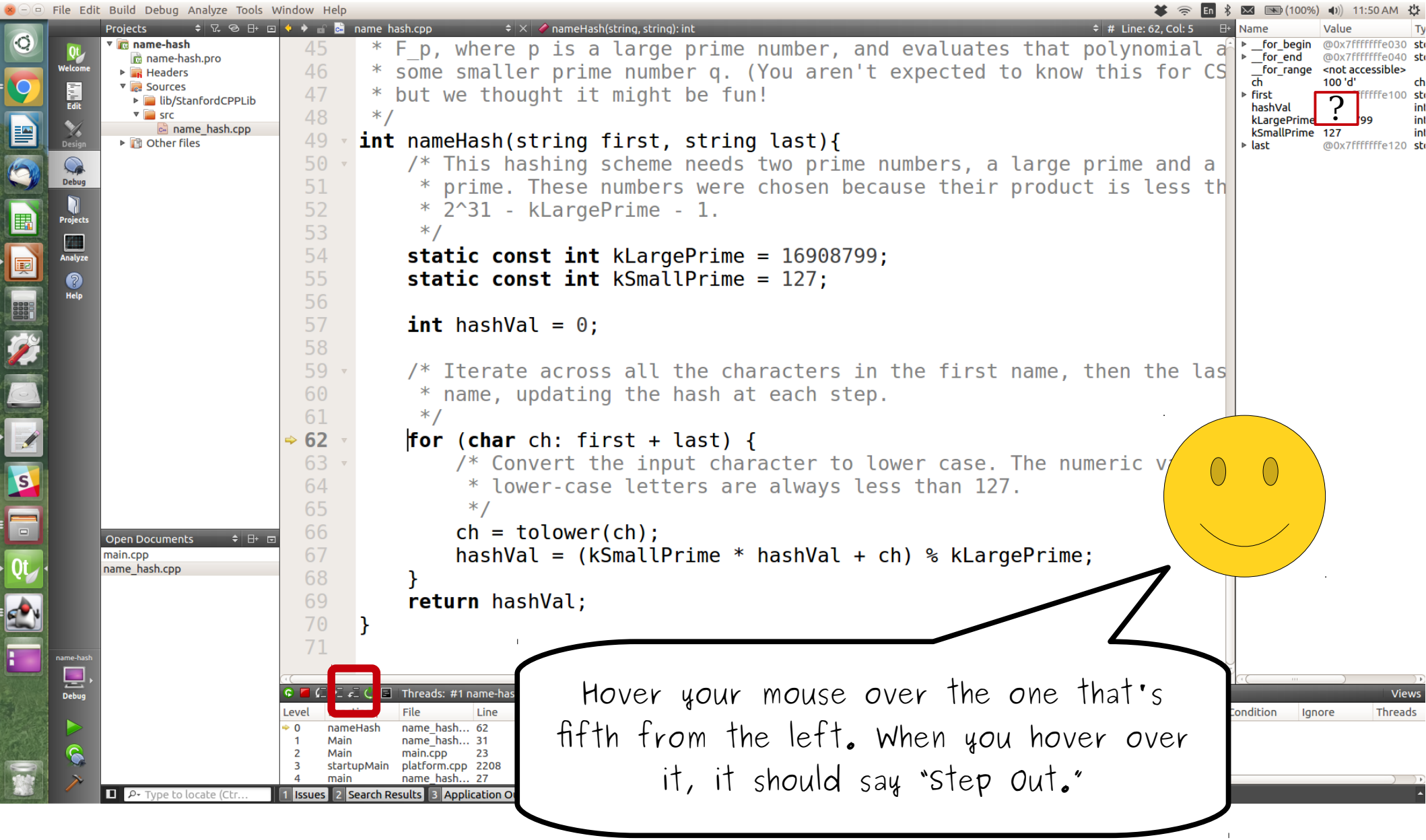

You'll now be at this point in the program. We've covered up the value of hashVal in this image, because at this point you should be able to see what hashVal is by reading the value in the side pane. This is the special value we want you to tell us when submitting the assignment!
?

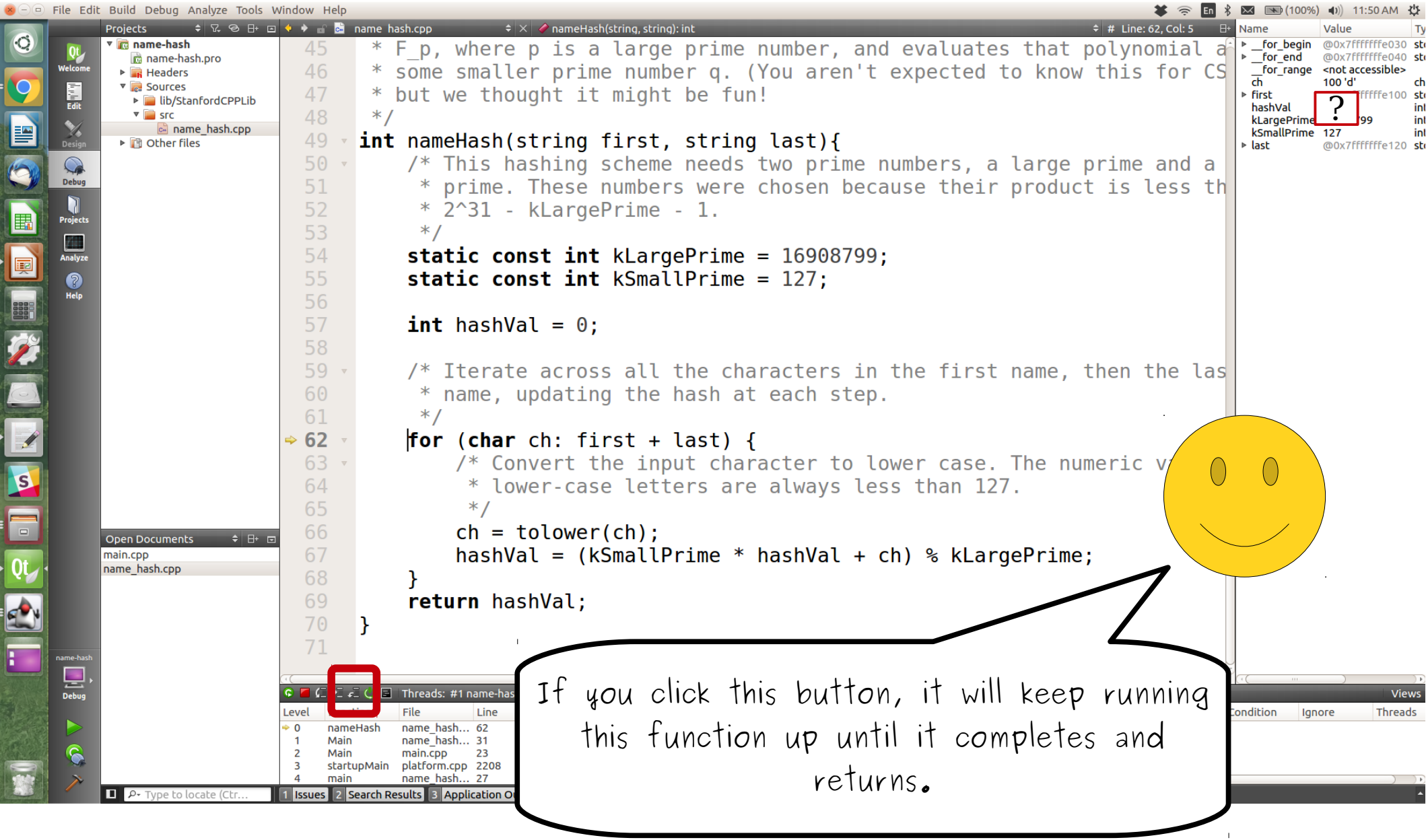

To finish up this section on the debugger, we'd like to show you two last little techniques that you might find useful when debugging programs.

