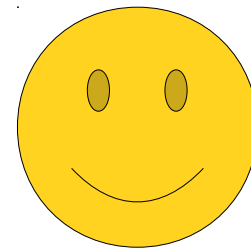


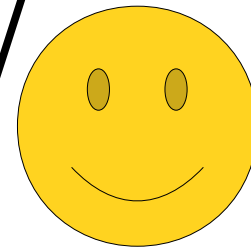
# Assignment 0: Using the Debugger

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

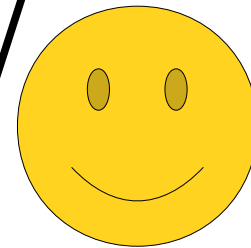
Hi everybody!



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

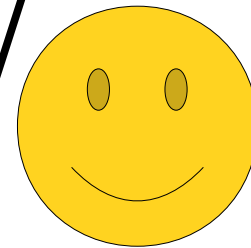


The debugger is a tool you can use to help see what your program is doing as you run it.

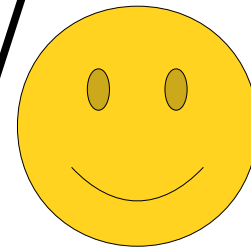




It's really useful for helping find errors in your programs, and the more practice you get with it, the easier it'll be to correct mistakes in the programs you write.



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 the
43 * treats
44 * It then
45 * F_p, where
46 * some smaller prime numbers (1000000007, 1000000009,
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 close to
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> # Line: 1, Col: 1

name-hash  
name-hash.pro  
Headers  
Sources  
lib/StanfordCPLib  
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  
this for CS106B,

time and a small  
is less than

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


Now, click the mouse!

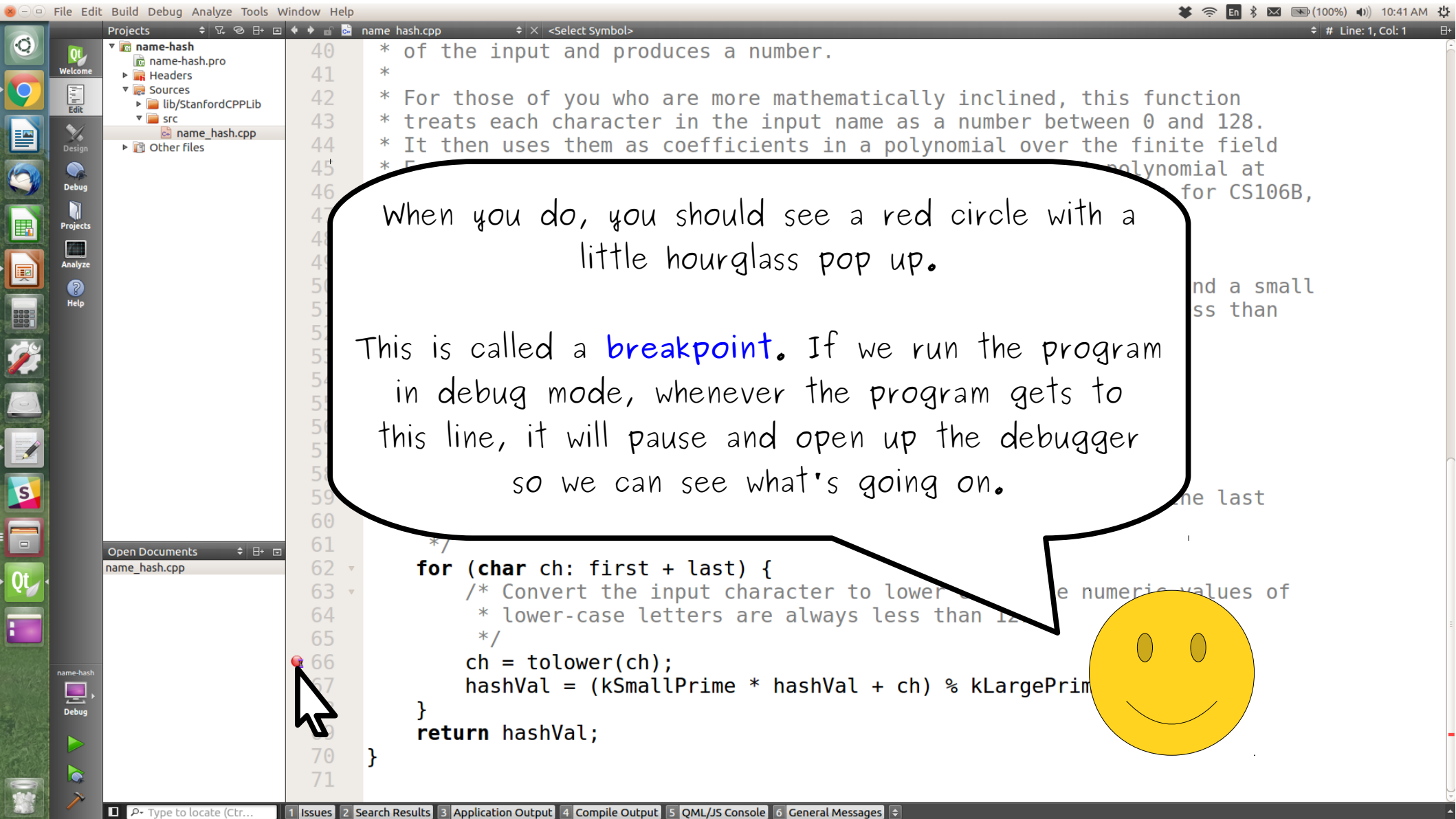
56  
57  
58  
59 /\* Iterate across all the characters in the first name to the last  
60 \* name, updating the hash at each step.  
61 \*/  
62 for (char ch: first + last) {  
63 /\* Convert the input character to lower case. values of  
64 \* lower-case letters are always less than 127.  
65 \*/  
66 ch = tolower(ch);  
67 hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;  
68 }  
69 return hashVal;  
70 }  
71

Open Documents  
name\_hash.cpp

name-hash  
Debug

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 <Select Symbol> # Line: 1, Col: 1

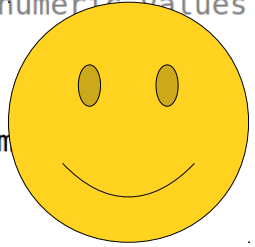
name-hash  
name-hash.pro  
Headers  
Sources  
lib/StanfordCPPLib  
src  
name\_hash.cpp  
Other files

Open Documents  
name\_hash.cpp

name-hash  
Debug

```
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     * prime less than  
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 values of  
64         * lower-case letters are always less than 128.  
65         */  
66         ch = tolower(ch);  
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
68     }  
69     return hashVal;  
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...



... 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.

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

```
41 // ...  
42 // ...  
43 /*  
44 * treats each character in the input name as a coefficient in a polynomial  
45 * It then uses them as coefficients in a polynomial of degree k-1 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!)  
49 */  
50 int nameHash(const std::string& name) {  
51     /*  
52     * ...  
53     * ...  
54     * ...  
55     * ...  
56     * ...  
57     * ...  
58     * ...  
59     * ...  
60     * ...  
61     * ...  
62     * ...  
63     * ...  
64     * ...  
65     * ...  
66     * ...  
67     */  
68     ch = tolower(ch);  
69     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
70 }
```

A console window is open, displaying the prompt "What is your first name? |".

The debugger window is open, showing the "Threads" tab with the following data:

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

The status bar at the bottom shows the following tabs: 1 Issues, 2 Search Results, 3 Application Output, 4 Compile Output, 5 QML/JS Console, 6 General Messages.





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

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

