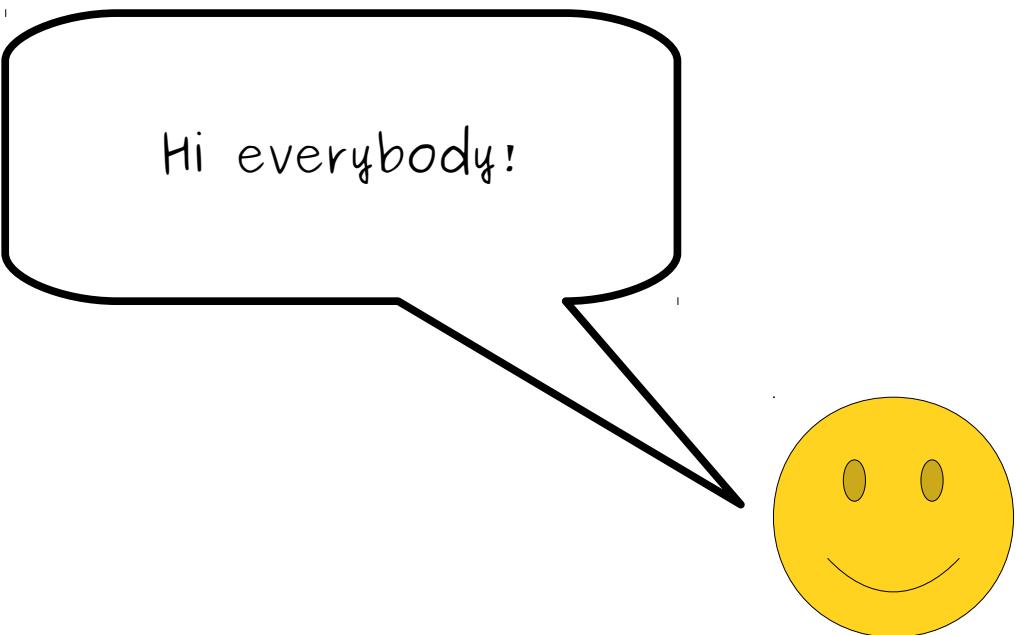
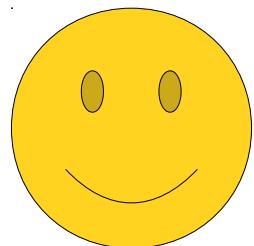


Assignment 0: Using the Debugger

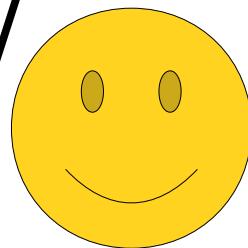
This assignment was written by Keith Schwartz. Thanks, Keith!



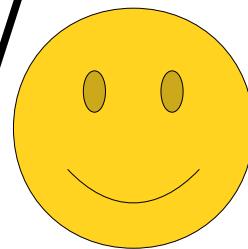
Hi everybody!



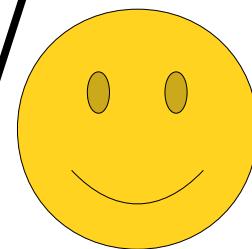
As part of Assignment 0, we'd like you to get a little bit of practice using the debugger in Qt Creator.



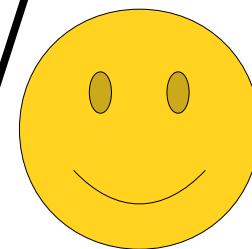
The debugger is a tool you can use to help see what your program is doing as you run it.



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.



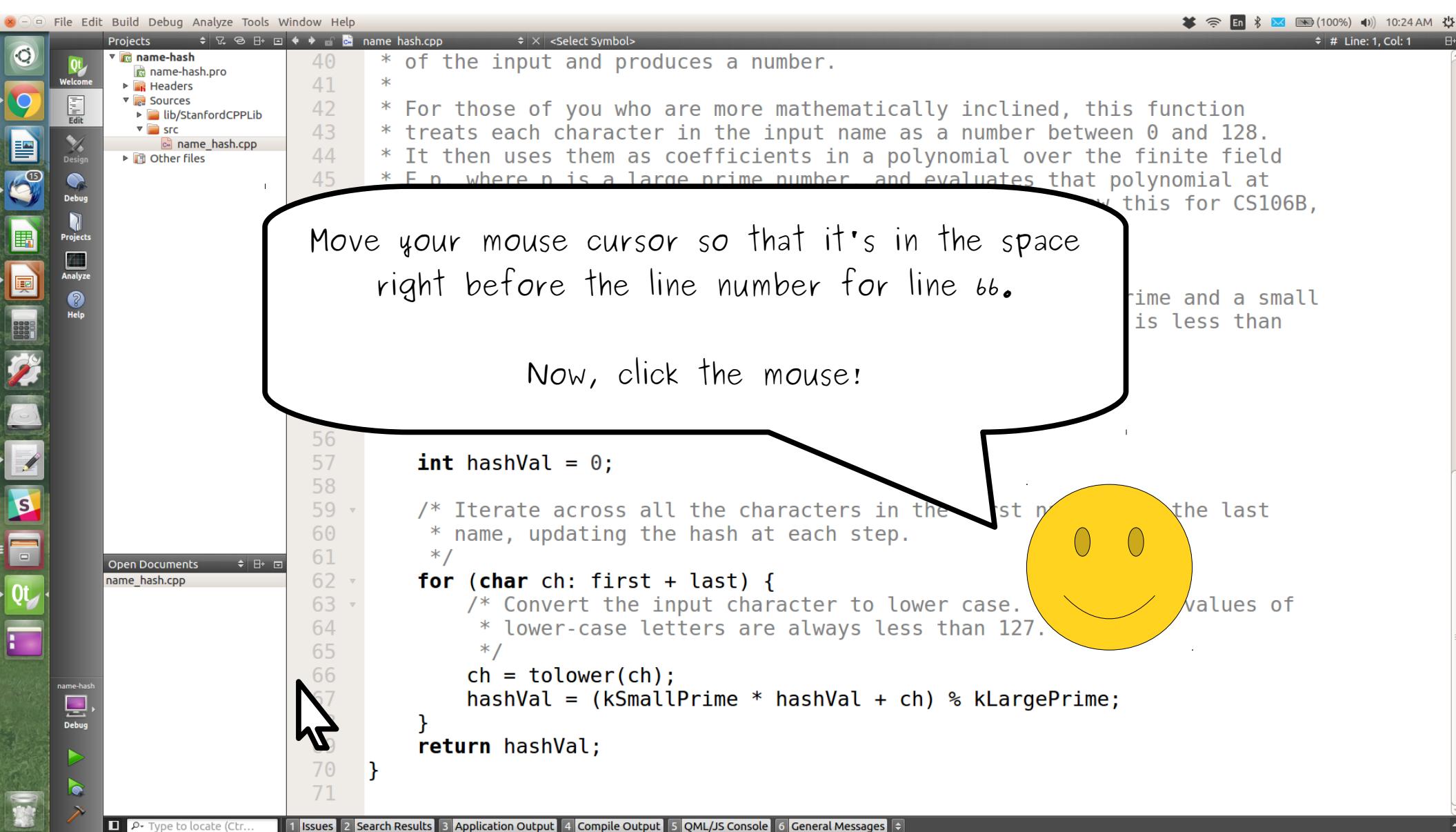
Think of this guide as a little tutorial walkthrough to help give you a sense of how to use the debugger and how to make sense of what you're seeing.



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.



```
40 * of the
41 *
42 * For tho
43 * treats
44 * It then
45 * F_p, wher
46 * some smaller prime numbers, a large prime, and
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
51     * prime. These numbers were chosen because their product is less
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 last
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 of
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 }
```



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp <Select Symbol>

name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

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 field
45 *
46 *
47 *
48 *
49 *
50 *
51 *
52 *
53 *
54 *
55 *
56 *
57 *
58 *
59 *
60 *
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. * Lower-case letters are always less than 128 */
64 /*
65 ch = tolower(ch);
66 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
67 }
68 return hashVal;
69 }

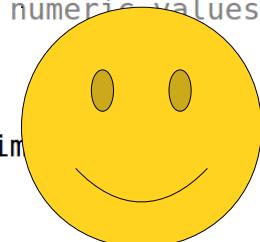
When you do, you should see a red circle with a little hourglass pop up.

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.

Open Documents name_hash.cpp

name-hash
Debug

Issues Search Results Application Output Compile Output QML/JS Console General Messages



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp <Select Symbol>

name hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

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 field
45 * F_p , where p is a large prime number, and evaluates that polynomial at
46 * some smaller prime number q . (You aren't expected to know this for CS106B,
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 small
51 * less than
52 *
53 *
54 *
55 *
56 *
57 *
58 *
59 *
60 */
61 for (char ch: first + last) {
62 /* Convert the input character to lower case. * Lower-case letters are always less than 128.
63 * Lower-case letters are always less than 128.
64 */
65 ch = tolower(ch);
66 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
67 }
68 return hashVal;
69 }
70 }
71 }

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...

Open Documents name_hash.cpp

name hash
Debug

Run in debug mode button

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages



A large black callout bubble surrounds the explanatory text, with a line pointing from the text to the "Run in debug mode" button in the bottom right corner of the interface.

... 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.



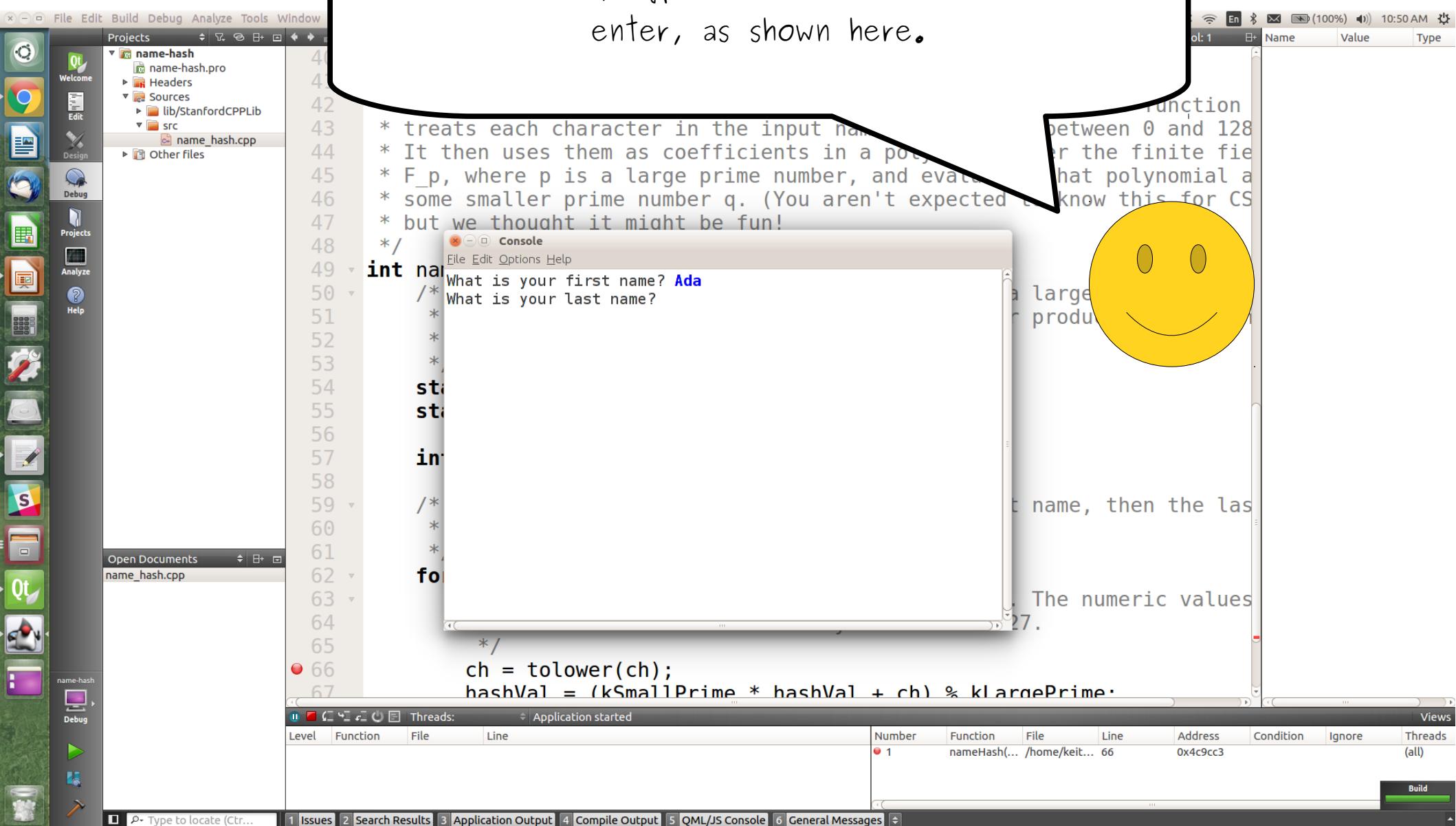
```
* treats each character in the input name
* It then uses them as coefficients in a polynomial
*  $F_p$ , where  $p$  is a large prime number, and evaluates it
* some smaller prime number  $q$ . (You aren't expected to know this for CS
* but we thought it might be fun!
*/
int nameHash(string name) {
    /* ... */
    std::istringstream iss(name);
    string s;
    int hashVal = 1;
    /* ... */
    for (char ch; iss > ch; ch = tolower(ch)) {
        /* ... */
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
}
```

What is your first name? |

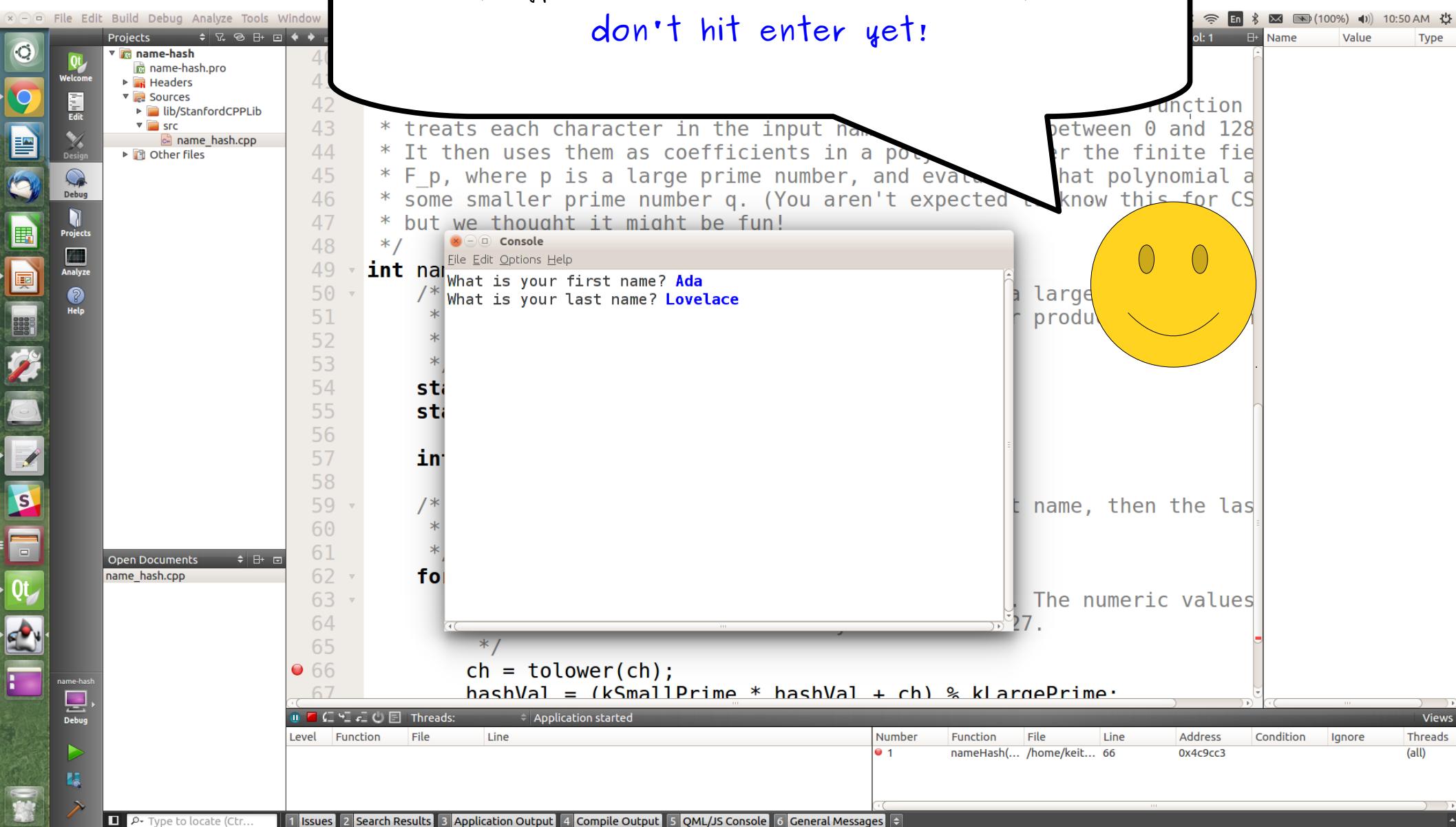
Threads: Application started

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
				1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		

In the meantime, type in the first name Ada and hit enter, as shown here.



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



The screenshot shows a Qt IDE interface with a code editor and a terminal window. The code editor displays a C++ file named `name_hash.cpp` containing a function for calculating a hash value based on a name. The terminal window shows the following interaction:

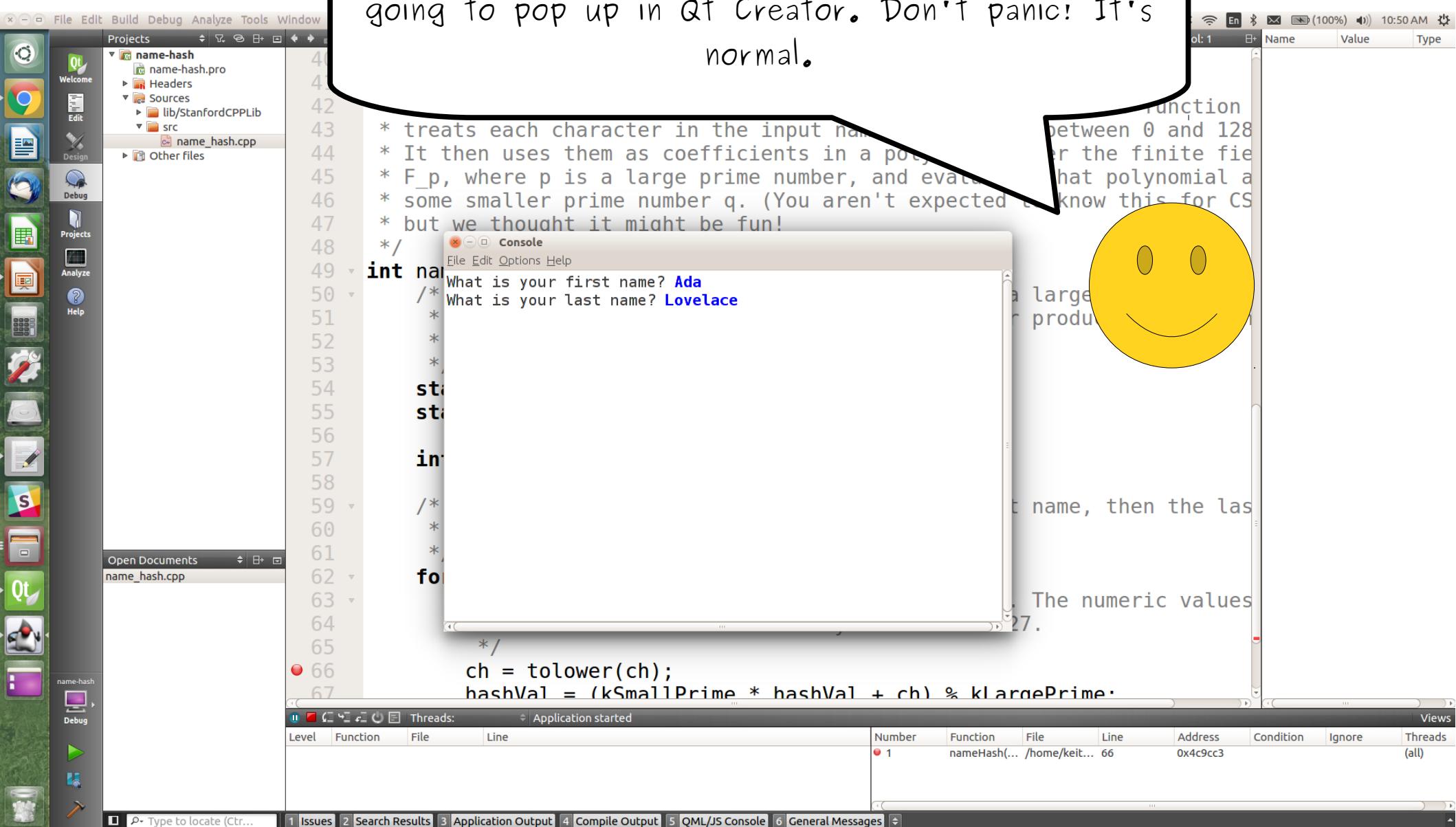
```
What is your first name? Ada
What is your last name? Lovelace
```

A large yellow smiley face icon is overlaid on the right side of the terminal window. A callout bubble points from the text "don't hit enter yet!" to the smiley face, suggesting that the user should not press enter after typing "Lovelace".

Below the terminal window, a table shows a single log entry:

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash(...)	/home/keit...	66	0x4c9cc3							(all)

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



The screenshot shows the Qt Creator IDE interface. The left sidebar contains icons for Welcome, Edit, Design, Projects, Analyze, and Help. The main window shows a project named "name-hash" with files like "name.hash.pro", "Headers", "Sources", "lib/StanfordCPPLib", and "src/name_hash.cpp". The code editor displays "name_hash.cpp" with the following content:

```
41
42
43 * treats each character in the input name
44 * It then uses them as coefficients in a polynomial
45 *  $F_p$ , where  $p$  is a large prime number, and evaluates it
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(const QString& name) {
50     /* ... */
51
52     static const int kSmallPrime = 101;
53     static const int kLargePrime = 104729;
54
55     int hashVal = 1;
56
57     /* ... */
58
59     for (int i = 0; i < name.length(); ++i) {
60         /* ... */
61
62         ch = tolower(ch);
63         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
64
65     }
66
67 }
```

A terminal window titled "Console" is open, showing the output of the program:

```
What is your first name? Ada
What is your last name? Lovelace
```

A large yellow smiley face icon is overlaid on the right side of the interface. The status bar at the bottom shows "Threads: Application started" and a list of tabs: Issues, Search Results, Application Output, Compile Output, QML/JS Console, and General Messages.