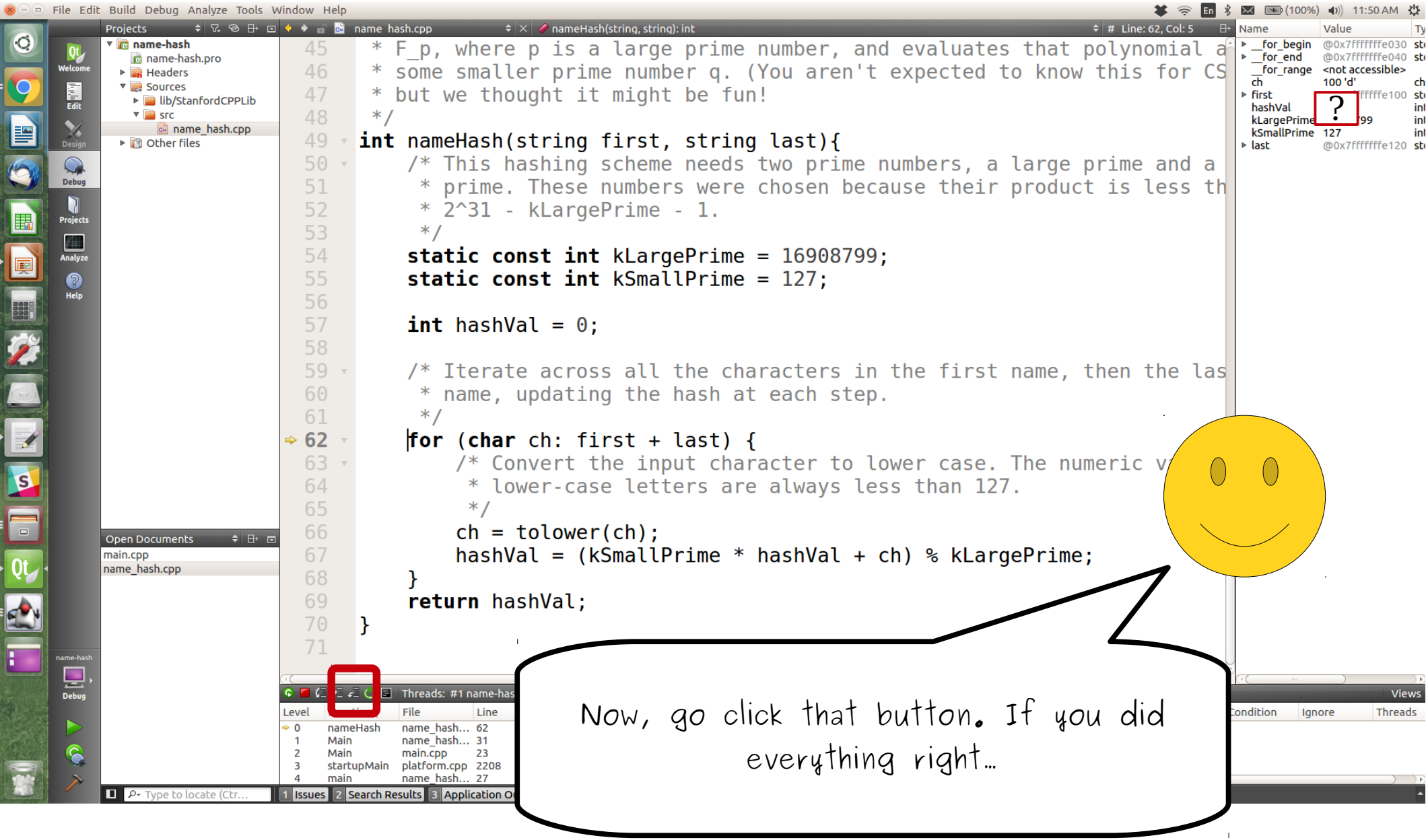

To start this off, click on the the breakpoint that we set earlier in the program. If you do…
?

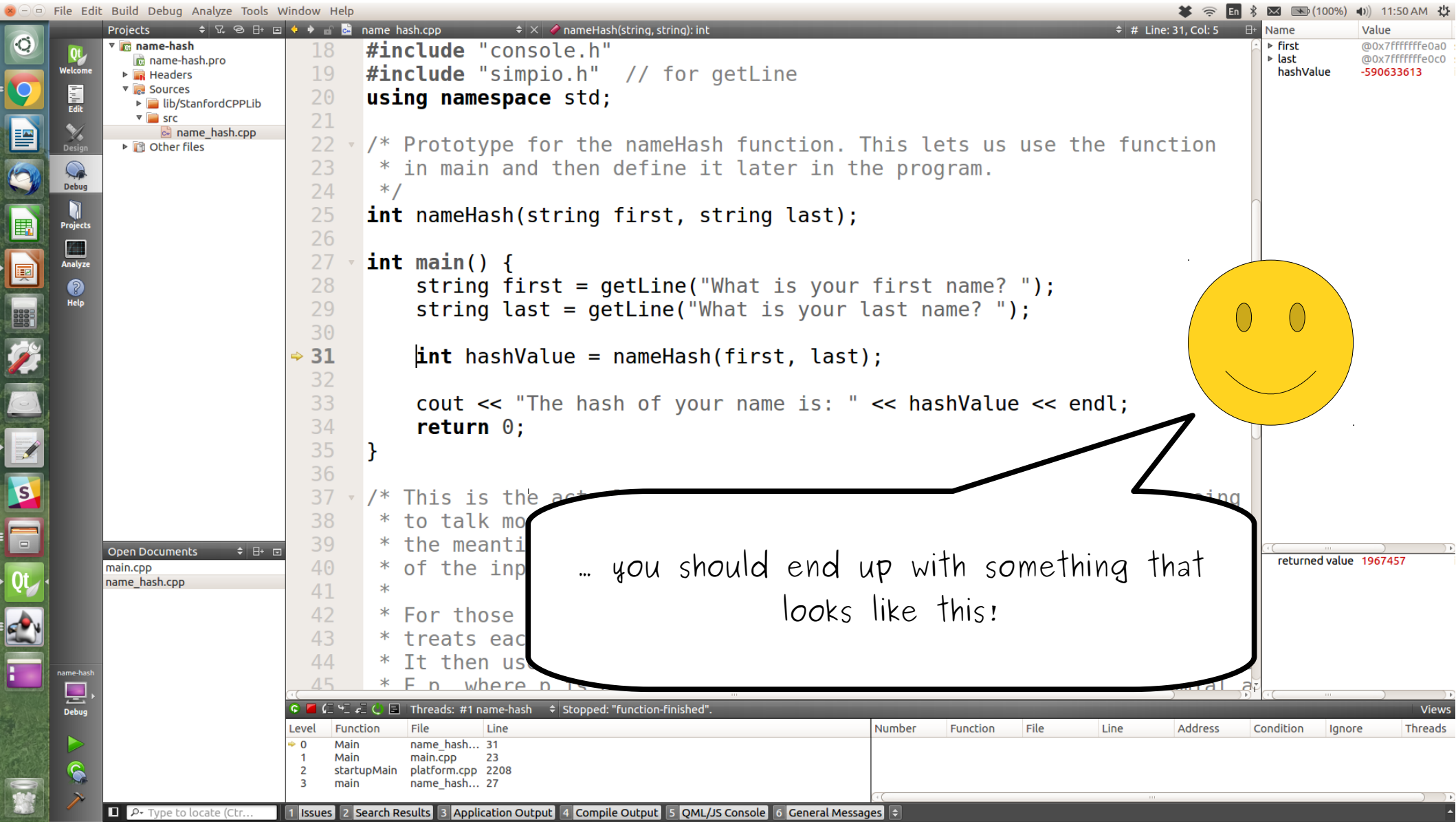

… it should clear the breakpoint. Now, if we were to run this program again in debug mode, it would not stop at this point, since nothing's telling it to!
?

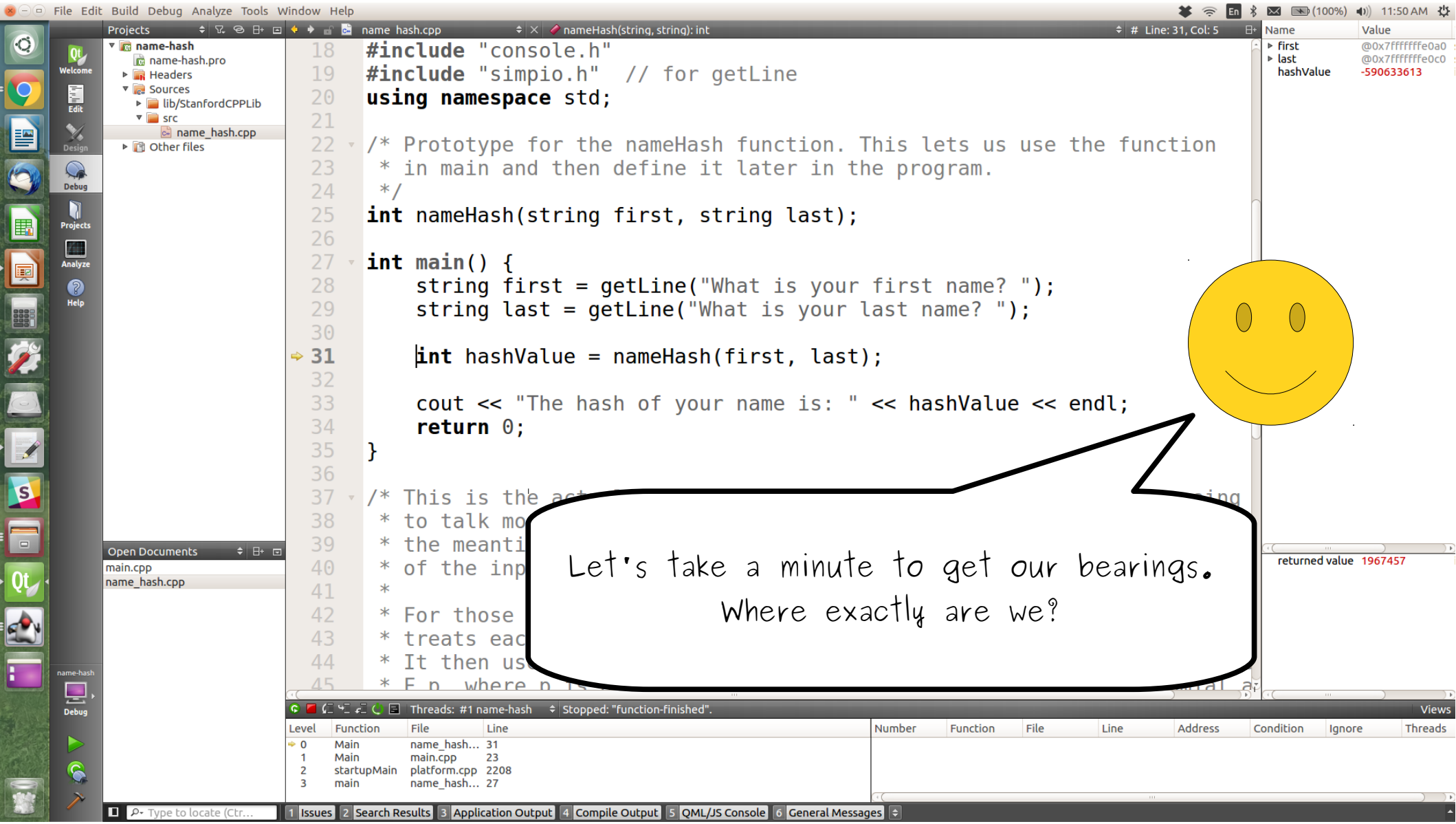

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 62, Col: 5
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric v
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
?
Level
Function
Line
0 nameHash 62
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main 27
Condition
Ignore
Threads
Views
Issues
Search Results
Now, take a look back at these buttons.