```
41 *
42 *
43 * treats each character in the input name as a coefficient in a polynomial
44 * It then uses them as coefficients in a polynomial of degree k-1 over the finite field
45 * 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
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 std::string& name) {
50     /*
51     *
52     *
53     *
54     *
55     *
56     *
57     *
58     *
59     */
60     /*
61     *
62     *
63     *
64     *
65     */
66     ch = tolower(ch);
67     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
```

A console window is open in the foreground, displaying the prompts:

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

A yellow smiley face is drawn next to the console window. The bottom status bar shows the 'Threads' panel with the following data:

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

The bottom status bar also includes a 'Build' button and a 'Type to locate (Ctrl...)' search bar.



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



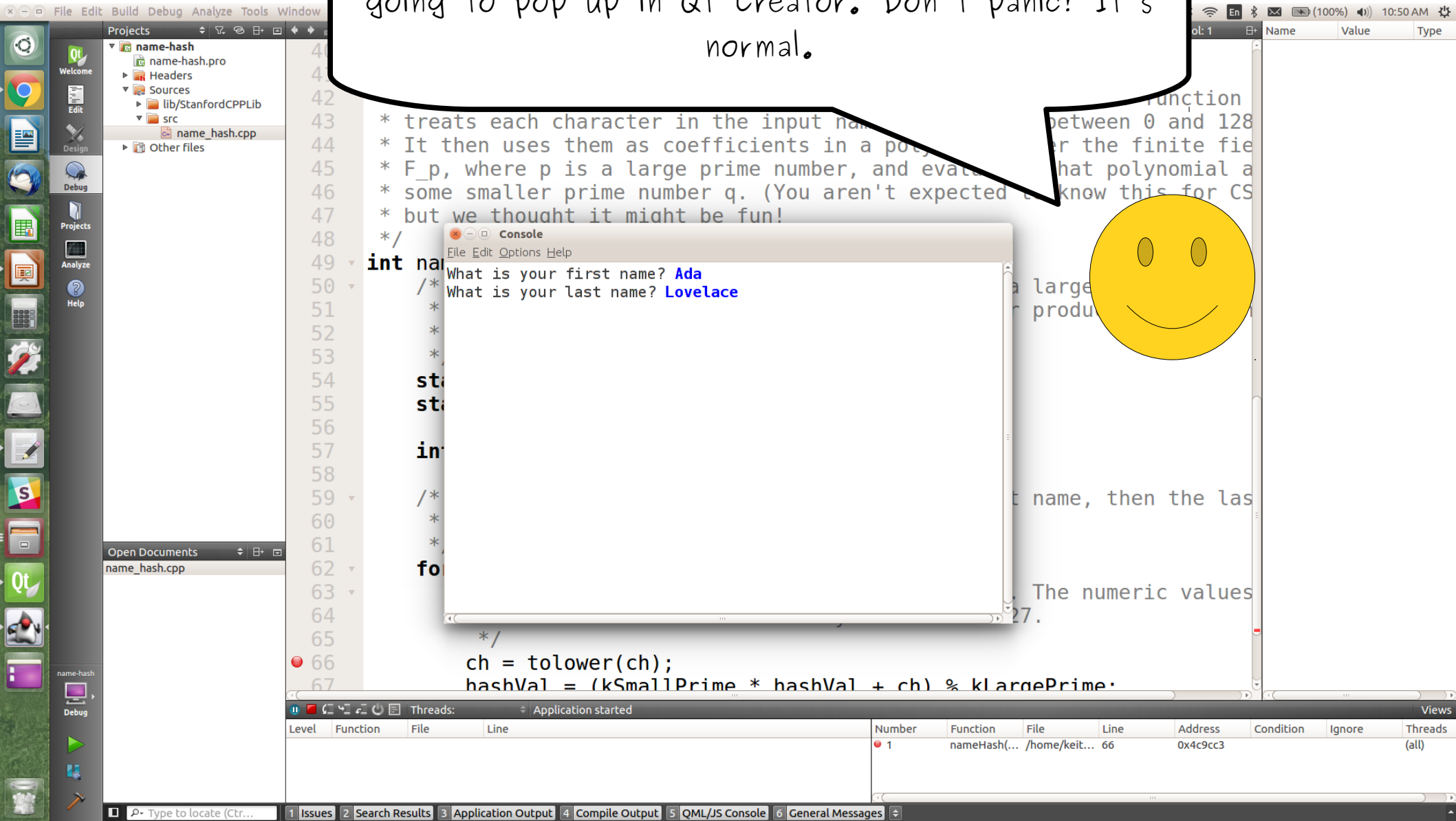
Qt Creator IDE interface showing a C++ project named "name-hash". The project structure includes "name-hash.pro", "Headers", "Sources", "lib/StanfordCPPLib", "src", and "name\_hash.cpp". The "name\_hash.cpp" file is open, displaying C++ code for a name hashing function. The code includes comments explaining the algorithm and the function signature: `int nameHash(const string& name)`. The function body starts with `int hashVal = 0;` and includes a loop to process each character in the input name. The code is partially visible, showing lines 41 through 67.

A console window is open, displaying the program's output: "What is your first name? Ada" and "What is your last name? Lovelace". The user has entered "Ada" for the first name and "Lovelace" for the last name, but has not yet pressed the Enter key.

The Qt Creator interface also shows a "Threads" panel at the bottom, indicating that the application has started. The "Threads" panel lists the function `nameHash(...)` at line 66, with address `0x4c9cc3`.

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

```
41 //
42 //
43 * treats each character in the input name as a coefficient in a polynomial
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!)
48 */
49 int nameHash(const std::string& name) {
50     /*
51     *
52     *
53     *
54     *
55     *
56     *
57     *
58     */
59     /*
60     *
61     *
62     *
63     */
64     for (int i = 0; i < name.length(); i++) {
65         /*
66         *
67         *
68         *
69         */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69 }
```

A console window is open in the foreground, displaying the following prompts and user input:

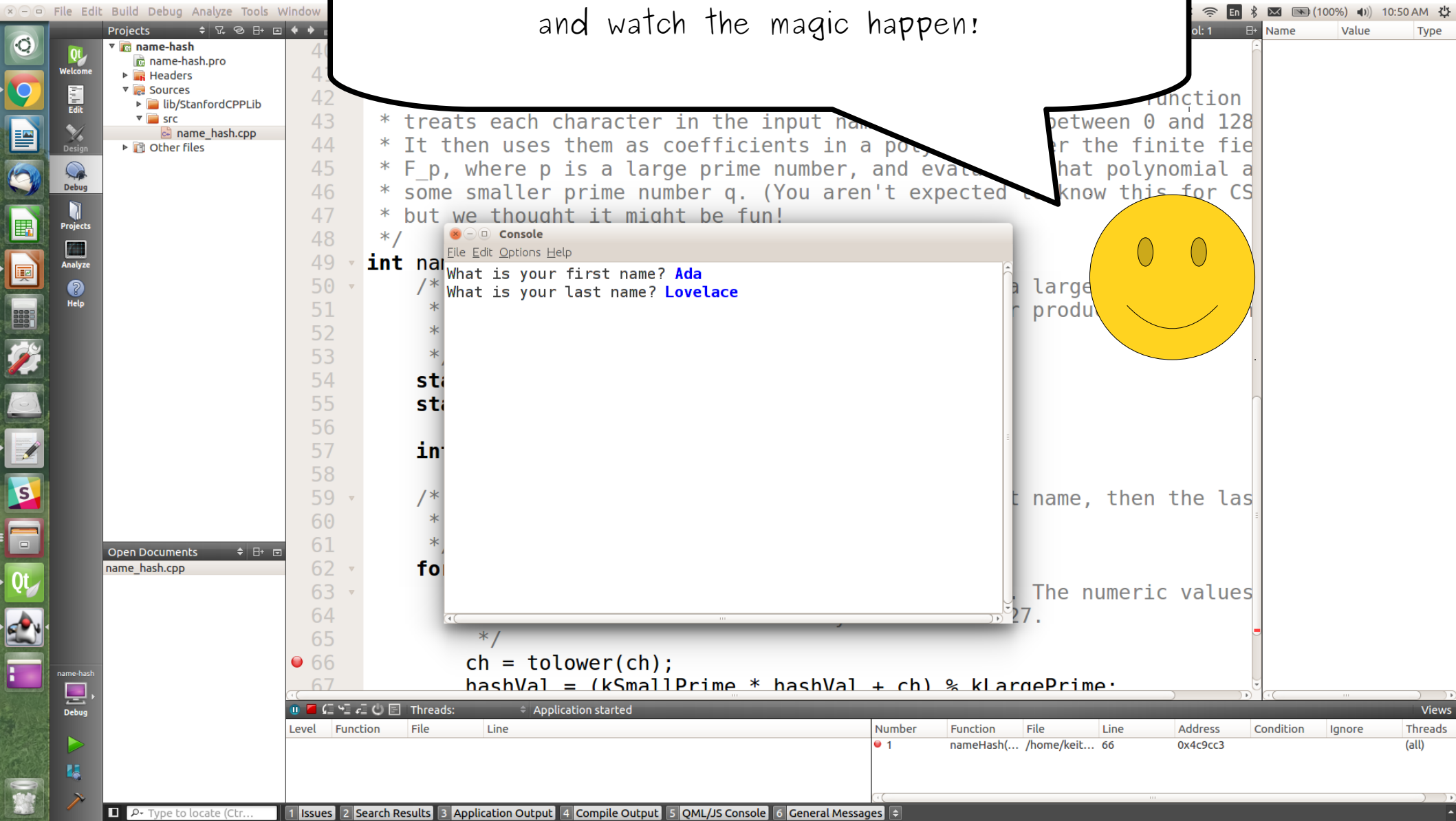
```
What is your first name? Ada
What is your last name? Lovelace
```

A yellow smiley face is drawn next to the console window.

The bottom status bar shows the following tabs: 1 Issues, 2 Search Results, 3 Application Output, 4 Compile Output, 5 QML/JS Console, 6 General Messages.

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

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



The screenshot shows the Qt Creator IDE with a project named 'name-hash'. The source file 'name\_hash.cpp' is open, showing a C++ program that calculates a hash for a name. The program prompts the user for their first and last names. A console window is open, showing the user input 'Ada' for the first name and 'Lovelace' for the last name. A yellow smiley face is drawn next to the console. The background code shows a function that calculates a hash for a name using a polynomial.

```
41 * treats each character in the input name as a coefficient in a polynomial of degree n-1, where n is the length of the name.
42 * It then uses them as coefficients in a polynomial of degree n-1, where n is the length of the name.
43 * 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
44 * but we thought it might be fun!)
45 */
46
47 int nameHash(const std::string& name) {
48     int hashVal = 0;
49     for (int i = 0; i < name.length(); i++) {
50         char ch = name[i];
51         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
52     }
53     return hashVal;
54 }
55
56 int main() {
57     std::string first, last;
58     std::cout << "What is your first name? ";
59     std::getline(std::cin, first);
60     std::cout << "What is your last name? ";
61     std::getline(std::cin, last);
62     int hashVal = nameHash(first + last);
63     std::cout << "The numeric value of the hash is: " << hashVal << std::endl;
64 }
65
66 ch = tolower(ch);
67 hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
```

Console Output:

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

Debug Console:

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

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

```
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         * lower-case letters are always less than 127.
65         */
66         ch = tolower(ch);
67         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68     }
69     return hashVal;
70 }
71 }
```



Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str

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

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



Qt IDE interface showing the source code of `name_hash.cpp` and the debugger window.

**Source Code:**

```
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           * 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  }
```

**Debugger Window:**

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								

**Variable Window:**

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str





File Edit Build Debug Analyze Tools Window Help

Projects name-hash name-hash.pro 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 th
52     * 2023-11-14 14:14:14
```

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.

```
61     for (char ch: first + last) {
62         /* Convert the input character to lower case. The numeric values
63         * lower-case letters are always less than 127.
64         */
65         ch = tolower(ch);
66         hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
67     }
68     return hashVal;
69 }
70 }
71 }
```

Open Documents name\_hash.cpp

Qt Debug

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	Main	name_hash...	31	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name_hash...	27								