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.



```
4
4
47
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 than
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 last
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         * for 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 }
```

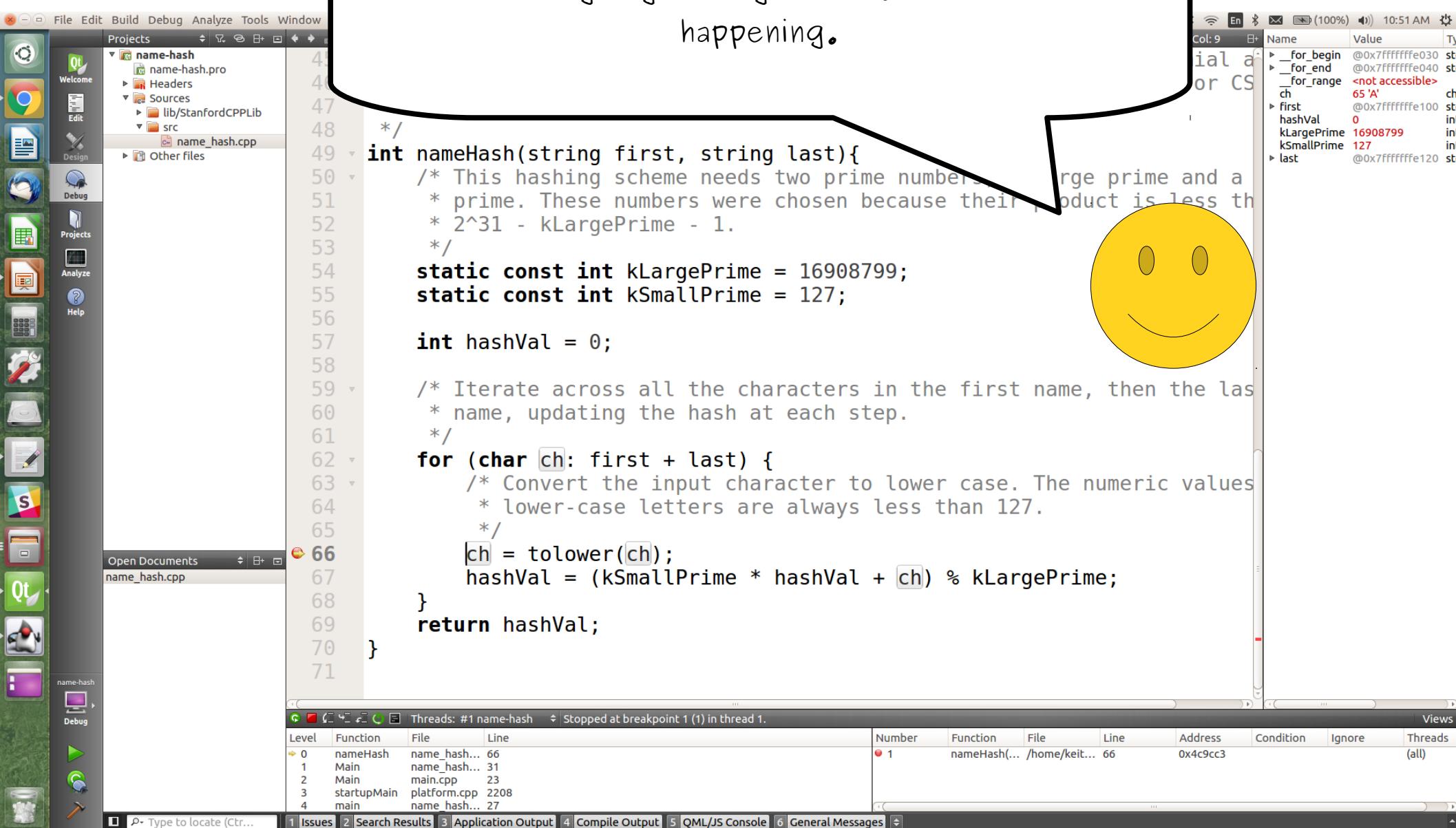
Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl+F)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

There's a lot going on right here. Let's see what's happening.



```
4
4
47
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 than
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 last
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         * for 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 }
```

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Issues Search Results Application Output Compile Output QML/JS Console General Messages

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

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 than
52 * 2^31. This is fine.
53 }

54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71 }

for (char ch: first + last) {
 /* Convert the input character to lower-case. Lower-case letters are always less than
 * than the uppercase letters. The numeric values of the lowercase letters are
 * greater than 127.
 */
 ch = tolower(ch);
 hashVal = (kSmallPrime * hashVal + ch);
}
return hashVal;

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)

Name Value

_for_begin @0x7fffffff030
_for_end @0x7fffffff040
_for_range <not accessible>
ch 65 'A'
first @0x7fffffff100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120

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

then the last

case. The numeric values of the lowercase letters are greater than 127.

me;

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

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.

```
for (char ch: first + last) {  
    /* Convert the input character  
     * lower-case letters are always less  
     */  
    ch = tolower(ch);  
    hashVal = (kSmallPrime * hashVal + ch);  
}  
return hashVal;
```

case. The numeric values are less than 127.



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

Name	Value
_for_begin	@0x7fffffff030
_for_end	@0x7fffffff040
_for_range	<not accessible>
ch	65 'A'
first	@0x7fffffff100
hashVal	0
kLargePrime	16908799
kSmallPrime	127
last	@0x7fffffff120

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
52 * 2^31. This is fine.
53 }

54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71

for (char ch: first + last) {
 /* Convert the input character to lower-case. Lower-case letters are always less
 * than the uppercase letters. The numeric values of the lowercase letters are
 * greater than the uppercase letters. For example, 'a' is 97 and 'A' is 65.
 */
 ch = tolower(ch);
 hashVal = (kSmallPrime * hashVal + ch);
}
return hashVal;

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)

Projects

name hash

- name-hash.pro
- Headers
- Sources
 - lib/StanfordCPPLib
 - src
 - name_hash.cpp
- Other files

Qt Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents name_hash.cpp

name hash

Debug

Type to locate (Ctrl...)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

Views

10:51 AM

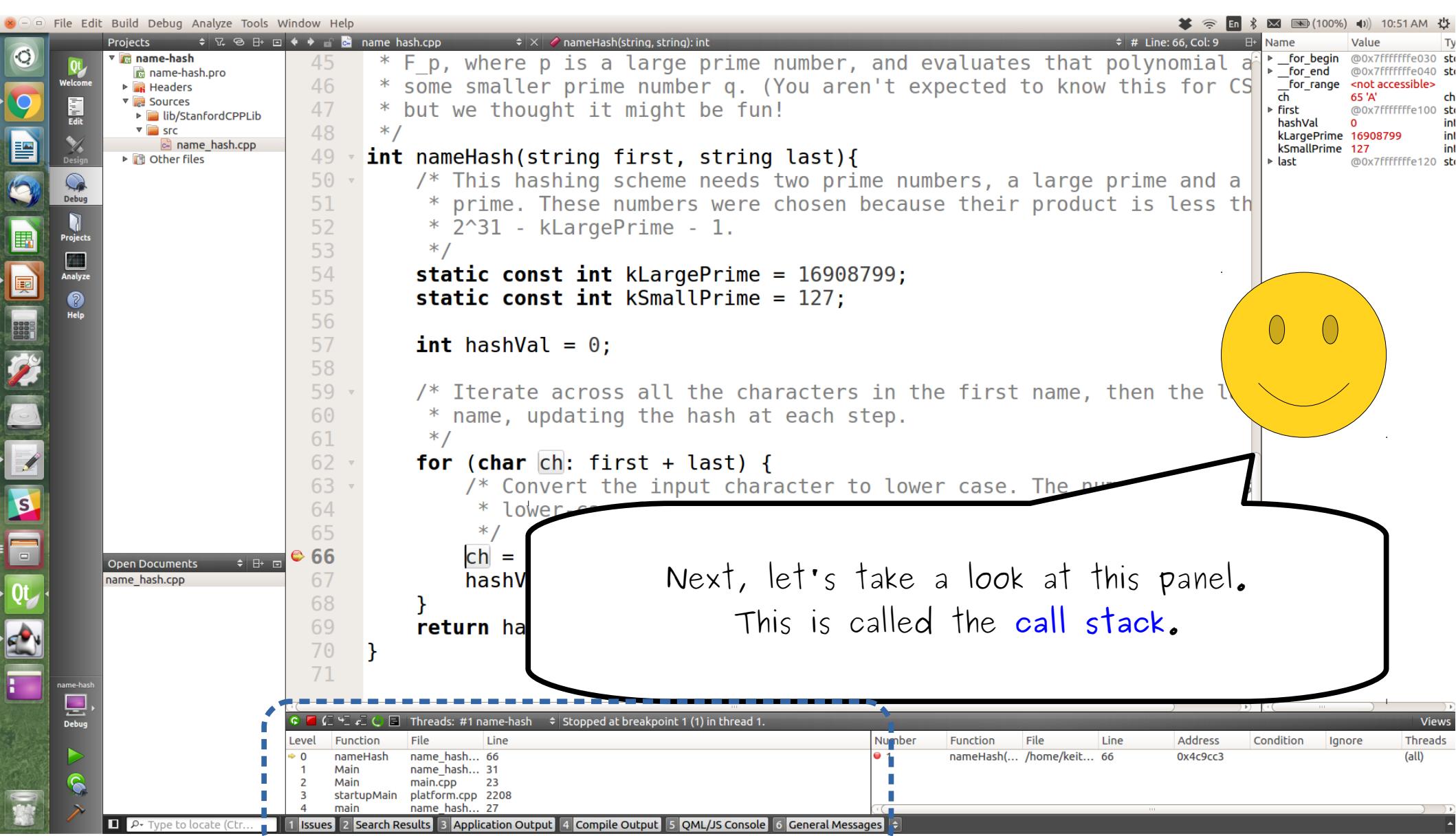
Whenever you pop up the debugger, it's good to figure out exactly where you are in the program that you're running, so you'll get into the habit of checking for this yellow arrow.

then the last

case. The numeric values are greater than 127.

me;





File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numbers
64 * lower case characters are mapped to the numbers
65 */
66 ch = tolower(ch);
67 hashVal = hashVal * kSmallPrime + ch;
68 }
69 return hashVal;
70 }
71 }

Open Documents name_hash.cpp

66

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/keith...	66	0x4c9cc3			(all)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

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

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numbers
64 * lower case.
65 */
66 ch = tolower(ch);
67 hashVal = (hashVal * kSmallPrime + ch) % kLargePrime;
68 }
69 return hashVal;
70 }
71 }

Open Documents name_hash.cpp

name hash

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

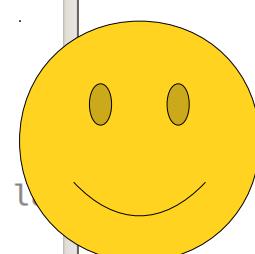
Number Function File Line Address Condition Ignore Threads

1 nameHash(... /home/keith... 66 0x4c9cc3 (all)

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

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!



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numbers
64 * lower case
65 */
66 ch = tolower(ch);
67 hashVal = (hashVal * kSmallPrime + ch) % kLargePrime;
68 }
69 return hashVal;
70 }
71 }

66

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/keith...	66	0x4c9cc3			(all)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

1 2 3 4 5 6

Qt Welcome Edit Design Debug Projects Analyze Help

name hash

name-hash.pro Headers Sources lib/StanfordCPPLib src name_hash.cpp Other files

Open Documents name_hash.cpp

name hash

Debug

Qt Projects Analyze Help

Smiley face icon

Handwritten note: 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?

The screenshot shows the Qt Creator IDE interface. The code editor displays a C++ file named `name_hash.cpp` with the following content:

```
45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
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 last
60     * name, updating the hash at each step.
61     */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The numbers
64         * lower case.
65         */
66         ch = tolower(ch);
67         hashVal = (hashVal * kSmallPrime + ch) % kLargePrime;
68     }
69
70     return hashVal;
71 }
```

A yellow smiley face is drawn on the right side of the screen, with a black line connecting it to a callout bubble. The bubble contains the handwritten text: "The call stack can tell us exactly that!"

The bottom of the interface shows the call stack window, which is currently empty. The status bar at the bottom indicates: "Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1." The bottom navigation bar includes tabs for Issues, Search Results, Application Output, Compile Output, QML/JS Console, and General Messages.

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numbers
64 * lower case
65 */
66 ch = tolower(ch);
67 hashVal = (hashVal * kSmallPrime + ch) % kLargePrime;
68 }
69 return hashVal;
70 }
71 }

66

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/keith...	66	0x4c9cc3			(all)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

Qt Welcome Edit Design Debug Projects Analyze Help

name hash

name-hash.pro Headers Sources lib/StanfordCPPLib src name_hash.cpp Other files

Open Documents name_hash.cpp

name hash

Debug

Smiley face icon

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`.

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numbers
64 * lower case characters are mapped to the numbers
65 */
66 ch = hashVal;
67 }
68 return hashVal;
69 }
70
71 }

66

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level Function File Line

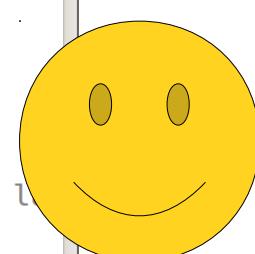
0 nameHash name.hash 66
1 Main name hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name hash... 27

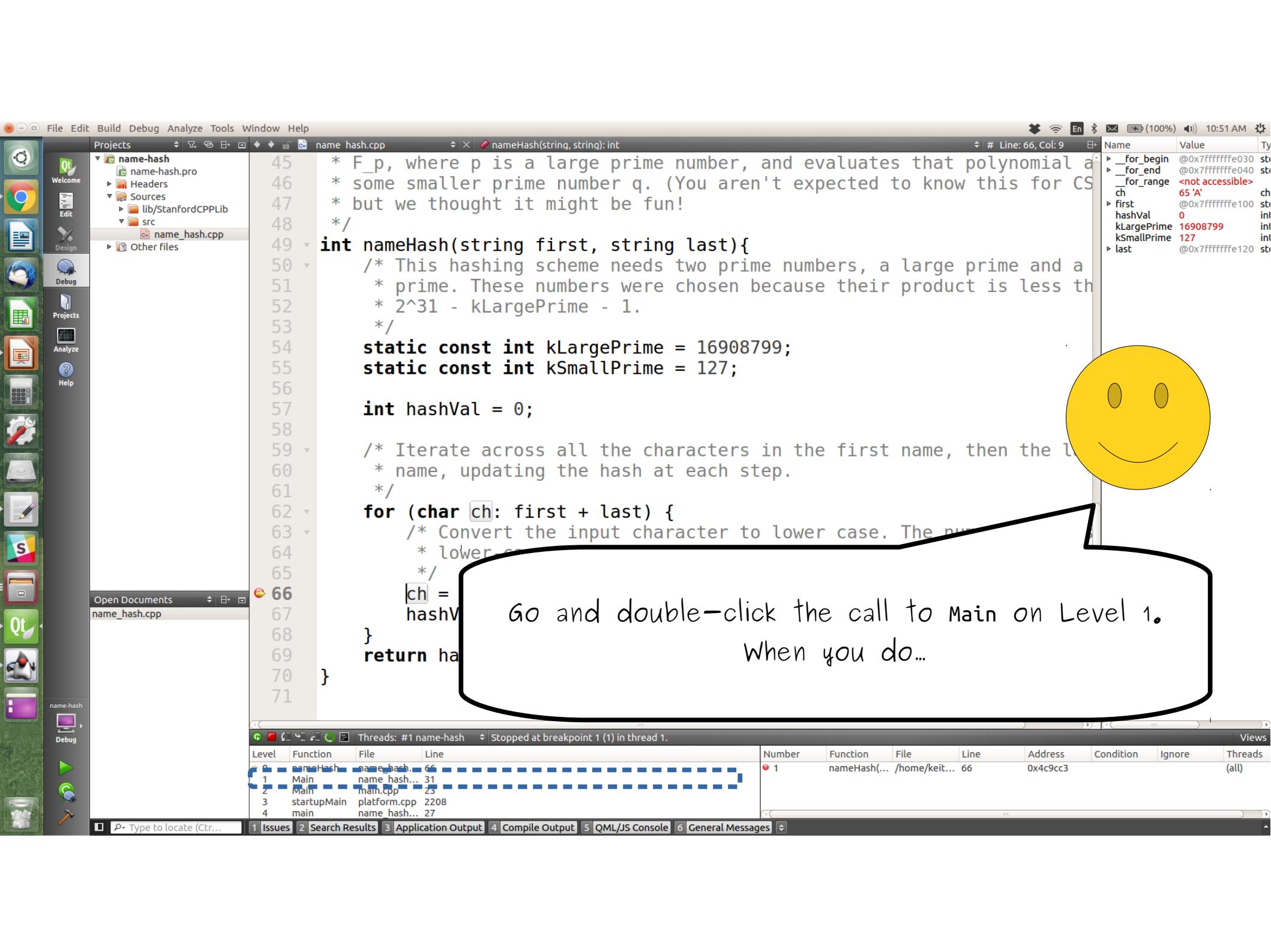
Number Function File Line Address Condition Ignore Threads

1 nameHash(... /home/keit... 66 0x4c9cc3 (all)

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console 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.cpp # Line: 31, Col: 5

name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

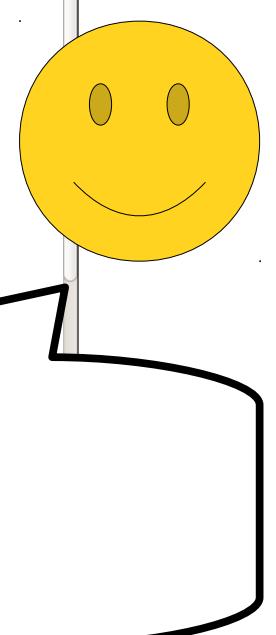
```
18 #include "console.h"
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the function
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string 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 actual
38 * to talk more
39 * the meaning
40 * of the input
41 *
42 * For those
43 * treats each
44 * It then uses
45 * F n where n is
```

Open Documents name_hash.cpp

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages



... you'll end up over here!

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp # Line: 31, Col: 5

name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

```
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 actual
38 * to talk more
39 * the meaning
40 * of the input
41 *
42 * For those
43 * treats each
44 * It then uses
45 * F n where n is
```

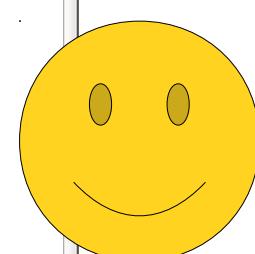
Open Documents name_hash.cpp

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Notice that the highlighted line here includes a call to the `nameHash` function. This is the part of the code that actually called `nameHash`, which is how we got to the line with the breakpoint!



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp # Line: 31, Col: 5

name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

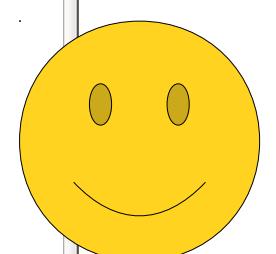
```
18 #include "console.h"
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for the nameHash function. This lets us use the function
23 * in main and then define it later in the program.
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string 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 actual
38 * to talk more
39 * the meaning
40 * of the input
41 *
42 * For those
43 * treats each
44 * It then uses
45 * For where n
```

Open Documents name_hash.cpp

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages



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!

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp # Line: 31, Col: 5

name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

```
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 actual
38 * to talk more
39 * the meaning
40 * of the input
41 *
42 * For those
43 * treats each
44 * It then uses
45 * F n where n is
```

Open Documents name_hash.cpp

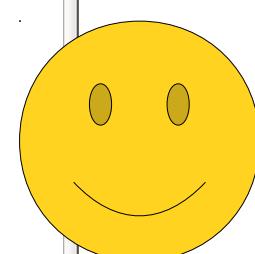
Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

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

Name Value Type

first @0x7fffffff0a0 std::int64_t
last @0x7fffffff0c0 std::int64_t
hashValue 766504679 int

You might notice that there's some more stuff in the call stack beyond just main and nameHash. What are those?



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp # Line: 31, Col: 5

name-hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

```
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 actual
38 * to talk more
39 * the meaning
40 * of the input
41 *
42 * For those
43 * treats each
44 * It then uses
45 * F n where n is
```

Open Documents name_hash.cpp

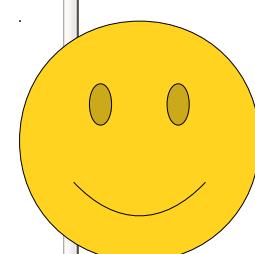
Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

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

Views

Issues Search Results Application Output Compile Output QML/JS Console General Messages

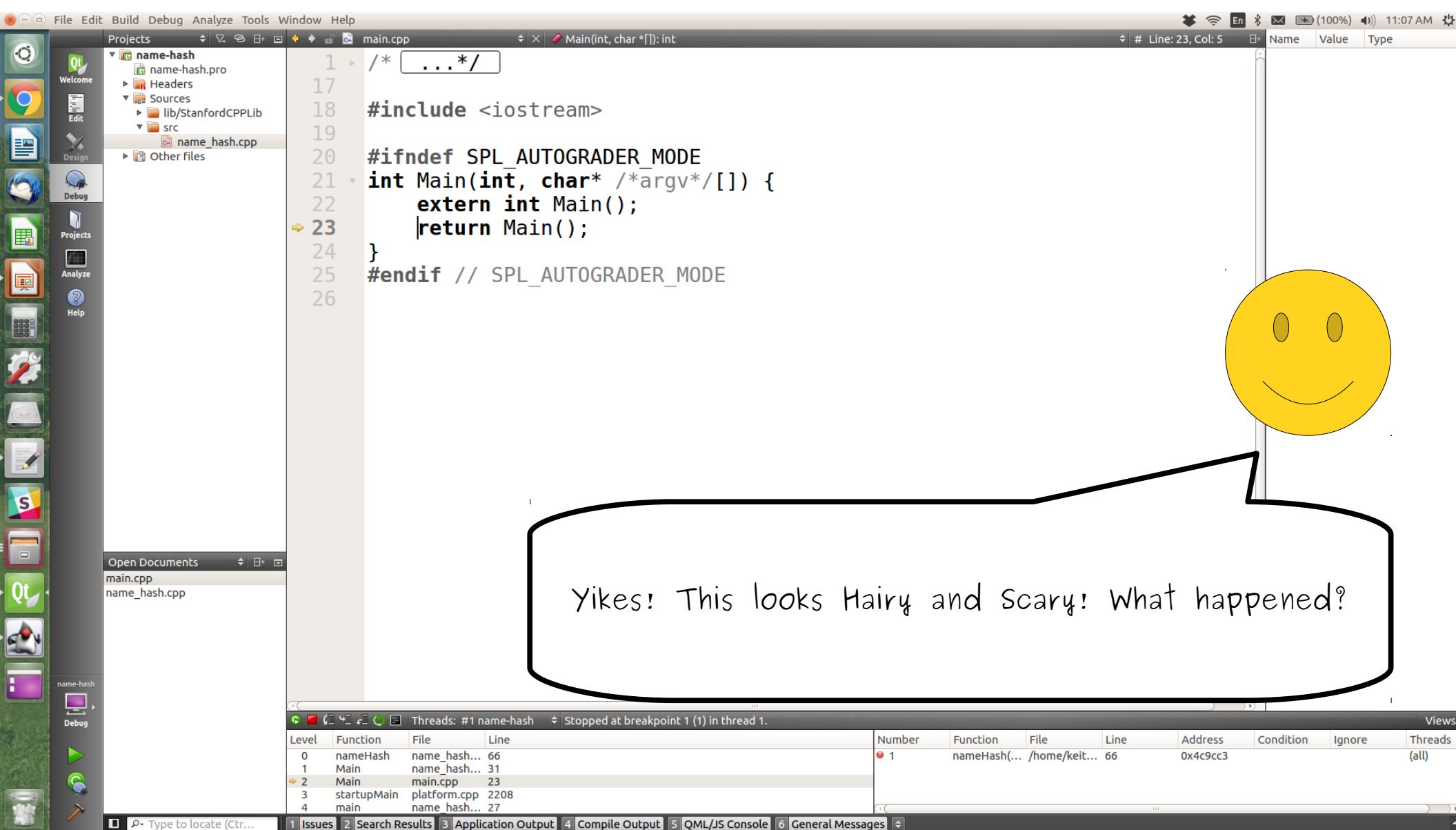
Let's find out! Double-click on the line marked "Main" on Level 2. When you do...