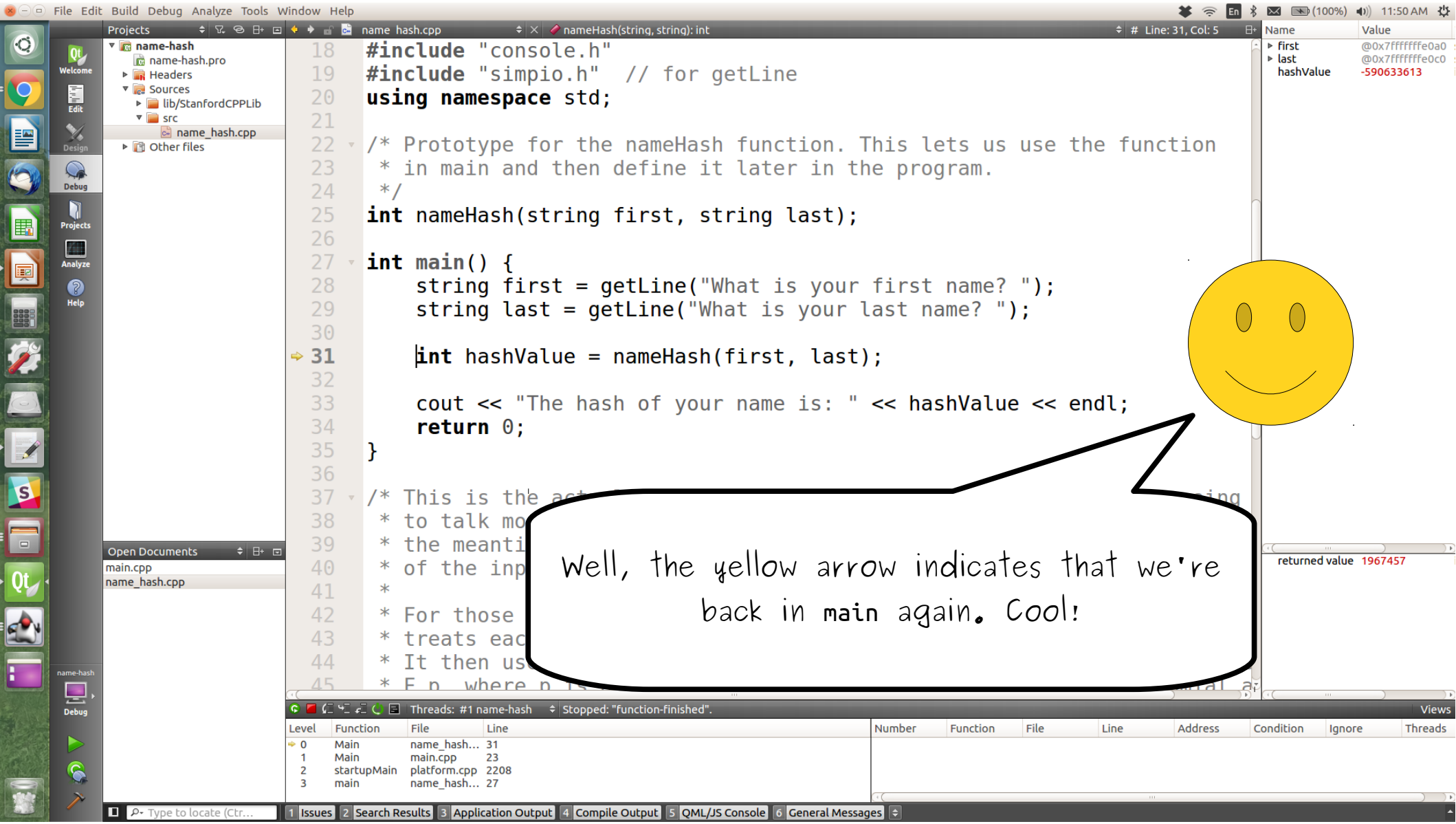

Hover your mouse over the one that's fifth from the left. When you hover over it, it should say "Step Out."

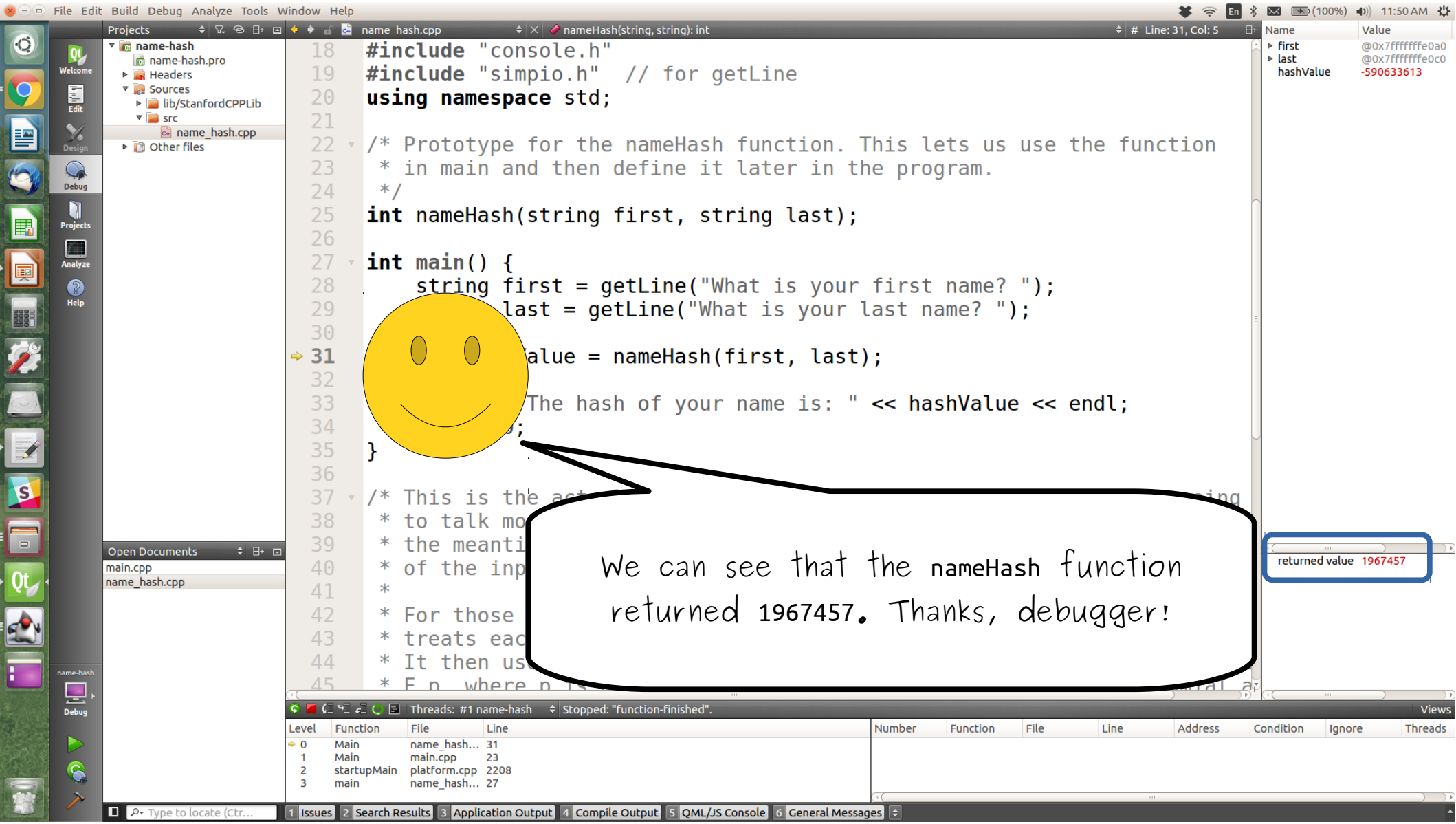

name_hash.cpp
nameHash(string, string): int
Line: 62, Col: 5
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
45 * F_p, where p is a large prime number, and evaluates that polynomial a
46 * some smaller prime number q. (You aren't expected to know this for CS
47 * but we thought it might be fun!
48 */
49 int nameHash(string first, string last){
50 /* This hashing scheme needs two prime numbers, a large prime and a
51 * prime. These numbers were chosen because their product is less th
52 * 2^31 - kLargePrime - 1.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal = 0;
58
59 /* Iterate across all the characters in the first name, then the las
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric v
64 * lower-case letters are always less than 127.
65 */
66 ch = tolower(ch);
67 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68 }
69 return hashVal;
70 }
71
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
?
Threads: #1 name-has
Level
File
Line
0 nameHash name_hash... 62
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Condition
Ignore
Threads
Views
Issues
Search Results
If you click this button, it will keep running this function up until it completes and returns.