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


Name Value Type  
\_for\_begin @0x7fffffff030 str  
\_for\_end @0x7fffffff040 str  
\_for\_range <not accessible> str  
ch 65 'A' ch  
first @0x7fffffff100 str  
hashVal 0 int  
kLargePrime 16908799 int  
kSmallPrime 127 int  
last @0x7fffffff120 str

then the last

case. The numeric values

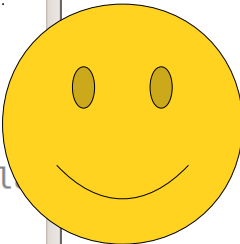
an 127.

me;









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 name-hash.pro 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 th
52     * 2^31 - kLargePrime - 1.
53     */
54     static const int kLargePrime = 16908799;
55     static const int kSmallPrime = 127;
56
57     int hashVal = 0;
58
59     /* Iterate across all the characters in the first name, then the l
60     * name, updating the hash at each step.
61     */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The num
64         * lower
65         */
66         ch =
67         hashV
68     }
69     return ha
70 }
71 }
```

Open Documents name\_hash.cpp

Qt Welcome Edit Design Debug Projects Analyze Help

name-hash

Debug

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	Main	name_hash...	31	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name_hash...	27								

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

Type to locate (Ctrl...)

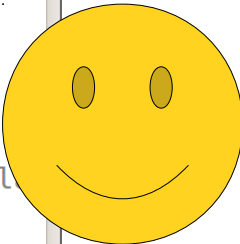
Variable View:

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str

The call stack can tell us exactly that!







Go and double-click the call to `Main` on Level 1 (your Qt Creator may tell you `Main` is on 2 instead of 1, but that is okay).

When you do...

Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

name\_hash.cpp

name-hash

Debug

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 implementation of the function
38  * to talk to the user and get the input
39  * the meaning of the input
40  * of the input
41  *
42  * For those who are not familiar with the
43  * treats each character as a number
44  * It then uses a formula to calculate the hash
45  * For those who are not familiar with the
```

Name	Value	Type
first	@0x7fffffff0a0	std::string
last	@0x7fffffff0c0	std::string
hashValue	766504679	int

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

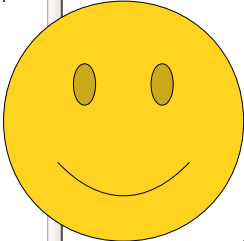
2 Search Results

3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages



... you'll end up over here!

Qt Creator IDE showing a C++ project named "name-hash". The main file, "name\_hash.cpp", is open and displays the following code:

```
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 implementation of the nameHash function.
38  * to talk more about the meaning of the input parameters,
39  * of the input parameters,
40  * For those who are not familiar with the function,
41  * treats each character as a number,
42  * It then uses a simple algorithm to calculate the hash value.
43  * For those who are not familiar with the function,
44  * It then uses a simple algorithm to calculate the hash value.
45  * For those who are not familiar with the function,
46  * It then uses a simple algorithm to calculate the hash value.
```

The line `int hashValue = nameHash(first, last);` (line 31) is highlighted with a yellow background. A yellow smiley face icon is positioned to the right of the code, with a speech bubble pointing to the highlighted line.

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!

The bottom of the IDE shows the "Threads" panel, indicating the program is stopped at breakpoint 1 (1) in thread 1. The "Stack" panel shows the following call stack:

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





Qt IDE interface showing a C++ project named "name-hash". The main editor displays the source file "name\_hash.cpp" with the following code:

```
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 implementation of the nameHash function.
38  * to talk more about the meaning of the input parameters
39  * For those who treat each parameter as a string
40  * It then uses the nameHash function to calculate the hash value
41  * For those who treat each parameter as a string
42  * It then uses the nameHash function to calculate the hash value
43  * For those who treat each parameter as a string
44  * It then uses the nameHash function to calculate the hash value
45  * For those who treat each parameter as a string
46  * It then uses the nameHash function to calculate the hash value
47  * For those who treat each parameter as a string
48  * It then uses the nameHash function to calculate the hash value
49  * For those who treat each parameter as a string
50  * It then uses the nameHash function to calculate the hash value
51  * For those who treat each parameter as a string
52  * It then uses the nameHash function to calculate the hash value
53  * For those who treat each parameter as a string
54  * It then uses the nameHash function to calculate the hash value
55  * For those who treat each parameter as a string
56  * It then uses the nameHash function to calculate the hash value
57  * For those who treat each parameter as a string
58  * It then uses the nameHash function to calculate the hash value
59  * For those who treat each parameter as a string
60  * It then uses the nameHash function to calculate the hash value
61  * For those who treat each parameter as a string
62  * It then uses the nameHash function to calculate the hash value
63  * For those who treat each parameter as a string
64  * It then uses the nameHash function to calculate the hash value
65  * For those who treat each parameter as a string
66  * It then uses the nameHash function to calculate the hash value
67  * For those who treat each parameter as a string
68  * It then uses the nameHash function to calculate the hash value
69  * For those who treat each parameter as a string
70  * It then uses the nameHash function to calculate the hash value
71  * For those who treat each parameter as a string
72  * It then uses the nameHash function to calculate the hash value
73  * For those who treat each parameter as a string
74  * It then uses the nameHash function to calculate the hash value
75  * For those who treat each parameter as a string
76  * It then uses the nameHash function to calculate the hash value
77  * For those who treat each parameter as a string
78  * It then uses the nameHash function to calculate the hash value
79  * For those who treat each parameter as a string
80  * It then uses the nameHash function to calculate the hash value
81  * For those who treat each parameter as a string
82  * It then uses the nameHash function to calculate the hash value
83  * For those who treat each parameter as a string
84  * It then uses the nameHash function to calculate the hash value
85  * For those who treat each parameter as a string
86  * It then uses the nameHash function to calculate the hash value
87  * For those who treat each parameter as a string
88  * It then uses the nameHash function to calculate the hash value
89  * For those who treat each parameter as a string
90  * It then uses the nameHash function to calculate the hash value
91  * For those who treat each parameter as a string
92  * It then uses the nameHash function to calculate the hash value
93  * For those who treat each parameter as a string
94  * It then uses the nameHash function to calculate the hash value
95  * For those who treat each parameter as a string
96  * It then uses the nameHash function to calculate the hash value
97  * For those who treat each parameter as a string
98  * It then uses the nameHash function to calculate the hash value
99  * For those who treat each parameter as a string
100 * It then uses the nameHash function to calculate the hash value
```

The "Open Documents" pane shows "name\_hash.cpp". The "Debug" pane shows the call stack:

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

The "Variables" pane shows the following variables:

Name	Value	Type
first	@0x7fffffffe0a0	std::string
last	@0x7fffffffe0c0	std::string
hashValue	766504679	int

A yellow smiley face is drawn next to the call stack, with a speech bubble containing the text:

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

Qt IDE interface showing a C++ project named "name-hash". The main editor displays the source code for "name\_hash.cpp". The code includes "console.h" and "simpio.h", and uses the "std" namespace. It defines a "nameHash" function and a "main" function. The "main" function prompts the user for their first and last names, calculates the hash, and prints the result. A breakpoint is set at line 31, which is highlighted with a yellow arrow.

```
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 implementation of the nameHash function.
38  * to talk more about the meaning of the input parameters,
39  * see the comments in the header file.
40  *
41  * For those who are new to C++, this function
42  * treats each character of the string as a
43  * separate entity. It then uses a
44  * simple algorithm to calculate the hash value.
45  * For more information, see the comments in the header file.
```

The "Open Documents" pane shows "name\_hash.cpp". The "Debug" pane at the bottom shows the execution stack, with the "Main" function highlighted. The "Variables" pane on the right shows the values of "first", "last", and "hashValue".

Let's find out! Double-click on the line marked "Main" on Level 2 (or maybe 3 -- click the next Main down the list). When you do...

Qt IDE interface showing a C++ project named "name-hash". The main window displays the source file "main.cpp" with the following code:

```
1  /* ... */
17
18 #include <iostream>
19
20 #ifndef SPL_AUTOGRADER_MODE
21 int Main(int, char* /*argv*/[]) {
22     extern int Main();
23     return Main();
24 }
25 #endif // SPL_AUTOGRADER_MODE
26
```

The code is annotated with a yellow smiley face and a speech bubble indicating a common mistake:

... you'll end up with something that looks like this.

The bottom panel shows the "Threads" view, indicating the program is stopped at breakpoint 1 (1) in thread 1. The "Stack" view shows the following call stack:

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

The bottom status bar shows the "Issues" tab selected, with a search bar and tabs for "Search Results", "Application Output", "Compile Output", "QML/JS Console", and "General Messages".

Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

Debug

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

/\* ... \*/

#include <iostream>

#ifndef SPL\_AUTOGRADER\_MODE

int Main(int, char\* /\*argv\*/[]) {

extern int Main();

return Main();

}

#endif // SPL\_AUTOGRADER\_MODE

# Line: 23, Col: 5

Name

Value

Type

Yikes! This looks Hairy and Scary! What happened?

Threads: #1 name-hash

Stopped at breakpoint 1 (1) in thread 1.

Level	Function	File	Line
0	nameHash	name_hash...	66
1	Main	name_hash...	31
2	Main	main.cpp	23
3	startupMain	platform.cpp	2208
4	main	name_hash...	27

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

Type to locate (Ctrl...

1 Issues

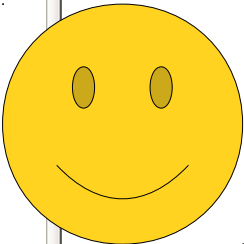
2 Search Results

3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages



Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

/\* ... \*/

#include <iostream>

#ifndef SPL\_AUTOGRADER\_MODE

int Main(int, char\* /\*argv\*/[]) {

extern int Main();

return Main();

}

#endif // SPL\_AUTOGRADER\_MODE

Qt

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...