The screenshot shows the Qt Creator IDE interface. The main window displays a C++ file named `main.cpp` with the following code:

```
1  /* ... */
2
3  #include <iostream>
4
5  #ifndef SPL_AUTOGRADER_MODE
6  int Main(int, char* /*argv*/[]) {
7      extern int Main();
8      return Main();
9  }
10 #endif // SPL_AUTOGRADER_MODE
```

The code includes a conditional compilation directive for the Stanford AutoGrader. A hand-drawn annotation is overlaid on the interface, consisting of a large yellow smiley face and a speech bubble. The speech bubble contains the text: "... you'll end up with something that looks like this." A line connects the smiley face to the bottom of the speech bubble.



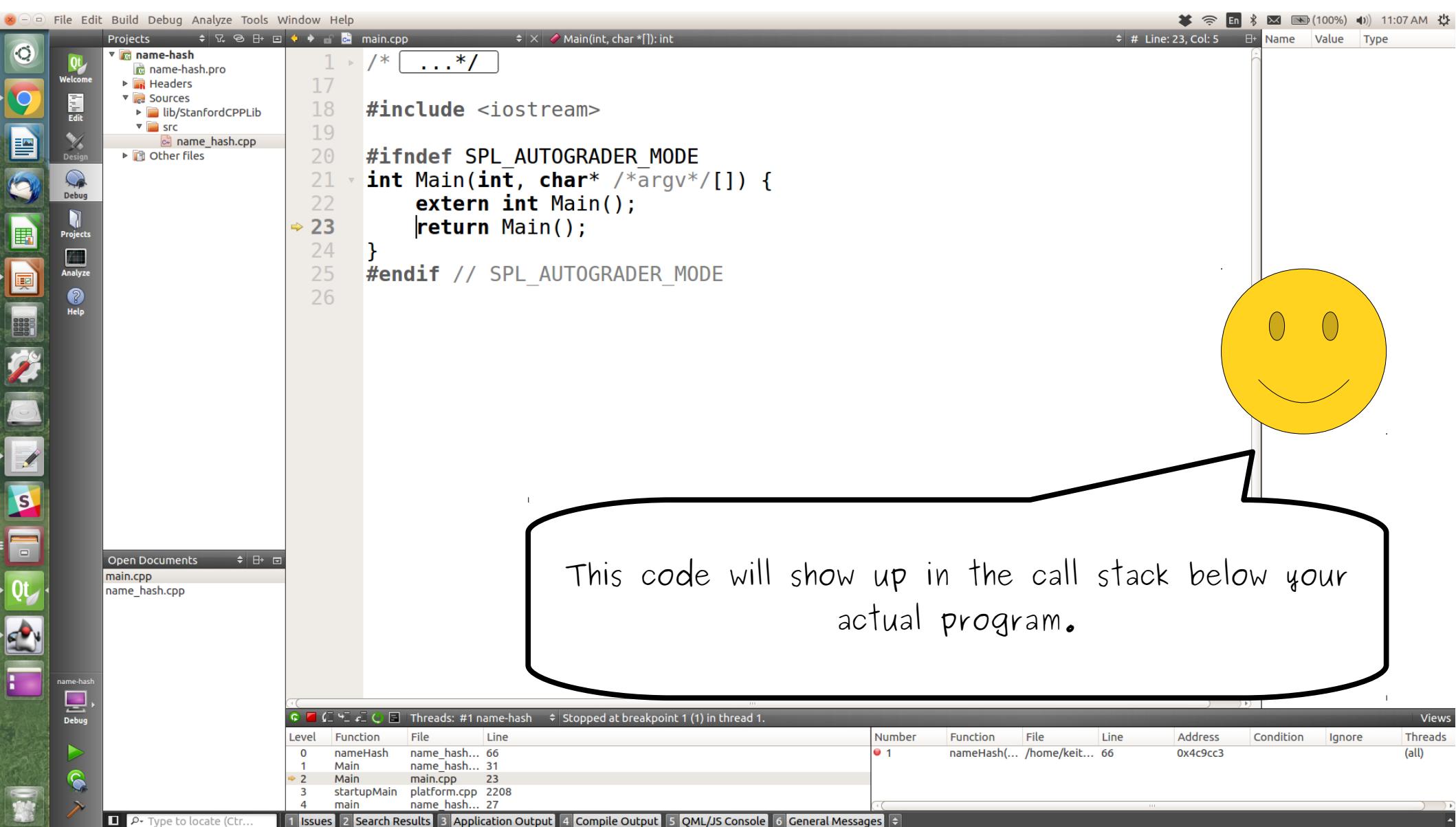
The screenshot shows the Qt Creator IDE interface. The code editor displays a portion of `main.cpp` with the following content:

```
1  /* ... */
2
3  #include <iostream>
4
5  #ifndef SPL_AUTOGRADER_MODE
6  int Main(int, char* /*argv*/[]) {
7      extern int Main();
8      return Main();
9  }
10 #endif // SPL_AUTOGRADER_MODE
```

A large yellow smiley face is drawn on the right side of the screen, with a speech bubble pointing towards the code. The speech bubble contains the following handwritten text:

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.

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.



The screenshot shows a Qt IDE interface with the following details:

- File Menu:** File, Edit, Build, Debug, Analyze, Tools, Window, Help.
- Projects View:** Shows a project named "name-hash" with files: name-hash.pro, Headers, Sources (lib/StanfordCPPLib, src), name_hash.cpp, and Other files.
- Code Editor:** The main window displays the "main.cpp" file. The code includes a header guard, an include directive for `<iostream>`, and a conditional block for `SPL_AUTOGRADER_MODE`. The conditional block contains a declaration of `extern int Main();` and a return statement. The line number 23 is highlighted.
- Callout Bubble:** A large, hand-drawn style callout bubble points from the right side of the code editor towards the text in the annotation.
- Annotation Text:** The text inside the callout bubble reads: "This code will show up in the call stack below your actual program."
- Call Stack View:** At the bottom, the "Threads" view shows the call stack: "Threads: #1 name-hash" and "Stopped at breakpoint 1 (1) in thread 1." The stack trace includes entries for `nameHash`, `Main`, `main`, `startupMain`, and `main`.
- Bottom Bar:** Includes tabs for Issues, Search Results, Application Output, Compile Output, QML/JS Console, and General Messages.

The screenshot shows the Qt Creator IDE interface. The main window displays a C++ file named `main.cpp` with the following code:

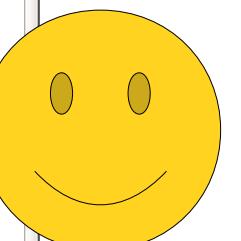
```
1  /* ... */
2
3  #include <iostream>
4
5  #ifndef SPL_AUTOGRADER_MODE
6  int Main(int, char* /*argv*/[]) {
7      extern int Main();
8      return Main();
9  }
10 #endif // SPL_AUTOGRADER_MODE
```

The code editor has a line 23 highlighted with a yellow arrow. The bottom status bar indicates: "Threads: #1 name-hash" and "Stopped at breakpoint 1 (1) in thread 1." The bottom navigation bar includes tabs for "Issues", "Search Results", "Application Output", "Compile Output", "QML/JS Console", and "General Messages".

A large, hand-drawn style callout bubble originates from the bottom right of the code editor, pointing towards the center of the annotation area. The bubble contains the following text:

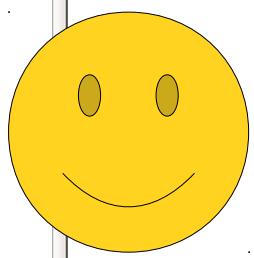
You shouldn't need to dig around this deep in the call stack, and if you do, it should probably be a message telling you to back up a bit back to code that you actually wrote.

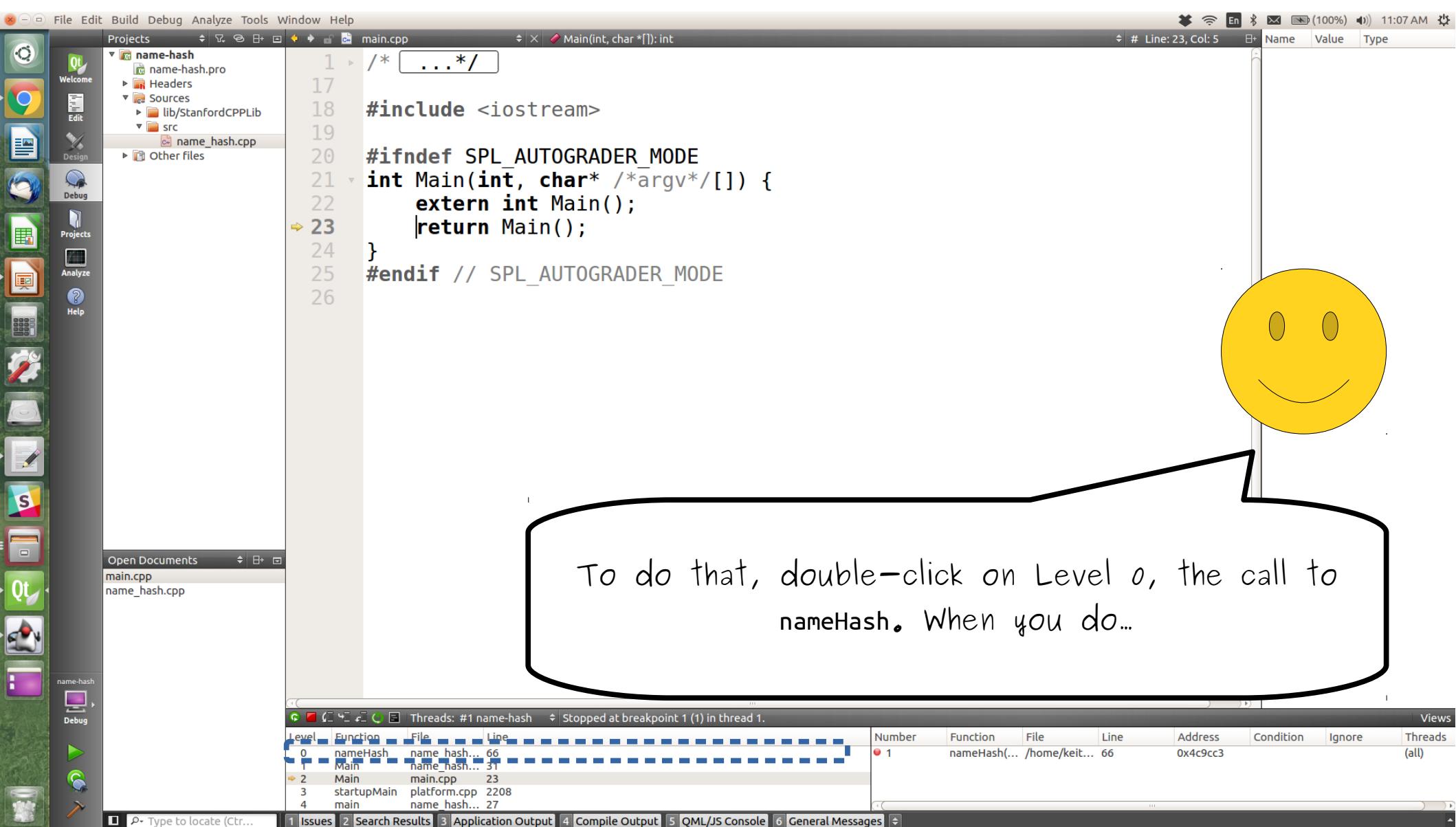
A large yellow smiley face icon is positioned in the upper right corner of the annotation area.



So let's jump back to the code that we actually wrote.

```
1  /* ... */
2
3  #include <iostream>
4
5  #ifndef SPL_AUTOGRADER_MODE
6  int Main(int, char* /*argv*/[]) {
7      extern int Main();
8      return Main();
9  }
10 #endif // SPL_AUTOGRADER_MODE
```





The screenshot shows a Qt IDE interface with the following components:

- Top Bar:** File, Edit, Build, Debug, Analyze, Tools, Window, Help.
- Projects View:** Shows a project named "name-hash" with files: name-hash.pro, Headers, Sources (lib/StanfordCPPLib, src), name_hash.cpp, and Other files.
- Code Editor:** Displays main.cpp with the following code:

```
1  /* ... */
2
3  #include <iostream>
4
5  #ifndef SPL_AUTOGRADER_MODE
6  int Main(int, char* /*argv*/[]) {
7      extern int Main();
8      return Main();
9  }
10 #endif // SPL_AUTOGRADER_MODE
```
- Open Documents:** main.cpp, name_hash.cpp.
- Threads View:** Shows a stack trace for thread 1, stopped at breakpoint 1 (line 66). The stack trace is:

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
- Registers View:** Shows a table with columns: Number, Function, File, Line, Address, Condition, Ignore, Threads. One entry is visible: 1, nameHash(... /home/keit... 66, 0x4c9cc3, (all)).
- Bottom Bar:** Issues, Search Results, Application Output, Compile Output, QML/JS Console, General Messages.

A yellow smiley face is positioned in the upper right area of the interface. A large, hand-drawn-style callout bubble originates from the bottom right, pointing towards the center of the code editor area. The text inside the bubble reads:

To do that, double-click on Level 0, the call to nameHash. When you do...

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
60 * name, updating the hash at each step.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numbers
64 * lower case characters are mapped to the same numbers as
65 * upper case characters.
66 */
67 ch = tolower(ch);
68 hashVal = (hashVal * kSmallPrime + ch) % kLargePrime;
69 }
70 return hashVal;
71 }

66

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Number Function File Line Address Condition Ignore Threads

0 nameHash name_hash... 66 0x4c9cc3 (all)

1 Main name_hash... 31

2 Main main.cpp 23

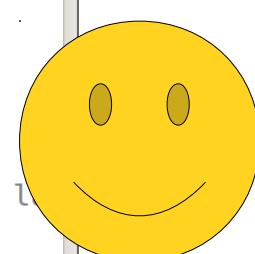
3 startupMain platform.cpp 2208

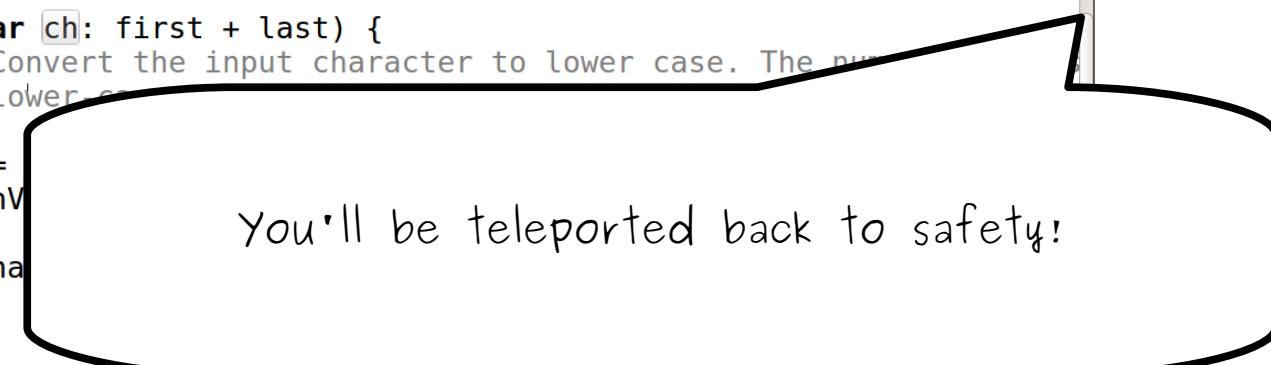
4 main name hash... 27

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

You'll be teleported back to safety!







File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * $2^{31} - kLargePrime - 1$.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal;
58
59 /* Iterates over each character in the string
60 * naming the character 'ch'.
61 */
62 for (char ch; ch != '\0'; ch = tolower(ch)){
63 /*
64 * Hash function: hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
65 */
66 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
67 }
68 return hashVal;
69 }
70
71 }

Line: 66, Col: 9

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 65 'A'
first @0x7fffffff100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120

Open Documents main.cpp name_hash.cpp

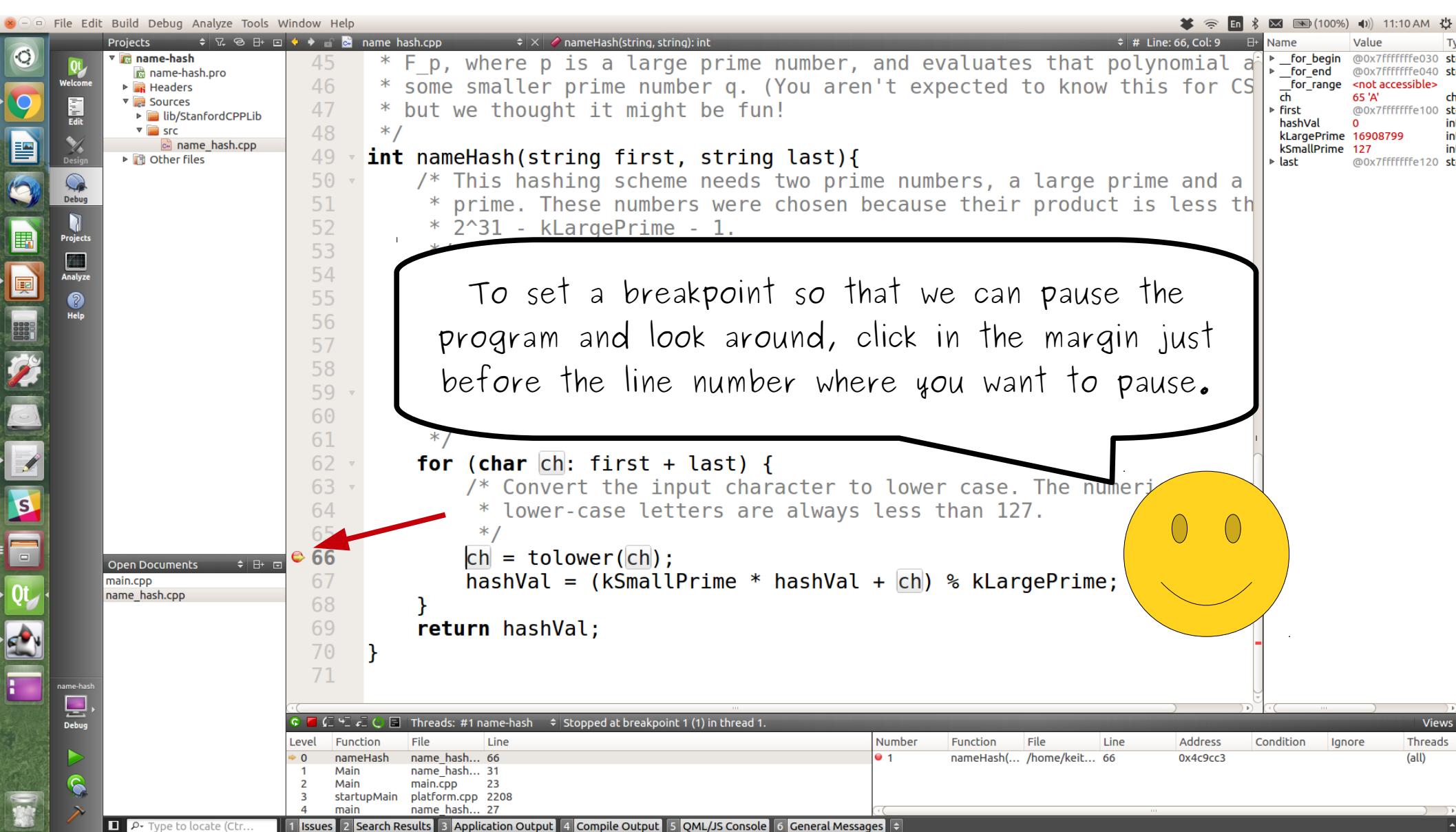
Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Let's quickly recap what we've seen so far.