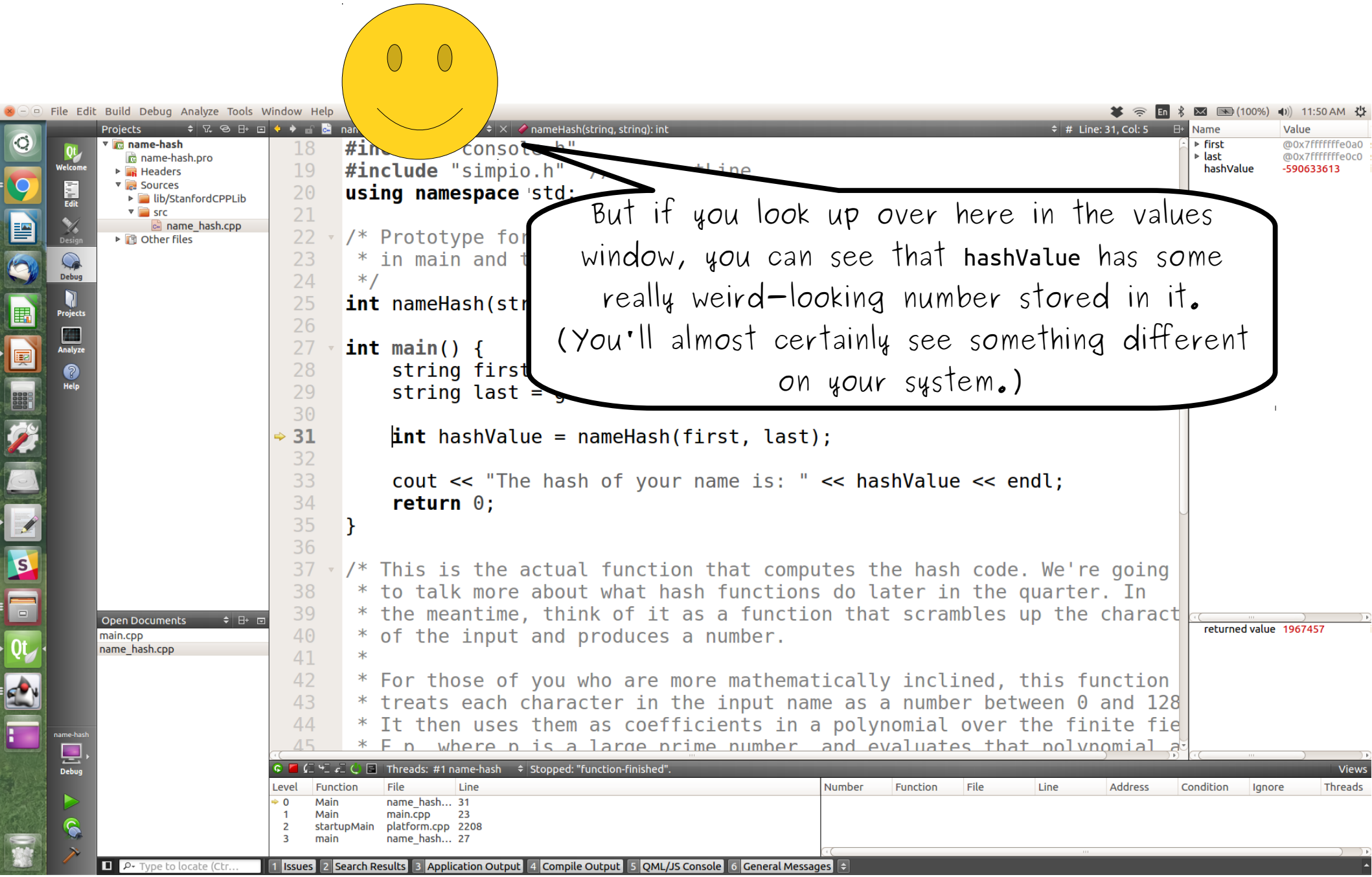

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name_hash.cpp
nameHash(string, string): int
Line: 62, Col: 5
* F_p, where p is a large prime number, and evaluates that polynomial a
* some smaller prime number q. (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string first, string last){
/* This hashing scheme needs two prime numbers, a large prime and a
* prime. These numbers were chosen because their product is less th
* 2^31 - kLargePrime - 1.
*/
static const int kLargePrime = 16908799;
static const int kSmallPrime = 127;
int hashVal = 0;
/* Iterate across all the characters in the first name, then the las
* name, updating the hash at each step.
*/
for (char ch: first + last) {
/* Convert the input character to lower case. The numeric v
* lower-case letters are always less than 127.
*/
ch = tolower(ch);
hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;
}
Name
Value
__for_begin
__for_end
__for_range
ch
first
hashVal
kLargePrime
kSmallPrime
last
?
Threads: #1 name-has
Level
File
Line
0 nameHash name_hash... 62
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name_hash... 27
Views
Condition
Ignore
Threads
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application O
Now, go click that button. If you did everything right…

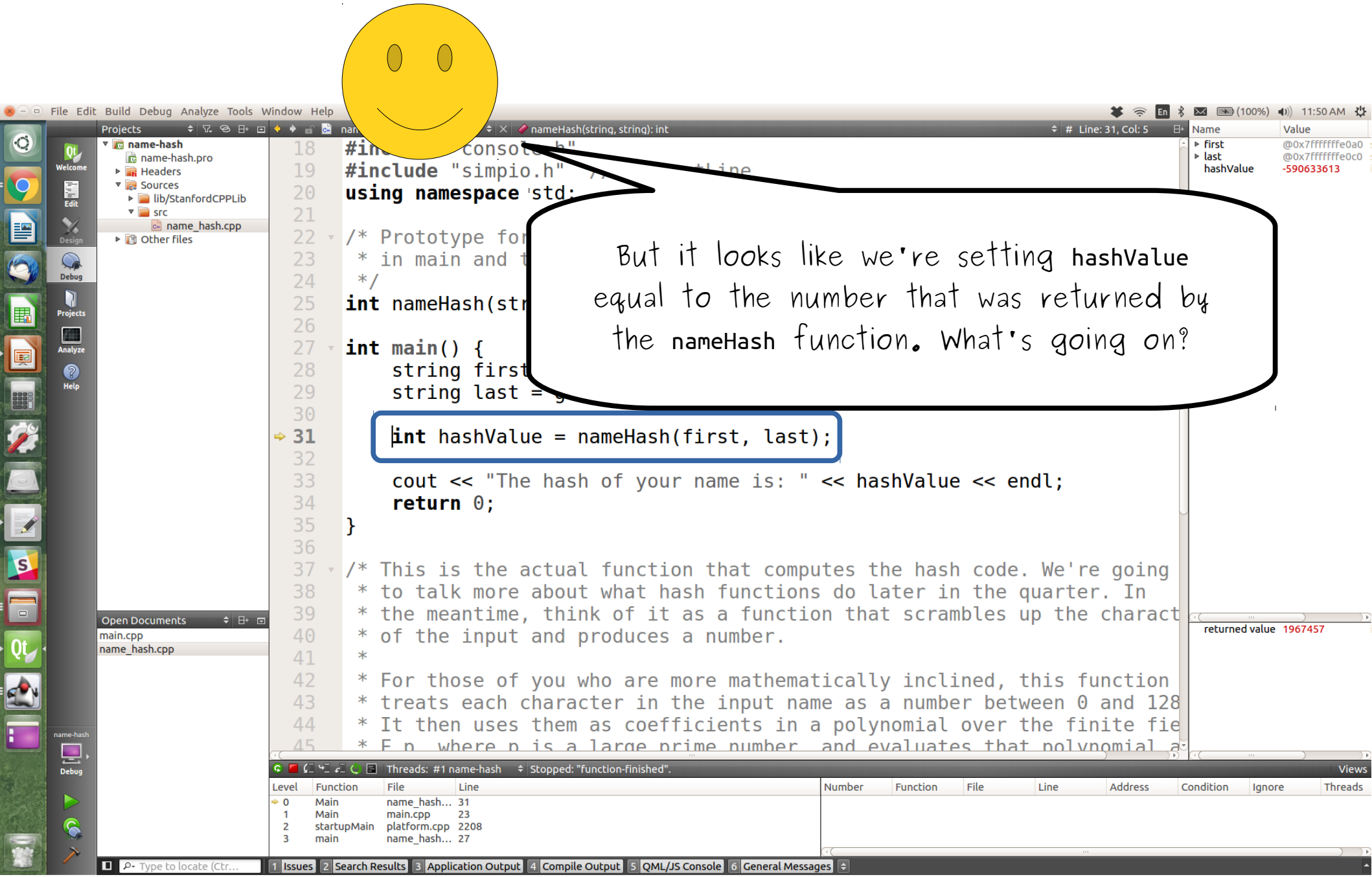

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Name
Value
first
last
hashValue
-590633613
returned value 1967457
… you should end up with something that looks like this!
Threads: #1 name-hash
Stopped: "function-finished".
Views
Level
Function
File
Line
0 Main name_hash... 31
1 Main main.cpp 23
2 startupMain platform.cpp 2208
3 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

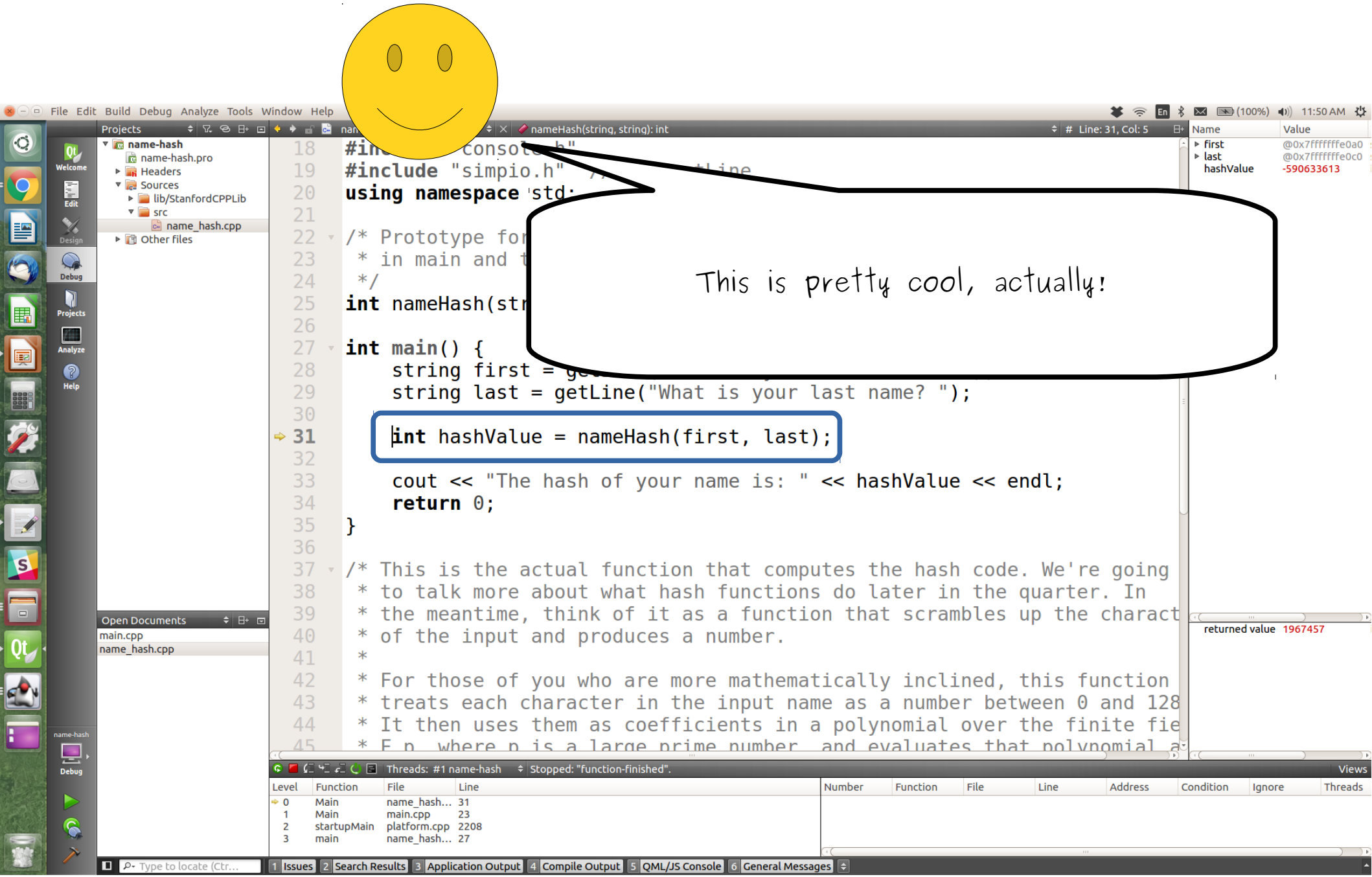

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Let's take a minute to get our bearings.
Where exactly are we?
Name
Value
first
last
hashValue
returned value 1967457
Threads: #1 name-hash
Stopped: "function-finished".
Views
Level
Function
File
Line
Number
Address
Condition
Ignore
Threads
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages


File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
string last = getLine("What is your last name? ");
int hashValue = nameHash(first, last);
cout << "The hash of your name is: " << hashValue << endl;
return 0;
}
Name
Value
first
last
hashValue
returned value 1967457
Well, the yellow arrow indicates that we're back in main again. Cool!
Threads: #1 name-hash
Stopped: "function-finished".
Views
Level
Function
File
Line
Number
Address
Condition
Ignore
Threads
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages


File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
name hash.cpp
nameHash(string, string): int
Line: 31, Col: 5
#include "console.h"
#include "simpio.h" // for getLine
using namespace std;
/* Prototype for the nameHash function. This lets us use the function
* in main and then define it later in the program.
*/
int nameHash(string first, string last);
int main() {
string first = getLine("What is your first name? ");
last = getLine("What is your last name? ");
alue = nameHash(first, last);
The hash of your name is: " << hashValue << endl;
}
Name
Value
first
last
hashValue
-590633613
returned value 1967457
We can see that the nameHash function returned 1967457. Thanks, debugger!
Threads: #1 name-hash
Stopped: "function-finished".
Views
Level
Function
File
Line
0 Main name_hash... 31
1 Main main.cpp 23
2 startupMain platform.cpp 2208
3 main name_hash... 27
Number
Function
File
Line
Address
Condition
Ignore
Threads
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

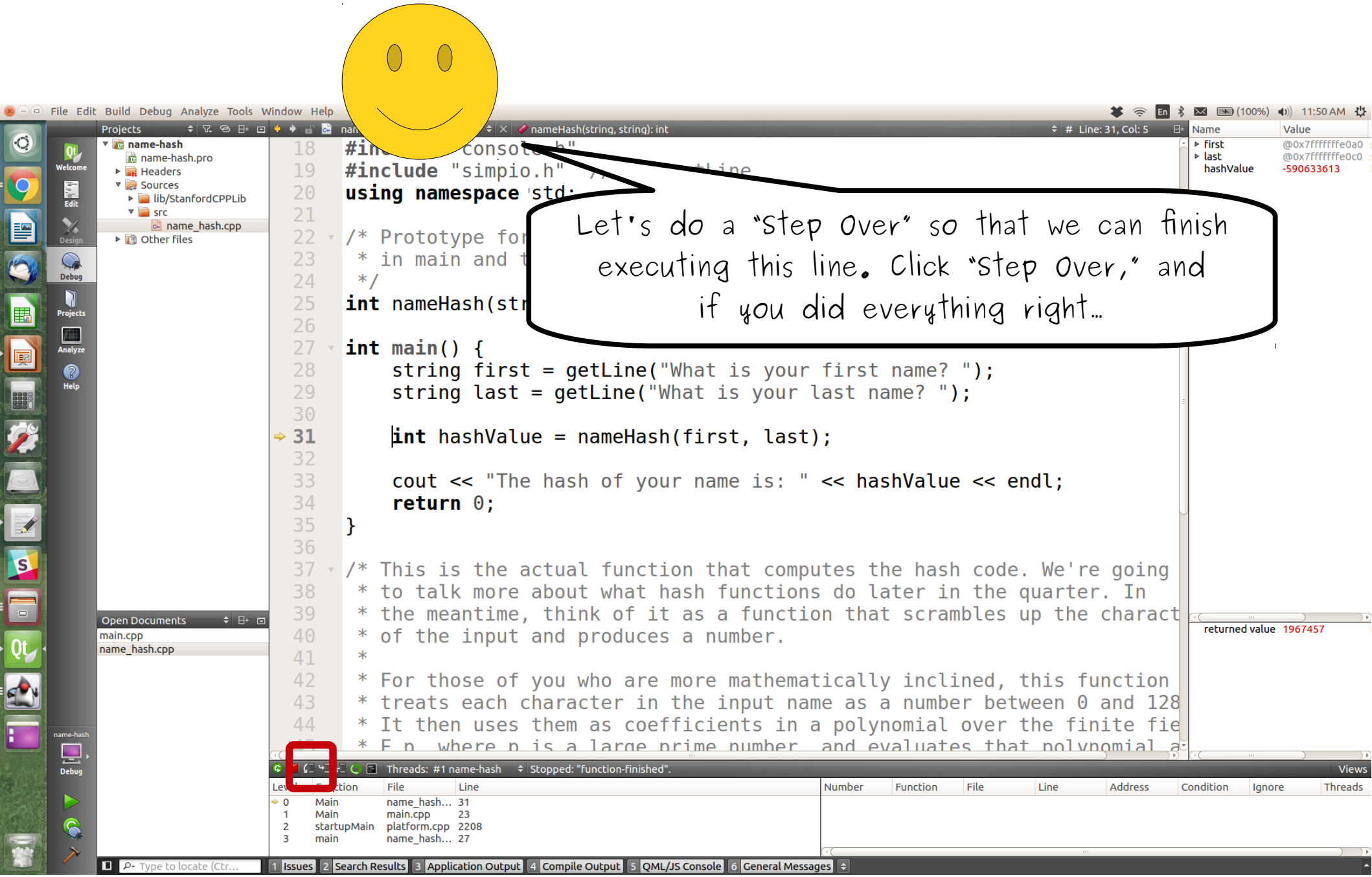

But if you look up over here in the values window, you can see that hashValue has some really weird-looking number stored in it. (You'll almost certainly see something different on your system.)

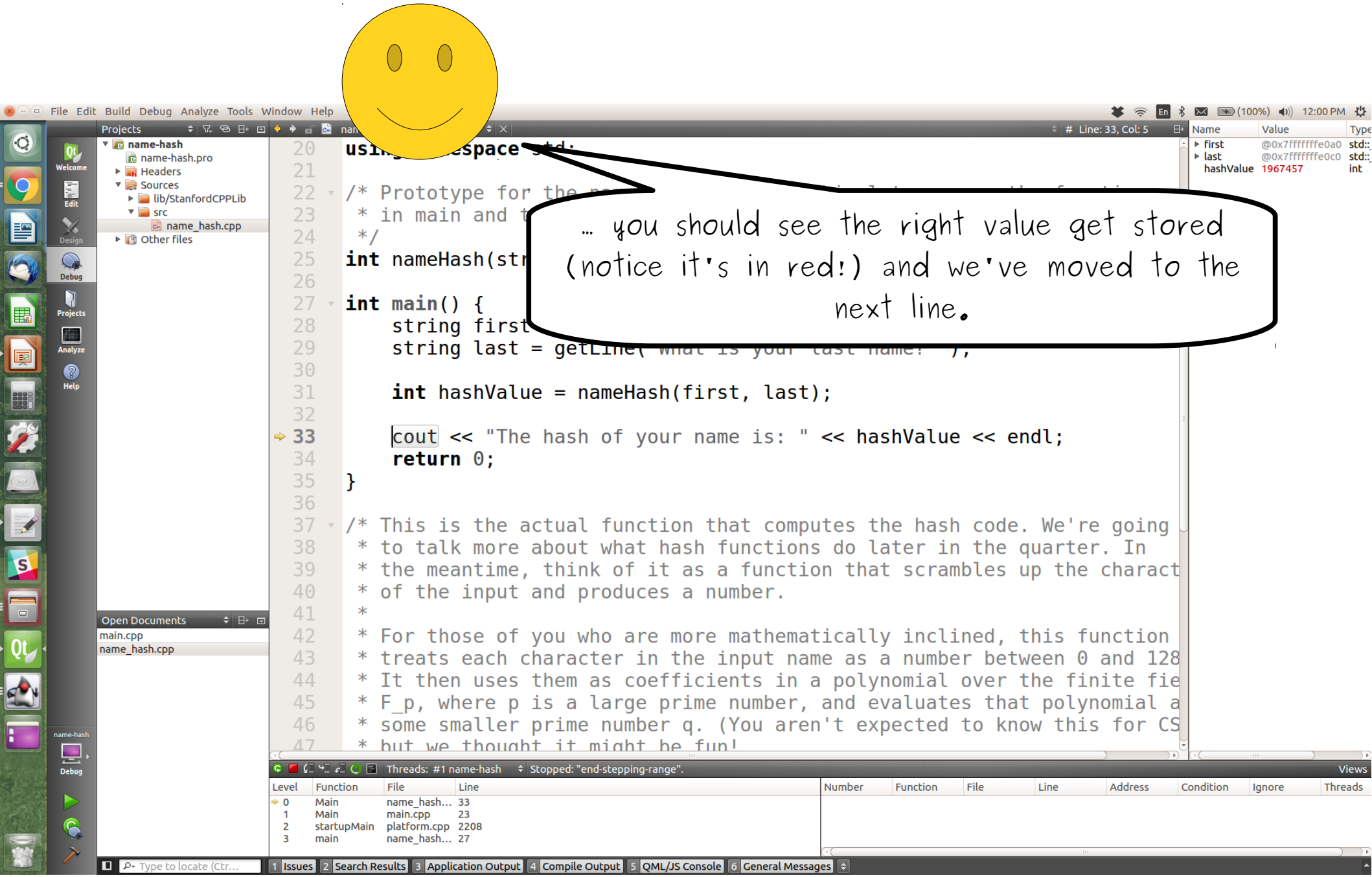

But it looks like we're setting hashValue equal to the number that was returned by the nameHash function. What's going on?