1 Issues

2 Search Results

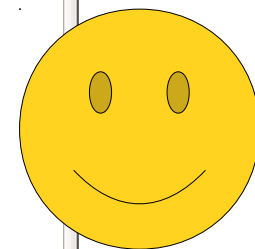
3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages

11:07 AM



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.



Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

/\* ... \*/

#include <iostream>

#ifndef SPL\_AUTOGRADER\_MODE

int Main(int, char\* /\*argv\*/[]) {

extern int Main();

return Main();

}

#endif // SPL\_AUTOGRADER\_MODE

Name

Value

Type

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	Main	name_hash...	31	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
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 code will show up in the call stack below your actual program.

Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

Debug

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

/\* ... \*/

#include <iostream>

#ifndef SPL\_AUTOGRADER\_MODE

int Main(int, char\* /\*argv\*/[]) {

extern int Main();

return Main();

}

#endif // SPL\_AUTOGRADER\_MODE

Qt

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...

1 Issues

2 Search Results

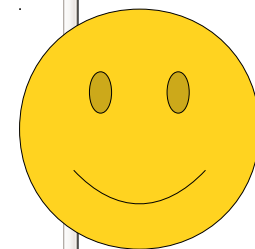
3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages

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.







Qt IDE interface showing a C++ project named "name-hash". The main editor displays the source file "main.cpp" with the following code:

```
1  /* ... */
17
18 #include <iostream>
19
20 #ifndef SPL_AUTOGRADER_MODE
21 int Main(int, char* /*argv*/[]) {
22     extern int Main();
23     return Main();
24 }
25 #endif // SPL_AUTOGRADER_MODE
26
```

The "Open Documents" list shows "main.cpp" and "name\_hash.cpp". The "Debug" console at the bottom displays the following stack trace:

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

The "Debug" console also shows the following information:

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

A yellow smiley face is drawn next to the stack trace, and a speech bubble contains the text:

To do that, double-click on Level 0 (or 1 -- see the image below), the call to nameHash. When you do...

Qt Creator IDE showing a C++ project named "name-hash". The main editor displays the source file "name\_hash.cpp" with the following code:

```
45  * F_p, where p is a large prime number, and evaluates that polynomial a
46  * some smaller prime number q. (You aren't expected to know this for CS
47  * but we thought it might be fun!
48  */
49  int nameHash(string first, string last){
50      /* This hashing scheme needs two prime numbers, a large prime and a
51       * prime. These numbers were chosen because their product is less th
52       * 2^31 - kLargePrime - 1.
53       */
54      static const int kLargePrime = 16908799;
55      static const int kSmallPrime = 127;
56
57      int hashVal = 0;
58
59      /* Iterate across all the characters in the first name, then the l
60       * name, updating the hash at each step.
61       */
62      for (char ch: first + last) {
63          /* Convert the input character to lower case. The num
64           * lower
65           */
66          ch =
67          hashV
68      }
69      return ha
70  }
71
```

The code is currently stopped at a breakpoint at line 66. A yellow smiley face is drawn next to the code, and a speech bubble contains the text: "You'll be teleported back to safety!"

The Qt Creator interface includes a sidebar with project files, a top menu bar (File, Edit, Build, Debug, Analyze, Tools, Window, Help), and a bottom status bar with tabs for Issues, Search Results, Application Output, Compile Output, QML/JS Console, and General Messages.

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

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)

Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

Debug

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

nameHash(string, string): int

# Line: 66, Col: 9

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

\* F\_p, where p is a large prime number, and evaluates that polynomial a

\* some smaller prime number q. (You aren't expected to know this for CS

\* but we thought it might be fun!

\*/

int nameHash(string first, string last){

/\* This hashing scheme needs two prime numbers, a large prime and a

\* prime. These numbers were chosen because their product is less th

\* 2^31 - kLargePrime - 1.

\*/

static const int kLargePrime = 16908799;

static const int kSmallPrime = 127;

int hashVal

/\* It

\* nam

\*/

for (c

/\*

\*/

\*/

ch = tolower(ch);

hashVal = (kSmallPrime \* hashVal + ch) % kLargePrime;

}

return hashVal;

}

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

Qt

Debug

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								

1 Issues

2 Search Results

3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages

Name

Value

Type

for\_begin

@0x7fffffff030

str

for\_end

@0x7fffffff040

str

for\_range

<not accessible>

ch

65 'A'

ch

first

@0x7fffffff100

str

hashVal

0

int

kLargePrime

16908799

int

kSmallPrime

127

int

last

@0x7fffffff120

str

Qt IDE interface showing a C++ project named "name-hash". The main editor displays the file "name\_hash.cpp" with the following code:

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

A red arrow points to line 66, indicating a breakpoint. A speech bubble contains the text:

To set a breakpoint so that we can pause the program and look around, click in the margin just before the line number where you want to pause.

A yellow smiley face is drawn next to the code.

The Qt IDE interface includes a sidebar with project files, a top menu bar (File, Edit, Build, Debug, Analyze, Tools, Window, Help), and a bottom status bar showing the current thread and breakpoint information.

The bottom status bar shows the following information:

- Threads: #1 name-hash
- Stopped at breakpoint 1 (1) in thread 1.
- Level: 0
- Function: nameHash
- File: name\_hash...
- Line: 66
- Number: 1
- Function: nameHash(...)
- File: /home/keit...
- Line: 66
- Address: 0x4c9cc3
- Condition: (all)
- Ignore: (all)
- Threads: (all)

Qt Creator IDE interface showing a C++ project named "name-hash". The main editor displays the source code for "name\_hash.cpp", which implements a name hashing function. A large speech bubble annotation is overlaid on the code, stating: "Once the breakpoint is reached, it will pull up all sorts of useful information." A yellow smiley face icon is positioned next to the speech bubble.

The code in "name\_hash.cpp" is as follows:

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

The right sidebar shows the "Name" and "Value" of variables in the current scope:

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

The bottom status bar indicates the program is stopped at breakpoint 1 (1) in thread 1. The bottom panel shows the "Threads" and "Stack" views.



Qt Creator IDE showing a C++ project named "name-hash". The main editor displays the file "name\_hash.cpp" with the following code:

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

A yellow arrow points to line 66, highlighting the variable `ch` in the assignment `ch = tolower(ch);`. A large speech bubble contains the text: "The yellow arrow points out where we are right now." A yellow smiley face is also present next to the arrow.

The right sidebar shows the "Name" pane with the following variables and values:

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

The bottom status bar shows the "Threads" pane with the following information:

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

The "Threads" pane also shows "Stopped at breakpoint 1 (1) in thread 1." The bottom status bar includes tabs for "Issues", "Search Results", "Application Output", "Compile Output", "QML/JS Console", and "General Messages".



Qt 5.12.0

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

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

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



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

Qt Creator IDE showing a C++ project named "name-hash". The main editor displays the file "name\_hash.cpp" with the following code:

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

A handwritten note in a speech bubble says: "Now, let's see how we can read the values of the variables in this function."

The right sidebar shows the "Name" pane with the following variables and their values:

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

The bottom status bar shows the "Threads" pane with the following information:

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

The "Threads" pane also shows "Stopped at breakpoint 1 (1) in thread 1."

The bottom status bar shows the "Issues" pane with the following information:

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

Look up at this panel over here.



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

```
45 * but we want to be fun!
46 */
47
48 static const int kLargePrime = 16908799;
49 static const int kSmallPrime = 127;
50
51 int hashVal = 0;
52
53 /* Iterate across all the characters in the first name, then the last
54 * name, updating the hash at each step.
55 */
56 for (char ch: first + last) {
57     /* Convert the input character to lower case. The numeric values
58     * lower-case letters are always less than 127.
59     */
60     ch = tolower(ch);
61     hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
62 }
63 return hashVal;
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								

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.



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

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



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

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	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								

1 Issues

2 Search Results

3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages



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



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

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	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

For example, here you can see the values of `kLargePrime` and `kSmallPrime`, which match the values they were declared with (you may have to click on the arrow next to "[statics]" on your computer to see these values).



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

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str

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
48 */
49
50 nameHash(string first, string last){
51     /* This hashing scheme needs two prime numbers, a large prime and a
52     * prime. These numbers were chosen because their product is less th
53     *  $2^{31}$  - kLargePrime - 1.
54     */
55     static const int kLargePrime = 16908799;
56     static const int kSmallPrime = 127;
57
58     int hashVal = 0;
59
60     /* Iterate across all the characters in the first name, then the las
61     * name, updating the hash at each step.
62     */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numeric values
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 }
```

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	str
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str

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 * but we
48 */
49
50 nameHash(string first, string last){
51     /* This hashing scheme needs two prime numbers, a large prime and a
52     * small prime. These numbers were chosen because their product is less th
53     *  $2^{31} - kLargePrime - 1$ .
54     */
55     static const int kLargePrime = 16908799;
56     static const int kSmallPrime = 127;
57
58     int hashVal = 0;
59
60     /* Iterate across all the characters in the first name, then the las
61     * name, updating the hash at each step.
62     */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numeric values
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 }
```

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str



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 * but we
48 */
49
50 nameHash(string first, string last){
51     /* This hashing scheme needs two prime numbers, a large prime and a
52     * small prime. These numbers were chosen because their product is less th
53     *  $2^{31} - kLargePrime - 1$ .
54     */
55     static const int kLargePrime = 16908799;
56     static const int kSmallPrime = 127;
57
58     int hashVal = 0;
59
60     /* Iterate across all the characters in the first name, then the las
61     * name, updating the hash at each step.
62     */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numeric values
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 }
```

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str

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 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 * but we
48 */
49
50 nameHash(string first, string last){
51     /* This hashing scheme needs two prime numbers, a large prime and a
52     * small prime. These numbers were chosen because their product is less th
53     *  $2^{31} - kLargePrime - 1$ .
54     */
55     static const int kLargePrime = 16908799;
56     static const int kSmallPrime = 127;
57
58     int hashVal = 0;
59
60     /* Iterate across all the characters in the first name, then the las
61     * name, updating the hash at each step.
62     */
63     for (char ch: first + last) {
64         /* Convert the input character to lower case. The numeric values
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 }
```

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str



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

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



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

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	str
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str



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



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.