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

name hash

- name-hash
- name-hash.pro
- Headers
- Sources
 - lib/StanfordCPPLib
 - src
- Other files

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 than
52 * 2^31 - kLargePrime - 1.
53 */
54
55 /*
56 * For each character in the string, convert it to lower case and add it to the
57 * hash value. The hash value is initialized to 0.
58 */
59 for (char ch: first + last) {
60 /* Convert the input character to lower case. The numeric
61 * lower-case letters are always less than 127.
62 */
63 ch = tolower(ch);
64 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
65 }
66 return hashVal;
67 }
68
69
70
71 }

Line: 66, Col: 9

Name Value

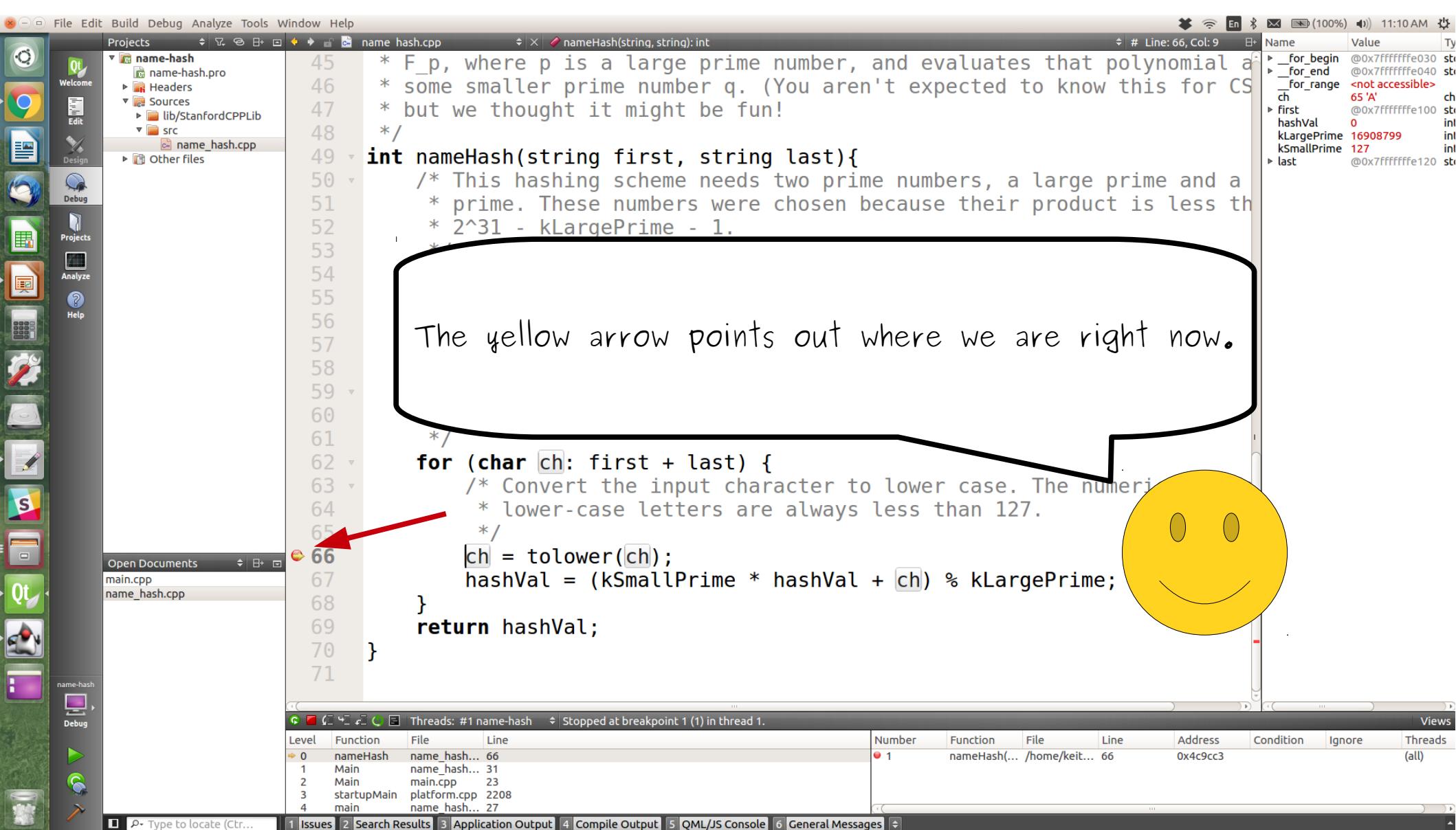
- for_begin @0x7fffffff030
- for_end @0x7fffffff040
- for_range <not accessible>
- ch 65 'A'
- first @0x7fffffff100
- hashVal 0
- kLargePrime 16908799
- kSmallPrime 127
- last @0x7fffffff120

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

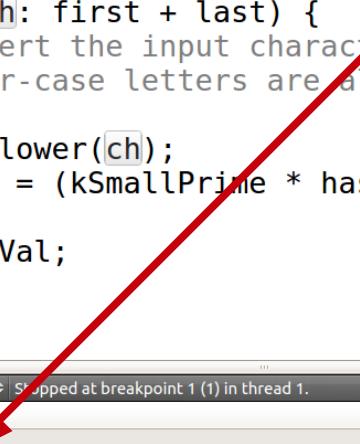
Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages



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



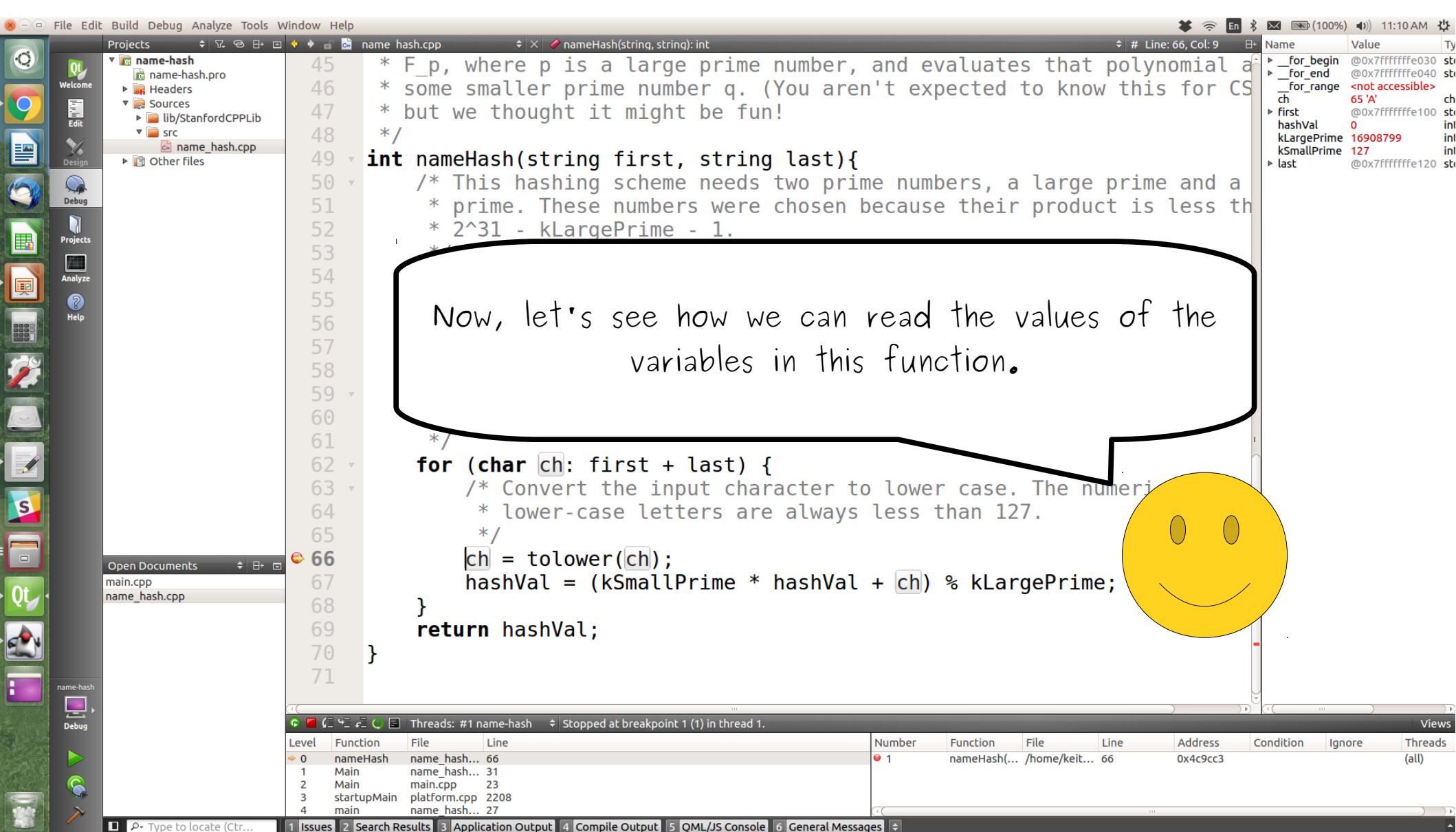
```
File Edit Build Debug Analyze Tools Window Help
Projects name hash.cpp nameHash(string, string): int
45 * F_p, where p is a large prime number, and evaluates that polynomial a
46 * some smaller prime number q. (You aren't expected to know this for CS
47 * but we thought it might be fun!
48 */
49 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 than
52     * 2^31 - kLargePrime - 1.
53     */
54
55
56
57
58
59
60
61
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The numeric
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 }
```

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

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages



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

```
45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52     * 2^31 - kLargePrime - 1.
53     */
54
55     /*
56     */
57
58     /*
59     */
60
61     /*
62     */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numerical
65         * lower-case letters are always less than 127.
66         */
67         ch = tolower(ch);
68         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69     }
70     return hashVal;
71 }
```

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)

Look up at this panel over here.



```
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 can't be run:
 */
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 than
 $2^{31} - kLargePrime - 1$.

 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;
}

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 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.



```
45
46
47 * but we
48 */ be run:
49
50     nameHash(string first, string last){
51
52     /* This hashing scheme needs two prime numbers, a large prime and a
53     prime. These numbers were chosen because their product is less than
54     2^31 - kLargePrime - 1.
55
56     static const int kLargePrime = 16908799;
57     static const int kSmallPrime = 127;
58
59     int hashVal = 0;
60
61     /* Iterate across all the characters in the first name, then the last
62     * name, updating the hash at each step.
63     */
64     for (char ch: first + last) {
65
66         /* Convert the input character to lower case. The numeric values
67         * lower-case letters are always less than 127.
68         */
69         ch = tolower(ch);
70         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
71     }
72     return hashVal;
73 }
```

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 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`.



If we ignore the weird-looking ones, we can see some nice, familiar names.



```
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
```

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 can't be run:
    */
    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 than
        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
            * for lower-case letters are always less than 127.
            */
            ch = tolower(ch);
            hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
        }
        return hashVal;
    }
```

11:10 AM

Name	Value
_for_begin	@0x7fffffff030
_for_end	@0x7fffffff040
_for_range	<not accessible>
ch	65 'A'
first	@0x7fffffff100
hashVal	0
kLargePrime	16908799
kSmallPrime	127
last	@0x7fffffff120

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

Type to locate (Ctrl...) 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.



```
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 can't be run:
 */
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 than
 $2^{31} - kLargePrime - 1$.
*/
 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;
}

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 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.



45
46
47 * but we do not want to be run:
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 than
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 last
60 name, updating the hash at each step. */**
61 **for (char ch: first + last) {**
62 **/* Convert the input character to lower case. The numeric values
63 of lower-case letters are always less than 127. */**
64 **ch** = **tolower(ch);**
65 hashVal = (kSmallPrime * hashVal + **ch**) % kLargePrime;
66 **}**
67 **return** hashVal;
68
69
70
71 }

66

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 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.



```
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
```

As we walk through the program one step at a time, we'll see these values change.

```
/* but we can't see the code yet because it hasn't been run:
 */
int nameHash(string first, string last){
    /* This hashing scheme needs two prime numbers, a large prime and a
     * smaller prime. These numbers were chosen because their product is less than
     * 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
         * for lower-case letters are always less than 127.
         */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
    return hashVal;
}
```

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 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.



```
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 can't be run:
*/
+ 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 than
 $2^{31} - kLargePrime - 1$.

 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
 * lower-case letters are always less than 127.
 */
 ch = tolower(ch);
 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
 }
 return hashVal;
}

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)

This loop is a **range-based for loop**. It says "for each character in the string `first + last`, do something with that character."



```
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 can't be run:
 */
+ 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 than
 $2^{31} - kLargePrime - 1$.
*/
 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;
}

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)

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Remember (from a while back) that we entered the name **Ada Lovelace**.



```
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 can't run:
*/
+ 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 than
 $2^{31} - kLargePrime - 1$.

 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
 * for lower-case letters are always less than 127.
 */
 ch = tolower(ch);
 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
 }
 return hashVal;

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)

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.



Name	Value	Type
► _for_begin	@0x7fffffe030	std::string
► _for_end	@0x7fffffe040	std::string
► _for_range	not accessible	
ch	65 'A'	char
first	@0x7fffffe100	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
► last	@0x7fffffe120	std::string

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.



```
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
```

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 do not yet know what will be run:  
    */  
    int nameHash(string first, string last){  
        /* This hashing scheme needs two prime numbers, a large prime and a  
        small prime. These numbers were chosen because their product is less than  
        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;  
    }
```

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

Debug

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)

Type to locate (Ctrl+F)

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!



```
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
```

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 can't be run:
    */
    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 than
            $2^{31} - kLargePrime - 1$ .
        */
        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;
    }
```

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

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	66	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 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.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
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 }

66

Open Documents main.cpp name_hash.cpp

Threads: #1 name-hash

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

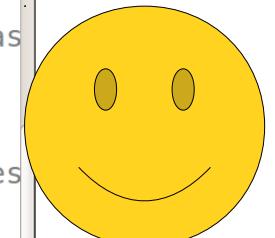
Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

11:10 AM

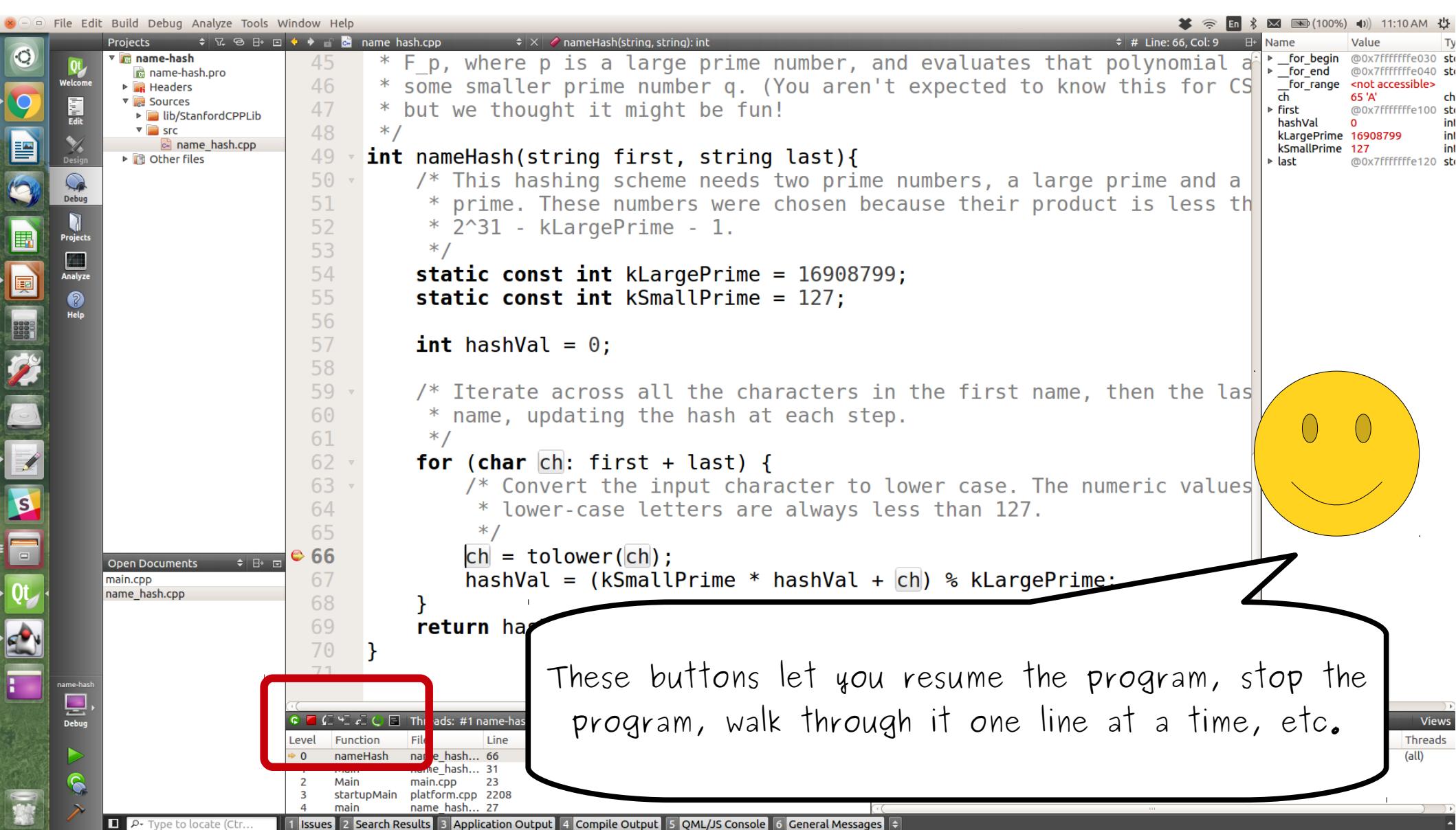
Qt Welcome Edit Design Debug Projects Analyze Help

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 65 'A'
first @0x7fffffff100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120



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.cpp nameHash(string, string): int

name hash

- name-hash
- name-hash.pro
- Headers
- Sources
 - lib/StanfordCPPLib
 - src
- name_hash.cpp
- Other files

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 than
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 last
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 }

66

Threads: #1 name-hash

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

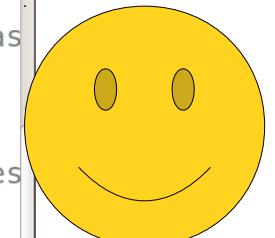
Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages

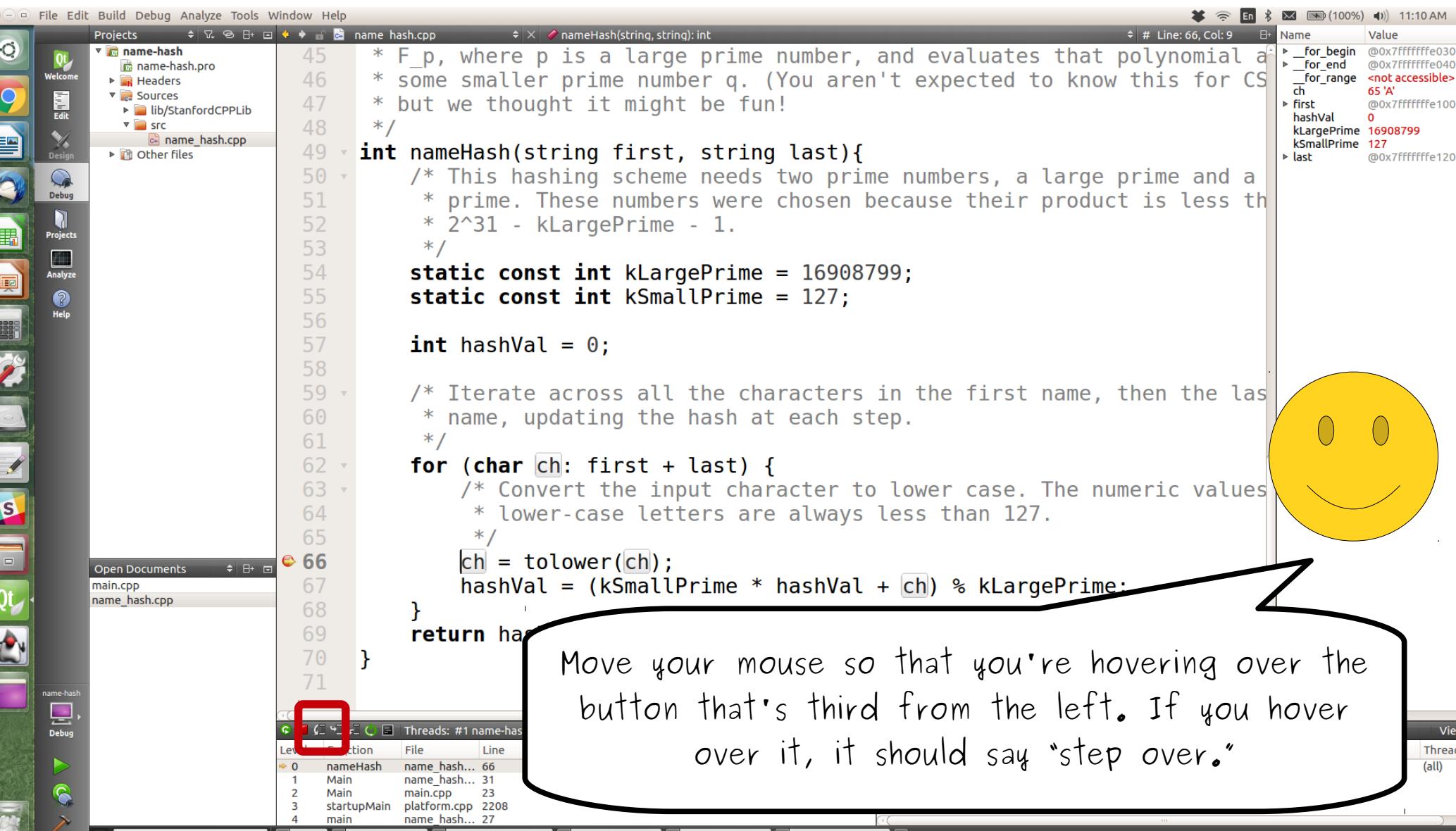
1 2 3 4 5 6

Name Value

- for_begin @0x7fffffff030
- for_end @0x7fffffff040
- for_range <not accessible>
- ch 65 'A'
- first @0x7fffffff100
- hashVal 0
- kLargePrime 16908799
- kSmallPrime 127
- last @0x7fffffff120

These buttons let you resume the program, stop the program, walk through it one line at a time, etc.





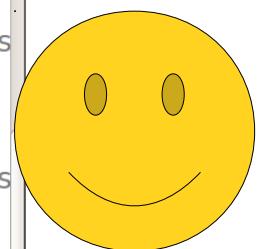
Move your mouse so that you're hovering over the button that's third from the left. If you hover over it, it should say "step over."

```
* F_p, where p is a large prime number, and evaluates that polynomial at
* 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 than
     * 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 for
         * lower-case letters are always less than 127.
        */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
    return hashVal;
}
```

Name	Value
_for_begin	@0x7fffffff030
_for_end	@0x7fffffff040
_for_range	<not accessible>
ch	65 'A'
first	@0x7fffffff100
hashVal	0
kLargePrime	16908799
kSmallPrime	127
last	@0x7fffffff120



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
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 last
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 }

66

Threads: #1 name-hash

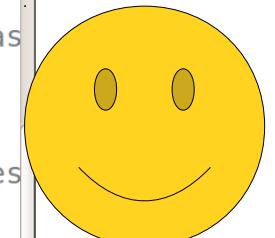
Level Function File Line

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 (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console 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...