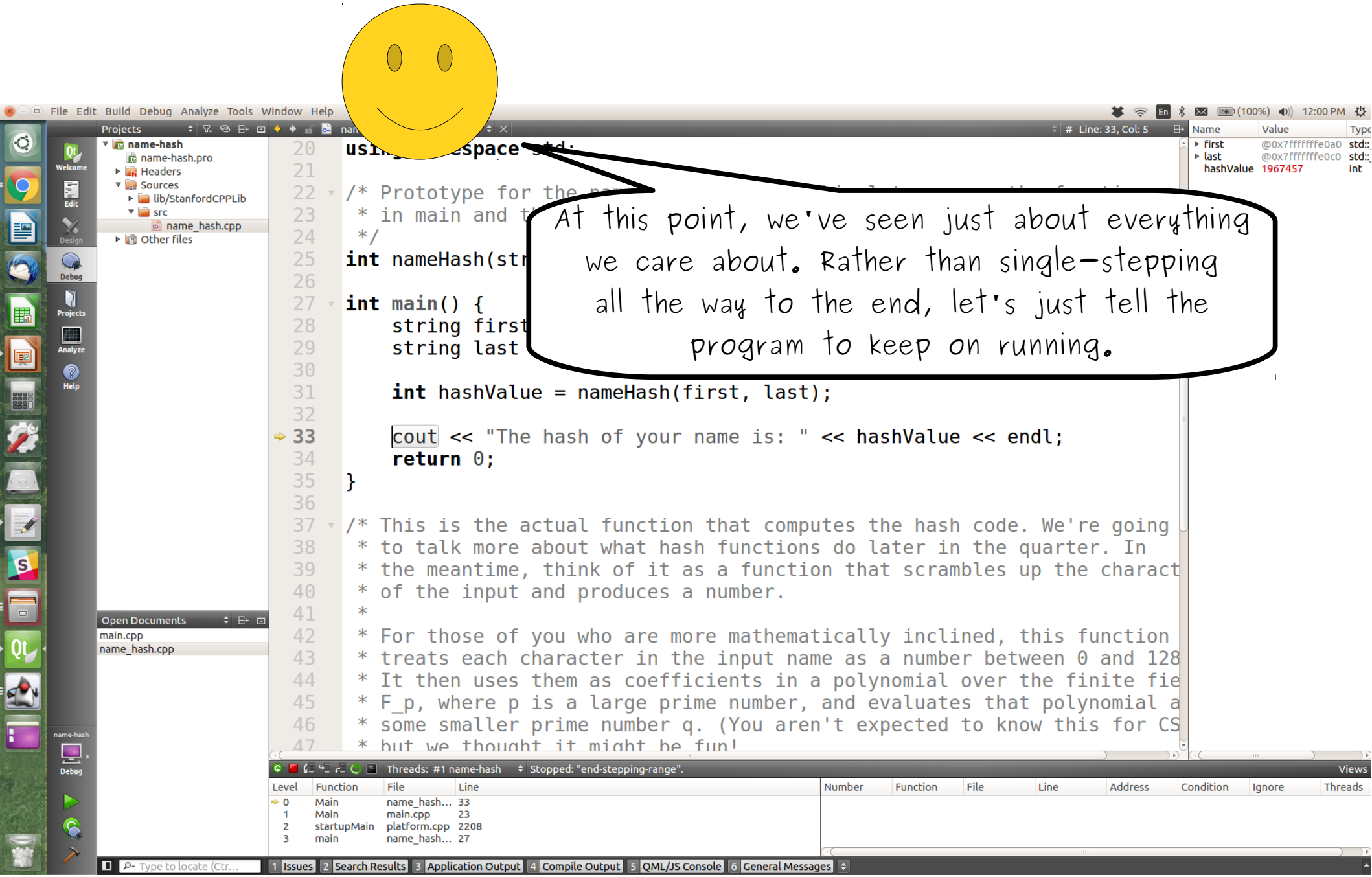

This is pretty cool, actually!

What's happened is that we've just returned from **nameHash** with a value, but since we're going through the program one step at a time, we haven't actually assigned that value to **hashValue** yet!

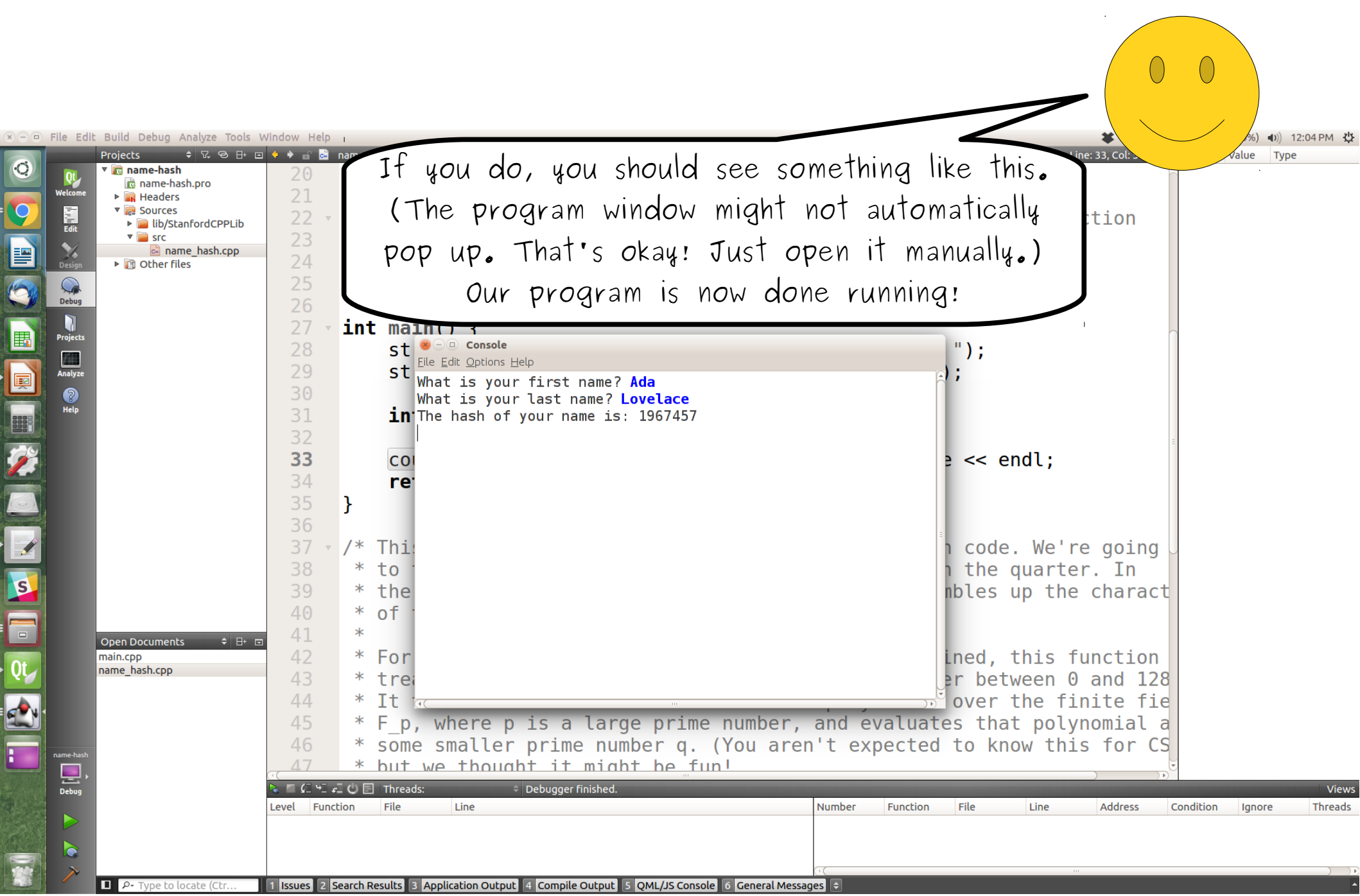
```
18  #in          consol    h"
19  #include "simpio.h"  //          tline
20  using namespace std;
21
22  /* Prototype for
23   * in main and t
24   */
25  int nameHash(str
26
27  int main() {
28      string first
29      string last = g
30
31      int hashValue = nameHash(first, last);
32
33      cout << "The hash of your name is: " << hashValue << endl;
34      return 0;
35  }
36
37  /* This is the actual function that computes the hash code. We're going
38   * to talk more about what hash functions do later in the quarter. In
39   * the meantime, think of it as a function that scrambles up the charact
40   * of the input and produces a number.
41   *
42   * For those of you who are more mathematically inclined, this function
43   * treats each character in the input name as a number between 0 and 128
44   * It then uses them as coefficients in a polynomial over the finite fie
45   * F_p, where p is a large prime number, and evaluates that polynomial a
```


Let's do a "Step Over" so that we can finish executing this line. Click "Step Over," and if you did everything right…