```
45
46
47 * but we
48 */
49
50 static const int kLargePrime = 16908799;
51 static const int kSmallPrime = 127;
52
53 int hashVal = 0;
54
55
56
57
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
```

Name	Value	Type
for_begin	@0x7fffffff030	std::string::size_type
for_end	@0x7fffffff040	std::string::size_type
for_range	not accessible	std::string::size_type
ch	65 'A'	char
first	@0x7ffffff00	std::string
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	std::string

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

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.



Qt IDE interface showing the source code of `name_hash.cpp` and the debug console.

**Source Code (name\_hash.cpp):**

```
45 * but we want to be fun!
46 */
47
48 /*
49  * nameHash(string first, string last){
50  * This hashing scheme needs two prime numbers, a large prime and a
51  * small prime. These numbers were chosen because their product is less th
52  *  $2^{31} - kLargePrime - 1$ .
53  */
54 static const int kLargePrime = 16908799;
55 static const int kSmallPrime = 127;
56
57 int hashVal = 0;
58
59 /* Iterate across all the characters in the first name, then the las
60  * name, updating the hash at each step.
61  */
62 for (char ch: first + last) {
63     /* Convert the input character to lower case. The numeric 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
```

**Debug Console:**

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								

**Memory View:**

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str



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!



Qt IDE interface showing the 'name-hash' project. The main editor displays the source code for 'name\_hash.cpp'. The code defines a hashing function 'nameHash' that takes two strings, 'first' and 'last', and returns a hash value. The function uses two prime numbers, 'kLargePrime' (16908799) and 'kSmallPrime' (127), to calculate the hash. The code is as follows:

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

The Qt IDE interface also shows the 'Open Documents' list with 'main.cpp' and 'name\_hash.cpp'. The 'Debug' console at the bottom shows the execution status: 'Threads: #1 name-hash' and 'Stopped at breakpoint 1 (1) in thread 1.' The 'Views' panel at the bottom right shows the current state of the program, including the 'Name' and 'Value' columns.

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str





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

Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

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 th
52       * 2^31 - kLargePrime - 1.
53       */
54      static const int kLargePrime = 16908799;
55      static const int kSmallPrime = 127;
56
57      int hashVal = 0;
58
59      /* Iterate across all the characters in the first name, then the las
60       * name, updating the hash at each step.
61       */
62      for (char ch: first + last) {
63          /* Convert the input character to lower case. The numeric 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	Action	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

1 Issues

2 Search Results

3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages

Name

Value

Type

__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	65 'A'	ch
first	@0x7fffffff100	str
hashVal	0	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str



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."



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...





...your window should look something like this.







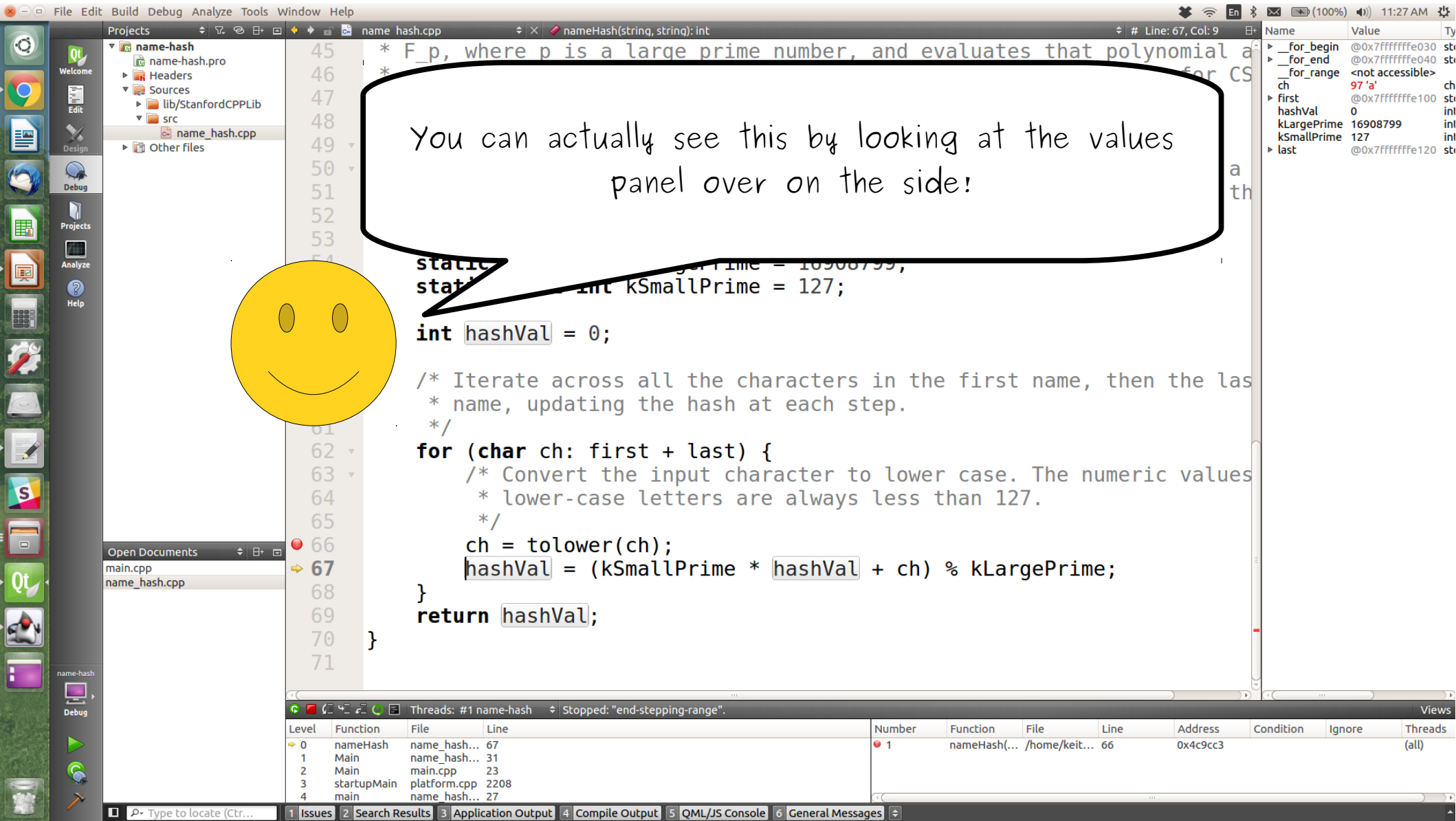
First, notice that our helpful Yellow Arrow friend is now pointing at line 67.



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

nameHash(string, string): int # Line: 67, Col: 9

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

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;

Notice that the value associated with ch has changed from 'A' to 'a' - it's now in lower-case!

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;

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...	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								

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



Qt 5.12.0

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

# Line: 67, Col: 9

Name Value Type

- \_\_for\_begin @0x7fffffff030 str
- \_\_for\_end @0x7fffffff040 str
- \_\_for\_range <not accessible> ch
- ch 97 'a' ch
- first @0x7fffffff100 str
- hashVal 0 int
- kLargePrime 16908799 int
- kSmallPrime 127 int
- last @0x7fffffff120 str

If you'll notice, this value is in red while all the other values are in black.