Views

Threads (all)

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

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 than
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 last
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 }

Open Documents main.cpp name_hash.cpp

name hash

Threads: #1 name-hash

Level Function File Line

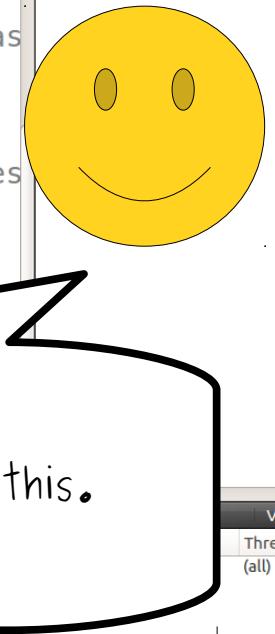
0 nameHash name_hash... 67
1 Main name_hash... 31
2 Main main.cpp 23
3 startupMain platform.cpp 2208
4 main name hash... 27

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

Name Value

_for_begin @0x7fffffe030
_for_end @0x7fffffe040
_for_range <not accessible>
ch 97 'a'
first @0x7fffffe100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffe120



...your window should look something like this.

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
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 last
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 }

Open Documents main.cpp name_hash.cpp

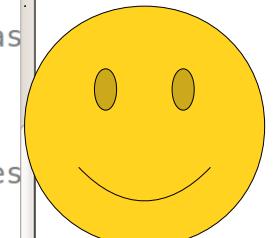
Threads: #1 name-hash

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

Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 97 'a'
first @0x7fffffff100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120



Okay! A few things have changed. Let's see what's going on.

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
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 last
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 }

Open Documents main.cpp name_hash.cpp

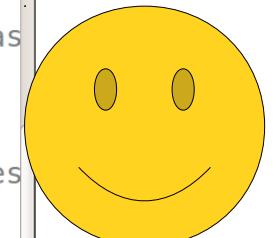
name hash

Threads: #1 name-hash

Views

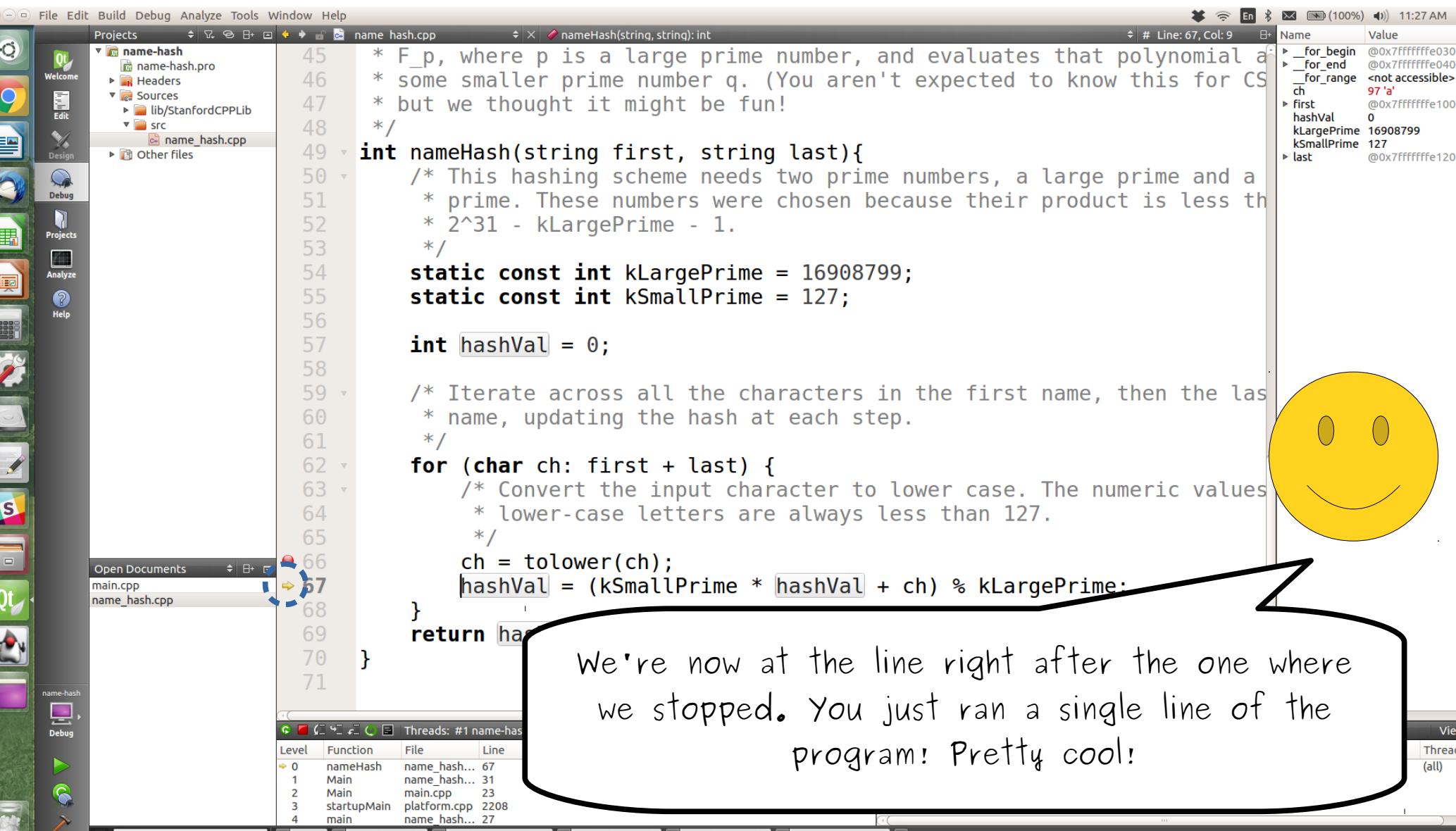
Threads (all)

Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages



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

For_begin @0x7fffffe030
For_end @0x7fffffe040
For_range <not accessible>
ch 97 'a'
First @0x7fffffe100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffe120



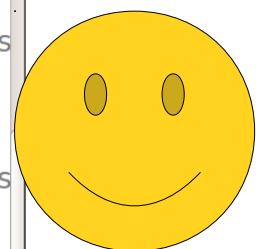
The screenshot shows the Qt Creator IDE interface. The main window displays a C++ file named `name_hash.cpp` with the following code:

```
* F_p, where p is a large prime number, and evaluates that polynomial at
* 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 than
     * 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 for
         * lower-case letters are always less than 127.
        */
        ch = tolower(ch);
        hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
    }
    return hashVal;
}
```

The code editor has a blue selection bar highlighting the line `hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;`. A large yellow smiley face icon is positioned on the right side of the screen. A black callout bubble with a hand-drawn style contains the text: "We're now at the line right after the one where we stopped. You just ran a single line of the program! Pretty cool!".



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

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 than
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 last
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 }

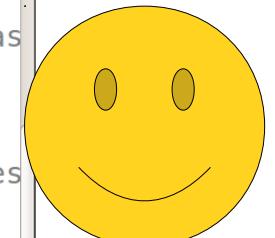
Threads: #1 name-hash

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

Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 97 'a'
first @0x7fffffff100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120



So what did that line of code do?

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

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 than
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 last
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 * for 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 }

Open Documents main.cpp name_hash.cpp

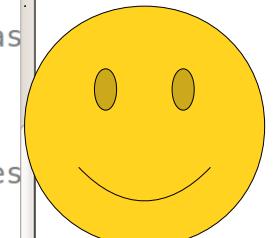
Threads: #1 name-hash

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

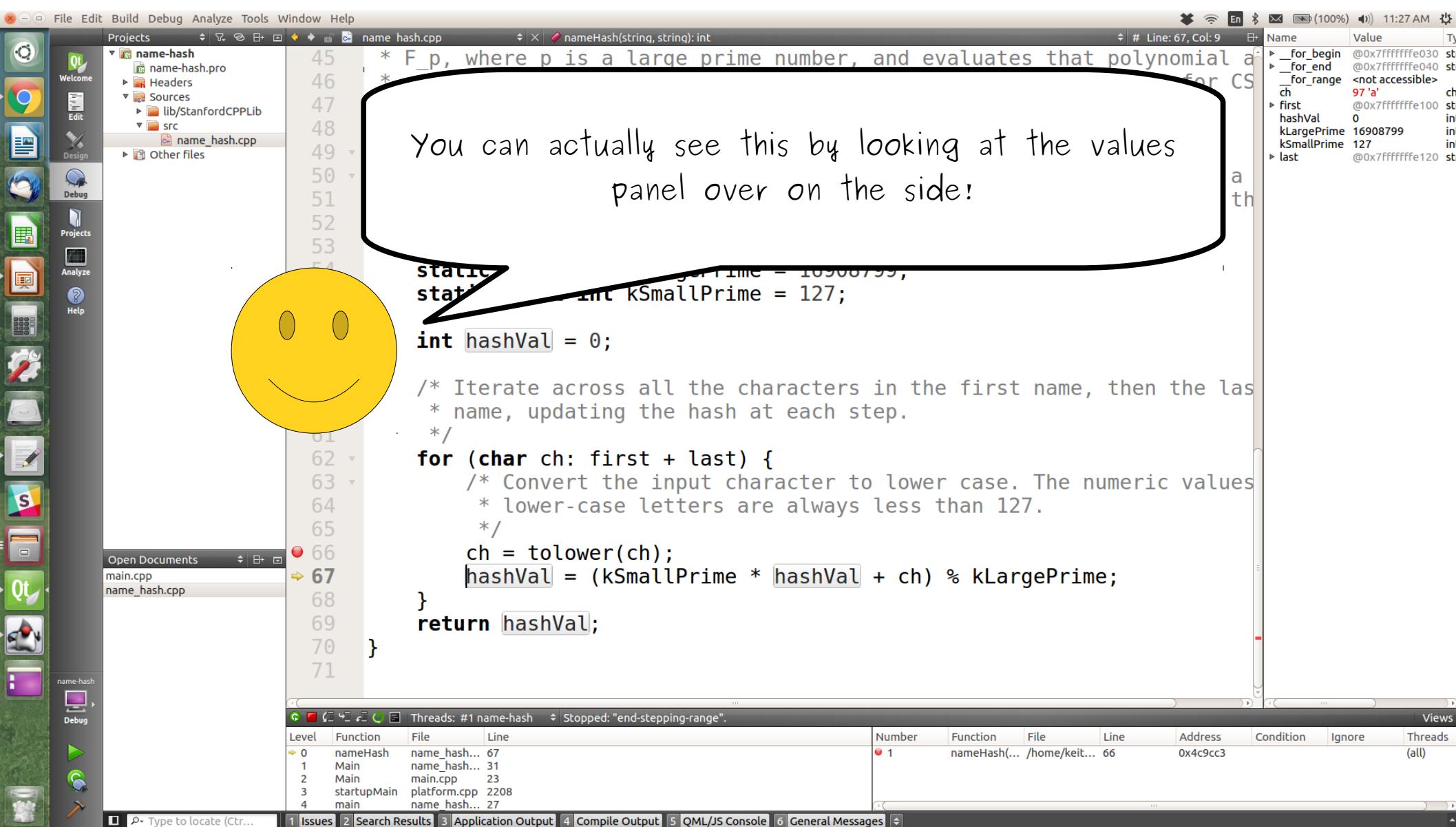
Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages

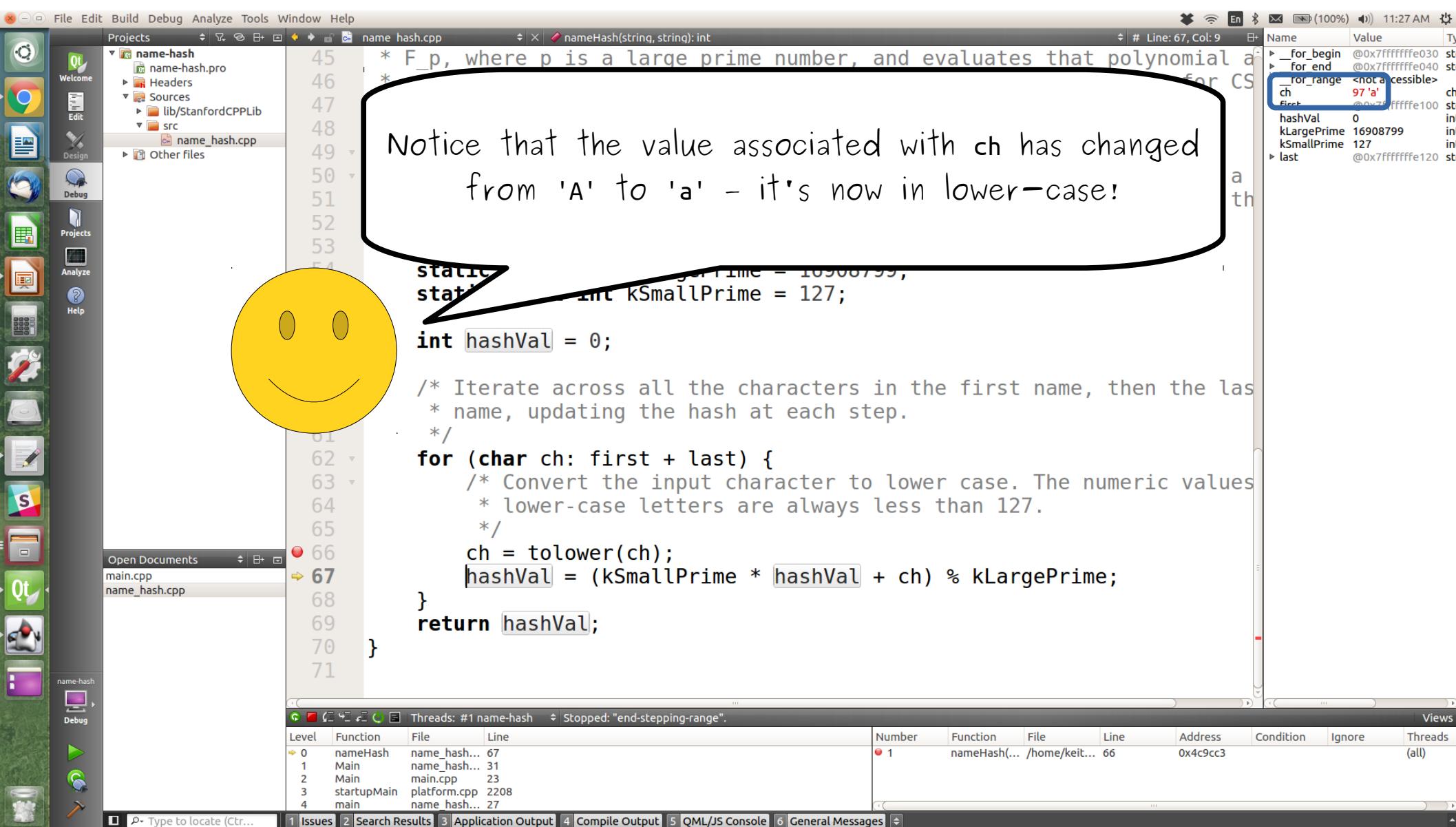
Name Value

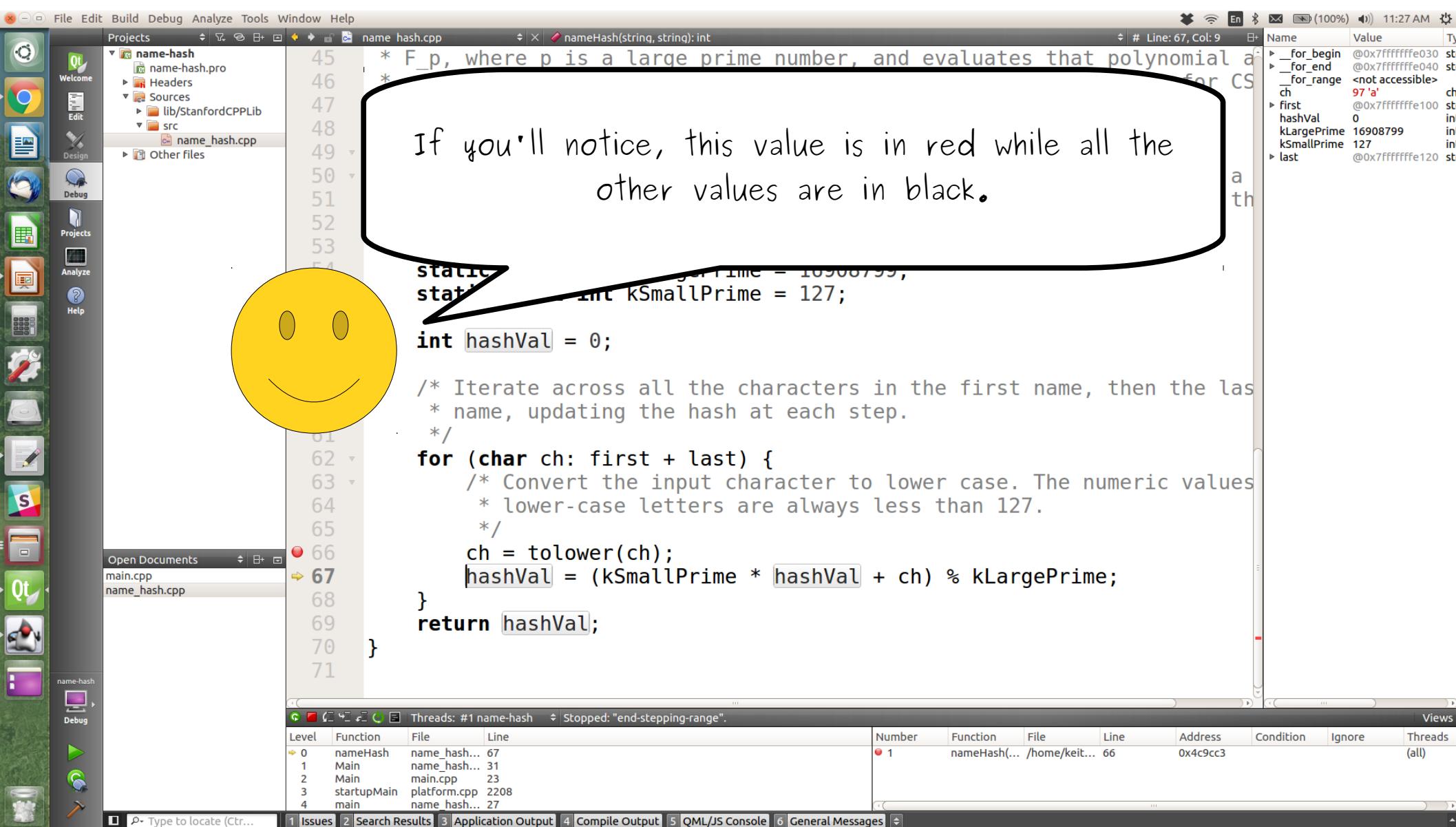
for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 97 'a'
first @0x7fffffff100
hashVal 0
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120

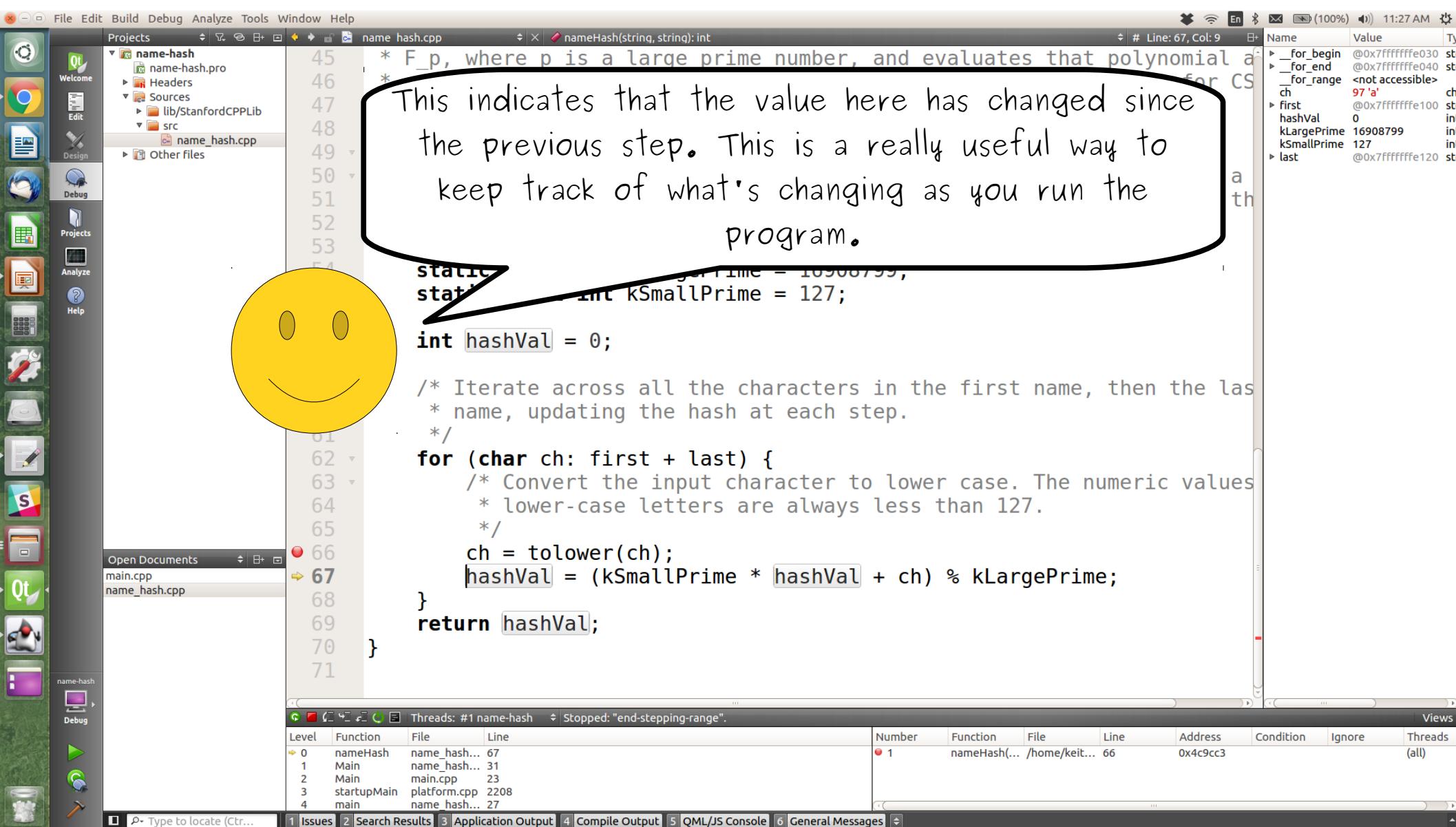


This line converts `ch` to lower case. The `tolower` function takes in a character and returns a lower-case version of it, so this overwrites `ch` with a lower-case version of itself.