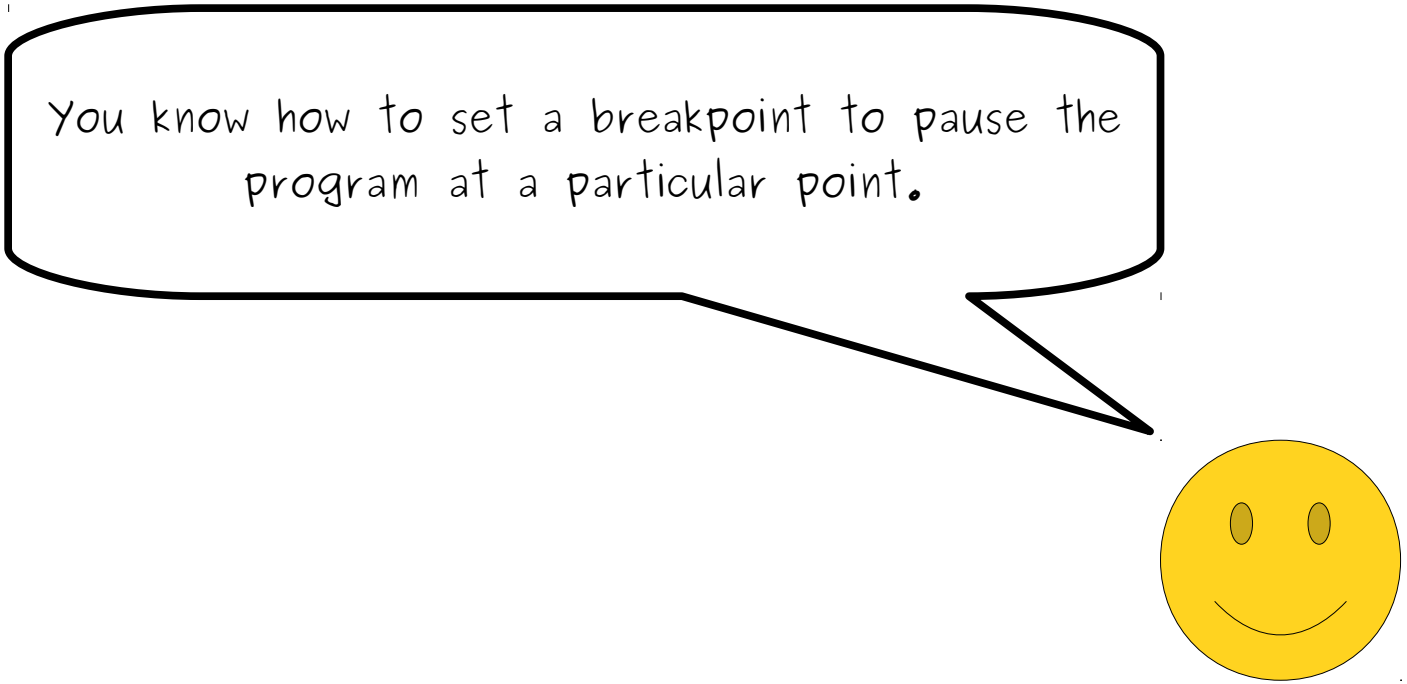

… you should see the right value get stored (notice it's in red!) and we've moved to the next line.

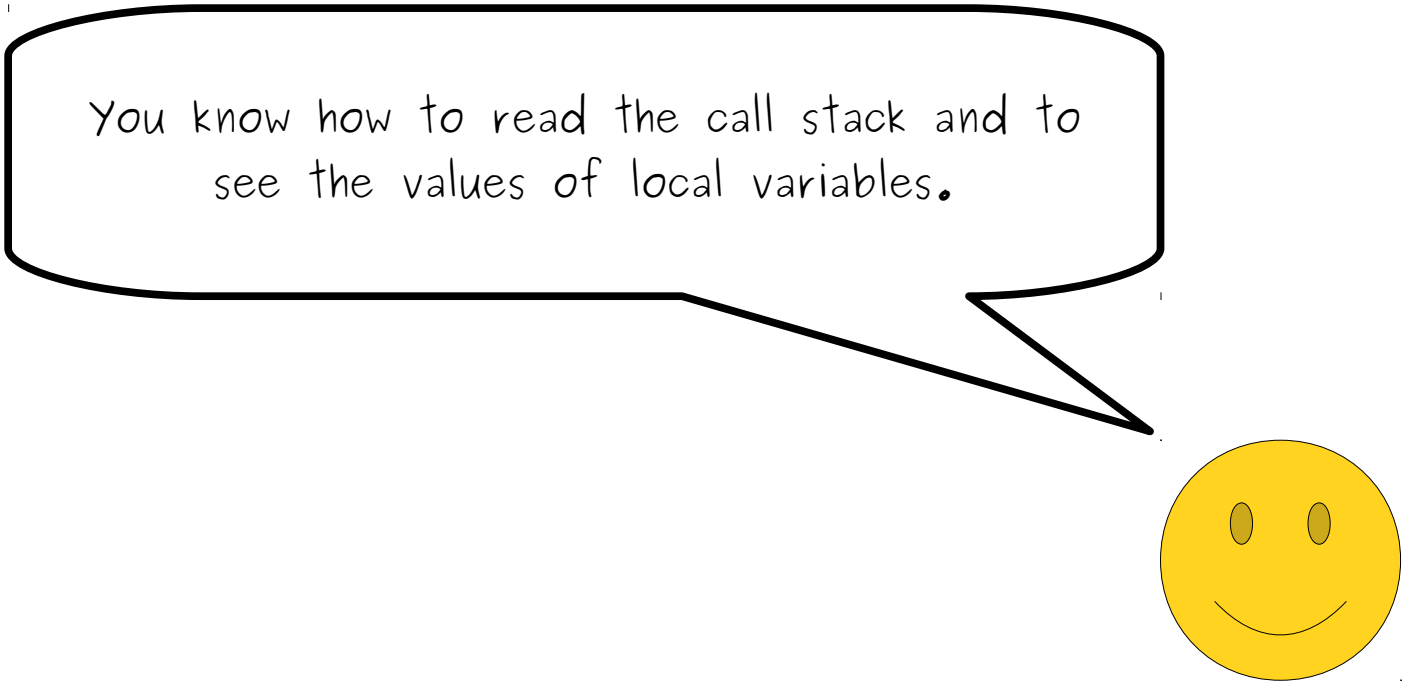

At this point, we've seen just about everything we care about. Rather than single-stepping all the way to the end, let's just tell the program to keep on running.

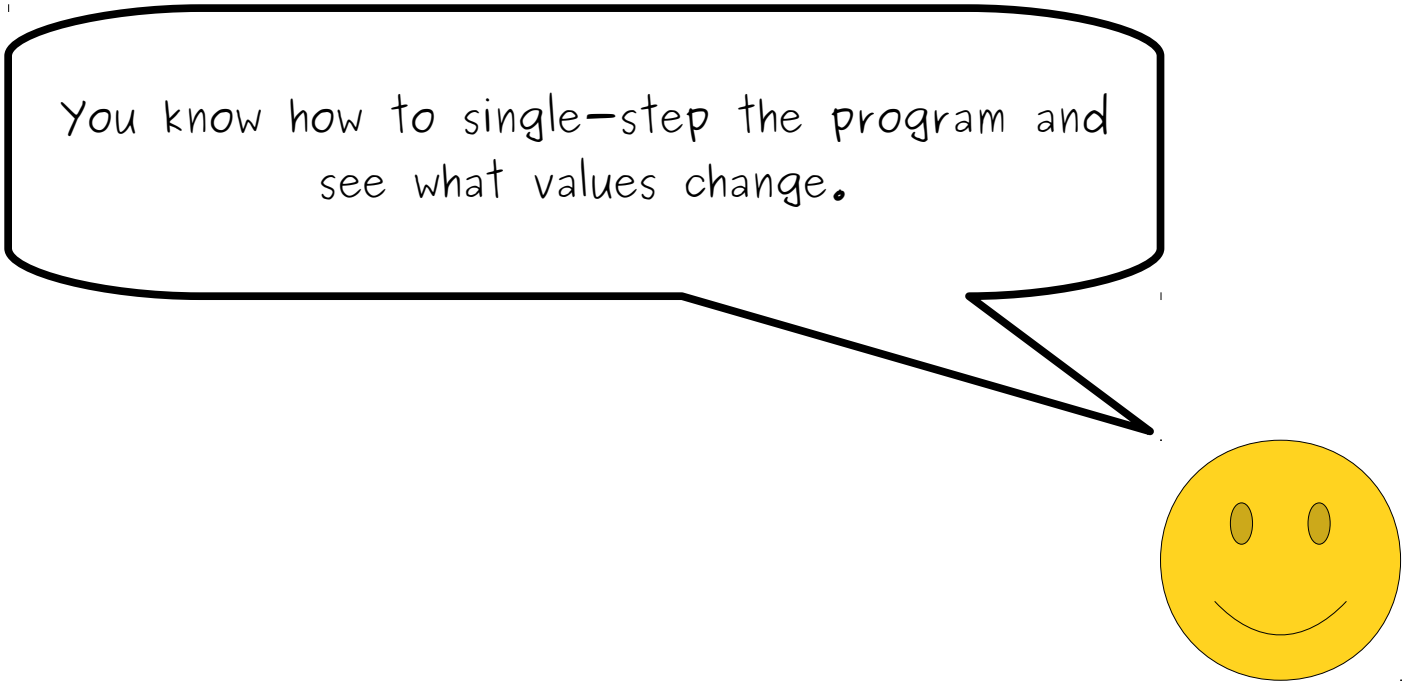

File Edit Build Debug Analyze Tools Window Help
Projects
name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files
Open Documents
main.cpp
name_hash.cpp
Line: 33, Col: 5
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the function
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28 string first = getLine("What is your first name? ");
29 string last = getLine("What is your last name? ");
30
31 int hashValue = nameHash(first, last);
32
33 cout << "The hash of your name is: " << hashValue << endl;
34 return 0;
35 }
36
37 /* This is the actual function that computes the
38 * to talk more about what hash functions do lat
39 * the meantime, think of it as a function that
40 * of the input and produces a number.
41 *
Name
Value
Type
first
last
hashValue
1967457
int
To do this, click on this button. If you hover over it, it says "Continue," and that button means "unpause the program and let it keep running from here."
Function
File
Line
Address
Condition
Ignore
Threads
Views
Welcome
Edit
Design
Debug
Projects
Analyze
Help
name-hash
Debug
Type to locate (Ctr...
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console
6 General Messages