```
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 static const int kLargePrime = 16908799,
56 static const int kSmallPrime = 127;
57
58 int hashVal = 0;
59
60 /* Iterate across all the characters in the first name, then the last
61 * name, updating the hash at each step.
62 */
63 for (char ch: first + last) {
64     /* Convert the input character to lower case. The numeric values
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 }
```

Open Documents

- main.cpp
- name\_hash.cpp

name-hash

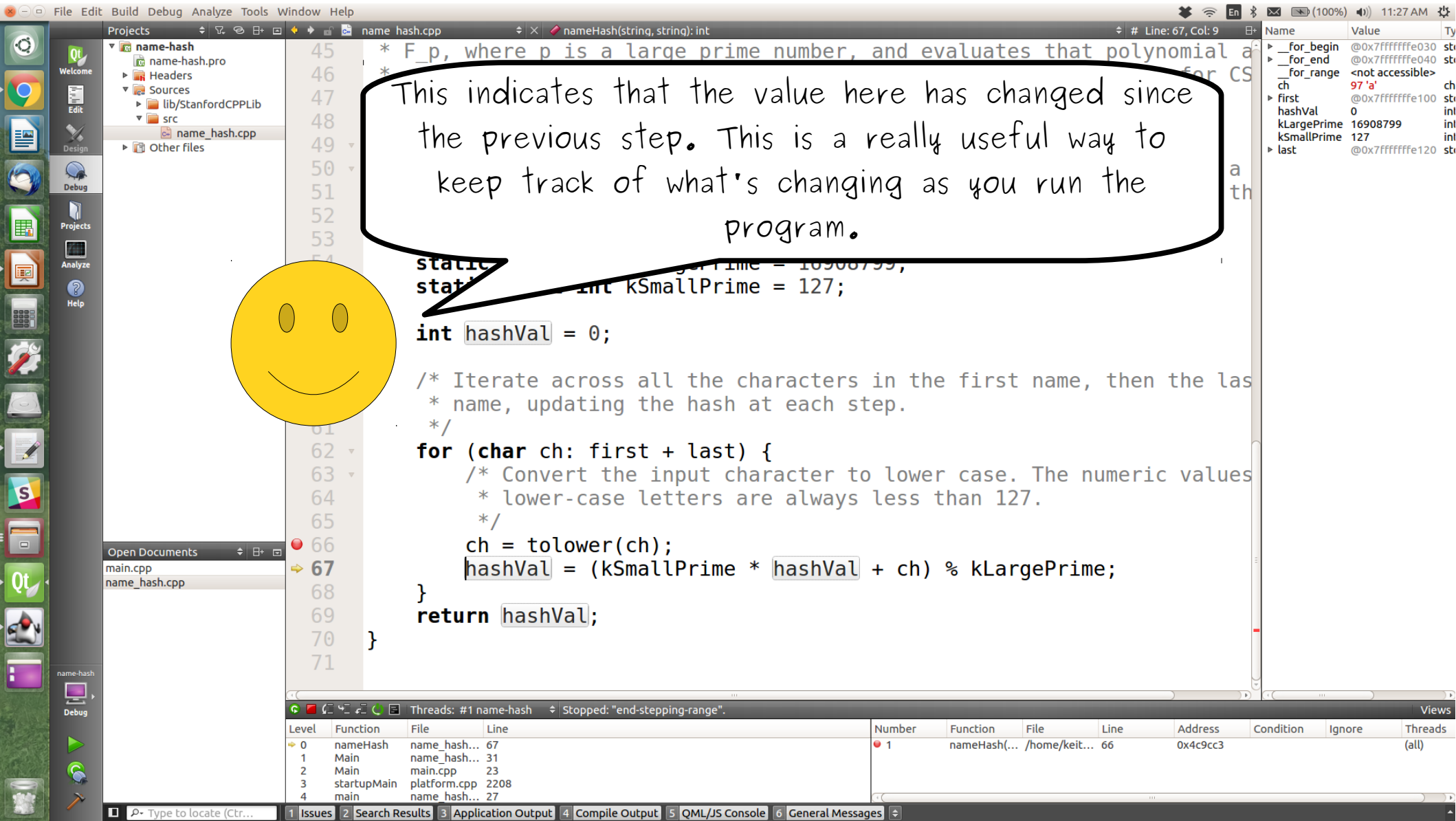
Debug

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

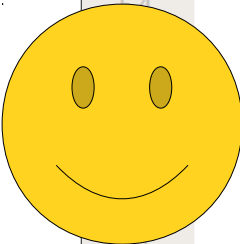
Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	67	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
1	Main	name_hash...	31								
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name_hash...	27								

Type 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
  - name-hash.pro
  - Headers
  - Sources
    - lib/StanfordCPPLib
    - src
      - name\_hash.cpp
  - Other files

Open Documents

- main.cpp
- name\_hash.cpp

name-hash

Debug

nameHash(string, string): int

# Line: 67, Col: 9

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

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

```
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: "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								

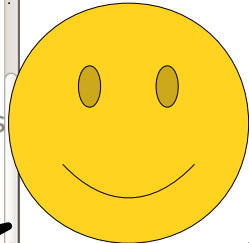
Type to locate (Ctrl+...)

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

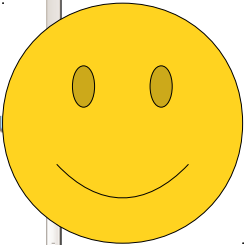






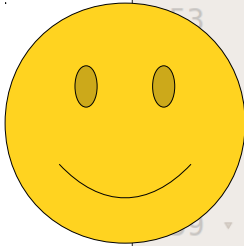






Let's see what's changed.

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.  
 */
```

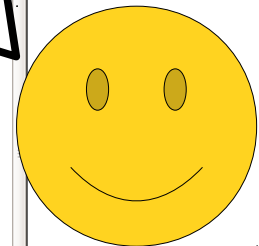
```
for (char ch: first + last) {  
    /* Convert the input character to lower case. The numeric values  
     * lower-case letters are always less than 127.  
     */  
    ch = tolower(ch);  
    hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;  
}  
return hashVal;
```

Name	Value	Type
__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	97 'a'	ch
first	@0x7fffffff100	str
hashVal	97	int
kLargePrime	16908799	int
kSmallPrime	127	int
last	@0x7fffffff120	str

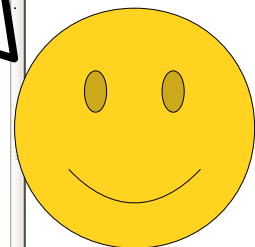
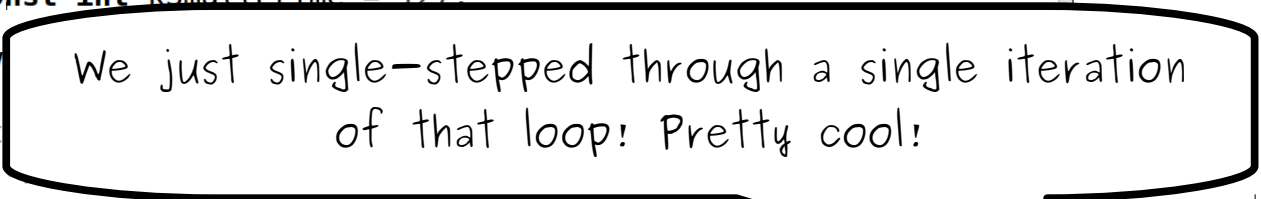
Threads: #1 name-hash Stopped: "end-stepping-range".

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								

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

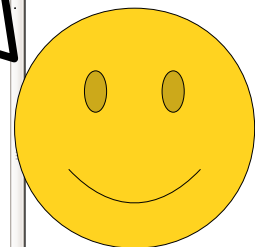


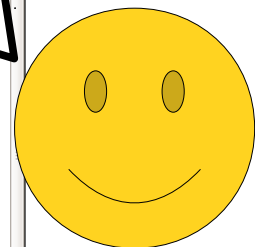


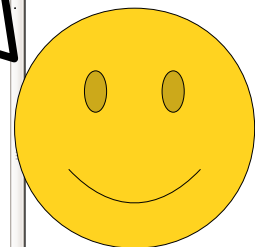












Qt 5.12.0

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


```
45 * F_p, where p is a large prime number, and evaluates that polynomial a
46 * some smaller prime number q. (You aren't expected to know this for CS
47 * but we thought it might be fun!
48 */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers, a large prime and a
51     * prime. These numbers were chosen because their product is less th
52     * 2^31 - kLargePrime - 1.
53     */
54     static const int kLargePrime = 16999799;
55     static co
56
57     int hashVal
58
59     /* Iterat
60     * name,
61     */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The numeric va
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	Main	name_hash...	31	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name_hash...	27								

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





Qt 5.12.0 - Qt Creator 4.10.0

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

```
45 * F_p, where p is a large prime number, and evaluates that polynomial a
46 * some smaller prime number q. (You aren't expected to know this for CS
47 * but we thought it might be fun!
48 */
49 int nameHash(string first, string last){
50     /* This hashing scheme needs two prime numbers, a large prime and a
51     * prime. These numbers were chosen because their product is less th
52     * 2^31 - kLargePrime - 1.
53     */
54     static const int kLargePrime = 16999799;
55     static const int kSmallPrime = 1000000007;
56
57     int hashVal = 0;
58
59     /* Iterate over the characters in the strings
60     * name, first, and last.
61     */
62     for (char ch: first + last) {
63         /* Convert the input character to lower case. The numeric value 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
```

Now, click "Step Over" one more time.



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	Main	name_hash...	31	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name_hash...	27								

Type to locate (Ctrl+Shift+F)

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



Qt 5.12.0

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

```
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
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 }
```

Threads: #1 name-hash Finished retrieving data

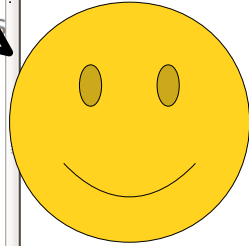
Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
0	nameHash	name_hash...	62								
1	Main	name_hash...	31	1	nameHash(...	/home/keit...	66	0x4c9cc3			(all)
2	Main	main.cpp	23								
3	startupMain	platform.cpp	2208								
4	main	name_hash...	27								

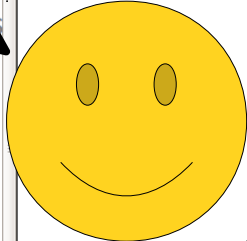
1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML/JS Console 6 General Messages

Name Value

Name	Value
__for_begin	@0x7fffffff030
__for_end	@0x7fffffff040
__for_range	<not accessible>
ch	100 'd'
first	fffffe100
hashVal	?
kLargePrime	99
kSmallPrime	127
last	0x7fffffff120

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!





Qt IDE interface showing a C++ project named "name-hash". The main editor displays the source code for `nameHash(string first, string last): int`. A breakpoint is set at line 62, which is highlighted with a blue arrow and a red dot. A speech bubble points to this breakpoint, containing the text: "To start this off, click on the the breakpoint that we set earlier in the program. If you do...". A yellow smiley face is also present next to the speech bubble.

The code in the editor is as follows:

```
45  * F_p, where p is a large prime number, and evaluates that polynomial a
46  * some smaller prime number q. (You aren't expected to know this for CS
47  * but we thought it might be fun!
48  */
49  int nameHash(string first, string last){
50      /* This hashing scheme needs two prime numbers, a large prime and a
51       * prime. These numbers were chosen because their product is less th
52       * 2^31 - kLargePrime - 1.
53
54
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
```

The Variables window on the right shows the following values:

Name	Value
__for_begin	@0x7fffffff030
__for_end	@0x7fffffff040
__for_range	<not accessible>
ch	100 'd'
first	fffffe100
hashVal	?
kLargePrime	99
kSmallPrime	127
last	@0x7fffffff120

The Debug console at the bottom shows the following threads:

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

Qt 5.12.0 (11:50 AM)

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

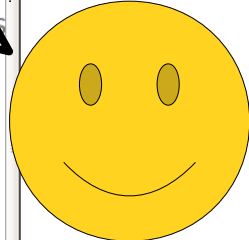
- main.cpp
- name\_hash.cpp

name-hash

Debug

```
45  * F_p, where p is a large prime number, and evaluates that polynomial a
46  * some smaller prime number q. (You aren't expected to know this for CS
47  * but we thought it might be fun!
48  */
49  int nameHash(string first, string last){
50      /* This hashing scheme needs two prime numbers, a large prime and a
51       * prime. These numbers were chosen because their product is less th
52       * 2^31 - kLargePrime - 1.
53
54
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
```

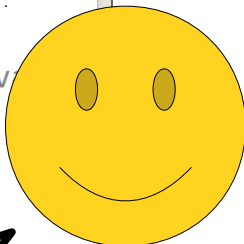
... it should clear the breakpoint. Now, if we were to run this program again in debug mode, it would not stop at this point, since nothing's telling it to!