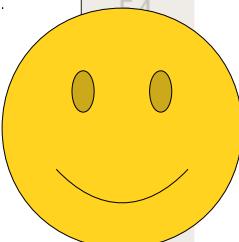








Now, let's take a look at line 67, where we are right now.



```
45 * F_p, where p is a large prime number, and evaluates that polynomial a
46 * for CS
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
```

static
static
int kLargePrime = 16908799,
int kSmallPrime = 127;

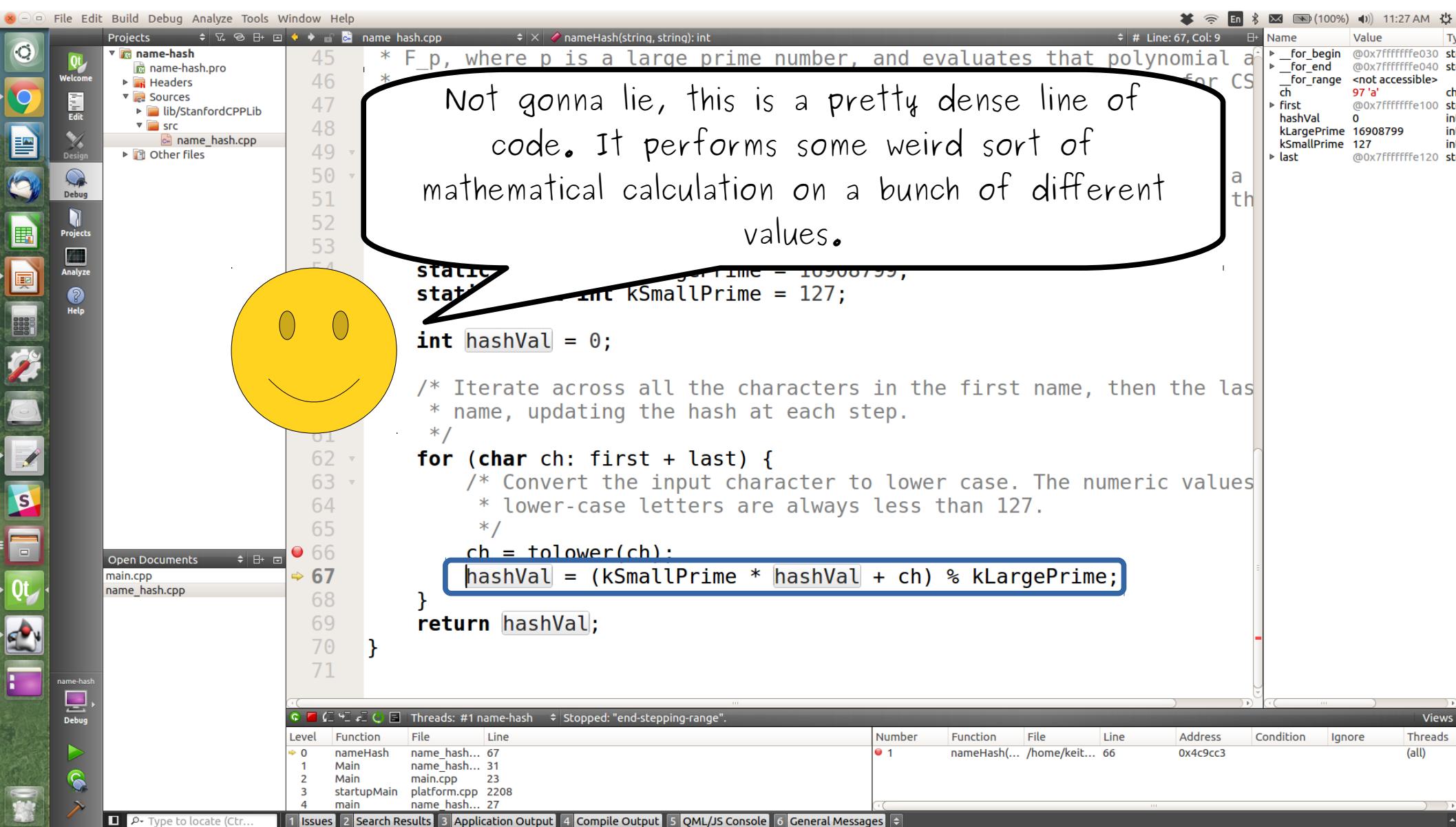
int hashVal = 0;

/* 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
 * lower-case letters are always less than 127.
 */
 ch = tolower(ch);
 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
}
return hashVal;

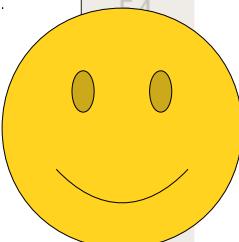
Threads: #1 name-hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	67	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...)



Fundamentally, though, it's just computing some weird function of some values and stashing it into `hashVal`.



```
45 * F_p, where p is a large prime number, and evaluates that polynomial a
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
```

Qt Projects

name hash

name hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name hash.cpp

Other files

File Edit Build Debug Analyze Tools Window Help

name hash(string, string): int

Line: 67, Col: 9

for CS

at

Value

Name

for_begin @0x7fffffff030

for_end @0x7fffffff040

for_range <not accessible>

ch 97 'a'

First @0x7fffffff100

hashVal 0

kLargePrime 16908799

kSmallPrime 127

last @0x7fffffff120

Projects

Analyze

Help

Qt

Open Documents

main.cpp

name hash.cpp

name hash

Debug

Threads: #1 name hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name hash...	67	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type

Views

Type to locate (Ctrl+F)

Issues

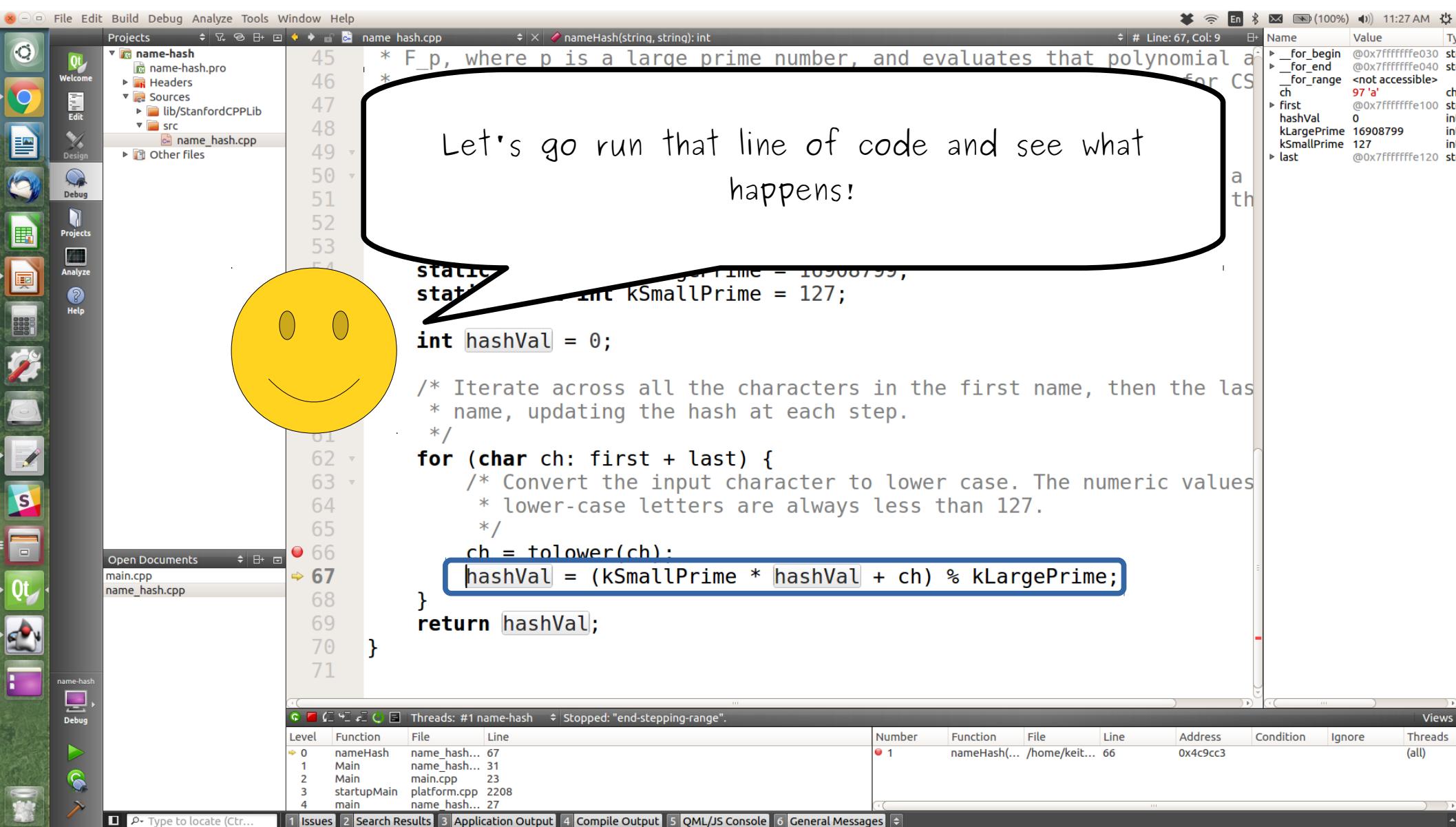
Search Results

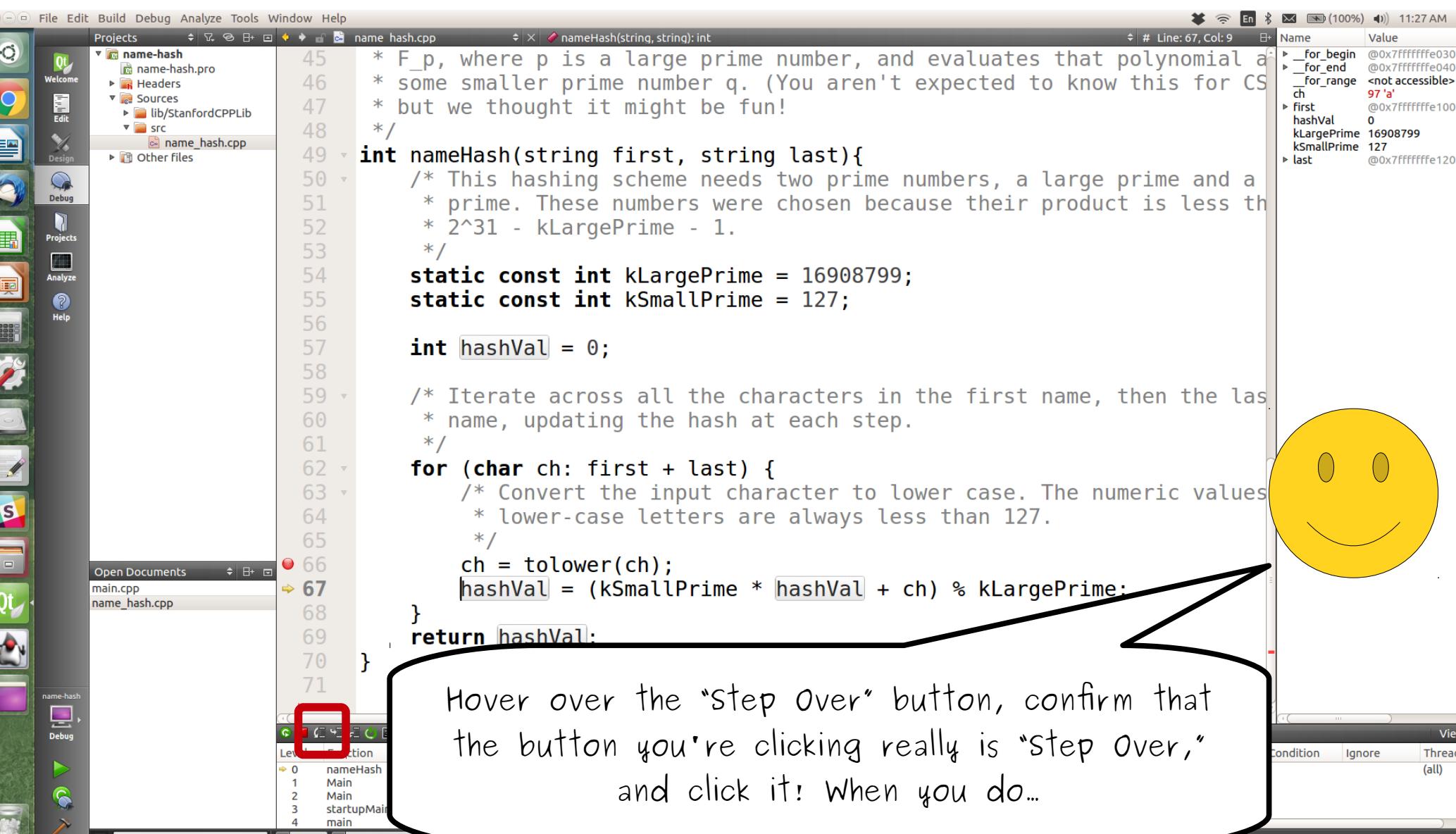
Application Output

Compile Output

QML/JS Console

General Messages





File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

name hash
name-hash.pro
Headers
Sources
lib/StanfordCPPLib
src
name_hash.cpp
Other files

45 * F_p , where p is a large prime number, and evaluates that polynomial at
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 than
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 last
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 }

66
67
68
69
70
71

Open Documents main.cpp name_hash.cpp

name hash
Debug

Step Over

0 nameHash
1 Main
2 Main
3 startupMain
4 main

Line: 67, Col: 9

Name Value Type

_for_begin @0x7fffffff030 std::string

_for_end @0x7fffffff040 std::string

_for_range <not accessible> std::string

ch 97 'a'

First @0x7fffffff100 std::string

hashVal 0 int

kLargePrime 16908799 int

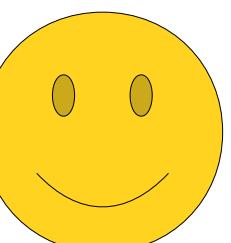
kSmallPrime 127 int

last @0x7fffffff120 std::string

Views Condition Ignore Threads (all)

1 Issues 2 Search Results

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.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
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 last
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 for
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 }

Open Documents main.cpp name_hash.cpp

name hash

Threads: #1 nar

Level Function File L

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

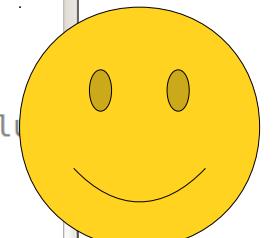
Issues Search Results Application Output Compile Output QML/JS Console General Messages

1 2 3 4 5 6

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 97 'a'
first @0x7fffffff100
hashVal 97
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120

... you'll end up with something like this!



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
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 last
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 for
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 }

Open Documents main.cpp name_hash.cpp

name hash

Threads: #1 nar

Level Function File L

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

Type to locate (Ctrl+F)

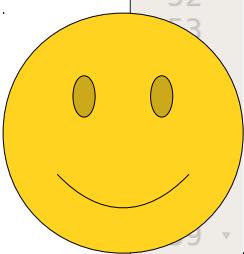
Issues Search Results Application Output Compile Output QML/JS Console General Messages

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 97 'a'
first @0x7fffffff100
hashVal 97
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120

Let's see what's changed.

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 last
 * 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;
```

Qt Creator IDE interface showing the code editor, project manager, and debugger. The debugger window shows a variable table with the following entries:

Name	Value
_for_begin	@0x7fffffff030
_for_end	@0x7fffffff040
_for_range	<not accessible>
ch	97 'a'
first	@0x7fffffff100
hashVal	97
kLargePrime	16908799
kSmallPrime	127
last	@0x7fffffff120

The hashVal variable is highlighted in red, indicating it has been modified. The debugger also shows a stack trace and a list of threads.

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * 2^31 - 1.
53 */
54 static const int kSmallPrime = 101;
55 static const int kLargePrime = 16908799;
56
57 int hashVal = kSmallPrime;
58
59 /* Iterate over each character in the string, updating the hash value.
60 * name, up to the end of the string.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

Open Documents main.cpp name_hash.cpp

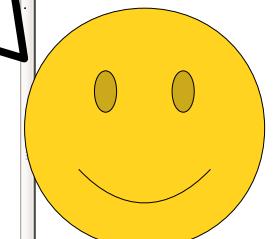
Threads: #1 name-hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	62	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

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.



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * 2^31 - kLargePrime - 1.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal;
58
59 /* Iterates over every character in the string. The character is converted
60 * to lowercase. The numeric value of the character is added to the hash
61 * value. The hash value is then converted back to a prime number.
62 */
63 for (char ch: first + last) {
64 /* Convert the input character to lower case. The numeric values of
65 * lower-case letters are always less than 127.
66 */
67 ch = tolower(ch);
68 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
69 }
70 return hashVal;
71 }

Qt Welcome Edit Design Debug Projects Analyze Help

Open Documents main.cpp name_hash.cpp

name-hash

Debug

Threads: #1 name-hash Stopped: "end-stepping-range".

Level Function 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

Number Function File Line Address Condition Ignore Threads

1 nameHash(... /home/keit... 66 0x4c9cc3 (all)

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

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



Name Value

for_begin @0x7fffffe030
for_end @0x7fffffe040
for_range <not accessible>
ch 97 'a'
first @0x7fffffe100
hashVal 97
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffe120

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * 2^31 - kLargePrime - 1.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal = 1;
58
59 /* Iterate over each character in the string.
60 * name, and multiply the current hash value by kSmallPrime
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

Line: 62, Col: 5

Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 97 'a'
first @0x7fffffff100
hashVal 97
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff120

Let's go do it again!

Threads: #1 name-hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	62	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...)

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.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * 2^31 - kLargePrime - 1.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal;
58 /* Iterates over each character in the string.
59 * name, adding its ASCII value to the hash value.
60 */
61
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

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

Threads: #1 name-hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	62	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Qt Welcome Edit Design Debug Projects Analyze Help

name hash

name-hash.pro Headers Sources lib/StanfordCPPLib src name_hash.cpp Other files

Open Documents main.cpp name_hash.cpp

name-hash

Debug

Qt

Help

Views

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

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 than
52 * $2^{31} - kLargePrime - 1$.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal;
58
59 /* Iterates over every character in the string 'name',
60 * name, and adds it to the hash value.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

66

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)

Name Value

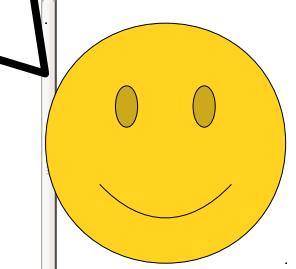
for_begin @0x7fffffe030
for_end @0x7fffffe040
for_range <not accessible>
ch 100 'd'
First @0x7fffffe100
hashVal 97
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffe120

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

Views

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 66, Col: 9

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * 2^31 - kLargePrime - 1.
53 */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal = 97;
58
59 /* Iterate over each character in the string.
60 * name, and add its ASCII value to the hash.
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

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

Threads: #1 name-hash Stopped at breakpoint 1 (1) in thread 1.

Views

Issues Search Results Application Output Compile Output QML/JS Console General Messages

name hash.cpp

name hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name_hash.cpp

Other files

Qt Welcome

Edit

Design

Debug

Projects

Analyze

Help

Qt

name hash

Debug

Open Documents

main.cpp name_hash.cpp

name hash

Debug

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)

Type to locate (Ctrl+F)

Issues Search Results Application Output Compile Output QML/JS Console General Messages

1 2 3 4 5 6

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

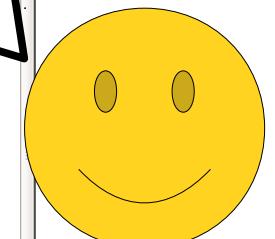
45 * F_p, where p is a large prime number, and evaluates that polynomial at
46 * some smaller prime number q. (You aren't expected to know this for CS
47 * but we thought it might be fun!
48 */
49 int nameHash(string first, string last){
50 /* This hashing scheme needs two prime numbers, a large prime and a
51 * prime. These numbers were chosen because their product is less than
52 * $2^{31} - kLargePrime - 1$.
53 */
54 static const int kSmallPrime = 109;
55 static const int kLargePrime = 16908799;
56
57 int hashVal = kSmallPrime;
58
59 /* Iterate over each character in the string.
60 * name, last
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

Threads: #1 name-hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	67	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages

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'.



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 67, Col: 9

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52 * $2^{31} - kLargePrime - 1$.
53 */
54 static const int kSmallPrime = 101;
55 static const int kLargePrime = 16908799;
56
57 int hashVal = kSmallPrime;
58
59 /* Iterate over each character in the string.
60 * name, last
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric values for
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 }

Now, click "Step Over" one more time.

Threads: #1 name-hash Stopped: "end-stepping-range".

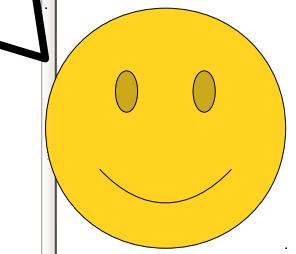
Level Function File Line

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

Number	Function	File	Line	Address	Condition	Ignore	Threads
1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)

Type to locate (Ctrl...) Issues Search Results Application Output Compile Output QML/JS Console General Messages



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

45 * F_p, where p is a large prime number, and evaluates that polynomial at
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
50
51
52
53
54
55
56
57
58
59
60
61 */
62 for (char ch: first + last) {
63 /* Convert the input character to lower case. The numeric
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 }

Open Documents main.cpp name_hash.cpp

Threads: #1 name-hash Finished retrieving data

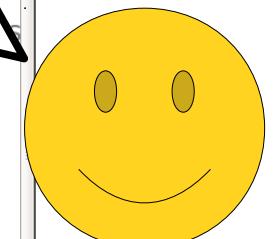
Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	62	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

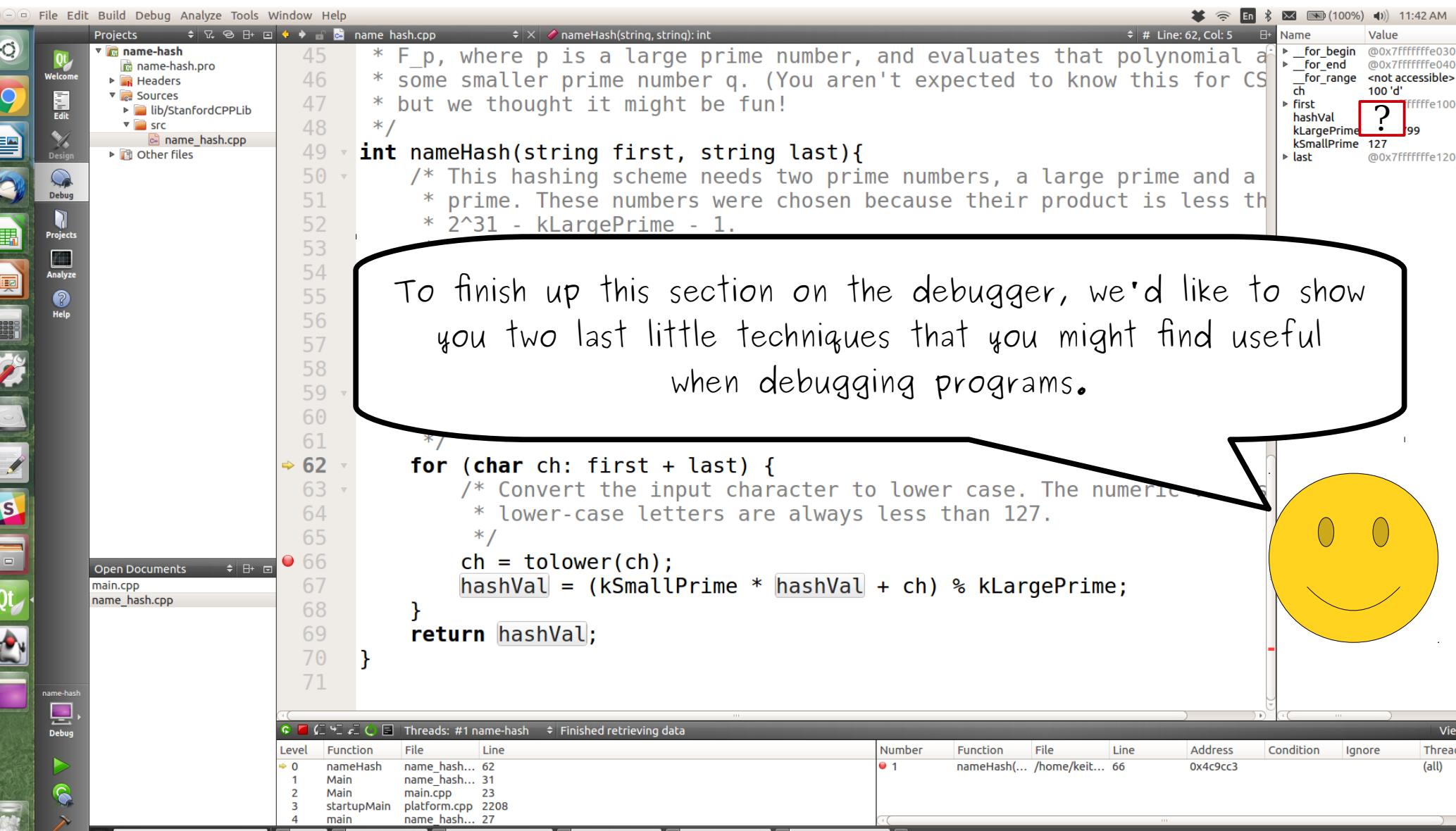
Type to locate (Ctrl...) 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

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.