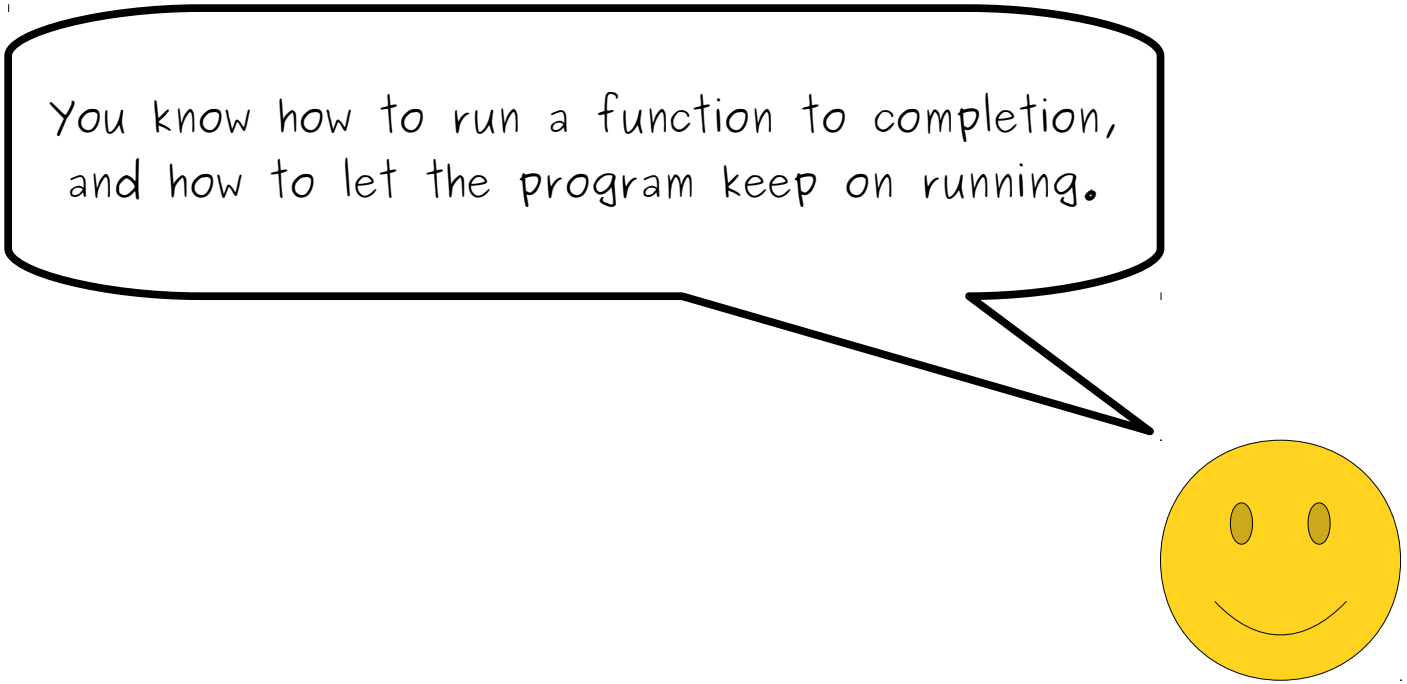

If you do, you should see something like this.
(The program window might not automatically pop up. That's okay! Just open it manually.)
Our program is now done running!
Console
File Edit Options Help
What is your first name? Ada
What is your last name? Lovelace
The hash of your name is: 1967457

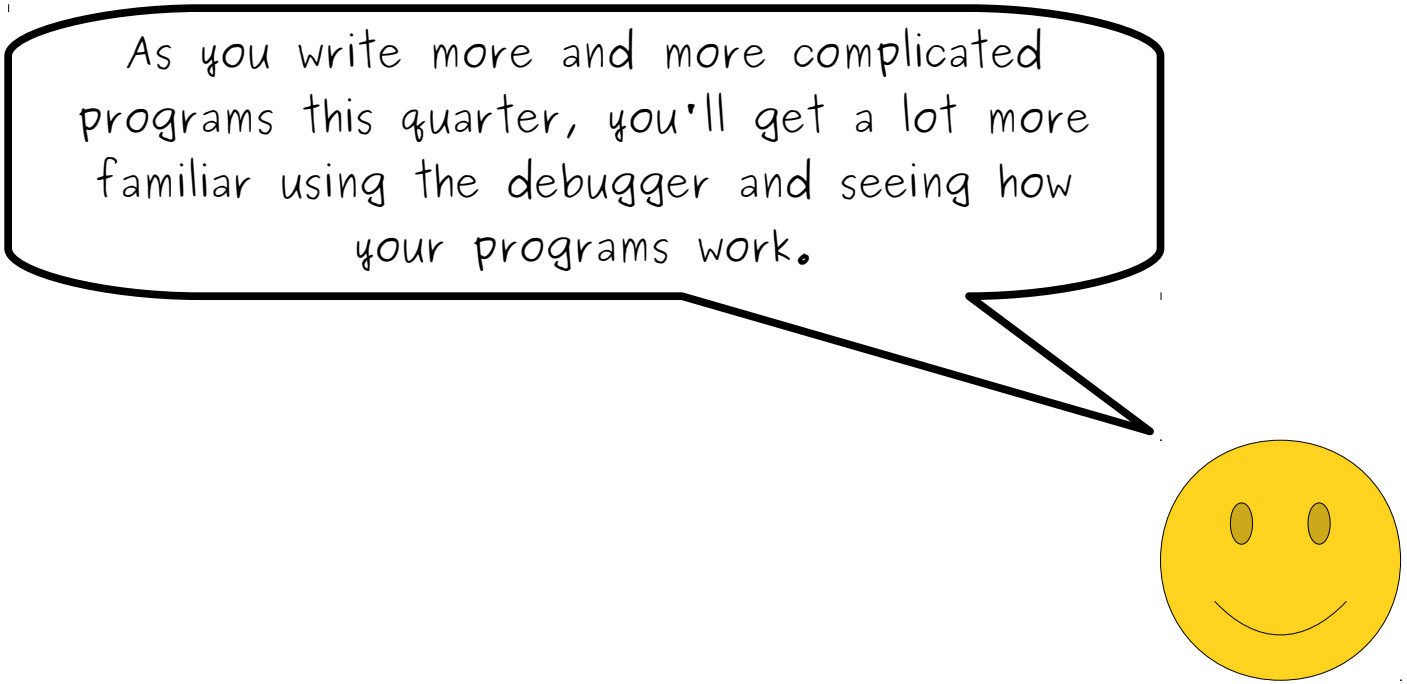

So there you have it! You've now gotten more familiar with the debugger!

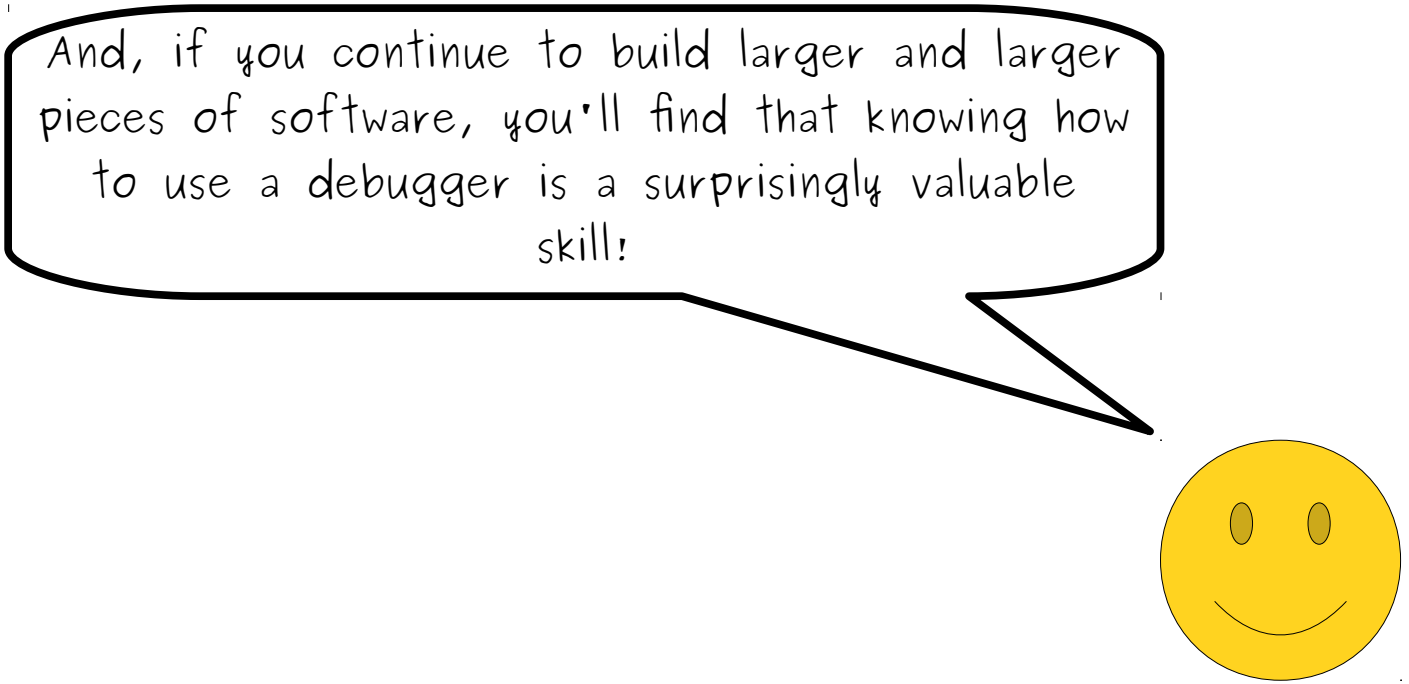

You know how to set a breakpoint to pause the program at a particular point.


You know how to read the call stack and to see the values of local variables.


You know how to single-step the program and see what values change.


You know how to run a function to completion,
and how to let the program keep on running.


As you write more and more complicated programs this quarter, you'll get a lot more familiar using the debugger and seeing how your programs work.


And, if you continue to build larger and larger pieces of software, you'll find that knowing how to use a debugger is a surprisingly valuable skill!


Hope this helps, and welcome to CS106B!