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

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

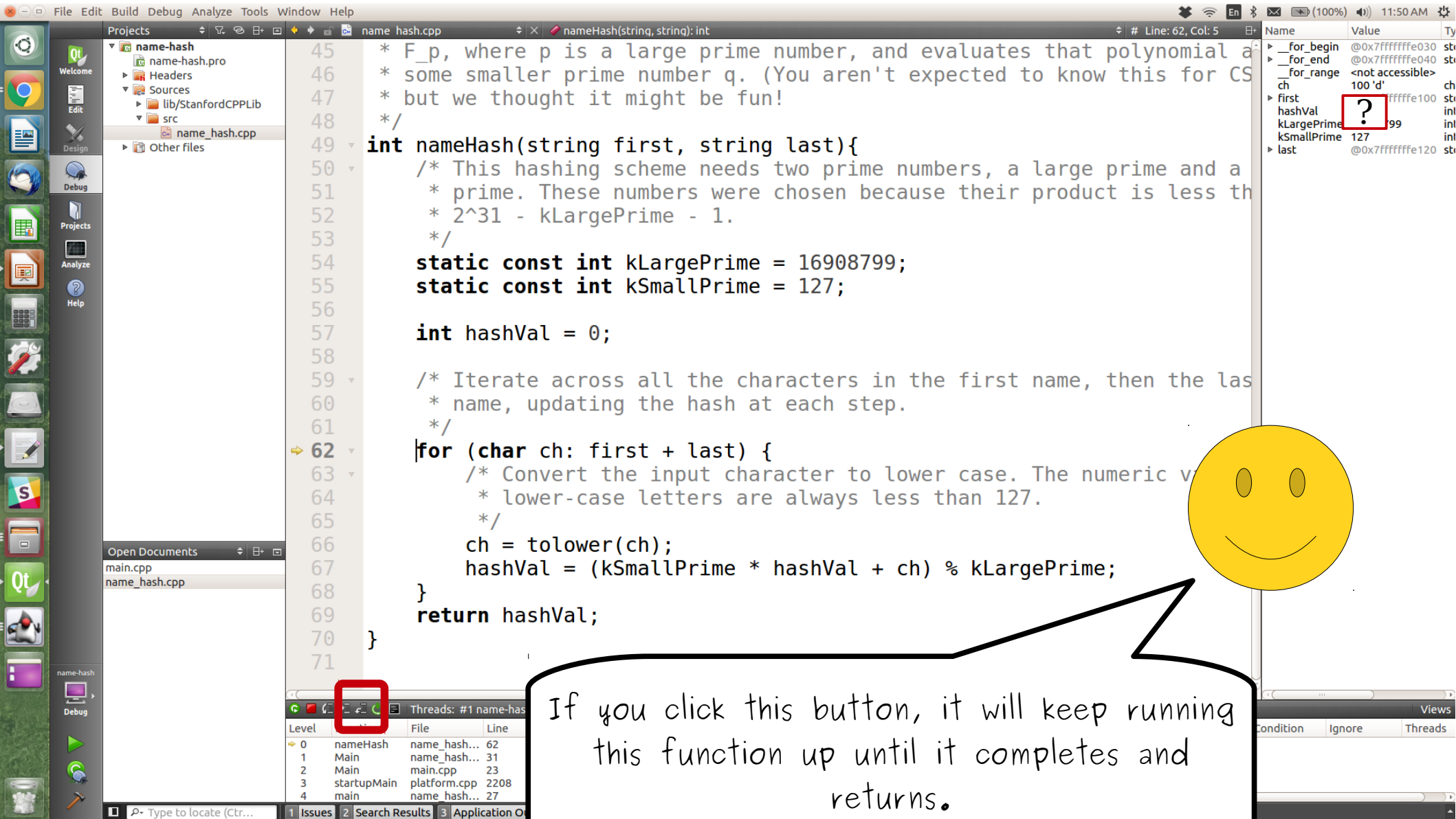


Now, take a look back at these buttons.









Qt

Welcome

Edit

Design

Debug

Projects

Analyze

Help

Open Documents

main.cpp

name\_hash.cpp

name-hash

name-hash.pro

Headers

Sources

lib/StanfordCPPLib

src

name\_hash.cpp

Other files

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 th
52       * 2^31 - kLargePrime - 1.
53       */
54      static const int kLargePrime = 16908799;
55      static const int kSmallPrime = 127;
56
57      int hashVal = 0;
58
59      /* Iterate across all the characters in the first name, then the las
60       * name, updating the hash at each step.
61       */
62      for (char ch: first + last) {
63          /* Convert the input character to lower case. The numeric v
64           * lower-case letters are always less than 127.
65           */
66          ch = tolower(ch);
67          hashVal = (kSmallPrime * hashVal + ch) % kLargePrime;
68      }
69      return hashVal;
70  }
71  }
```

Threads: #1 name-has

Level		File	Line
0	nameHash	name_hash...	62
1	Main	name_hash...	31
2	Main	main.cpp	23
3	startupMain	platform.cpp	2208
4	main	name_hash...	27

1 Issues

2 Search Results

3 Application O

Name

Value

Type

__for_begin	@0x7fffffff030	str
__for_end	@0x7fffffff040	str
__for_range	<not accessible>	ch
ch	100 'd'	ch
first	fffffe100	str
hashVal	?	int
kLargePrime	99	int
kSmallPrime	127	int
last	@0x7fffffff120	str



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

Qt Creator IDE showing a C++ project named "name-hash". The main window displays the source code for "name\_hash.cpp". The code includes "console.h" and "simpio.h", and uses the "std" namespace. It defines a "nameHash" function and a "main" function. The "main" function prompts the user for their first and last names, calculates the hash value, and prints it.

```
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 implementation of the nameHash function.
38  * to talk more about the meaning of the input parameters,
39  * see the documentation.
40  * For those who are new to C++,
41  * treats each character as a number.
42  * It then uses a simple algorithm to calculate the hash value.
43  * For more information, see the documentation.
44  * For more information, see the documentation.
45  */
```

The right sidebar shows the "Name" and "Value" of variables. The "first" variable has a value of "@0x7fffffff0a0", the "last" variable has a value of "@0x7fffffff0c0", and the "hashValue" variable has a value of "-590633613".

A yellow smiley face is drawn next to the code. A speech bubble points to the "hashValue" variable, containing the text: "... you should end up with something that looks like this!".

The bottom status bar shows the "Threads" panel, indicating that the program has stopped at "function-finished". The "Level" panel shows the call stack, with the top level being "Main" in "name\_hash.cpp" at line 31.

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

Level	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				

Qt Creator IDE interface showing a C++ project named "name-hash". The main editor displays the source file "name\_hash.cpp" with the following code:

```
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 implementation of the nameHash function.
38  * to talk more about the meaning of the input parameters,
39  * of the input parameters,
40  * of the input parameters,
41  * For those who are not familiar with the function,
42  * treats each character as a number,
43  * It then uses a simple algorithm to calculate the hash value.
44  * For those who are not familiar with the function,
45  * For those who are not familiar with the function,
```

The right sidebar shows the "Name" and "Value" of variables:

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

A yellow smiley face is drawn next to the code. A speech bubble points to the code with the text:

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

The bottom status bar shows the "Threads" panel with the following data:

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

The bottom status bar also shows the "Application Output" panel with the following text:

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

Qt Creator IDE interface showing a C++ project named "name-hash". The main editor displays the source file "name\_hash.cpp" with the following code:

```
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 implementation of the nameHash function.
38  * to talk more about the meaning of the input parameters,
39  * see the comments in the header file.
40  * For those who are new to C++,
41  * treats each character as a number.
42  * It then uses a simple algorithm to calculate the hash value.
43  * For more information, see the header file.
```

A yellow arrow points to line 31, indicating the call to the `nameHash` function. A speech bubble points to this line with the text: "Well, the yellow arrow indicates that we're back in main again. Cool!"

The right sidebar shows the "Name" and "Value" of variables:

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

The bottom status bar shows the "Threads" panel, indicating the program has stopped at "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



Qt Creator IDE interface showing a C++ project named "name-hash". The main editor displays the source code for "name\_hash.cpp".

```
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 }
35
36 /* This is the actual implementation of the nameHash function.
37  * to talk more about the meaning of the input parameters,
38  * see the comments in the header file.
39  * For those who are interested, the function
40  * treats each character of the input string as a digit
41  * It then uses a base-26 conversion to calculate the hash.
42  * For example, the name "John" would be converted to
43  * 10^3 + 15^2 + 8^1 + 4^0 = 1000 + 225 + 8 + 4 = 1237.
```



We can see that the nameHash function returned 1967457. Thanks, debugger! (well, unfortunately this doesn't work on a Mac...)

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

returned value 1967457

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

1 Issues

2 Search Results

3 Application Output

4 Compile Output

5 QML/JS Console

6 General Messages





But if you look 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.)

Qt IDE interface showing the source code for `name_hash.cpp` and the Qt Debug Console.

**Source Code:**

```
18 #include <iostream>
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for nameHash
23  * in main and the function definition
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
```

**Qt Debug Console:**

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

Qt Debug Console Output:

```
returned value 1967457
```



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

```
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 code. We're going  
* to talk more about what hash functions do later in the quarter. In  
* the meantime, think of it as a function that scrambles up the character  
* of the input and produces a number.  
*  
* For those of you who are more mathematically inclined, this function  
* treats each character in the input name as a number between 0 and 128  
* It then uses them as coefficients in a polynomial over the finite field  
* F_p where p is a large prime number and evaluates that polynomial at
```

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...

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



This is pretty cool, actually!

Qt IDE interface showing a C++ project named "name-hash". The main window displays the source code for "name\_hash.cpp". The code includes a header file "simpio.h" and uses the "std" namespace. It defines a function "nameHash" and a "main" function. The "main" function prompts the user for their last name, computes a hash value using "nameHash", and prints it. The "nameHash" function is a placeholder for a more complex implementation.