```
45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
52     *  $2^{31} - kLargePrime - 1$ .
53
54
55
56
57
58
59
60
61 */
62 for (char ch: first + last) {
63     /* Convert the input character to lower case. The numeric values for
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 }
```

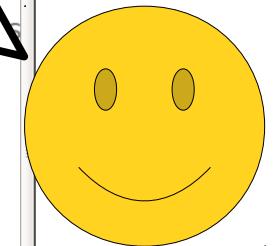
Qt Creator IDE interface elements:

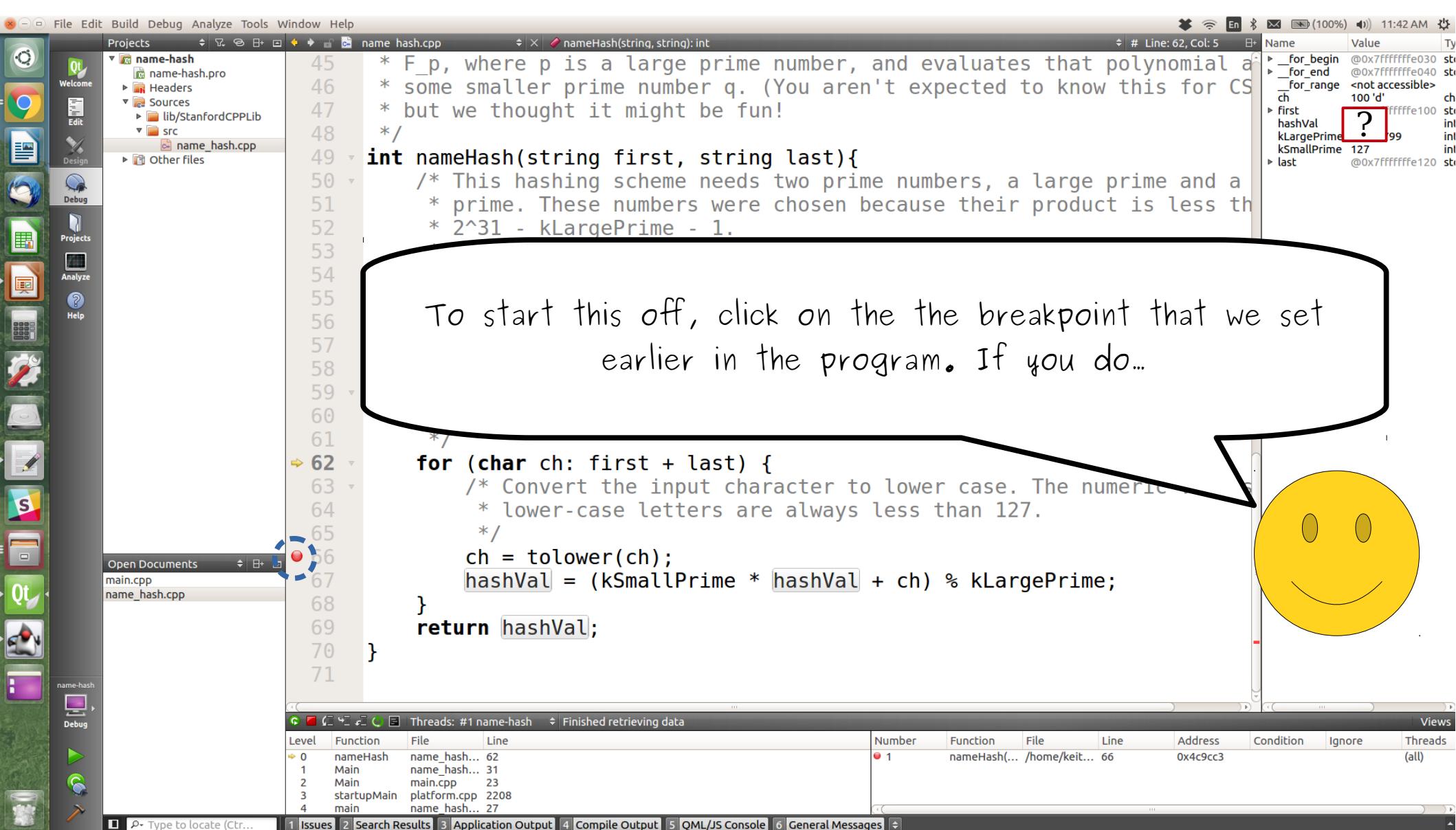
- File, Edit, Build, Debug, Analyze, Tools, Window, Help menu.
- Projects sidebar with icons for Welcome, Edit, Design, Debug, Projects, Analyze, and Help.
- Open Documents list showing main.cpp and name_hash.cpp.
- Code editor showing the nameHash function implementation.
- Breakpoints: A red circle is on line 66.
- Threads: #1 name-hash: Finished retrieving data.
- Call Stack: Shows the call stack with levels 0 to 4.
- Registers: Shows registers with values for _for_begin, _for_end, _for_range, first, hashVal, kLargePrime, kSmallPrime, and last.
- Registers table:

Name	Value
_for_begin	@0x7fffffff030
_for_end	@0x7fffffff040
_for_range	<not accessible>
ch	100 'd'
first	99
hashVal	fffffe100
kLargePrime	127
kSmallPrime	127
last	@0x7fffffff120

- Issues, Search Results, Application Output, Compile Output, QML/JS Console, General Messages tabs.
- Views: Shows the current view is Threads.

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...

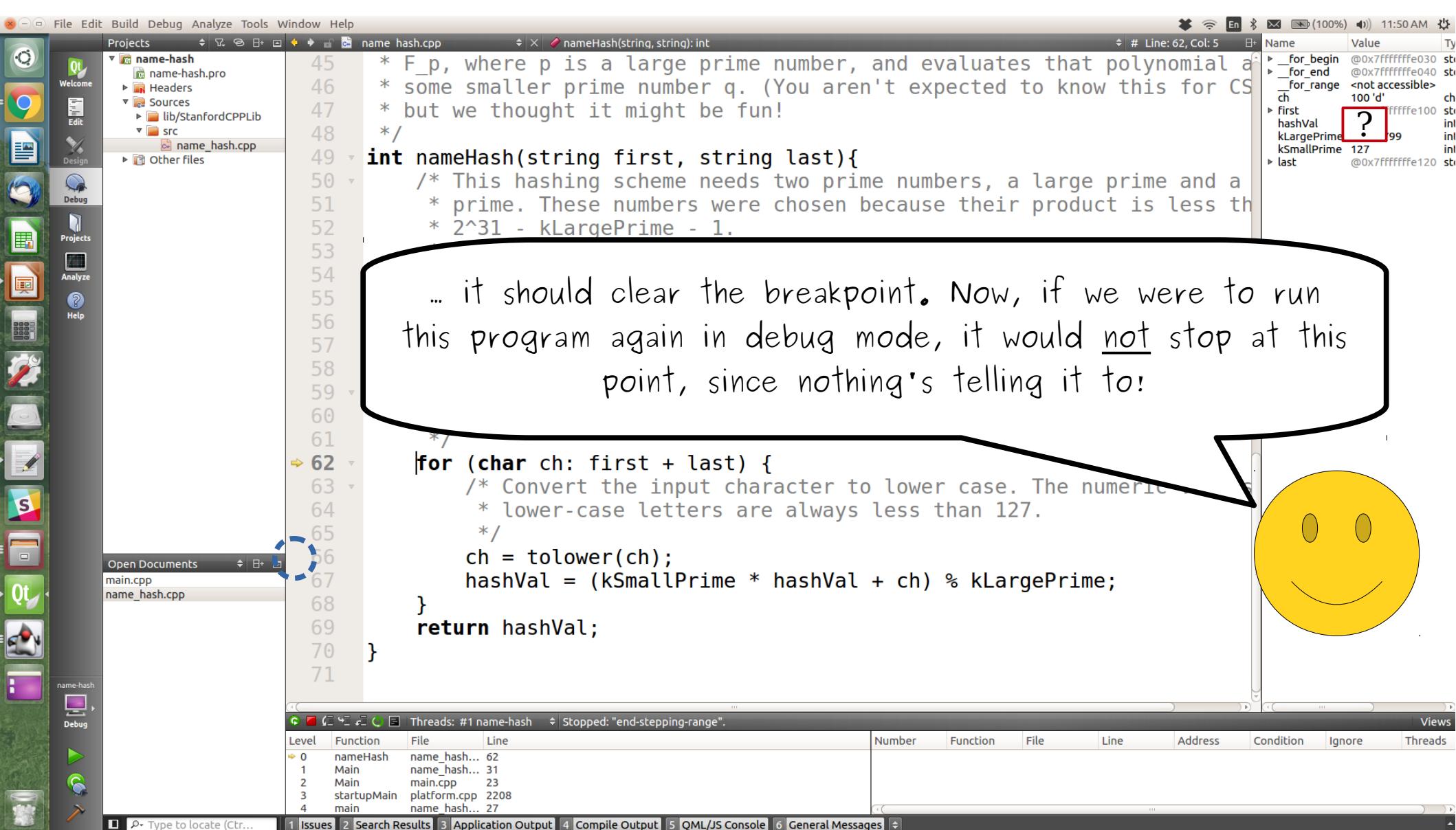
```
45 * F_p, where p is a large prime number, and evaluates that polynomial at
46 * some smaller prime number q. (You aren't expected to know this for CS
47 * but we thought it might be fun!
48 */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers, a large prime and a
51     * prime. These numbers were chosen because their product is less than
52     *  $2^{31} - kLargePrime - 1$ .
53
54
55
56
57
58
59
60
61
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The numeric values for
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 }
```

For_begin @0x7fffffff030
For_end @0x7fffffff040
For_range <not accessible>
ch 100 'd'
first hashVal ? 99
kLargePrime 127
kSmallPrime 127
last @0x7fffffff120

Threads: #1 name-hash Finished retrieving data

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	62	1	nameHash(...	/home/keit...	66	0x4c9cc3	(all)		
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name hash...	27								

Type to locate (Ctrl+F) 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

name hash.cpp nameHash(string, string): int

45 * F_p , where p is a large prime number, and evaluates that polynomial at
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 than
52 * $2^{31} - kLargePrime - 1$.
53 */
54 /*
55 * it should clear the breakpoint. Now, if we were to run
56 * this program again in debug mode, it would not stop at this
57 * point, since nothing's telling it to!
58 */
59 /*
60 * Convert the input character to lower case. The numeric values for
61 * lower-case letters are always less than 127.
62 */
63 **for** (**char** ch: first + last) {
64 ch = **tolower**(ch);
65 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
66 }
67 **return** hashVal;
68 }
69
70
71 }

Open Documents main.cpp name_hash.cpp

Threads: #1 name-hash Stopped: "end-stepping-range".

Level Function 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

Name Value

__for_begin @0x7fffffff030
__for_end @0x7fffffff040
__for_range <not accessible>
ch
first
hashVal
kLargePrime 100 'd'
kSmallPrime 127
last @0x7fffffff120

?

... 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!

0 0

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

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 than
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 last
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 for
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 }

Open Documents main.cpp name_hash.cpp

name-hash

Threads: #1 name-has

Level	Function	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

Type to locate (Ctrl...) Issues Search Results Application O

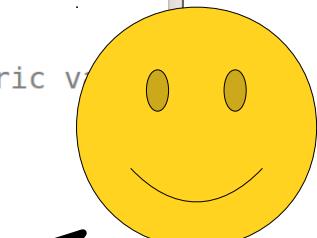
Views

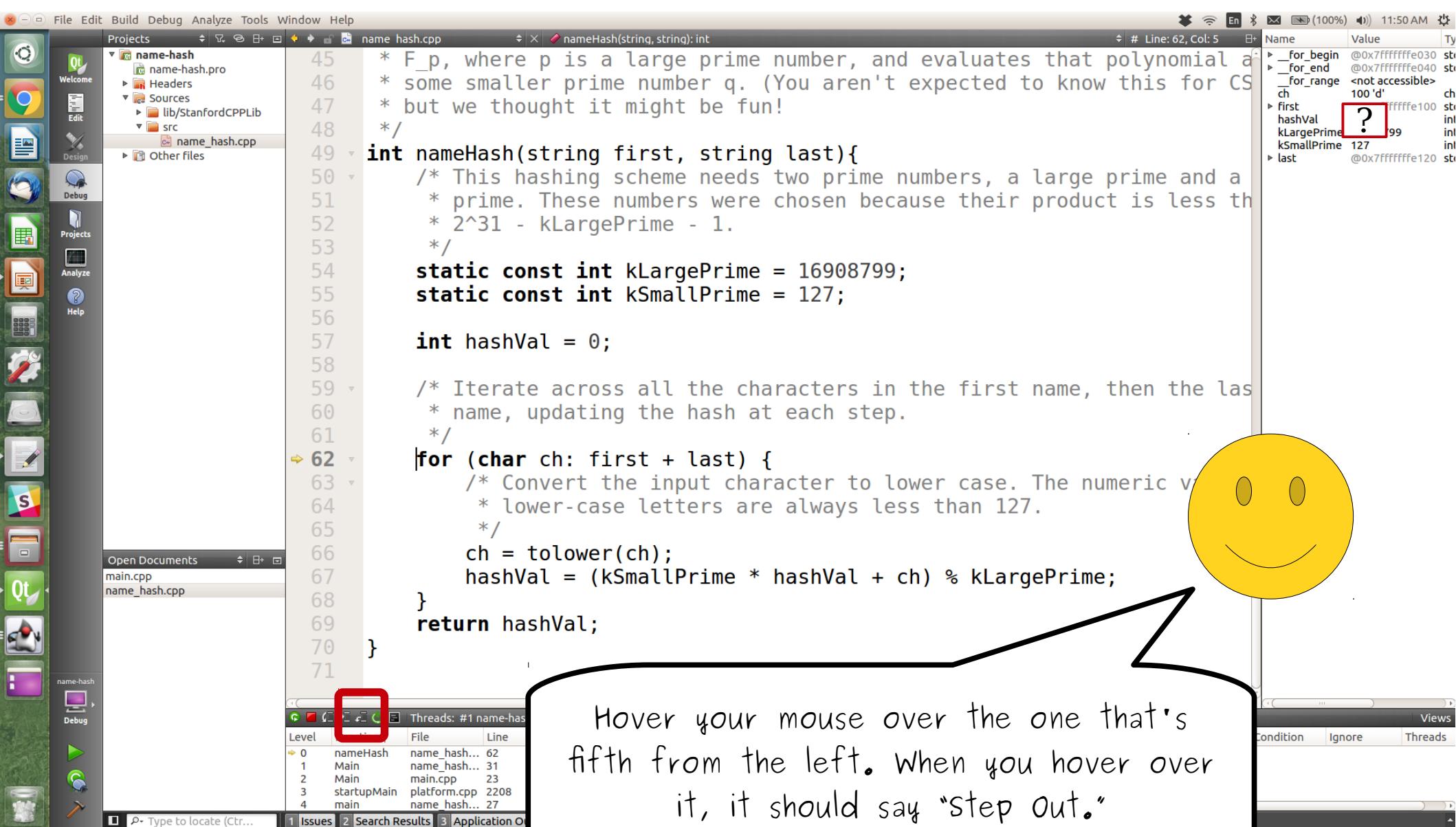
Name Value

for_begin @0x7fffffff030
for_end @0x7fffffff040
for_range <not accessible>
ch 100 'd'
first hashVal
kLargePrime 16908799
kSmallPrime 127
last @0x7fffffff0120

?

Now, take a look back at these buttons.





File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

Line: 62, Col: 5

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 than
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 last
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 for
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 }

Threads: #1 name-ha

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

Type to locate (Ctrl...)

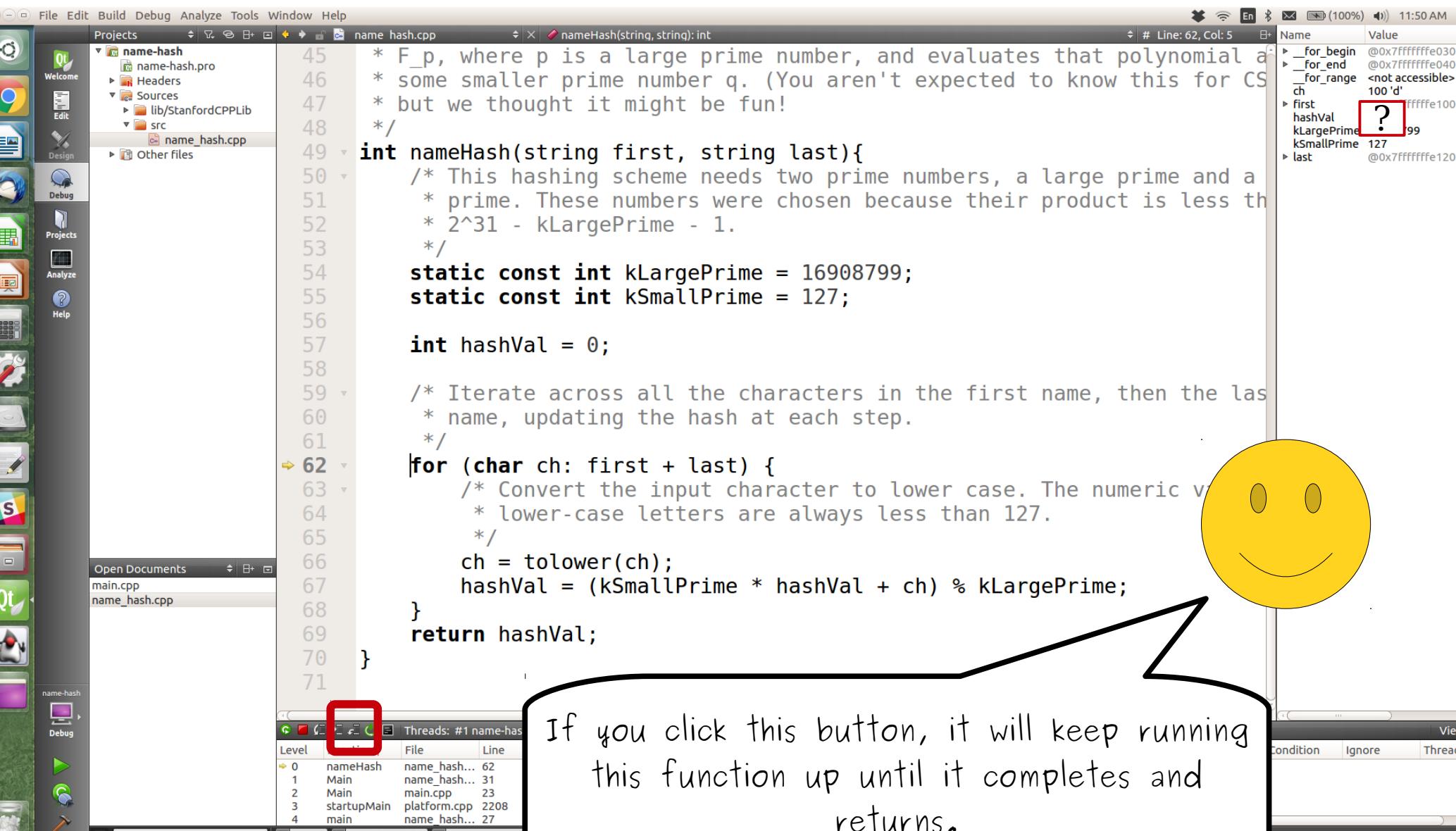
Issues Search Results Application O

Views

Condition Ignore Threads

?

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



If you click this button, it will keep running this function up until it completes and returns.