```
18 #include "simpio.h"
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for
23  * in main and
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_p where p is a large prime number and evaluates that polynomial at
```

The Qt IDE interface also shows a "Name" and "Value" table on the right side of the editor, displaying the values of the variables "first", "last", and "hashValue".

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

The "Open Documents" list on the left shows "main.cpp" and "name\_hash.cpp". The "Debug" console at the bottom shows the execution flow, indicating that the program has finished execution.

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

Qt IDE interface showing the "name-hash" project. The main window displays the source code for "name\_hash.cpp". The code includes a header file "simpio.h" and uses the "std" namespace. It defines a function "nameHash" and a "main" function. The "main" function prompts the user for their last name, computes a hash value using "nameHash", and prints it. The "nameHash" function is a placeholder for a more complex implementation.

```
18 #include "simpio.h"
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for
23  * in main and
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_p where p is a large prime number and evaluates that polynomial at
```

The Qt IDE interface also shows a "Name" and "Value" table on the right side of the editor, displaying the values of the variables "first", "last", and "hashValue".

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

The "Open Documents" list on the left shows "main.cpp" and "name\_hash.cpp". The "Debug" console at the bottom shows the execution flow, indicating that the program has finished execution.

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

Qt IDE interface showing the "name-hash" project. The main window displays the source code for "name\_hash.cpp". The code includes a header file "simpio.h" and uses the "std" namespace. It defines a function "nameHash" and a "main" function. The "main" function prompts the user for their last name, computes a hash value using "nameHash", and prints it. The "nameHash" function is a placeholder for a more complex implementation.

```
18 #include "simpio.h"
19 #include "simpio.h"
20 using namespace std;
21
22 /* Prototype for
23  * in main and
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_p where p is a large prime number and evaluates that polynomial at
```

The Qt IDE interface also shows a "Name" and "Value" table on the right side of the editor, displaying the values of the variables "first", "last", and "hashValue".

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

The "Open Documents" list on the left shows "main.cpp" and "name\_hash.cpp". The "Debug" console at the bottom shows the execution flow, indicating that the program has finished execution.

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



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!

```
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 code. We're going  
* to talk more about what hash functions do later in the quarter. In  
* the meantime, think of it as a function that scrambles up the character  
* of the input and produces a number.  
*  
* For those of you who are more mathematically inclined, this function  
* treats each character in the input name as a number between 0 and 128  
* It then uses them as coefficients in a polynomial over the finite field  
*  $F_p$  where  $p$  is a large prime number and evaluates that polynomial at
```

Name	Value
first	@0x7fffffff0a0
last	@0x7fffffff0c0
hashValue	-590633613

returned value 1967457

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

1

2

3

4

5

6

Issues

Search Results

Application Output

Compile Output

OML/JS Console

General Messages



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

Qt IDE interface showing a C++ project named "name-hash". The main window displays the source code for "name\_hash.cpp". The code includes a function prototype for "nameHash", a "main" function, and the implementation of "nameHash". The "main" function prompts the user for their first and last names, computes the hash, and prints the result. The "nameHash" function is a complex mathematical function that computes the hash code.

The "main" function is as follows:

```
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;
}
```

The "nameHash" function is as follows:

```
int nameHash(string, string): int
/* This is the actual function that computes the hash code. We're going
 * to talk more about what hash functions do later in the quarter. In
 * the meantime, think of it as a function that scrambles up the character
 * of the input and produces a number.
 *
 * For those of you who are more mathematically inclined, this function
 * treats each character in the input name as a number between 0 and 128.
 * It then uses them as coefficients in a polynomial over the finite field
 * F_p where p is a large prime number and evaluates that polynomial at
```

The "Qt" sidebar on the left shows the project structure, including "name-hash.pro", "Headers", "Sources", "lib/StanfordCPPLib", "src", and "name\_hash.cpp". The "Open Documents" list shows "main.cpp" and "name\_hash.cpp".

The "Debug" sidebar at the bottom shows the "Threads" panel, which is currently empty. The "Stack" panel shows the current stack frame: "Main" at line 31 of "name\_hash.cpp".

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

The "Qt" sidebar on the left shows the project structure, including "name-hash.pro", "Headers", "Sources", "lib/StanfordCPPLib", "src", and "name\_hash.cpp". The "Open Documents" list shows "main.cpp" and "name\_hash.cpp".

The "Debug" sidebar at the bottom shows the "Threads" panel, which is currently empty. The "Stack" panel shows the current stack frame: "Main" at line 31 of "name\_hash.cpp".





Qt Creator IDE interface showing a C++ project named "name-hash". The main editor displays the source code for "name\_hash.cpp". A yellow smiley face sticker is placed over the top left of the IDE window.

The code in the editor is as follows:

```
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 = 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_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!)
```

The Variables panel on the right shows the following variables:

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

The Debug Console at the bottom shows the following output:

```
Threads: #1 name-hash Stopped: "end-stepping-range".
Level Function File Line
0 Main name_hash... 33
1 Main main.cpp 23
2 startupMain platform.cpp 2208
3 main name_hash... 27
```

The bottom status bar shows the following tabs: Issues, Search Results, Application Output, Compile Output, QML/JS Console, General Messages.





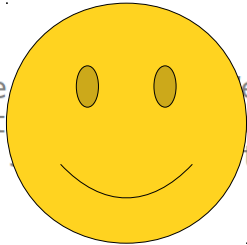
At this point, we've seen just about everything we care about. Rather than single-stepping all the way to the end, let's just tell the program to keep on running.

Qt Creator IDE interface showing a C++ project named "name-hash". The main editor displays the source code for "name\_hash.cpp". The code includes a namespace declaration, a prototype for the "nameHash" function, and the "main" function. The "main" function calls "nameHash" and prints the result. The "nameHash" function is partially visible, showing a comment about its purpose.

```
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!)
```

The Qt Creator interface also shows a "Projects" pane on the left with the project structure, an "Open Documents" pane, and a "Debug" pane at the bottom. The "Debug" pane shows the execution stack, indicating the program is stopped at line 33 of "name\_hash.cpp".

Level	Function	File	Line
0	Main	name_hash...	33
1	Main	main.cpp	23
2	startupMain	platform.cpp	2208
3	main	name_hash...	27





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!

Qt Creator IDE interface showing the 'name-hash' project. The console window displays the program's output:

```
What is your first name? Ada
What is your last name? Lovelace
The hash of your name is: 1967457
```

The source code for 'name\_hash.cpp' is visible in the background, showing a main function that prompts for a name and calculates a hash.

Debugger finished.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
-------	----------	------	------	--------	----------	------	------	---------	-----------	--------	---------

Qt Creator IDE interface showing the 'name-hash' project. The console window displays the program's output:

```
What is your first name? Ada
What is your last name? Lovelace
The hash of your name is: 1967457
```

The source code for 'name\_hash.cpp' is visible in the background, showing a main function that prompts for a name and calculates a hash.

Debugger finished.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
-------	----------	------	------	--------	----------	------	------	---------	-----------	--------	---------

Qt Creator IDE interface showing the 'name-hash' project. The console window displays the program's output:

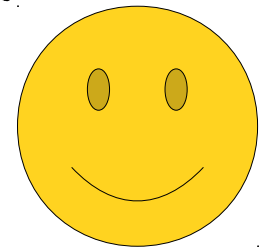
```
What is your first name? Ada
What is your last name? Lovelace
The hash of your name is: 1967457
```

The source code for 'name\_hash.cpp' is visible in the background, showing a main function that prompts for a name and calculates a hash.

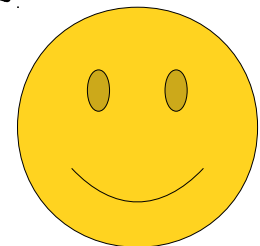
Debugger finished.

Level	Function	File	Line	Number	Function	File	Line	Address	Condition	Ignore	Threads
-------	----------	------	------	--------	----------	------	------	---------	-----------	--------	---------

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

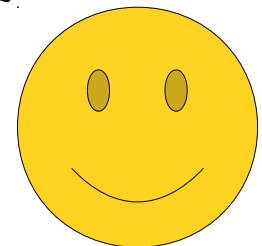


You know how to set a breakpoint to pause the program at a particular point.

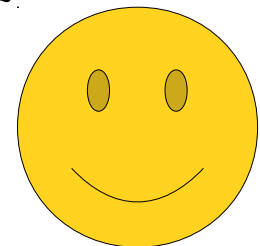




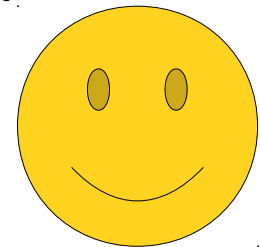
You know how to read the call stack and to  
see the values of local variables.



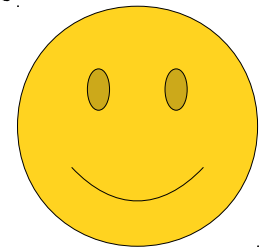
You know how to single-step the program and see what values change.



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!