```
45 * F_p, where p is a large prime number, and evaluates that polynomial at
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 than
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 last
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 for
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 }
```

Threads: #1 name-has

Level	File	Line
0	nameHash	name_hash...
1	Main	name_hash...
2	Main	main.cpp
3	startupMain	platform.cpp
4	main	name_hash...

File Edit Build Debug Analyze Tools Window Help

Projects

name hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name_hash.cpp

Other files

name

Value

for_begin @0x7fffffff030

for_end @0x7fffffff040

for_range <not accessible>

ch 100 'd'

first hashVal ?

kLargePrime 99

kSmallPrime 127

last @0x7fffffff120

Open Documents

main.cpp

name_hash.cpp

Qt Welcome

Edit

Design

Debug

Projects

Analyze

Help

Type to locate (Ctrl+F)

Issues

Search Results

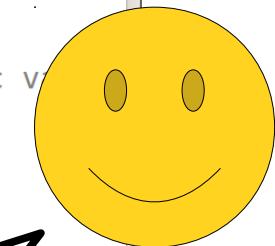
Application O

Views

Condition

Ignore

Threads



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.cpp nameHash(string, string): int

Line: 62, Col: 5

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 than
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 last
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 for
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 }

Threads: #1 name-hash

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

Type to locate (Ctrl...) Issues Search Results Application

Projects

name hash

- name-hash.pro
- Headers
- Sources
 - lib/StanfordCPPLib
 - src
 - name_hash.cpp
- Other files

Qt Welcome

Edit

Design

Debug

Projects

Analyze

Help

Qt

name hash

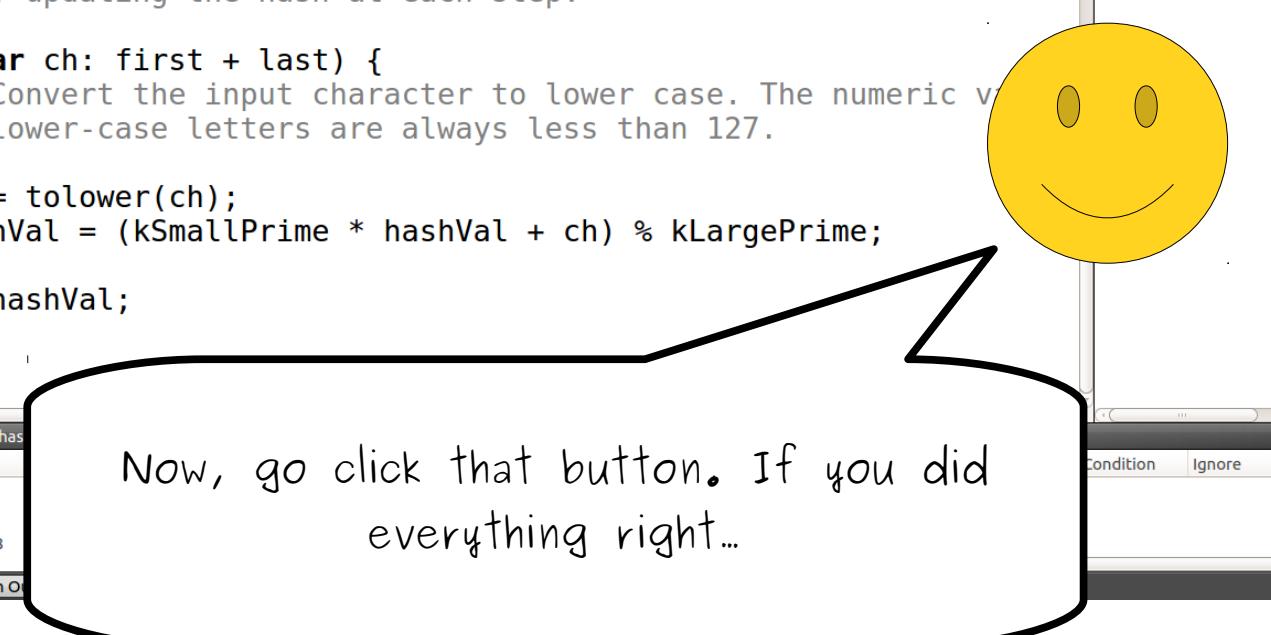
Debug

Threads: #1 name-hash

Name	Value
_for_begin	@0x7fffffff030
_for_end	@0x7fffffff040
_for_range	<not accessible>
ch	100 'd'
first	99
hashVal	fffffe100
kLargePrime	127
kSmallPrime	127
last	@0x7fffffff0120

?

Now, go click that button. If you did everything right...



File Edit Build Debug Analyze Tools Window Help

Projects 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;
}

/* This is the act
 * to talk mo
 * the meanti
 * of the inp
 *
 * For those
 * treats each
 * It then us
 * F n where n

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Threads: #1 name-hash Stopped: "function-finished".

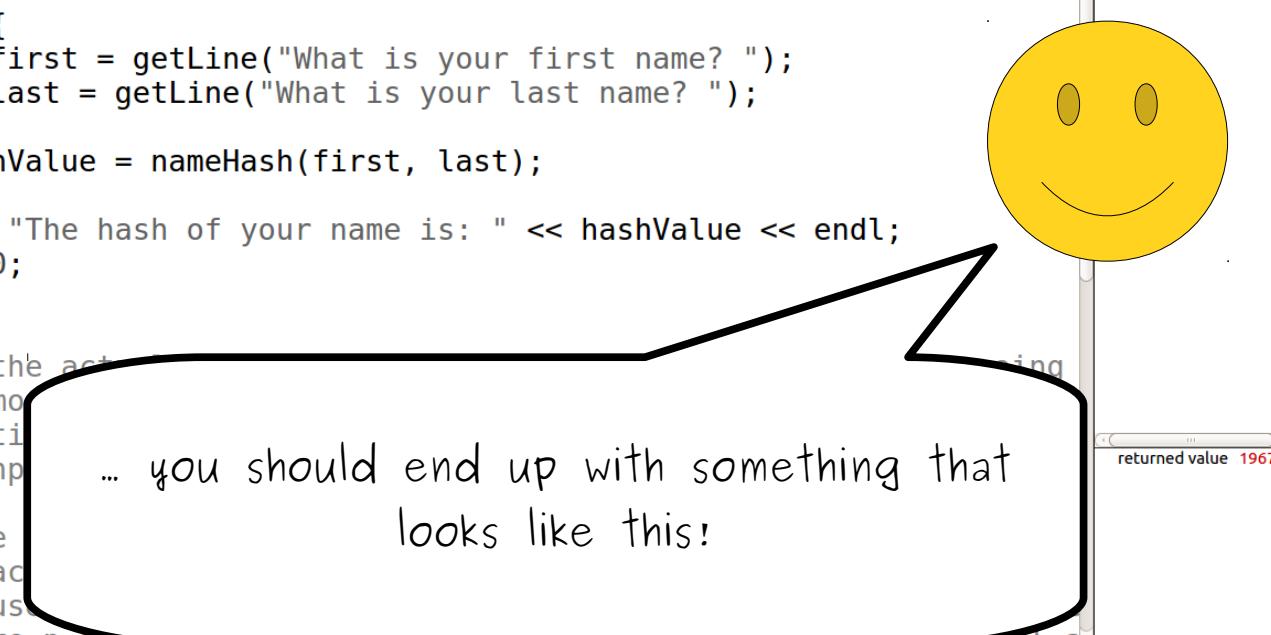
Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	31								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

11:50 AM

first @0x7fffffff0a0
last @0x7fffffff0c0
hashValue -590633613

returned value 1967457

... you should end up with something that looks like this!



File Edit Build Debug Analyze Tools Window Help

Projects 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;
}

/* This is the act
 * to talk mo
 * the meanti
 * of the inp
 *
 * For those
 * treats eac
 * It then us
 * F n where n

18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Views

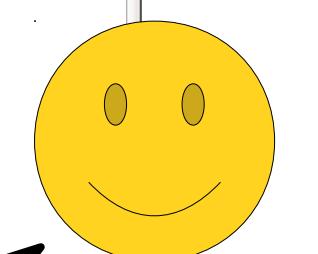
Threads: #1 name-hash Stopped: "function-finished".

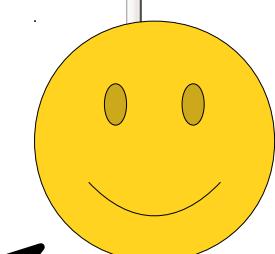
Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	31								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

first @0x7fffffff0a0
last @0x7fffffff0c0
hashValue -590633613

returned value 1967457

Let's take a minute to get our bearings.
Where exactly are we?





Well, the yellow arrow indicates that we're back in `main` again. Cool!

```
#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);

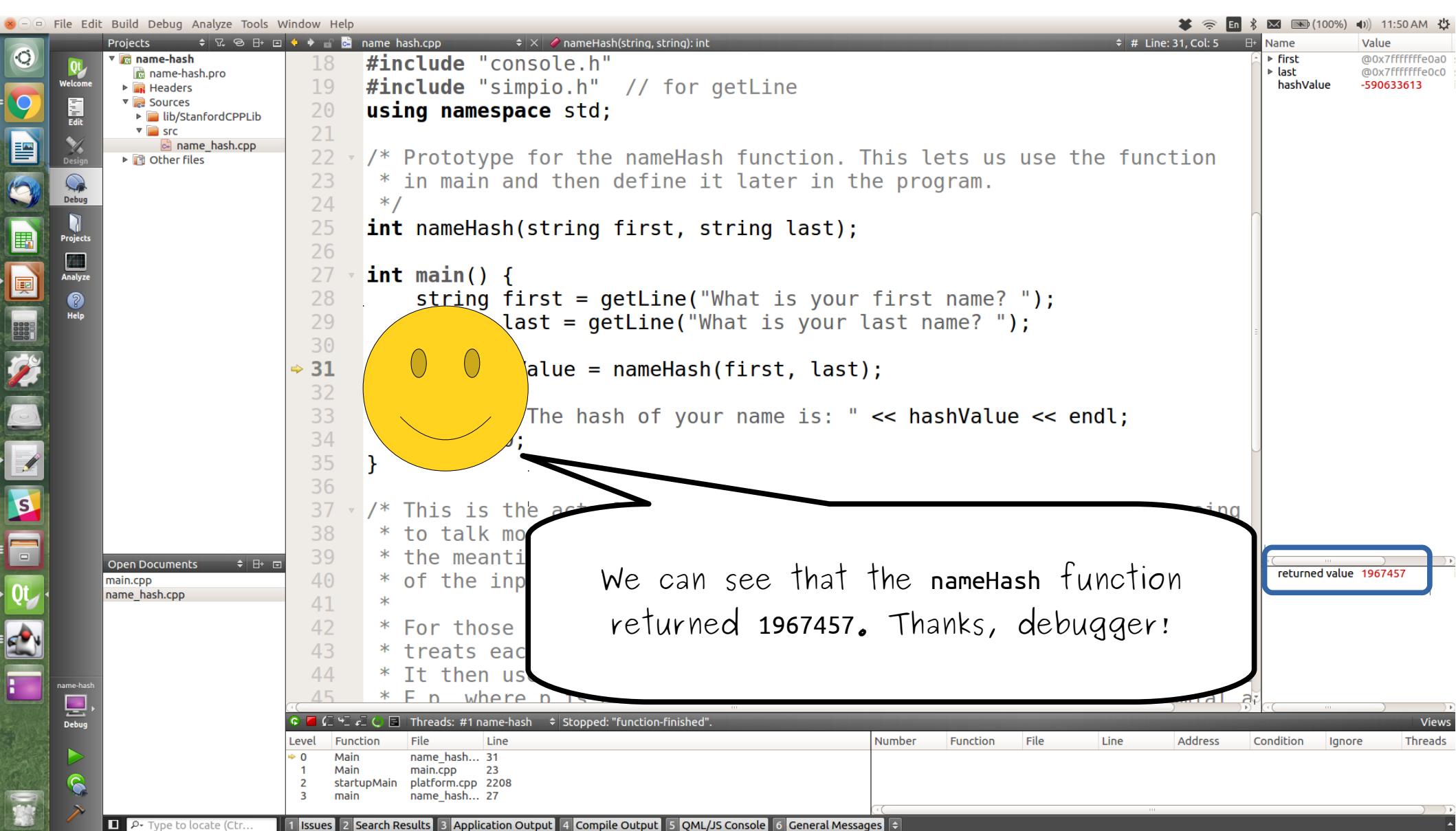
int main() {
    string first = getLine("What is your first name? ");
    string last = getLine("What is your last name? ");

    int hashValue = nameHash(first, last);

    cout << "The hash of your name is: " << hashValue << endl;
    return 0;
}

/* This is the act
 * to talk more
 * the meaning
 * of the input
 *
 * For those
 * treats each
 * It then uses
 * Finally where n
```

Level	Function	File	Line
0	Main	name_hash...	31
1	Main	main.cpp	23
2	startupMain	platform.cpp	2208
3	main	name_hash...	27



File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp nameHash(string, string): int

#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? ");
 value = nameHash(first, last);
 cout << "The hash of your name is: " << hashValue << endl;

/* This is the actual
 * to talk more
 * the meaning
 * of the input
 *
 * For those
 * treats each
 * It then uses
 * F n where n is

Threads: #1 name-hash Stopped: "function-finished".

Level Function File Line

0 Main name_hash... 31
1 Main main.cpp 23
2 startupMain platform.cpp 2208
3 main name_hash... 27

Name Value

first @0x7fffffff0a0
last @0x7fffffff0c0
hashValue -590633613

returned value 1967457

Type to locate (Ctrl+F)

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

nameHash(string, string): int

```
18 #include <console.h>
19 #include "simpio.h"
20 // Line
21 using namespace std;
22 /* Prototype for
23 * in main and the
24 */
25 int nameHash(string,
26
27 int main() {
28     string first;
29     string last = " ";
30
31     int hashValue = nameHash(first, last);
32
33     cout << "The hash of your name is: " << hashValue << endl;
34     return 0;
35
36 /* This is the actual function that computes the hash code. We're going
37 * to talk more about what hash functions do later in the quarter. In
38 * the meantime, think of it as a function that scrambles up the characters
39 * of the input and produces a number.
40 *
41 * For those of you who are more mathematically inclined, this function
42 * treats each character in the input name as a number between 0 and 128
43 * It then uses them as coefficients in a polynomial over the finite field
44 *  $F_n$  where  $n$  is a large prime number and evaluates that polynomial at
45 */
```

11:50 AM (100%)

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.)

Views

Threads: #1 name-hash Stopped: "function-finished".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	31								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

Type to locate (Ctrl+F)

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

nameHash(string, string): int

```
18 #include <conio.h>
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for
23  * in main and the
24  */
25 int nameHash(string,
26
27 int main() {
28     string first;
29     string last = " ";
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 characters
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 field
45 *  $F_n$  where  $n$  is a large prime number and evaluates that polynomial at
```

Line: 31, Col: 5

Name	Value
first	0x7fffffff0a0
last	0x7fffffff0c0
hashValue	-590633613

returned value 1967457

Threads: #1 name-hash Stopped: "function-finished".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	31								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

Type to locate (Ctrl+F)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages



This is pretty cool, actually!

```
18 #include "simpio.h" // for getLine
19 #include "console.h" // for cout
20 using namespace std;
21
22 /* Prototype for
23  * in main and the
24  */
25 int nameHash(string, string);
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 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 characters
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 field
45 *  $F_n$  where  $n$  is a large prime number and evaluates that polynomial at
```

This is pretty cool, actually!

returned value 1967457

Level	Function	File	Line
0	Main	name_hash...	31
1	Main	main.cpp	23
2	startupMain	platform.cpp	2208
3	main	name_hash...	27



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!

```
File Edit Build Debug Analyze Tools Window Help
Projects name-hash nameHash(string, string): int
18 #include <conio.h>
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for
23  * in main and the
24  */
25 int nameHash(string, string);
26
27 int main() {
28     string first;
29     string last = " ";
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 characters
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 field
45 *  $F_n$  where  $n$  is a large prime number and evaluates that polynomial at
46 *  $x = 256$ . The result is the hash value.
47 */
48
49 int nameHash(string first, string last) {
50     int hashValue = 0;
51
52     for (int i = 0; i < first.length(); i++) {
53         hashValue = hashValue * 256 + first[i];
54     }
55
56     for (int i = 0; i < last.length(); i++) {
57         hashValue = hashValue * 256 + last[i];
58     }
59
60     return hashValue;
61 }
```

Views

Threads: #1 name-hash Stopped: "function-finished".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	31								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

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

nameHash(string, string): int

```
18 #include <conio.h>
19 #include "simpio.h" // for getLine
20 using namespace std;
21
22 /* Prototype for nameHash()
23  * in main and in nameHash()
24  */
25 int nameHash(string, string);
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 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 characters
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 field
45 *  $F_n$  where  $n$  is a large prime number and evaluates that polynomial at
46 *  $x = 256$ .
```

Threads: #1 name-hash Stopped: "function-finished".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	31								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

Views

Type to locate (Ctrl+F)

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

```
20 using namespace std;
21
22 /* Prototype for the nameHash function
23 * in main and the nameHash function
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
35     return 0;
36 }
37
38 /* This is the actual function that computes the hash code. We're going
39 * to talk more about what hash functions do later in the quarter. In
40 * the meantime, think of it as a function that scrambles up the characters
41 * of the input and produces a number.
42 *
43 * For those of you who are more mathematically inclined, this function
44 * treats each character in the input name as a number between 0 and 128
45 * It then uses them as coefficients in a polynomial over the finite field
46 *  $F_p$ , where  $p$  is a large prime number, and evaluates that polynomial at
47 * some smaller prime number  $q$ . (You aren't expected to know this for CS
48 * but we thought it might be fun!)
```

Line: 33, Col: 5

Name	Value	Type
first	@0x7fffffff0a0	std::string
last	@0x7fffffff0c0	std::string
hashValue	1967457	int

Threads: #1 name-hash Stopped: "end-stepping-range".

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	33								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

Type to locate (Ctrl+F)

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

```
20 using namespace std;
21
22 /* Prototype for the nameHash function
23 * in main and the nameHash function
24 */
25 int nameHash(string first, string last);
26
27 int main() {
28     string first;
29     string last;
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 characters
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 field
45 *  $F_p$ , where  $p$  is a large prime number, and evaluates that polynomial at
46 * some smaller prime number  $q$ . (You aren't expected to know this for CS
47 * but we thought it might be fun!)
```

Threads: #1 name-hash Stopped: "end-stepping-range".

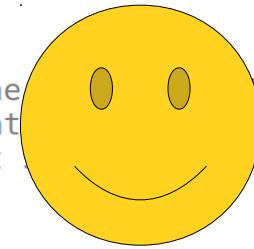
Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	Main	name_hash...	33								
1	Main	main.cpp	23								
2	startupMain	platform.cpp	2208								
3	main	name_hash...	27								

Type to locate (Ctrl+F)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

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.

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."



```
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;
}

/* This is the actual function that computes the hash. We're going to talk more about what hash functions do later. In the meantime, think of it as a function that takes two strings of the input and produces a number.
 */


```

Views

Function File Line Address Condition Ignore Threads

File Edit Build Debug Analyze Tools Window Help

Projects name hash.cpp # Line: 33, Col: 5

name hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name_hash.cpp

Other files

Qt Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name_hash.cpp

name hash

Debug

Type to locate (Ctrl...)

Issues

Search Results

Application Output

Compile Output

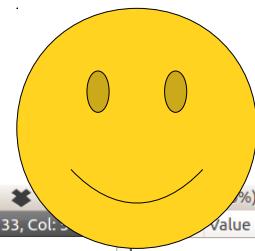
QML/JS Console

General Messages

first @0xfffffffffe0a0 std::int32_t

last @0x7fffffff0e0c0 std::int32_t

hashValue 1967457 int



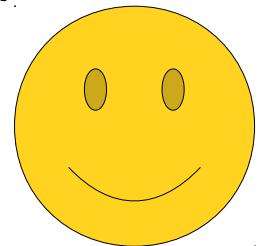
If you do, you should see something like this.
(The program window might not automatically
pop up. That's okay! Just open it manually.)

Our program is now done running!

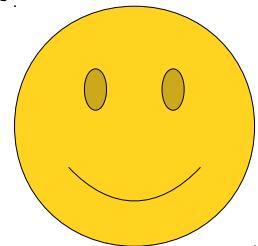
```
int main() {
    std::cout << "What is your first name? Ada";
    std::cout << "What is your last name? Lovelace";
    std::cout << "The hash of your name is: 1967457";
    std::cout << endl;
}
```

The screenshot shows the Qt Creator IDE interface. On the left is the project tree for 'name-hash' with files 'name-hash.pro', 'Headers', 'Sources' (containing 'lib/StanfordCPPLib' and 'src'), and 'name_hash.cpp'. The 'src' folder is selected. The main window shows the code for 'main.cpp' and 'name_hash.cpp'. A terminal window is open, displaying the program's output: 'What is your first name? Ada', 'What is your last name? Lovelace', and 'The hash of your name is: 1967457'. Below the terminal, the status bar says 'Debugger finished.' The bottom navigation bar has tabs for 'Issues', 'Search Results', 'Application Output', 'Compile Output', 'QML/JS Console', and 'General Messages'.

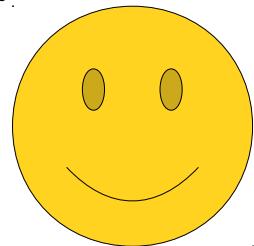
So there you have it! You've now gotten more
familiar with the debugger!



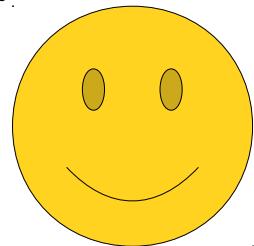
You know how to set a breakpoint to pause the program at a particular point.



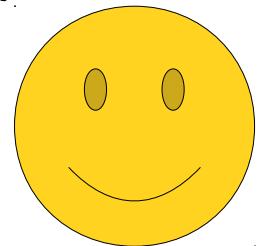
You know how to read the call stack and to see the values of local variables.



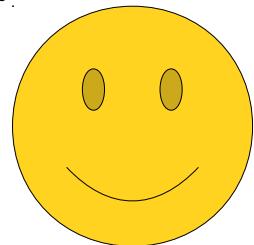
You know how to single-step the program and
see what values change.



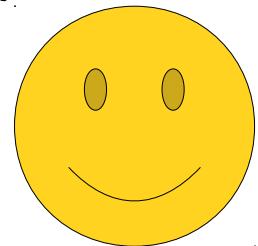
You know how to run a function to completion,
and how to let the program keep on running.



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!

